Trajectory Generation for Minimum Closed-Loop State Sensitivity

Paolo Robuffo Giordano[1], Quentin Delamare[1] and Antonio Franchi[2]

***Abstract*— In this paper we propose a novel general method to let a dynamical system fulfil at best a control task when the nominal parameters are not perfectly known. The approach is based on the introduction of the novel concept of *closed-loop sensitivity*, a quantity that relates parameter variations to deviations of the closed-loop trajectory of the system/controller pair. This new definition takes into account the dependency of the control inputs from the system states and nominal parameters as well as from the controller dynamics. The reference trajectory to be tracked is taken as optimization variable, and the dynamics of both the sensitivity and of its gradient are computed analytically along the system trajectories. We then show how this computation can be effectively exploited for solving trajectory optimization problems aimed at generating a reference trajectory that minimizes a norm of the closed-loop sensitivity. The theoretical results are validated via an extensive campaign of Monte Carlo simulations for two relevant robotic systems: a unicycle and a quadrotor UAV.**

I. INTRODUCTION

One of the major challenges for automatic/intelligent machines is the need to operate in uncertain real world conditions, with robotics being perhaps the discipline mostly concerned by this (unavoidable) issue. Indeed, robot decisions and control actions are, directly or indirectly, based on some model of the 'world' which is, unavoidably, only an approximation of the reality. A classical example is that of parametric uncertainty in the context of execution of a motion task (e.g., a mobile robot needing to reach a location, or a manipulator needing to grasp an object). In these cases, a valid model of the robot may be available, but the parameters may be uncertain, thus potentially generating a substantial discrepancy between the ideal and real behavior of the controlled, i.e, closed-loop, system.

The main design philosophy in tackling this fundamental issue has in general been to devise controllers tailored to the specific system at hand, and able to ensure a stable and satisfactory behavior also in presence of model uncertainty. A first classical way, which stems from the adaptive control paradigm [1], is to estimate the nominal parameters online while the system is performing the control task. This is obtained by introducing an additional dynamics in the controller devoted to the parameter estimation. However, estimating the parameters *online* may not be an easy task, since it typically requires to determine, and follow, trajectories that are 'exciting' enough: these trajectories may be cumbersome or even in conflict with the main control task. Furthermore, the parameters estimation dynamics may introduce unwanted transient behaviors, and in general guaranteeing stability (and satisfactory performance) of the coupled system/estimation dynamics is far from trivial.

Another classical approach has been the use of robust controllers [2]: these are typically static control laws that ensure a certain degree of insensitivity to a bounded model uncertainty (e.g., H-infinity loop shaping, valid only for linear systems, or sliding mode control, applicable also to nonlinear systems, but only with a sufficiently fast actuator dynamics). However, such controller needs to be specifically tailored for each considered system by exploiting the parametric uncertainty upper bound. Passivity-based methods [3] can also be considered as a robust control action exploiting some structural energetic properties of the system invariant to variations in the parameters. However, in most (if not all) of the aforementioned cases, and especially the last one, robustness is obtained at the expense of accuracy/performance in the execution of the control task. A tradeoff is indeed always present: controllers that can "exactly realize" a generic motion task must also rely on the assumption of perfect knowledge of the nominal parameters, while controllers that can provide some degree of robustness cannot then guarantee an exact/accurate execution of the control task, but, in many cases, only the so-called 'practical stabilization'.

The goal of this paper is to propose a new point of view that reverses the perspective on tackling motion control tasks under parametric uncertainties: rather than striving to design a sophisticated controller with some robustness guarantees for a specific system, we propose to attain robustness (for any choice of the control action) by suitably shaping the reference motion trajectory so as to minimize the state sensitivity to parameter uncertainty of the resulting *closed-loop* system. Indeed, we believe that a shift of focus from classical control design to a *control-aware* trajectory generation can result in a much more general approach compared to those focusing on the design of a specific controller performing well only for a restricted class of systems.

The proposed approach belongs to a recent trend whose goal is to improve, by means of trajectory optimization algorithms, the performance of a control task for uncertain dynamical systems. For instance, a large number of works has focused on the observability of the system states or the parameters of the model. In these works the typical goal is to find trajectories that enhance the state/parameter estimation convergence speed or accuracy, see, e.g., [4]–[12] and reference therein. However, estimating the parameters online can, again, introduce undesired transients and coupled estimation/system dynamics that can be hard to analyze. Other interesting recent works have instead focused on the generation of feedforward trajectories meant to minimize the state sensitivity of a system, see, e.g., [13]–[15] and

[1]CNRS, Univ Rennes, Inria, IRISA, Rennes, France, {prg,quentin.delamare}@irisa.fr
[2]LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France, antonio.franchi@laas.fr

references therein. However, the main limitation of these approaches is in their 'open-loop' nature, as they do not fully take into account the effects of parametric uncertainty on the coupled system/controller pair (i.e., the 'closed-loop' system).

The aim of this work is to extend the latter approaches for generating desired trajectories able to minimize the sensitivity of the *closed-loop* system to parametric uncertainties. The presented machinery (and associated optimization) is *agnostic* of the particular choice of control strategy in the sense that, given any choice of the control action, the proposed sensitivity minimization will optimize the tracking performance for the chosen system-controller pair. One can then, for instance, choose and fine tune a control law for a nominal model of the system, and then generate optimized trajectories that fulfill some given task and whose realization by the chosen controller results as insensitive as possible to variations in the nominal parameters. This is achieved by extending the 'open-loop' optimization proposed in [15] to the 'closed-loop' case, in which the coupling system/controller, and the dependency of the controller (in both its feedforward and feedback terms) on the (possibly uncertain) parameters, is fully taken into account.

The paper is structured as follows: in Sect. III we derive the closed-loop state sensitivity w.r.t. parametric variations along the system trajectories. This sensitivity is then exploited in Sect. IV for proposing two optimization problems for two representative cost functions, namely the terminal and the integral sensitivity. This machinery is then tested in two case studies involving a ground and aerial robot tasked to track a reference trajectory for their position (Sect. V). The proposed sensitivity optimization framework is validated via extensive Monte Carlo simulations in Sect. VI: the results fully support the ability of the sensitivity optimization to improve the *closed-loop* performance of the chosen tracking tasks. Finally, Sect. VII concludes the paper.

II. PROBLEM STATEMENT

Consider a generic class of nonlinear system

$$\dot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q},\, \boldsymbol{u},\, \boldsymbol{p}) \tag{1}$$

where $\boldsymbol{q} \in \mathbb{R}^{n_q}$ is the state, $\boldsymbol{u} \in \mathbb{R}^{n_u}$ is the input vector, and $\boldsymbol{p} \in \mathbb{R}^{n_p}$ the vector of *system* parameters of the class, whose real value is general not precisely known. With reference to a robot, these can include, for instance, mass, inertia, location of the center of mass, calibration parameters such as lengths, relative poses between sensors and actuators, actuator characteristics, and so forth.

Consider now a desired trajectory $\boldsymbol{r}_d(t) \in \mathbb{R}^{n_r}$, $t \in [t_0,\, t_f]$, which should be followed by a n_r-dimensional output function of the state $\boldsymbol{r}(\boldsymbol{q})$, where $n_r \leq n_q$. For example, $\boldsymbol{r}(\boldsymbol{q})$ could be the end-effector pose of a manipulator arm or the pose of a mobile robot. We assume that a tracking controller exists for solving the tracking problem $\boldsymbol{e}(t) = \boldsymbol{r}_d(t) - \boldsymbol{r}(\boldsymbol{q}(t)) \rightarrow \boldsymbol{0}$, This control action can, in general, also have internal states (e.g., an integral action, or dynamic extensions). The general form of the tracking controller can be written then as

$$\begin{cases} \dot{\boldsymbol{\xi}} &=& \boldsymbol{g}(\boldsymbol{\xi},\, \boldsymbol{q},\, \boldsymbol{r}_d(t),\, \boldsymbol{p}_c) \\ \boldsymbol{u} &=& \boldsymbol{h}(\boldsymbol{\xi},\, \boldsymbol{q},\, \boldsymbol{r}_d(t),\, \boldsymbol{p}_c) \end{cases}, \tag{2}$$

where $\boldsymbol{\xi} \in \mathbb{R}^{n_\xi}$ are the controller internal states. Here $\boldsymbol{p}_c \in \mathbb{R}^{n_p}$ represents the vector of *nominal* parameters used in the control loop which, in general, may differ from the 'real' value of $\boldsymbol{p}$ because of poor knowledge of the system model.

If $\boldsymbol{p} = \boldsymbol{p}_c$, then the controller is assumed to be able to perfectly realize the tracking task, in the sense that if $\boldsymbol{e}(t_0) = \dot{\boldsymbol{e}}(t_0) = \ddot{\boldsymbol{e}}(t_0) = \ldots = \boldsymbol{0}$, then $\boldsymbol{e}(t) \equiv \boldsymbol{0}\ \forall t \in [t_0,\, t_f]$. We then call this case the *nominal* one: the controller has perfect knowledge of the real system parameters and the tracking task can be exactly realized (i.e., the tracking error can remain identically zero under the correct choice of initial conditions for $\boldsymbol{\xi}(t_0)$). If, on the other hand, $\boldsymbol{p} \neq \boldsymbol{p}_c$, then the tracking task cannot (in general) be perfectly realized: the tracking error $\boldsymbol{e}(t)$ may not converge to zero or it may have unwanted transient phases (by first growing and then reducing again) because of the wrong value of the nominal parameters $\boldsymbol{p}_c$ w.r.t. the real value of $\boldsymbol{p}$. We call all these cases the *perturbed* ones.

The goal of this paper is to consider the optimization of the closed-loop performance of (1–2) in terms of minimizing the tracking error $\boldsymbol{e}(t)$ under possible parametric uncertainties, that is, discrepancies between the real (but possibly unknown) value of $\boldsymbol{p}$ and its nominal value $\boldsymbol{p}_c$ used in the control action. We note that, once a particular controller (2) is chosen, the optimization variables (free parameters) upon which one could still act for optimizing the closed-loop performance of system (1–2) are (i) the shape of the desired trajectory $\boldsymbol{r}_d(t)$ (possibly under constraints), (ii) the initial condition $\boldsymbol{\xi}(t_0)$ of the controller states, (iii) the control gains or other control parameters. Nevertheless, we consider here that the initial condition $\boldsymbol{\xi}(t_0)$ and the control gains are 'given' (e.g., because they have already been tuned for the nominal case). The only optimization variables left in our context are then the parameters describing the shape of $\boldsymbol{r}_d(t)$: therefore the problem addressed in this paper becomes the one of finding the *optimal desired trajectory* $\boldsymbol{r}_d(t)$ whose tracking by (1–2) will be *most insensitive* to discrepancies between $\boldsymbol{p}$ and $\boldsymbol{p}_c$.

We now detail the machinery needed to perform the sought optimization. For obtaining a finite-dimensional problem, we consider the desired trajectory $\boldsymbol{r}_d(t)$ to belong to a parametric class of curves $\boldsymbol{r}_d(t) = \boldsymbol{r}_d(\boldsymbol{a},\, t)$, where $\boldsymbol{a} \in \mathbb{R}^{n_a}$ represents the parameter vector describing a particular member in the chosen class of curves. Therefore, vector $\boldsymbol{a}$ becomes the optimization variable.

III. CLOSED-LOOP STATE SENSITIVITY

A. Definition

Consider the quantity

$$\boldsymbol{\Pi}(t) = \left.\frac{\partial \boldsymbol{q}(t)}{\partial \boldsymbol{p}}\right|_{\boldsymbol{p}=\boldsymbol{p}_c} \in \mathbb{R}^{n_q \times n_p} \tag{3}$$

evalauted along the closed-loop trajectories of (1–2). This represents the sensitivity of the state evolution w.r.t. changes in the parameter vector $\boldsymbol{p}$ evaluated on the *nominal* value $\boldsymbol{p} = \boldsymbol{p}_c$. Intuitively, minimization of some norm of $\boldsymbol{\Pi}(t)$ w.r.t. the optimization variables $\boldsymbol{a}$ would then result in a desired trajectory $\boldsymbol{r}_d(\boldsymbol{a}, t)$ whose tracking makes the *closed-loop* state evolution $\boldsymbol{q}(t)$ in the perturbed case ($\boldsymbol{p} \neq \boldsymbol{p}_c$) to be as close as possible to the evolution in the nominal case ($\boldsymbol{p} = \boldsymbol{p}_c$), assuming the perturbation is small enough. Since the nominal case implies a perfect tracking of the desired trajectory $\boldsymbol{r}_d(\boldsymbol{a}, t)$ by assumption, one would then obtain an improved tracking of $\boldsymbol{r}_d(\boldsymbol{a}, t)$ in the perturbed case too.

The quantity $\boldsymbol{\Pi}(t)$ cannot be, in general, computed in closed-form, but its dynamics can be given a closed-form expression. This has, for instance, been detailed in [15] for an *open-loop* version of the problem considered in this paper, in which the (indirect) dependency of the control inputs $\boldsymbol{u}(\cdot)$ on the system parameters is not taken into account. In this case, since $\boldsymbol{u}(\cdot)$ is considered independent from $\boldsymbol{p}$, the dynamics of $\boldsymbol{\Pi}$ takes the expression

$$\dot{\boldsymbol{\Pi}} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{q}}\boldsymbol{\Pi} + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{p}}, \qquad \boldsymbol{\Pi}(t_0) = \boldsymbol{0}, \tag{4}$$

where $\boldsymbol{f}(\cdot)$ is the system dynamics (1).

Evaluation of the *closed-loop* sensitivity considered in this paper is, however, more involved since the control input $\boldsymbol{u}(\cdot)$ is a function of both the system states $\boldsymbol{q}$ and the control states $\boldsymbol{\xi}$, see (2), which are, in turn, indirectly affected by variations in $\boldsymbol{p}$. Therefore, (4) must be suitably extended for considering this dependency. To this end, let

$$\boldsymbol{\Pi}_\xi(t) = \left.\frac{\partial \boldsymbol{\xi}(t)}{\partial \boldsymbol{p}}\right|_{\boldsymbol{p}=\boldsymbol{p}_c} \in \mathbb{R}^{n_\xi \times n_p} \tag{5}$$

represent the sensitivity of the *controller states* w.r.t. the parameter vector, again evaluated for $\boldsymbol{p} = \boldsymbol{p}_c$. By exploiting the fact that

$$\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{p}}\right) = \frac{\partial}{\partial \boldsymbol{p}}\dot{\boldsymbol{q}} \quad \text{and} \quad \frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial \boldsymbol{\xi}}{\partial \boldsymbol{p}}\right) = \frac{\partial}{\partial \boldsymbol{p}}\dot{\boldsymbol{\xi}} \tag{6}$$

and by leveraging (1–2), after some derivations one can obtain the following system of matrix differential equations[1]

$$\begin{cases} \dot{\boldsymbol{\Pi}} = \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{q}}\boldsymbol{\Pi} + \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{p}} + \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\left(\dfrac{\partial \boldsymbol{h}}{\partial \boldsymbol{q}}\boldsymbol{\Pi} + \dfrac{\partial \boldsymbol{h}}{\partial \boldsymbol{\xi}}\boldsymbol{\Pi}_\xi\right), \ \boldsymbol{\Pi}(t_0) = \boldsymbol{0} \\ \dot{\boldsymbol{\Pi}}_\xi = \dfrac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}}\boldsymbol{\Pi} + \dfrac{\partial \boldsymbol{g}}{\partial \boldsymbol{\xi}}\boldsymbol{\Pi}_\xi, \ \boldsymbol{\Pi}_\xi(t_0) = \boldsymbol{0} \end{cases}. \tag{7}$$

Therefore the evolution of $\boldsymbol{\Pi}(t)$ (and of $\boldsymbol{\Pi}_\xi(t)$) can be obtained by forward integration of (7) over the time interval of interest. Note that the terms $\partial \boldsymbol{h}/\partial \boldsymbol{p} = \boldsymbol{0}$ and $\partial \boldsymbol{g}/\partial \boldsymbol{p} = \boldsymbol{0}$ since $\partial \boldsymbol{p}_c/\partial \boldsymbol{p} = \boldsymbol{0}$: the sensitivity is evaluated w.r.t. variations in the system parameters $\boldsymbol{p}$, and the nominal parameters $\boldsymbol{p}_c$ are constant[2] w.r.t. variations in $\boldsymbol{p}$.

B. Gradient

Having addressed the evaluation of the (closed-loop) state sensitivity $\boldsymbol{\Pi}(t)$, we now discuss how to obtain an expression for the gradient of $\boldsymbol{\Pi}$ w.r.t. the optimization variables $\boldsymbol{a}$, that is, the quantity $\partial \boldsymbol{\Pi}/\partial \boldsymbol{a}$. Indeed, this gradient will be generally needed by any optimization procedure, such as the one detailed in the next Sect. IV. Note that $\partial \boldsymbol{\Pi}/\partial \boldsymbol{a}$ is a tensor quantity: for simplifying the derivations we then work out the expression for the gradient $\partial \boldsymbol{\Pi}/\partial a_i$ w.r.t. each individual i-th component of $\boldsymbol{a}$.

Let then

$$\boldsymbol{\Pi}_{a_i}(t) = \left.\frac{\partial \boldsymbol{\Pi}(t)}{\partial a_i}\right|_{\boldsymbol{p}=\boldsymbol{p}_c} \in \mathbb{R}^{n_q \times n_p} \tag{8}$$

be the sought (matrix) gradient of the system state sensitivity w.r.t. the optimization variable a_i, and let also

$$\boldsymbol{\Pi}_{\xi_{a_i}}(t) = \left.\frac{\partial \boldsymbol{\Pi}_\xi(t)}{\partial a_i}\right|_{\boldsymbol{p}=\boldsymbol{p}_c} \in \mathbb{R}^{n_\xi \times n_p} \tag{9}$$

be the (matrix) gradient of the controller state sensitivity w.r.t. a_i. Analogously to $\boldsymbol{\Pi}_\xi$, the quantity $\boldsymbol{\Pi}_{\xi_{a_i}}$ is introduced since it is needed for evaluating $\boldsymbol{\Pi}_{a_i}$.

We also define the following quantities

$$\boldsymbol{\Gamma}_i(t) = \left.\frac{\partial \boldsymbol{q}(t)}{\partial a_i}\right|_{\boldsymbol{p}=\boldsymbol{p}_c} \in \mathbb{R}^{n_q} \tag{10}$$

$$\boldsymbol{\Gamma}_{\xi_i}(t) = \left.\frac{\partial \boldsymbol{\xi}(t)}{\partial a_i}\right|_{\boldsymbol{p}=\boldsymbol{p}_c} \in \mathbb{R}^{n_\xi} \tag{11}$$

as the gradients of the *system/control* states w.r.t. changes in the optimization variables a_i. These quantities are also needed for evaluating $\boldsymbol{\Pi}_{a_i}$. By adopting the same reasoning leading to (7), one can show that the dynamics of $\boldsymbol{\Gamma}_i$ and $\boldsymbol{\Gamma}_{\xi_i}$ (along the system trajectories) takes the expression

$$\begin{cases} \dot{\boldsymbol{\Gamma}}_i = \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{q}}\boldsymbol{\Gamma}_i + \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\left(\dfrac{\partial \boldsymbol{h}}{\partial \boldsymbol{q}}\boldsymbol{\Gamma}_i + \dfrac{\partial \boldsymbol{h}}{\partial \boldsymbol{\xi}}\boldsymbol{\Gamma}_{\xi_i} + \dfrac{\partial \boldsymbol{h}}{\partial a_i}\right), \ \boldsymbol{\Gamma}_i(t_0) = \boldsymbol{0} \\ \dot{\boldsymbol{\Gamma}}_{\xi_i} = \dfrac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}}\boldsymbol{\Gamma}_i + \dfrac{\partial \boldsymbol{g}}{\partial \boldsymbol{\xi}}\boldsymbol{\Gamma}_{\xi_i} + \dfrac{\partial \boldsymbol{g}}{\partial a_i}, \ \boldsymbol{\Gamma}_{\xi_i}(t_0) = \boldsymbol{0} \end{cases}, \tag{12}$$

which allows evaluating $\boldsymbol{\Gamma}_i(t)$ and $\boldsymbol{\Gamma}_{\xi_i}(t)$ by forward integration analogously to $\boldsymbol{\Pi}(t)$ and $\boldsymbol{\Pi}_\xi(t)$ in (7).

For ease of notation we finally introduce several terms of interest: we first rewrite (7) in the more compact form

$$\begin{cases} \dot{\boldsymbol{\Pi}} = \boldsymbol{f}_{\boldsymbol{q}}\boldsymbol{\Pi} + \boldsymbol{f}_{\boldsymbol{p}} + \boldsymbol{f}_{\boldsymbol{u}}\left(\boldsymbol{h}_{\boldsymbol{q}}\boldsymbol{\Pi} + \boldsymbol{h}_{\boldsymbol{\xi}}\boldsymbol{\Pi}_\xi\right), \ \boldsymbol{\Pi}(t_0) = \boldsymbol{0} \\ \dot{\boldsymbol{\Pi}}_\xi = \boldsymbol{g}_{\boldsymbol{q}}\boldsymbol{\Pi} + \boldsymbol{g}_{\boldsymbol{\xi}}\boldsymbol{\Pi}_\xi, \ \boldsymbol{\Pi}_\xi(t_0) = \boldsymbol{0} \end{cases}, \tag{13}$$

and then define vector $\boldsymbol{u}_{a_i} \in \mathbb{R}^{n_u}$ as

$$\boldsymbol{u}_{a_i} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{q}}\boldsymbol{\Gamma}_i + \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{\xi}}\boldsymbol{\Gamma}_{\xi_i} + \frac{\partial \boldsymbol{h}}{\partial a_i} \tag{14}$$

from (12). Finally let $\boldsymbol{A}(\boldsymbol{x}) \in \mathbb{R}^{n_1 \times n_2}$ be a matrix function of a vector quantity $\boldsymbol{x} \in \mathbb{R}^{n_3}$: we will use the shorthand $\dfrac{\partial \boldsymbol{A}}{\partial \boldsymbol{x}} \circ \boldsymbol{b}$, with $\boldsymbol{b} \in \mathbb{R}^{n_3}$, to denote the tensor product

$$\frac{\partial \boldsymbol{A}}{\partial \boldsymbol{x}} \circ \boldsymbol{b} = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\left(\frac{\partial [\boldsymbol{A}(\boldsymbol{x})]_{ij}}{\partial \boldsymbol{x}}\boldsymbol{b}\right)\boldsymbol{\iota}_i\boldsymbol{\kappa}_j^T \in \mathbb{R}^{n_1 \times n_2}$$

where $[\boldsymbol{A}(\boldsymbol{x})]_{ij}$ is the (i, j)-th element of matrix $\boldsymbol{A}$, and $\boldsymbol{\iota}_i$ and $\boldsymbol{\kappa}_i$ are the i-th canonical basis vectors of $\mathbb{R}^{n_1}$ and $\mathbb{R}^{n_2}$, respectively.

[1]Note how (4) can be seen as a special case of (7) when $\partial \boldsymbol{u}(\cdot)/\partial \boldsymbol{p} = \boldsymbol{0}$.
[2]If the nominal parameters $\boldsymbol{p}_c$ were updated online, they would appear as additional control states in vector $\boldsymbol{\xi}$.

With all these settings, differentiating (7) w.r.t. a_i yields the following system

$$\begin{cases} \dot{\boldsymbol{\Pi}}_{a_i} = \left(\frac{\partial \boldsymbol{f}_{\boldsymbol{q}}}{\partial \boldsymbol{q}} \circ \boldsymbol{\Gamma}_i + \frac{\partial \boldsymbol{f}_{\boldsymbol{q}}}{\partial \boldsymbol{u}} \circ \boldsymbol{u}_{a_i}\right) \boldsymbol{\Pi} + \boldsymbol{f}_{\boldsymbol{q}} \boldsymbol{\Pi}_{a_i} + \frac{\partial \boldsymbol{f}_{\boldsymbol{p}}}{\partial \boldsymbol{q}} \circ \boldsymbol{\Gamma}_i + \\ \quad \frac{\partial \boldsymbol{f}_{\boldsymbol{p}}}{\partial \boldsymbol{u}} \circ \boldsymbol{u}_{a_i} + \left(\frac{\partial \boldsymbol{f}_{\boldsymbol{u}}}{\partial \boldsymbol{q}} \circ \boldsymbol{\Gamma}_i + \frac{\partial \boldsymbol{f}_{\boldsymbol{u}}}{\partial \boldsymbol{u}} \circ \boldsymbol{u}_{a_i}\right) (\boldsymbol{h}_{\boldsymbol{q}} \boldsymbol{\Pi} + \boldsymbol{h}_{\boldsymbol{\xi}} \boldsymbol{\Pi}_\xi) + \\ \quad \boldsymbol{f}_{\boldsymbol{u}} \left(\left(\frac{\partial \boldsymbol{h}_{\boldsymbol{q}}}{\partial \boldsymbol{q}} \circ \boldsymbol{\Gamma}_i + \frac{\partial \boldsymbol{h}_{\boldsymbol{q}}}{\partial \boldsymbol{\xi}} \circ \boldsymbol{\Gamma}_{\xi_i} + \frac{\partial \boldsymbol{h}_{\boldsymbol{q}}}{\partial a_i}\right) \boldsymbol{\Pi} + \right. \\ \quad \left. \left(\frac{\partial \boldsymbol{h}_{\boldsymbol{\xi}}}{\partial \boldsymbol{q}} \circ \boldsymbol{\Gamma}_i + \frac{\partial \boldsymbol{h}_{\boldsymbol{\xi}}}{\partial \boldsymbol{\xi}} \circ \boldsymbol{\Gamma}_{\xi_i} + \frac{\partial \boldsymbol{h}_{\boldsymbol{\xi}}}{\partial a_i}\right) \boldsymbol{\Pi}_\xi + \boldsymbol{h}_{\boldsymbol{q}} \boldsymbol{\Pi}_{a_i} + \boldsymbol{h}_{\boldsymbol{\xi}} \boldsymbol{\Pi}_{\xi_{a_i}} \right) \\ \dot{\boldsymbol{\Pi}}_{\xi_{a_i}} = \left(\frac{\partial \boldsymbol{g}_{\boldsymbol{q}}}{\partial \boldsymbol{q}} \circ \boldsymbol{\Gamma}_i + \frac{\partial \boldsymbol{g}_{\boldsymbol{q}}}{\partial \boldsymbol{\xi}} \circ \boldsymbol{\Gamma}_{\xi_i} + \frac{\partial \boldsymbol{g}_{\boldsymbol{q}}}{\partial a_i}\right) \boldsymbol{\Pi} + \\ \quad \left(\frac{\partial \boldsymbol{g}_{\boldsymbol{\xi}}}{\partial \boldsymbol{q}} \circ \boldsymbol{\Gamma}_i + \frac{\partial \boldsymbol{g}_{\boldsymbol{\xi}}}{\partial \boldsymbol{\xi}} \circ \boldsymbol{\Gamma}_{\xi_i} + \frac{\partial \boldsymbol{g}_{\boldsymbol{\xi}}}{\partial a_i}\right) \boldsymbol{\Pi}_\xi + \boldsymbol{g}_{\boldsymbol{q}} \boldsymbol{\Pi}_{a_i} + \boldsymbol{g}_{\boldsymbol{\xi}} \boldsymbol{\Pi}_{\xi_{a_i}} \end{cases} \tag{15}$$

with initial conditions $\boldsymbol{\Pi}_{a_i}(t_0) = \boldsymbol{0}$ and $\boldsymbol{\Pi}_{\xi_{a_i}}(t_0) = \boldsymbol{0}$.

Summarizing, in order to optimize some function of the state sensitivity matrix $\boldsymbol{\Pi}$, one can benefit from an expression of its gradient $\boldsymbol{\Pi}_{a_i}$ w.r.t. the optimization variables a_i. This gradient can be obtained by forward integrating (15) together with system (12) for obtaining $\boldsymbol{\Gamma}_i$ and $\boldsymbol{\Gamma}_{\xi_i}$, and system (7) for obtaining $\boldsymbol{\Pi}$ and $\boldsymbol{\Pi}_\xi$. Note that, as explained, one also needs to propagate the gradient of the controller state sensitivity $\boldsymbol{\Pi}_{\xi_{a_i}}$ for correctly evaluating $\boldsymbol{\Pi}_{a_i}$.

IV. SENSITIVITY OPTIMIZATION

For the sake of illustrating how one can exploit the closed-loop state sensitivity (and its gradient) introduced so far, we consider two possible optimization problems of interest: given the system dynamics (1), a reference trajectory $\boldsymbol{r}_d(\boldsymbol{a},\, t)$ defined over a given time interval $t \in [t_0,\, t_f]$, and the tracking controller (2), find the optimal parameter vector $\boldsymbol{a}_{opt}$ such that

$$\boldsymbol{a}_{opt} = \arg\min_{\boldsymbol{a} \in \mathcal{A}} \|\boldsymbol{\Pi}(t_f)\| \tag{16}$$

or

$$\boldsymbol{a}_{opt} = \arg\min_{\boldsymbol{a} \in \mathcal{A}} \int_{t_0}^{t_f} \|\boldsymbol{\Pi}(\tau)\| \mathrm{d}\tau \tag{17}$$

where $\|\cdot\|$ is a suitable norm for the state sensitivity matrix $\boldsymbol{\Pi}$, and $\mathcal{A}$ is the set of possible values for the optimization variables $\boldsymbol{a}$. The first problem (16) focuses on optimizing the (perturbed) tracking performance of $\boldsymbol{r}_d(\boldsymbol{a},\, t)$ at the final time t_f: this may be relevant when, for instance, needing to reach or grasp a specific object/location at t_f with high accuracy. The second problem aims at optimizing the average (perturbed) tracking performance of $\boldsymbol{r}_d(\boldsymbol{a},\, t)$ during the whole trajectory duration: this may be relevant when one wants to minimize deviations from the desired trajectory for all $t \in [t_0,\, t_f]$ (e.g., in order to avoid obstacle collisions).

Problems (16–17) are constrained minimization problems that can be addressed with any suitable off-the-shelf solver. In this work we adopt a simple gradient-based optimization algorithm for the sake of illustrating how to exploit the various quantities introduced in Sect. III. To this end, we consider a scenario in which initial and final values are given for $\boldsymbol{r}_d(\boldsymbol{a},\, t)$ and a number of its time derivatives (e.g., given initial/final position, velocity, acceleration, and so on). These constraints, defining the admissible set $\mathcal{A}$, can be written in matrix form as $\boldsymbol{M}\boldsymbol{a} = \boldsymbol{d}$, where $\boldsymbol{d}$ is the given set of initial and final values for $\boldsymbol{r}_d(\boldsymbol{a},\, t)$. Vector $\boldsymbol{a}$ can then be optimized with a null-space approach by starting from an initial guess satisfying the constraint and implementing the update law[3]

$$\dot{\boldsymbol{a}} = k_1 \boldsymbol{M}^\dagger (\boldsymbol{d} - \boldsymbol{M}\boldsymbol{a}) + k_2 (\boldsymbol{I} - \boldsymbol{M}^\dagger \boldsymbol{M}) \boldsymbol{\nu} \tag{18}$$

with vector $\boldsymbol{\nu} \in \mathbb{R}^{n_a}$ being the negative gradient of the cost functions in (16–17), and $k_1 > 0$, $k_2 > 0$ suitable gains.

As for the choice of an appropriate matrix norm $\|\cdot\|$, many possibilities exist (e.g., determinant, trace, condition number). In this work we chose the trace operator of the 'squared' version of matrix $\boldsymbol{\Pi}$, that is, $\|\boldsymbol{\Pi}\| = \frac{1}{2}\mathrm{Tr}(\boldsymbol{\Pi}^T\boldsymbol{\Pi})$ as matrix norm, thus resulting in

$$\boldsymbol{\nu}_i = -\frac{1}{2}\frac{\partial \mathrm{Tr}(\boldsymbol{\Pi}^T(t_f)\boldsymbol{\Pi}(t_f))}{\partial a_i} = -\mathrm{Tr}(\boldsymbol{\Pi}^T(t_f)\boldsymbol{\Pi}_{a_i}(t_f))$$

for Problem (16) and

$$\boldsymbol{\nu}_i = -\frac{1}{2}\frac{\partial \int_{t_0}^{t_f} \mathrm{Tr}(\boldsymbol{\Pi}^T(\tau)\boldsymbol{\Pi}(\tau))\mathrm{d}\tau}{\partial a_i} = -\frac{1}{2}\int_{t_0}^{t_f} \mathrm{Tr}\left(\frac{\partial \boldsymbol{\Pi}^T(\tau)\boldsymbol{\Pi}(\tau)}{\partial a_i}\mathrm{d}\tau\right) =$$
$$= -\,\mathrm{Tr}\left(\int_{t_0}^{t_f} \boldsymbol{\Pi}^T(\tau)\boldsymbol{\Pi}_{a_i}(\tau)\mathrm{d}\tau\right)$$

for Problem (17).

V. CASE STUDIES

We have considered two case studies for illustrating the results of the optimization problems (16–17): a unicycle (differential drive) and a planar quadrotor. This section discusses the detailed expressions of (1–2) for both robots, and the next section presents a number of simulation results.

A. Unicycle

Let $\boldsymbol{q} = [x\ y\ \theta] \in \mathbb{R}^3$ be the unicycle state in a world frame $\mathcal{F}_W = \{\boldsymbol{O}_W;\ \boldsymbol{x}_W,\ \boldsymbol{y}_W\}$, with $(x,\, y)$ being the planar position in $\mathcal{F}_W$ and θ the unicycle heading, see Fig. 1(right). Let also $(v,\, \omega)$ be the unicycle linear/angular velocities and $(\omega_R,\, \omega_L)$ the right/left wheel spinning velocities: as well-known, these are related by

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{2b} & -\frac{r}{2b} \end{bmatrix} \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} = \boldsymbol{S} \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix}$$

where r stands for the wheel radius and b for the distance between the wheels. The actual inputs of a unicycle (in the differential drive configuration) are $(\omega_R,\, \omega_L)$: we thus set $\boldsymbol{u} = [\omega_R\ \omega_L]^T$ as the unicycle control inputs. Because of this choice, any uncertainty in the calibration parameters r and b directly affects the system dynamics. Therefore we take $\boldsymbol{p} = [r\ b]^T$ as the vector of system parameters w.r.t. which the closed-loop state sensitivity will be evaluated With these settings, the unicycle dynamics has the expression

$$\dot{\boldsymbol{q}} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \boldsymbol{S}\boldsymbol{u} = \boldsymbol{f}(\boldsymbol{q},\, \boldsymbol{u},\, \boldsymbol{p}).$$

The chosen control task is that of letting the output $\boldsymbol{r}(\boldsymbol{q}) = [x\ y]^T \in \mathbb{R}^2$ (the unicycle planar position) tracking

[3]Since Problems (16) and (17) are in general non-convex, the update law (18) can only guarantee convergence towards a local minimum.

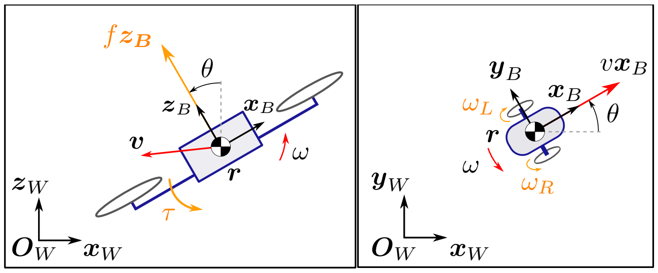

Fig. 1: Illustration of the main quantities characterizing the unicycle model (right) and quadrotor model (left)

a reference trajectory $\boldsymbol{r}_d(t) \in \mathbb{R}^2$. We solve this tracking control task by implementing a dynamic feedback linearization (DFL) controller (with integral action), with the aim of guaranteeing the best possible tracking performance in the nominal case ($\boldsymbol{p}_c = \boldsymbol{p}$), see, e.g., [16]. This control strategy can be summarized as follows: let $\boldsymbol{\xi} = [\xi_v \ \xi_x \ \xi_y]^T \in \mathbb{R}^3$ be the controller states, where ξ_v represents the dynamic extension of the unicycle linear velocity v, and (ξ_x, ξ_y) the states of the integral action. Differentiating twice the unicycle position $\boldsymbol{r}(\boldsymbol{q})$ yields

$$\ddot{\boldsymbol{r}} = \begin{bmatrix} \cos(\theta) & -\xi_v \sin(\theta) \\ \sin(\theta) & \xi_v \cos(\theta) \end{bmatrix} \begin{bmatrix} \dot{v} \\ \omega \end{bmatrix} = \boldsymbol{A}(\boldsymbol{q}, \boldsymbol{\xi}) \begin{bmatrix} \dot{v} \\ \omega \end{bmatrix}$$

Define the following vectors

$$\begin{cases} \dot{\boldsymbol{r}}_\xi = [\cos(\theta)\xi_v \ \ \sin(\theta)\xi_v]^T \\ \boldsymbol{\xi}_{xy} = [\xi_x \ \xi_y]^T \\ \boldsymbol{\eta} = \ddot{\boldsymbol{r}}_d + k_v(\dot{\boldsymbol{r}}_d - \dot{\boldsymbol{r}}_\xi) + k_p(\boldsymbol{r}_d - \boldsymbol{r}) + k_i \boldsymbol{\xi}_{xy} \end{cases} \tag{19}$$

where $k_v > 0$, $k_p > 0$ and $k_i > 0$ are suitable control gains. Following [16], the dynamics of control states can then be written as

$$\dot{\boldsymbol{\xi}} = \begin{bmatrix} [\ 1 \quad 0\ ]\boldsymbol{A}^{-1}\boldsymbol{\eta} \\ \boldsymbol{r}_d - \boldsymbol{r} \end{bmatrix} = \boldsymbol{g}(\boldsymbol{\xi}, \boldsymbol{q}, \boldsymbol{r}_d(t)) \tag{20}$$

and the unicycle control inputs as

$$\boldsymbol{u} = \boldsymbol{S}_c^{-1} \begin{bmatrix} \xi_v \\ [\ 0 \quad 1\ ]\boldsymbol{A}^{-1}\boldsymbol{\eta} \end{bmatrix} = \boldsymbol{h}(\boldsymbol{\xi}, \boldsymbol{q}, \boldsymbol{r}_d(t), \boldsymbol{p}_c), \tag{21}$$

Note that in (21) the 'calibration' matrix $\boldsymbol{S}_c$ is (correctly) evaluated on the nominal parameters $\boldsymbol{p}_c$.

B. Planar quadrotor

Let $\mathcal{F}_W = \{\boldsymbol{O}_W;\ \boldsymbol{x}_W,\ \boldsymbol{z}_W\}$ be a world frame and $\mathcal{F}_B = \{\boldsymbol{O}_B;\ \boldsymbol{x}_B,\ \boldsymbol{z}_B\}$ the body frame attached to the quadrotor center of mass, with $\boldsymbol{z}_B$ aligned with the thrust direction, see Fig. 1(left). In the planar quadrotor case the state consists of the quadrotor position $\boldsymbol{r} = (x, z)$ and linear velocity $\boldsymbol{v} = (v_x, v_z)$ in $\mathcal{F}_W$, and of the quadrotor body orientation θ and angular velocity ω with, thus, $\boldsymbol{q} = [\boldsymbol{r}^T \ \boldsymbol{v}^T \ \theta \ \omega]^T \in \mathbb{R}^6$. Similarly to the previous unicycle case, one can define the 'ideal' inputs (f, τ) (total thrust and torque) and the 'actual' inputs (w_R, w_L) (right/left propeller speeds): these are related by

$$\begin{bmatrix} f \\ \tau \end{bmatrix} = \begin{bmatrix} k_f & k_f \\ k_\tau & -k_\tau \end{bmatrix} \begin{bmatrix} w_R \\ w_L \end{bmatrix} = \boldsymbol{T} \begin{bmatrix} w_R \\ w_L \end{bmatrix}, \tag{22}$$

where k_f and k_τ are, in first approximation, calibration parameters depending on the propeller characteristics, see, e.g., [17]. Throughout the following developments, we will then take $\boldsymbol{u} = [w_R \ w_L]^T$ as the quadrotor control inputs.

The quadrotor dynamical model considered in this work is

$$\begin{cases} \dot{\boldsymbol{p}} &= \boldsymbol{v} \\ \dot{\boldsymbol{v}} &= \begin{bmatrix} 0 \\ -g \end{bmatrix} + \dfrac{f}{m} \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix} - \boldsymbol{B}(\theta)\boldsymbol{v} \\ \dot{\theta} &= \omega \\ \dot{\omega} &= \dfrac{\tau}{I} \end{cases} \tag{23}$$

with m and I being the quadrotor mass and inertia, and g the gravity acceleration magnitude. Note that the control inputs $\boldsymbol{u} = [w_R \ w_L]^T$ enter in (23) via (22). This then induces a dependency on the propeller characteristics (k_f, k_τ). Furthermore, the term $\boldsymbol{B}(\theta)\boldsymbol{v}$ is meant to model a *body-frame* air drag with possible different magnitudes along the horizontal/vertical quadrotor axes $\boldsymbol{x}_B$ and $\boldsymbol{z}_B$. Letting $\boldsymbol{R}(\theta) \in SO(2)$ be the 2D rotation matrix from $\mathcal{F}_W$ to $\mathcal{F}_B$, matrix $\boldsymbol{B}(\theta)$ is defined as

$$\boldsymbol{B}(\theta) = \boldsymbol{R}(\theta) \begin{bmatrix} b_x & 0 \\ 0 & b_z \end{bmatrix} \boldsymbol{R}^T(\theta) \tag{24}$$

where $b_x \geq 0$ and $b_z \geq 0$ are the body-frame drag coefficients along $\boldsymbol{x}_B$ and $\boldsymbol{z}_B$, respectively. Finally, the system parameter vector considered for the closed-loop sensitivity optimization is taken as

$$\boldsymbol{p} = \left[\frac{k_f}{m} \quad \frac{k_\tau}{I} \quad b_x \quad b_z\right]^T \in \mathbb{R}^4.$$

Indeed, as clear from (22–23), the quadrotor dynamics is only affected by the ratios k_f/m and k_τ/I and not by the individual values of these parameters.

The control task is, again, tracking of a reference trajectory $\boldsymbol{r}_d(t)$ for the quadrotor position $\boldsymbol{r}$. Analogously to the unicycle case, we implemented a DFL controller (with integral term) for obtaining the best possible tracking performance of $\boldsymbol{r}_d(t)$ in the nominal case ($\boldsymbol{p}_c = \boldsymbol{p}$), see, e.g., [18]. In the quadrotor case, the controller states are $\boldsymbol{\xi} = [\xi_f \ \xi_{df} \ \xi_x \ \xi_z] \in \mathbb{R}^4$, where ξ_f and ξ_{df} represent the two dynamic extensions of the thrust input f, and (ξ_x, ξ_z) are again the states of the integral action.

The proposed controller can be briefly described as follows: differentiating four times the quadrotor position $\boldsymbol{r}$ yields

$$\ddddot{\boldsymbol{r}} = \boldsymbol{A}(\boldsymbol{q}, \boldsymbol{\xi}, \boldsymbol{p}_c) \begin{bmatrix} \ddot{f} \\ \tau \end{bmatrix} + \boldsymbol{b}(\boldsymbol{q}, \boldsymbol{\xi}, \boldsymbol{p}_c) \tag{25}$$

where the detailed expressions of $\boldsymbol{A}(\boldsymbol{q}, \boldsymbol{\xi}, \boldsymbol{p}_c) \in \mathbb{R}^{2\times 2}$ and $\boldsymbol{b}(\boldsymbol{q}, \boldsymbol{\xi}, \boldsymbol{p}_c) \in \mathbb{R}^2$ can be found in [18]. Let now

$$\begin{cases} \ddot{\boldsymbol{r}}_\xi &= \begin{bmatrix} 0 \\ -g_c \end{bmatrix} + \dfrac{\xi_f}{m_c} \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix} - \boldsymbol{B}(\theta)\boldsymbol{v} \\ \dddot{\boldsymbol{r}}_\xi &= \dfrac{\xi_{df}}{m_c} \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix} - \dfrac{\xi_f}{m_c} \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \omega - \dot{\boldsymbol{B}}\boldsymbol{v} - \boldsymbol{B}\ddot{\boldsymbol{r}}_\xi \\ \boldsymbol{\xi}_{xz} &= [\xi_x \ \xi_z]^T \\ \boldsymbol{\eta} &= \ddddot{\boldsymbol{r}}_d + k_j(\dddot{\boldsymbol{r}}_d - \dddot{\boldsymbol{r}}_\xi) + k_a(\ddot{\boldsymbol{r}}_d - \ddot{\boldsymbol{r}}_\xi) + k_v(\dot{\boldsymbol{r}}_d - \boldsymbol{v}) + \\ &\quad + k_p(\boldsymbol{r}_d - \boldsymbol{r}) + k_i \boldsymbol{\xi}_{xz} \end{cases} \tag{26}$$

where $k_j > 0$, $k_a > 0$, $k_v > 0$, $k_d > 0$, $k_p > 0$ and $k_i > 0$ are suitable control gains. The dynamics of the control states can be written as

$$\begin{bmatrix} \dot{\xi}_f \\ \dot{\xi}_{df} \\ \dot{\boldsymbol{\xi}}_{xz} \end{bmatrix} = \begin{bmatrix} \xi_{df} \\ [\,1 \quad 0\,]\,\boldsymbol{A}^{-1}(\boldsymbol{\eta} - \boldsymbol{b}) \\ \boldsymbol{r}_d - \boldsymbol{r} \end{bmatrix} = \boldsymbol{g}(\boldsymbol{\xi},\, \boldsymbol{q},\, \boldsymbol{r}_d(t),\, \boldsymbol{p}_c) \tag{27}$$

and the quadrotor control inputs are given by

$$\boldsymbol{u} = \boldsymbol{T}_c^{-1} \begin{bmatrix} \xi_f \\ [\,0 \quad 1\,]\,\boldsymbol{A}^{-1}(\boldsymbol{\eta} - \boldsymbol{b}) \end{bmatrix} = \boldsymbol{h}(\boldsymbol{\xi},\, \boldsymbol{q},\, \boldsymbol{r}_d(t),\, \boldsymbol{p}_c). \tag{28}$$

VI. RESULTS

We now discuss a statistical analysis in which we considered four possible state sensitivities:

1) Unicycle — sensitivity of the state $\boldsymbol{q}$ w.r.t. the two wheel parameters $(r,\, b)$;
2) Quadrotor — sensitivity of the state $\boldsymbol{q}$ w.r.t. the two drag parameters $(b_x,\, b_y)$;
3) Quadrotor — sensitivity of the state $\boldsymbol{q}$ w.r.t. the 'mass' parameter k_f/m;
4) Quadrotor — sensitivity of the state $\boldsymbol{q}$ w.r.t. the 'inertia' parameter k_τ/I.

Other combinations are clearly possible, e.g., state sensitivity w.r.t. all the parameters $\boldsymbol{p}$ at the same time.
For each of these cases we further considered four subcases: DFL controller *with* or *without* integral action (denoted as I and NI), and optimization of problem (16) or of problem (17) (denoted as TF and TI). For the ease of exposition we will then refer to an individual subcase with the code i-A-B, where $i = 1\ldots4$ refers to the four considered state sensitivities, A$\in\{$I, NI$\}$ to the presence/absence of the integral term in the DFL controller, and B$\in\{$TF, TI$\}$ to the optimization problem (16) or (17). Therefore, as illustration, 3-I-TF will denote the quadrotor state sensitivity w.r.t. k_f/m evaluated for the DFL controller with integral action and optmized at t_f as in (16).

Each of the sixteen combinations i-A-B was tested by running $N = 1000$ simulations in which the system under consideration (unicycle or quadrotor) was either tracking a non-optimal reference trajectory $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{n\_opt})$, where $\boldsymbol{a}_{n\_opt}$ was simply taken as $\boldsymbol{a}_{n\_opt} = \boldsymbol{M}^\dagger \boldsymbol{d}$ (see (18)), or the optimal one $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{opt})$ obtained by the solution of problem (16) or (17) (depending on the subcase) with initial guess $\boldsymbol{a} = \boldsymbol{a}_{n\_opt}$. In all cases, a polynomial of order 15 in the variable t was considered for each component of $\boldsymbol{r}_d(t,\, \boldsymbol{a})$.

In each run the model parameter(s) under consideration were generated by randomly perturbing the (true) system values. In particular, all nominal parameters except the drag coefficients were drawn from a uniform distribution with range 80% to 120% of the true system values. The drag parameters were instead drawn from a uniform distribution with range $[0\ 0.2]$.
The non-optimal reference trajectory $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{n\_opt})$ was always the same across all subcases i-A-B: a rest-to-rest motion with initial/final zero velocities, accelerations, jerks and snaps (the latter two only for the quadrotor case), and lasting 5 [s]. The optimized trajectory $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{opt})$ was, instead, different for each subcase i-A-B because of the different conditions tested, but it was obviously the same across the 1000 runs of each subcase. Finally, the DFL control gains were the same across all subcases (one set for the I condition with $k_i > 0$, and another set for the NI condition with $k_i = 0$), and chosen so as to obtain real and negative closed-loop poles.

Consider now a particular subcase i-A-B tested over the N runs and let:

- $\boldsymbol{q}^{n\_opt}_{nom}(t)$ represent the (closed-loop) state evolution in the *nominal* case ($\boldsymbol{p}_c = \boldsymbol{p}$) when tracking the *non-optimal* reference trajectory $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{n\_opt})$ ($\boldsymbol{q}^{n\_opt}_{nom}(t)$ is the same for all N runs);
- $\boldsymbol{q}^{n\_opt}_{pert,\,k}(t)$ represent the (closed-loop) state evolution in the k-th *perturbed* run ($\boldsymbol{p}_c \neq \boldsymbol{p}$), $k = 1\ldots N$, when again tracking the *non-optimal* reference trajectory $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{n\_opt})$;
- $\boldsymbol{q}^{opt}_{nom}(t)$ represent the (closed-loop) state evolution in the *nominal* case ($\boldsymbol{p}_c = \boldsymbol{p}$) when tracking the *optimal* reference trajectory $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{opt})$ ($\boldsymbol{q}^{opt}_{nom}(t)$ is the same for all N runs);
- $\boldsymbol{q}^{opt}_{pert,\,k}(t)$ represent the (closed-loop) state evolution in the k-th *perturbed* run ($\boldsymbol{p}_c \neq \boldsymbol{p}$), $k = 1\ldots N$, when again tracking the *optimal* reference trajectory $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{opt})$.

Finally, define the state evolution errors $\boldsymbol{e}^{n\_opt}_k(t) = \boldsymbol{q}^{n\_opt}_{nom}(t) - \boldsymbol{q}^{n\_opt}_{pert,\,k}(t)$ in the *non-optimal* case and $\boldsymbol{e}^{opt}_k(t) = \boldsymbol{q}^{opt}_{nom}(t) - \boldsymbol{q}^{opt}_{pert,\,k}(t)$ in the *optimal* case, and consider the quantities

$$\begin{cases} E^{n\_opt}_{TF,\,k} &= \|\boldsymbol{e}^{n\_opt}_k(t_f)\| \\ E^{n\_opt}_{TI,\,k} &= \int_{t_0}^{t_f} \|\boldsymbol{e}^{n\_opt}_k(\tau)\| \mathrm{d}\tau \\ E^{opt}_{TF,\,k} &= \|\boldsymbol{e}^{opt}_k(t_f)\| \\ E^{opt}_{TI,\,k} &= \int_{t_0}^{t_f} \|\boldsymbol{e}^{opt}_k(\tau)\| \mathrm{d}\tau \end{cases} \tag{29}$$

Let us focus on problem (16) (the other one being equivalent): if tracking the optimized trajectory $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{opt})$ obtained from (16) results in a smaller sensitivity norm $\|\boldsymbol{\Pi}(t_f)\|$ at t_f (as claimed), then one should expect the non-optimal state error norm $E^{n\_opt}_{TF,\,k}$ to be 'statistically larger' than the optimal $E^{opt}_{TF,\,k}$ over the N runs. In other words, the perturbed state evolution should consistently deviate less at t_f from the nominal one when following the optimal reference trajectory $\boldsymbol{r}_d(t,\, \boldsymbol{a}_{opt})$. Analogous considerations clearly hold for problem (17) and the quantities $E^{n\_opt}_{TI,\,k}$, $E^{opt}_{TI,\,k}$ as well.

Figures 2–5 illustrate the results of this statistical analysis for all the considered subcases across the N runs[4]. In particular, Fig. 2 reports the normalized histograms of $E^{opt}_{TF,\,k}$ (blue) vs. $E^{n\_opt}_{TF,\,k}$ (orange) for the cases 1-NI-TF (top left) and 1-I-TF (top right), and of $E^{opt}_{TI,\,k}$ (blue) vs. $E^{n\_opt}_{TI,\,k}$ (orange) for the cases 1-NI-TI (bottom left) and 1-I-TI (bottom right). The following Figs. 3–5 follow exactly the same

[4]A selection of some representative runs are shown in the attached video.

Case 1	μ^{opt}	$\mu^{n\text{-}opt}$	%	σ^{opt}	$\sigma^{n\text{-}opt}$	%
1-NI-TF	0.011	0.162	1373.8%	0.008	0.1065	1237.3%
1-I-TF	0.0077	0.097	1150.2%	0.0053	0.053	910%
1-NI-TI	0.119	0.138	16%	0.064	0.078	21.5%
1-I-TI	0.049	0.075	54.1%	0.026	0.044	66.1%

Case 2	μ^{opt}	$\mu^{n\text{-}opt}$	%	σ^{opt}	$\sigma^{n\text{-}opt}$	%
2-NI-TF	0.035	0.138	290.6%	0.016	0.048	192.4%
2-I-TF	0.022	0.086	292%	0.011	0.034	209.4%
2-NI-TI	0.40	0.462	15.2%	0.155	0.176	13.6%
2-I-TI	0.192	0.264	37.2%	0.0759	0.102	34%

Case 3	μ^{opt}	$\mu^{n\text{-}opt}$	%	σ^{opt}	$\sigma^{n\text{-}opt}$	%
3-NI-TF	1.286	1.332	3.4%	0.752	0.781	3.7%
3-I-TF	0.046	0.103	119.5%	0.032	0.06	87.9%
3-NI-TI	4.96	5	0.91%	2.97	3	0.98%
3-I-TI	1.572	1.575	0.18%	0.941	0.942	0.17%

Case 4	μ^{opt}	$\mu^{n\text{-}opt}$	%	σ^{opt}	$\sigma^{n\text{-}opt}$	%
4-NI-TF	0.001	0.007	523.8%	0.0007	0.0045	515.8%
4-I-TF	0.0013	0.007	449.3%	0.0008	0.004	396%
4-NI-TI	0.0193	0.029	49.8%	0.0113	0.017	49.6%
4-I-TI	0.012	0.017	41%	0.0065	0.001	53%

TABLE I: Mean/standard deviations (μ^{opt}, σ^{opt}) and ($\mu^{n\text{-}opt}$, $\sigma^{n\text{-}opt}$) of the various histograms shown in Figs. 2–5 together with the relative improvements in the optimal vs. non-optimal cases.

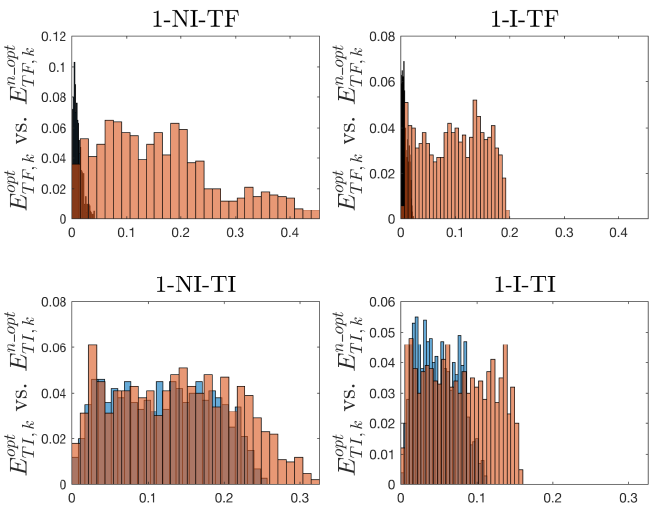

Fig. 2: Case 1: unicycle – sensitivity of $\boldsymbol{q}$ w.r.t. (r, b). Top row: $E^{opt}_{TF,k}$ (blue histogram) vs. $E^{n\text{-}opt}_{TF,k}$ (orange histogram) for the cases 1-NI-TF (left) and 1-I-TF (right). Bottom row: $E^{opt}_{TI,k}$ (blue histogram) vs. $E^{n\text{-}opt}_{TI,k}$ (orange histogram) for the cases 1-NI-TI (left) and 1-I-TI (right).

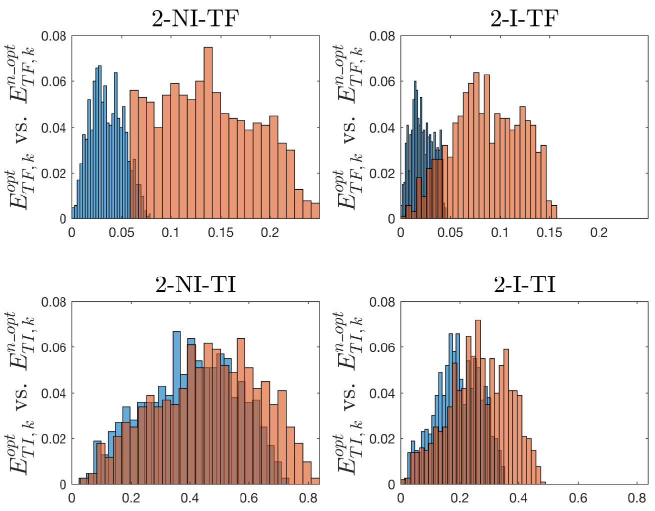

Fig. 3: Case 2: quadrotor – sensitivity of $\boldsymbol{q}$ w.r.t. (b_x, b_y). Top row: $E^{opt}_{TF,k}$ (blue histogram) vs. $E^{n\text{-}opt}_{TF,k}$ (orange histogram) for the cases 2-NI-TF (left) and 2-I-TF (right). Bottom row: $Ei^{opt}_{TI,k}$ (blue histogram) vs. $E^{n\text{-}opt}_{TI,k}$ (orange histogram) for the cases 2-NI-TI (left) and 2-I-TI (right).

pattern for the remaining cases 2, 3, 4. These histograms are normalized so that the height of each bin represents

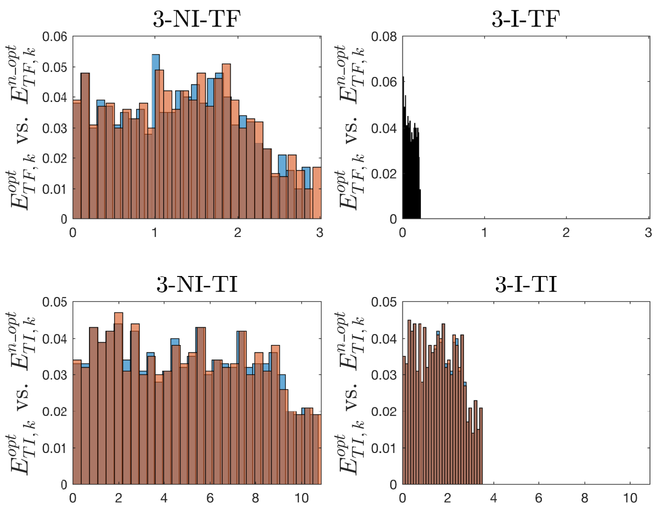

Fig. 4: Case 3: quadrotor – sensitivity of $\boldsymbol{q}$ w.r.t. k_f/m. Top row: $E^{opt}_{TF,k}$ (blue histogram) vs. $E^{n\text{-}opt}_{TF,k}$ (orange histogram) for the cases 3-NI-TF (left) and 3-I-TF (right). Bottom row: $E^{opt}_{TI,k}$ (blue histogram) vs. $E^{n\text{-}opt}_{TI,k}$ (orange histogram) for the cases 3-NI-TI (left) and 3-I-TI (right).

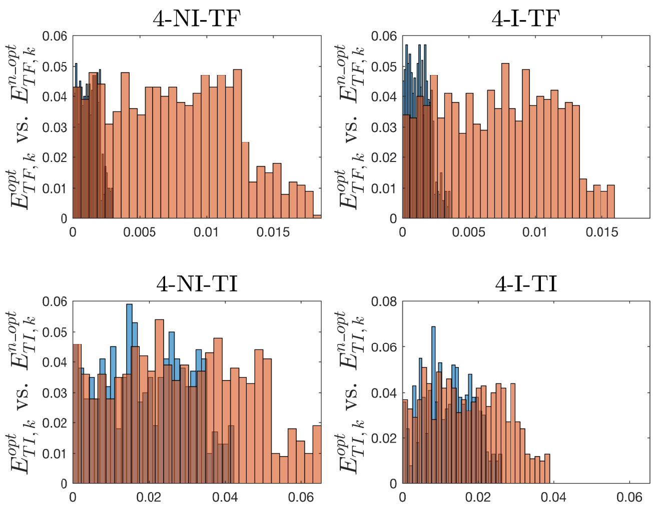

Fig. 5: Case 4: quadrotor – sensitivity of $\boldsymbol{q}$ w.r.t. k_τ/I. Top row: $E^{opt}_{TF,k}$ (blue histogram) vs. $E^{n\text{-}opt}_{TF,k}$ (orange histogram) for the cases 4-NI-TF (left) and 4-I-TF (right). Bottom row: $E^{opt}_{TI,k}$ (blue histogram) vs. $E^{n\text{-}opt}_{TI,k}$ (orange histogram) for the cases 4-NI-TI (left) and 4-I-TI (right).

the probability of having a tracking error norm that falls within the bin bounds[5]. Furthermore, Table I reports for all tested conditions the mean/standard deviations (μ^{opt}, σ^{opt}) and ($\mu^{n\text{-}opt}$, $\sigma^{n\text{-}opt}$) of the various histograms shown in Figs. 2–5, together with the relative improvements in the optimal vs. non-optimal cases.

We can then note the following facts: the state error norms in the optimal cases *always* resulted smaller (in both the mean and variance) w.r.t. the non-optimal cases in *all* the tested conditions. Therefore, the proposed optimization of the reference trajectory $\boldsymbol{r}_d(t, \boldsymbol{a})$ was able to reduce the average tracking error (at t_f or over the whole trajectory) by also making it more predictable (by reducing its variance). Note that this is true not only for all conditions NI (no integral term), as one could have expected, but also for all conditions I (integral term): therefore, despite the beneficial action of an integral term in compensating for parametric uncertainties, the proposed optimization is still able to further improve the overall tracking performance. Furthermore, one can also note how the improvements in the tracking error performance (both mean and variance) are always larger in the TF conditions than in the TI conditions. This can be explained as follows: in the TF conditions, the optimization has the possibility to generate a suitable 'maneuver' for eventually recovering the tracking error norm at the final time t_f (while possibly accepting an increased tracking error before t_f). On the other hand, the TI conditions weight the tracking error norm over the *whole trajectory*, thus leaving less room for the optimization to improve the tracking performance (e.g., contrarily to the TF cases, a maneuver that temporarily increases the tracking error would result in a poor final performance).

Coming to the individual cases, we can note that in cases 1, 2 and 4 the sensitivity optimization is quite consistently able to produce a significant improvement in the tracking error norm performance in all conditions (with higher/lower improvements depending on the specific cases). The same is not true, however, for case 3: here, only the I-TF condition resulted in a significant improvement of the tracking error norm (119% in mean and 87% in variance), while the other conditions had negligible improvements. This result can be explained by considering that case 3 involved the quadrotor state sensitivity w.r.t. the 'mass' parameter k_f/m, and variations in this parameter directly affect the possibility for compensating for the gravitational acceleration $[0\ g]^T$ in (23) (a *constant* drift term). In the NI conditions the DFL controller cannot compensate for $[0\ g]^T$ *whatever* the shape of the reference trajectory. On the other hand, in the I-TF condition the optimization has the possibility to produce a 'maneuver' that suitably slows down the quadrotor before reaching the final pose at t_f: this maneuver grants enough time to the integral term for compensating from the wrong k_f/m and, thus, allow to subsequently reach the correct pose at t_f. Indeed, note that in case 4 (sensitivity w.r.t. a similar parameter, the 'inertia' k_τ/I), the optimization can significantly improve the performance in all conditions since, in this case, no drift term is present in the states directly affected by k_τ/I.

[5]Therefore, the histograms can be seen as an approximation of the 'probability distribution' of the tracking error norms resulting from the parameters being drawn from a uniform distribution.

VII. CONCLUSIONS

We believe that the reported results provide a strong and successful validation of the proposed closed-loop sensitivity minimization for the sake of rendering a given system/control pair as insensitive as possible to parametric uncertainties. Future developments will involve considering optimization problems more complex than (16–17) by, e.g., taking also into account limited actuation or other concurrent objectives (e.g., minimize energy or time). We are also working towards an experimental validation of the approach.

REFERENCES

[1] K. J. Astrom and B. Wittenmark, *Adaptive Control*, 2nd ed. Prentice Hall, 1994.

[2] K. Zhou and J. C. Doyle, *Essentials of Robust Control*, 1st ed. Prentice Hall, 1997.

[3] B. Brogliato, B. Maschke, R. Lozano, and O. Egeland, "Passivity-based control," in *Dissipative Systems Analysis and Control*, ser. Communications and Control Engineering. Springer, 2007.

[4] J. van den Berg, P. Abbeel, and K. Goldberg, "QG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information," *The International Journal of Robotics Research*, vol. 30, no. 7, p. 895913, 2011.

[5] S.Ponda, R.Kolacinski, and E.Frazzoli, "Trajectory optimization for target localization using small unmanned aerial vehicles," in *AIAA Guidance, Navigation, and Control Conference*, 2012.

[6] A. Censi, A. Franchi, L. Marchionni, and G. Oriolo, "Simultaneous maximum-likelihood calibration of odometry and sensor parameters," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 475–492, 2013.

[7] B. T. Hinson, M. K. Binder, and K. A. Morgansen, "Path planning to optimize observability in a planar uniform flow field," in *American Control Conference*, 2013, p. 13921399.

[8] A. D. Wilson, J. A. Schultz, and T. D. Murphey, "Trajectory optimization for well-conditioned parameter estimation," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 1, pp. 28–36, 2015.

[9] A. Franchi, A. Petitti, and A. Rizzo, "Decentralized parameter estimation and observation for cooperative mobile manipulation of an unknown load using noisy measurements," in *2015 IEEE Int. Conf. on Robotics and Automation*, Seattle, WA, May 2015, pp. 5517–5522.

[10] L. M. Miller, Y. Silverman, M. A. MacIver, and T. D. Murphey, "Ergodic exploration of distributed information," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 36–52, 2016.

[11] K. Hausmana, J. Preiss, G. S. Sukhatme, , and S. Weiss, "Observability- aware trajectory optimization for self-calibration with application to uavs," *IEEE Robotics and Automation Letters*, 2017.

[12] P. Salaris, R. Spica, P. Robuffo Giordano, and P. Rives, "Online optimal active sensing control," in *IEEE Int. Conf. on Robotics and Automation*, 2017.

[13] S. Candido and S. Hutchinson, "Minimum uncertainty robot path planning using a pomdp approach," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2010, p. 14081413.

[14] ——, "Minimum uncertainty robot navigation using information-guided pomdp planning," in *IEEE Int. Conf. on Robotics and Automation*, 2011, p. 61026108.

[15] A. Ansari and T. D. Murphey, "Minimum sensitivity control for planning with parametric and hybrid uncertainty," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 823–839, 2015.

[16] G. Oriolo, A. De Luca, and M. Vendittelli, "WMR control via dynamic feedback linearization: Design, implementation and experimental validation," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 6, pp. 835–852, 2002.

[17] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *IEEE Robotics & Automation Magazine*, vol. 19, no. 3, pp. 20–32, 2012.

[18] V. Mistler, A. Benallegue, and N. M'Sirdi, "Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback," in *Proc. of the 2001 IEEE Int. Worksop on Robot and Human Interactive Communication*, 2001, pp. 586–593.