Experimental Validation of Sensitivity-Aware Trajectory Planning for a Redundant Robotic Manipulator Under Payload Uncertainty

Ali Srour[1], Antonio Franchi[2,3], Paolo Robuffo Giordano[1], Marco Cognetti[4]

***Abstract*—In this paper, we experimentally validate the recent concepts of *closed-loop state and input sensitivity* in the context of robust manipulation control for a robot manipulator. Our objective is to assess how optimizing trajectories with respect to sensitivity metrics can enhance the closed-loop system's performance w.r.t. model uncertainties, such as those arising from payload variations during precise manipulation tasks. We conduct a series of experiments to validate our optimization approach across different trajectories, focusing primarily on evaluating the precision of the manipulator's end-effector at critical moments where high accuracy is essential. Our findings offer valuable insights into improving the closed-loop robustness of the robot's state and inputs against physical parametric uncertainties that could otherwise degrade the system's performance.**

***Index Terms*—Optimization and Optimal Control; Planning under Uncertainty; Manipulation Planning**

I. Introduction

ROBOTS are now essential in daily applications, creating a growing demand for improving their robustness and accuracy. With the surge in online shopping, there exists a growing push to automate package handling which can be of various sizes and weights using robots. Thus, ensuring safe, reliable, and accurate package handling is crucial to enhance robot efficiency and meet operational demands effectively. Real-world physical perturbations can occur if the payload package is not fully grasped or if the dynamic parameters, such as mass and inertia, are not measured precisely. Accurate measurement and compensation of these payload parameters are essential in the robot model for precise manipulation.

Manuscript received: July, 13, 2024; Revised September, 27, 2024; Accepted November, 21, 2024.

This paper was recommended for publication by Editor L.Pallottino upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the project ANR-20-CE33-0003 "CAMP", the Horizon EU research and innovation programme [grant agreement No. 101120732] AUTOASSESS, and the Chaire de Professeur Junior grant no. ANR-22-CPJ1-0064-01. Experiments presented in this paper were carried out thanks to a platform of the Robotex 2.0 French research infrastructure.

[1]A. Srour and P. Robuffo Giordano are with CNRS, Univ Rennes, Inria, IRISA, Campus de Beaulieu, 35042 Rennes Cedex, France. email: {ali.srour,prg}@irisa.fr

[2]A. Franchi is with the Robotics and Mechatronics Department, Electrical Engineering, Mathematics, Computer Science (EEMCS) Faculty, University of Twente, 7500 AE Enschede, The Netherlands. email: a.franchi@utwente.nl

[3]A. Franchi is also with the Department of Computer, Control and Management Engineering, Sapienza University of Rome, 00185 Rome, Italy. email: antonio.franchi@uniroma1.it

[4]M. Cognetti is with LAAS-CNRS, Université de Toulouse, CNRS, UPS, Toulouse, France LAAS-CNRS, email: mcognetti@laas.fr

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

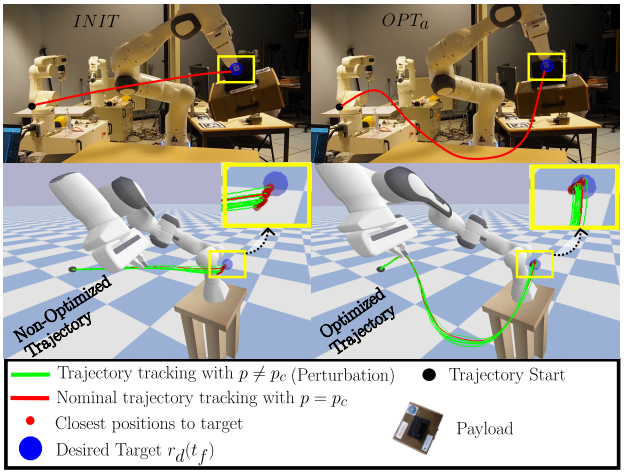

Fig. 1. This figure illustrates the comparison between a non-optimized trajectory ($INIT$ on the left) and one from our framework (OPT_a on the right), designed for robustly handling parametric payload uncertainties of the payload (i.e., the box grasped by the robot). Nominal trajectories (i.e., when the parameters are at their nominal values) are displayed in red while the ones with altered parameters are depicted in green. The aim is to maximize the robot end-effector precision at a target point (blue dot). The closest positions (red dots) are the robot's final positions across 15 real-world experiments, showing improved precision using our approach. A video of experiments is available at https://youtu.be/5dukBiuuOBs.

Furthermore, the payload may also have an influence on collision detection [1] and system safety [2]. Estimating payload package parameters, as described in [1], [3], involves using optimization problems or filtering methods along with proprioceptive sensors, as in [4]. In addition, some works, such as [5], employ vision and reinforcement learning to estimate the payload distribution. However, these methods often necessitate performing pre-task motions (under some excitation requirements), which can delay the manipulation task or, anyway, create the need to perform cumbersome maneuvers that are not related to the task realization.

In the literature, adaptive control [6] and robust control [7] are used either to estimate payload parameters online or to balance performance and robustness in the face of payload parametric uncertainty. A robust control is typically designed to handle worst-case scenarios and large perturbations, with tuning gains optimized for such conditions. However, this trades off robustness with performance, and can lead to unnecessary poor results when the payload parameters happen to be close to their nominal values. Another approach involves using Model Predictive Control (MPC), as discussed in [8].

This method utilizes the system's dynamic model to predict its behavior. However, the effectiveness of MPC relies heavily on the accuracy of the system model and the precise measurement of its parameters. Any uncertainties in these parameters can significantly degrade the controller's performance. An interesting approach is in [9], where a motion planner and controller uses reachability analysis for non-prehensile manipulation of unsecured objects. Finally, a fast time-scaling optimization based on iterative learning is proposed in [10] for handling model and execution uncertainties in contact tasks.

Recent strategies to mitigate the effects of parameter uncertainty in a robot model, as introduced in [11]–[14], exploit the *closed-loop state/input sensitivity* metrics to evaluate the impact of model uncertainties on robot behavior. By designing an "optimized" feedforward desired trajectory that minimizes these sensitivity metrics, robustness against parametric uncertainties is intrinsically embedded in the reference trajectory itself, *without* the need of employing specific (e.g., robust) control strategies. Indeed, the sensitivity machinery is agnostic to the particular controller, which can also be the one provided by the robot manufacturer. These versatile concepts can be applied to any system or controller under very mild assumptions (differentiability of robot dynamics and control action).

Further advancements include the work in [15], which investigates optimal initialization of an energy tank for passivity-based control in the context of robotic manipulators pushing payloads, and [16], which examines the effects of controller selection, gain tuning, and reference trajectory design on reducing parametric sensitivity for quadrotor flight control. Recent experimental validation of these methods, conducted on a quadrotor with uncertain payloads, demonstrated enhanced system robustness when navigating through confined spaces, such as the center of a window [17].

Similar to previous works, [18] employed sensitivity analysis metrics to study how model parameters affect end-effector position deviations. These studies, conducted in an open-loop manner, primarily aimed to identify the most influential parameters impacting state deviations to improve robot design. Authors of [19]–[21] focused on optimizing throwing motions by minimizing the sensitivity to dynamic parameters, static joint friction, and initial positioning errors. Their goal was to achieve robust tossing configurations despite parametric uncertainties. However, they modeled the object being tossed as a point mass and used a linearized system model for the sensitivity calculations.

To the best of our knowledge, the concept of *closed-loop state/input sensitivity* has not previously been applied to the case of torque control of manipulator arms (with full dynamics model) in the presence of uncertain payloads. A significant contribution of this study is to introduce, for the first time, an experimental assessment and validation of sensitivity-based trajectory generation focused on torque-controlled manipulators. The experiments utilize the Franka Emika Panda Robot, a widely adopted 7-DOF torque-controlled manipulator in the research community. Our objective is to apply the *closed-loop state/input sensitivity* (and related measures) for evaluating how a proper reference trajectory planning, designed to be robust against uncertainties in payload parameters, can improve performance of a standard manipulator with a standard controller and achieve more precise manipulation.

The remainder of this paper is structured as follows: In Sect. II, we recall the main notions of *closed-loop state/input sensitivity* with details about the manipulator model and tracking controller. In Sect. III, we present the reader with the optimization problem proposed. Sect. IV discusses the simulation and the experimental results on the controller performance for three different case studies. Finally, Sect. V concludes the paper.

II. METHODOLOGY

We consider the dynamic model of a robot in joint space that can be expressed by the following Euler-Lagrange equation:

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u}, \tag{1}$$

where $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \mathbf{R}^{n_q}$ are the joint angles, velocities and accelerations of the robot, respectively, $\boldsymbol{B}(\boldsymbol{q}) \in \mathbf{R}^{n_q \times n_q}$, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) \in \mathbf{R}^{n_q \times n_q}$, and $\boldsymbol{g}(\boldsymbol{q}) \in \mathbf{R}^{n_q}$ are the positive definite inertia matrix, the Coriolis matrix and the configuration dependent gravity vector, and $\boldsymbol{u} \in \mathbf{R}^{n_u}$ is the vector of motor torques. Finally, n_q corresponds to the robot's total degrees of freedom (DoF) and n_u to the dimension of the input space.

A. Load Dynamic Parameters

Consider the manipulator modeled as in (1) whose last end-effector reference frame is referred to as n_e . The robot has to manipulate a payload $\mathcal{L}$ whose dynamic parameters are the mass $m_{\mathcal{L}}$, the center of mass (**CoM**) denoted as $\boldsymbol{r}_{c\mathcal{L}}$, and the symmetric inertia tensor $\boldsymbol{I}_{\mathcal{L}}$, all expressed at the last frame n_e. When the robot is grasping the payload, the latter can be seen as an additional link of the robot, modifying the last link dynamic parameters as [1]:

$$\begin{cases} m_{n_e} \rightarrow m_{n_e} + m_{\mathcal{L}} \\ r_{c_{xn_e}} \rightarrow \dfrac{r_{c_{xn_e}} m_{n_e} + r_{c_{x\mathcal{L}}} m_{\mathcal{L}}}{m_{n_e} + m_{\mathcal{L}}} \\ r_{c_{yn_e}} \rightarrow \dfrac{r_{c_{yn_e}} m_{n_e} + r_{c_{y\mathcal{L}}} m_{\mathcal{L}}}{m_{n_e} + m_{\mathcal{L}}} \\ r_{c_{zn_e}} \rightarrow \dfrac{r_{c_{zn_e}} m_{n_e} + r_{c_{z\mathcal{L}}} m_{\mathcal{L}}}{m_{n_e} + m_{\mathcal{L}}} \end{cases} \tag{2}$$

$$\boldsymbol{I}_{ne} \rightarrow \boldsymbol{I}_{ne} + \boldsymbol{I}_{\mathcal{L}} \tag{3}$$

where $m_{n_e}, \boldsymbol{r}_{c_{n_e}}, \boldsymbol{I}_{n_e}$ are the mass, center of mass and the inertia tensor of the last link of the robot. As depicted from (2), the modified **CoM** is computed by summing the robot last link's **CoM** $\boldsymbol{r}_{c_{n_e}} = (r_{c_{xn_e}}, r_{c_{yn_e}}, r_{c_{zn_e}})$ and the payload's **CoM** $\boldsymbol{r}_{c_{\mathcal{L}}} = (r_{c_{x\mathcal{L}}}, r_{c_{y\mathcal{L}}}, r_{c_{z\mathcal{L}}})$, weighted by their respective masses and both expressed in the end-effector frame n_e. For the calculation of the modified inertia in (3), a direct sum of the robot link's inertia tensor $\boldsymbol{I}_{n_e}$ and the payload's inertia tensor $\boldsymbol{I}_{\mathcal{L}}$ is only valid if both tensors are represented in the same reference frame. If the payload's inertia tensor is defined in a frame attached to its **CoM** – denoted as ${}^{\mathcal{L}}\boldsymbol{I}_{\mathcal{L}}$ – the Steiner theorem can be applied to convert it to the n_e frame

$$\boldsymbol{I}_{\mathcal{L}} = {}^{n_e}\mathbf{R}_{\mathcal{L}}\, {}^{\mathcal{L}}\boldsymbol{I}_{\mathcal{L}}\, {}^{n_e}\mathbf{R}_{\mathcal{L}}^{T} + m_{\mathcal{L}} \boldsymbol{S}^{T}(\boldsymbol{r}_{c_{\mathcal{L}}}) \boldsymbol{S}(\boldsymbol{r}_{c_{\mathcal{L}}}) \tag{4}$$

where ${}^{n_e}\mathbf{R}_{\mathcal{L}}$ is the rotation matrix between n_e and the payload's reference frame and $\boldsymbol{S}(\boldsymbol{r}_{c_{\mathcal{L}}})$ is the skew-symmetric matrix obtained from $\boldsymbol{r}_{c_{\mathcal{L}}}$.

B. Sensitivity Analysis

Notions of *closed-loop state/input sensitivity* were first introduced in [11], [12] and further developed in [13], [15]–[17], [22], [23]. These concepts are general but have been primarily applied to aerial robots in the context of navigation tasks. In this paper, we aim to illustrate how the concept of sensitivity can be adapted to consider uncertain payload parameters, denoted as $\boldsymbol{p}$, while accounting for the full dynamics of a manipulator robot. Specifically, we consider the following vector of parameters $\boldsymbol{p} = [m_{\mathcal{L}}, \boldsymbol{r}_{c\mathcal{L}}, \boldsymbol{i}_{\mathcal{L}}] \in \mathbf{R}^{10}$ [1] as *uncertain*[1], where $\boldsymbol{i}_{\mathcal{L}} = [I_{\mathcal{L}_{xx}}, I_{\mathcal{L}_{xy}}, I_{\mathcal{L}_{xz}}, I_{\mathcal{L}_{yy}}, I_{\mathcal{L}_{yz}}, I_{\mathcal{L}_{zz}}] \in \mathbf{R}^6$ is the vector containing all the lower triangular elements of $\boldsymbol{I}_{\mathcal{L}}$.

Modifying the last link parameters as shown in (2) and (3), the dynamic model of the robot becomes explicitly dependent on $\boldsymbol{p}$ and can be expressed as

$$\boldsymbol{B}(\boldsymbol{q},\boldsymbol{p})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{p})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q},\boldsymbol{p}) = \boldsymbol{u}. \tag{5}$$

Let $\boldsymbol{x} = [\boldsymbol{q}, \dot{\boldsymbol{q}}]^T \in \mathbb{R}^{2n_q}$ be the state of the system and $\boldsymbol{s}$ be a quantity of interest to be controlled. For example, it can be the pose of the robot end-effector. In this case, $\boldsymbol{s} = (\boldsymbol{r}_e, \boldsymbol{\phi}_e)$ where $\boldsymbol{r}_e$ is the EE position and $\boldsymbol{\phi}_e$ is the EE orientation (e.g., axis/angle as in our paper). We consider the tracking task of a desired trajectory $\boldsymbol{s}_d(\boldsymbol{a}, t)$ defined for $t \in [t_0, t_f]$ and function of a finite set of trajectory parameters $\boldsymbol{a} \in \mathbb{R}^{n_a}$. Even if the sensitivity matrix can be evaluated on any pair of systems/controllers, we focus our attention in this paper, without loss of generality, on a (standard) joint space torque-controller based on feedback linearization with null space projection:

$$\begin{aligned}\boldsymbol{u} = (\boldsymbol{J}_s\boldsymbol{B}(\boldsymbol{q},\boldsymbol{p}_c)^{-1})^{\dagger}\big(\ddot{\boldsymbol{s}}_d + \boldsymbol{D}_s(\dot{\boldsymbol{s}}_d - \dot{\boldsymbol{s}}) + \boldsymbol{K}_s(\boldsymbol{s}_d - \boldsymbol{s}) \\ + \boldsymbol{e}_{\boldsymbol{s}_{int}} - \dot{\boldsymbol{J}}_s(\boldsymbol{q})\dot{\boldsymbol{q}}\big) + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{p}_c) + \boldsymbol{g}(\boldsymbol{q},\boldsymbol{p}_c) + \boldsymbol{P}^{\perp}\boldsymbol{\tau}_{\boldsymbol{N}}.\end{aligned} \tag{6}$$

In the previous equation, $\boldsymbol{J}_s$ denotes the task Jacobian and $\boldsymbol{\tau}_{\boldsymbol{N}} = -k_d\boldsymbol{B}\dot{\boldsymbol{q}}, k_d > 0$ represents a secondary priority torque, designed to introduce damping into the robot's motion. This torque is projected into the null space of the primary (tracking) task using the projector $\boldsymbol{P}^{\perp} = (\mathbb{I} - \boldsymbol{J}_s^{\top}\bar{\boldsymbol{J}}_s^{\top})$, which is based on the dynamically consistent inertia-weighted pseudo-inverse [24]. In particular, $\bar{\boldsymbol{J}}_s = \boldsymbol{B}^{-1}\boldsymbol{J}_s^{\top}(\boldsymbol{J}_s\boldsymbol{B}^{-1}\boldsymbol{J}_s^{\top})^{-1}$, and $\boldsymbol{K}_s, \boldsymbol{D}_s$ are the diagonal matrices corresponding to the proportional and derivative gains respectively. Finally,

$$\boldsymbol{e}_{\boldsymbol{s}_{int}} = \boldsymbol{K}_I \int (\boldsymbol{s}_d - \boldsymbol{s})\mathrm{d}t \tag{7}$$

is an integral action with $\boldsymbol{K}_I$ a positive definite diagonal integral gain matrix. It is worth mentioning that the controller $\boldsymbol{u}$ in (6) is evaluated at the *nominal* values of the parameters $\boldsymbol{p}_c$ (which may differ from the actual $\boldsymbol{p}$ because of inaccuracies in the payload parameters).

[1]This is motivated by the fact that we consider the robot dynamic model (without payload) to be reasonably well identified via a preliminary calibration procedure. However, in our context, the payload to be grasped is considered coarsely known in advance and, therefore, with uncertain parameters.

Let $\boldsymbol{X} = [\boldsymbol{r}_e, \boldsymbol{v}_e]^T \in \mathbb{R}^6$ denote the end-effector (EE) state, where $\boldsymbol{r}_e \in \mathbb{R}^3$ represents its position and $\boldsymbol{v}_e \in \mathbb{R}^3$ its linear velocity. In this paper, we are interested in minimizing the sensitivity ${}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}}$ (at a specific time) of the end-effector state $\boldsymbol{X}$ w.r.t. the parameters $\boldsymbol{p}$. This is motivated by the fact that we aim to generate precise trajectories for the robot end-effector in the presence of parameter uncertainties.

The computation of this sensitivity can be divided into two steps: (i) the computation of the robot closed-loop sensitivity ${}^{\boldsymbol{x}}\boldsymbol{\Pi}_{\boldsymbol{p}}$ as in [11], [12]; and (ii) the computation of the sensitivity ${}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{x}}$ of the EE state $\boldsymbol{X}$ w.r.t. the robot state $\boldsymbol{x}$. The overall sensitivity can be written as

$${}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}} = {}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{x}}{}^{\boldsymbol{x}}\boldsymbol{\Pi}_{\boldsymbol{p}}\ , \tag{8}$$

where ${}^{\boldsymbol{x}}\boldsymbol{\Pi}_{\boldsymbol{p}}(t) = \frac{\partial \boldsymbol{x}(t)}{\partial \boldsymbol{p}}\Big|_{\boldsymbol{p}=\boldsymbol{p}_c}$ and ${}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{x}}(t) = \frac{\partial \boldsymbol{X}(t)}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{p}=\boldsymbol{p}_c}$. Furthermore, we also define the input sensitivity as (see [12])

$$\boldsymbol{\Theta}(t) = \frac{\partial \boldsymbol{u}(t)}{\partial \boldsymbol{p}}\Big|_{\boldsymbol{p}=\boldsymbol{p}_c}. \tag{9}$$

Matrix ${}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}}(t)$ quantifies how variations of the parameters $\boldsymbol{p}$, around a nominal value $\boldsymbol{p}_c$, will affect the evolution of the state $\boldsymbol{X}$ (in closed-loop). Analogously, $\boldsymbol{\Theta}(t)$ relates how variations of parameters $\boldsymbol{p}$ would affect the inputs $\boldsymbol{u}$. A closed-form expression for matrices ${}^{\boldsymbol{x}}\boldsymbol{\Pi}_{\boldsymbol{p}}(t)$ and $\boldsymbol{\Theta}(t)$ is not, in general, available. However, one can obtain a closed-form expression for their dynamics and thus obtain the behavior of ${}^{\boldsymbol{x}}\boldsymbol{\Pi}_{\boldsymbol{p}}(t)$ and $\boldsymbol{\Theta}(t)$ via forward integration of an ODE along the system trajectories [11], [12].

Matrices ${}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}}(t)$ and $\boldsymbol{\Theta}(t)$ can then be used for several purposes in the context of trajectory optimization, for obtaining suitable sensitivity metrics to be optimized or an estimation of the envelope of "perturbed trajectories" for the states/inputs. To this end, assume that each parameter p_i can vary in a given range δp_i centered at a nominal p_{c_i}

$$p_i \in [p_{c_i} - \delta p_i,\ p_{c_i} + \delta p_i] \tag{10}$$

and define the diagonal weight matrix as $\boldsymbol{W} = \mathrm{diag}(\delta p_i^2)$. Letting $\Delta\boldsymbol{p} = \boldsymbol{p} - \boldsymbol{p}_c$, an ellipsoid in the parameter space centered at $\boldsymbol{p}_c$ and with semi-axes δp_i can be expressed as

$$\Delta\boldsymbol{p}^T\boldsymbol{W}^{-1}\Delta\boldsymbol{p} \leq 1. \tag{11}$$

Following the derivations in [23], from (8)–(9) and (11), one can obtain the corresponding ellipsoid in the state space

$$\Delta\boldsymbol{X}^T({}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}}\boldsymbol{W}{}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}}^T)^{-1}\Delta\boldsymbol{X} \leq 1 \tag{12}$$

and in input space

$$\Delta\boldsymbol{u}^T(\boldsymbol{\Theta}\boldsymbol{W}\boldsymbol{\Theta}^T)^{-1}\Delta\boldsymbol{u} \leq 1, \tag{13}$$

where $\Delta\boldsymbol{X} = \boldsymbol{X} - \boldsymbol{X}_{nom}$, with $\boldsymbol{X}_{nom}$ the nominal EE state, and $\Delta\boldsymbol{u} = \boldsymbol{u} - \boldsymbol{u}_{nom}$, with $\boldsymbol{u}_{nom}$ being the evolution of (6) in the nominal case $\boldsymbol{p} = \boldsymbol{p}_c$.

The state and input space ellipsoids can be exploited to define a sensitivity-based norm and 'tubes' that envelope the behavior of the state/input trajectories in the perturbed cases. For example, one can consider the worst-case scenario corresponding to the largest deviation of the EE state $\boldsymbol{X}$ when

the parameters lie in their bounds as in (10). This corresponds to defining the norm as

$$\|^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}}\|_W = \lambda_{\max}(^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}}\boldsymbol{W}^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}}^T), \tag{14}$$

where $\lambda_{\max}(\boldsymbol{A})$ denotes the maximum eigenvalue of a matrix $\boldsymbol{A}$. Furthermore, one can also exploit (12)–(13) for obtaining the tubes of perturbed trajectories for the individual components of the EE states and the inputs. By referring to [23] for additional details, for each direction of interest in the input space, one can obtain the "tube radius" $r_i(t)$ as

$$u_{nom,i}(t) - r_i(t) \leq u_i(t) \leq u_{nom,i}(t) + r_i(t), \tag{15}$$

where $u_{nom,i}(t)$ is the behavior of the input $u_i(t)$ in the nominal case $\boldsymbol{p} = \boldsymbol{p}_c$. Equation (15) bounds from above/below the envelope of perturbed inputs when the parameter uncertainty is bounded as in (10), and an analogous upper/lower bound can also be obtained for the generic EE state component $X_i(t), i \in [1, ..., 6]$.

III. OPTIMIZATION PROBLEM

In this paper, we consider the following trajectory optimization problem

$$\begin{aligned} \boldsymbol{a}^* = \arg\min_{\boldsymbol{a}} \|^{\boldsymbol{X}}\boldsymbol{\Pi}_{\boldsymbol{p}}(\,\bar{t}\,)\|_W \\ s.t. \quad \boldsymbol{M}\boldsymbol{a} = \boldsymbol{b} \\ U_{min,i} \leq u_{nom,i}(t) - r_i(t)\ \forall i \in [1, ..., n_u],\ \forall t \in [t_0,\, t_f] \\ u_{nom,i}(t) + r_i(t) \leq U_{max,i}\ \forall i \in [1, ..., n_u],\ \forall t \in [t_0,\, t_f]. \end{aligned} \tag{16}$$

We seek the optimal value $\boldsymbol{a}^*$ of the shape parameter $\boldsymbol{a}$ of the reference trajectory $\boldsymbol{s}_d(\boldsymbol{a}\,, t)$ for minimizing the sensitivity norm (14) at a specific time $\bar{t}$. Minimization of this cost will increase the robustness (thus reducing the sensitivity) of the closed-loop system at $\bar{t}$ against parameter uncertainties of the payload. The constraints consist of given initial/final conditions for $\boldsymbol{s}_d(\boldsymbol{a}\,, t)$, represented by the linear constraints $\boldsymbol{M}\boldsymbol{a} = \boldsymbol{b}$, and constraints that bound the envelope of perturbed inputs within actuation limits $U_{min,i} \leq U_{max,i}$, ensuring that the tracking of the optimized reference trajectory will be feasible for any value of the uncertain parameters $\boldsymbol{p}$ in the range (10). Note that these constraints leverage the "input tubes" as described in (15). Additionally, constraints such as obstacle avoidance and workspace limitations can be incorporated by leveraging state tubes, as demonstrated in [13], [14]. Also, we focus only on the sensitivity of the end effector's (EE) position and linear velocity with respect to the uncertain payload parameters $\boldsymbol{p}$. While it is possible to extend this analysis to include the sensitivity of other states, such as the EE's orientation and angular velocity, doing so would increase computation time and, furthermore, it has been shown that for a tossing task, the most relevant states are the position/linear velocity w.r.t. the rotational ones [25]–[27].

IV. ANALYSIS

A series of simulations and experiments have been carried out to evaluate the effectiveness of the proposed trajectory generation derived from solving (16). Three trajectories $\boldsymbol{s}_d(\boldsymbol{a}, t) = (\boldsymbol{r}_d(\boldsymbol{a}, t), \boldsymbol{\phi}_d(\boldsymbol{a}, t))$ were obtained, where $\boldsymbol{r}_d(\boldsymbol{a}, t)$ represents the desired Cartesian EE position and $\boldsymbol{\phi}_d(\boldsymbol{a}, t)$ represents the desired EE orientation. These trajectories – hereafter referred to as Traj1, Traj2, and Traj3 – comply with the initial and final state constraints (e.g., initial/final position, velocity, and acceleration constraints) and input saturations as in (16).

In Traj1 and Traj2 the sensitivity of the EE position is minimized at the final time t_f by solving the optimization problem (16) for $\bar{t} = t_f$ in order to increase the accuracy of the EE position at t_f where the robot should arrive at rest (the orientation is left free to be determined by the optimizer). Conversely, in Traj3, the sensitivity of the EE position is minimized at a specific time $t_w < t_f$ that corresponds to the time at which the robot is passing over a point of interest while moving – by solving (16) for $\bar{t} = t_w$. Its objective is to increase the accuracy of the EE position and linear velocity while the robot is moving.

We start by generating a first trajectory that will be referred to as $INIT$ which is constructed using piecewise Bezier curves as discussed in [16], [17]. This trajectory is obtained through a preliminary optimization process that minimizes the *snap* of $\boldsymbol{s}_d(\boldsymbol{a}\,, t)$ over the whole time interval. The goal is to generate a smooth trajectory with minimal curvature changes while satisfying nominal state and input constraints. To achieve this objective, the $INIT$ trajectories are obtained by solving the following optimization problem

$$\begin{aligned} \boldsymbol{a} = \arg\min_{\boldsymbol{a}} \left(\int_{t_0}^{t_f} \left\| \frac{d^4\boldsymbol{r}_d(\boldsymbol{a}, \tau)}{dt^4} \right\|^2 + \left\| \frac{d^4\boldsymbol{\phi}_{\boldsymbol{d}}(\boldsymbol{a}, \tau)}{dt^4} \right\|^2 \mathrm{d}\tau \right) \\ s.t. \quad \boldsymbol{M}\boldsymbol{a} = \boldsymbol{b} \\ U_{min,i} \leq u_{nom,i}(t) \leq U_{max,i}\ \forall i \in [1, \ldots, n_u],\ \forall t \in [t_0,\, t_f]. \end{aligned} \tag{17}$$

Our framework then modifies this trajectory by solving (16) with $INIT$ as an initial guess. The resulting trajectory is denoted as OPT_a. More in detail, our framework is implemented in Python and utilizes the COBYLA [28] nonlinear optimizer from the nlopt toolbox by employing CasADi for symbolic system representation [29]. Within this framework, the offline optimization of trajectories, as described in Sect. II, typically requires an average of 30 minutes per trajectory.

Our main objective is to assess the improvements in accuracy obtained within our framework for reaching better target attainment (desired position $\boldsymbol{r}_d(\bar{t})$ and desired linear velocity $\boldsymbol{v}_d(\bar{t})$) at a specific time along the trajectory. We also want to evaluate the impact of the integral action in (7) for real-world applications. To this aim, we will compare the results in the case $\boldsymbol{K}_I = 0$ and $\boldsymbol{K}_I \neq 0$.

In the following subsections, the perturbations of the payload's parameters are introduced by uniformly sampling the inner volume of the ellipsoid (11) in simulation. However, in real-world experiments, the perturbations were obtained by actually altering the payload $\mathcal{L}$ by changing the mass and weight distribution which leads to modify the inertia as well (see examples in Fig. 2, and the video).

The nominal parameters were obtained from a CAD model of the package and experimental measurements. These parameters are defined as $\boldsymbol{p}_c = [m_{\mathcal{L}_c}, \boldsymbol{r}_{c\mathcal{L}_c}, \boldsymbol{i}_{\mathcal{L}_c}]$, where the

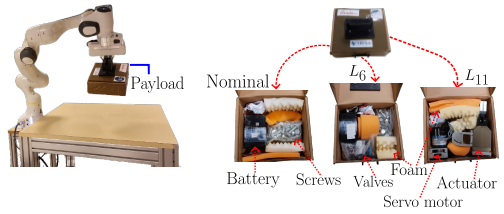

Fig. 2. Franka Emika Panda Robot while moving the box that contains a certain payload with a specific distribution. The nominal state of the payload is displayed (left) followed by the physical perturbation examples for the payloads L_6 and L_{11} aiming at changing the mass, center of the mass, and inertia of the payload.

nominal mass, the center of mass, and the inertia tensor are given by $m_{\mathcal{L}_c} = 2$ kg, $\boldsymbol{r}_{c\mathcal{L}_c} = [0.0, 0.0, 0.025]$ m, and $\boldsymbol{i}_{\mathcal{L}_c} = [7.4, -0.0001, 0.1, 11.4, -0.005, 1.6] \times 10^{-3}$ kg·m^2. Precise measurements of these parameters inside the robot model allow for precise manipulation. The uncertainty intervals δp_i for $\boldsymbol{W}$ are chosen relative to the nominal parameters $\boldsymbol{p}_c$ as $\delta m_{\mathcal{L}} = 0.2 m_{\mathcal{L}_c}$, $\delta \boldsymbol{r}_{cx\mathcal{L},cy\mathcal{L}} = 0.15l$, $\delta \boldsymbol{r}_{cz\mathcal{L}} = 0.05h$, and $\delta \boldsymbol{i}_{\mathcal{L}} = 0.2 \boldsymbol{i}_{\mathcal{L}_c}$, where $h = 7cm$ is the box height and $l = 20cm$ is the box length. Additionally, the gains used in the torque controller in (6) are defined as $\boldsymbol{K}_s = 480\, \boldsymbol{I}_6$, $\boldsymbol{D}_s = 60\, \boldsymbol{I}_6$, and $\boldsymbol{K}_I = 60\, \boldsymbol{I}_6$, where $\boldsymbol{I}_6$ is the 6×6 identity matrix.

A. *Sensitivity Optimization at $\bar{t} = t_f$*

In this section, we present the results when minimizing the sensitivity at $\bar{t} = t_f$. Fig. 3 shows the comparison between $INIT$ and OPT_a when no integral action is considered in the controller, i.e., $\boldsymbol{K}_I = 0$ in (7). We performed $N_{sim} = 30$ simulations ("Simulation Framework" label in Fig. 3) and $N_{exp} = 15$ real-world experiments ("Real Experiments" label in Fig. 3) where $\boldsymbol{p}$ was altered as described in Sect. IV.

In particular, Fig. 3 reports the results for Traj1 (1st and 3rd rows) and Traj2 (2nd and 4th rows). In the same figure, the nominal case trajectories (i.e., $\boldsymbol{p} = \boldsymbol{p}_c$) are reported in red while the "perturbed" ones (i.e., $\boldsymbol{p} \neq \boldsymbol{p}_c$) are depicted in green. The red dots represent, per each simulation/experiment, the point $\boldsymbol{r}_e(t_f)$ that is closest to the desired final position $\boldsymbol{r}_d(t_f)$. The results show that, in the nominal case, the robot is capable of tracking the reference trajectory with high accuracy. On the contrary, the tracking performance degrades when the payload parameters are altered, as can be seen by the sparsity of the final robot position in the $INIT$ column of Fig. 3. However, the performances improve with our approach, as evident by the OPT_a column of Fig. 3, where the red dots are much closer to the desired final position. This trend is also confirmed by Fig. 4 that shows the distance to the target (i.e., $||\boldsymbol{r}_d(t_f) - \boldsymbol{r}_e(t_f)||$) for the simulation (top) and experiment (bottom) cases. In particular, $INIT$ shows an average error of 0.012 m for Traj1 and 0.01 m for Traj2 in simulation. In contrast, the error is reduced to 0.008 m for Traj1 and 0.006 m for Traj2 for OPT_a. A similar behavior can be seen in real-world experiments. In fact, $INIT$ shows an average error of 0.022 m for Traj1 and 0.03 m for Traj2 with a large variance. On the contrary, OPT_a shows significantly better target attainment where the distance to the target drops to 0.011 m for Traj1 (50% increased accuracy, see Fig. 1) and

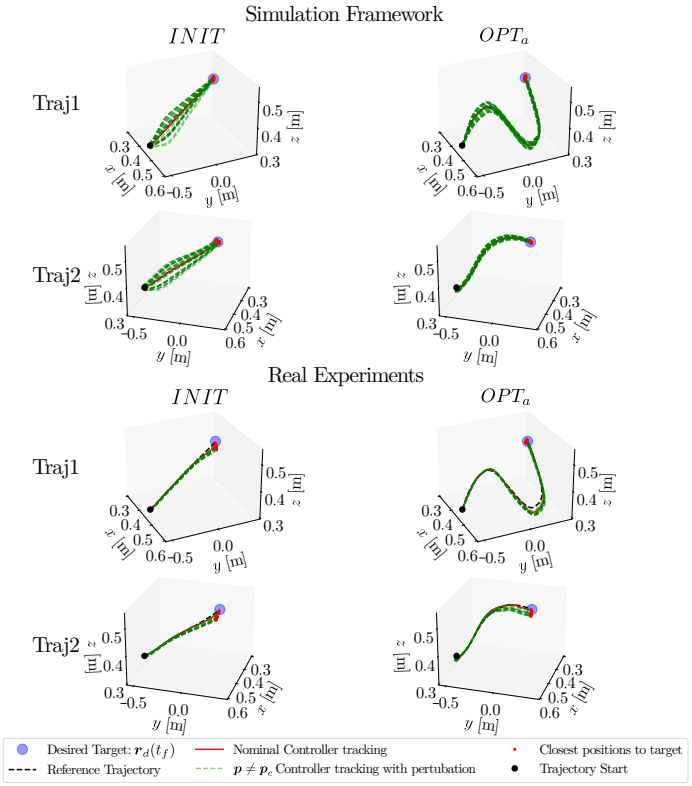

Fig. 3. Sensitivity results for $\bar{t} = t_f$ for two trajectories: Traj1 (1st and 3rd rows) and Traj2 (2nd and 4th rows) for $INIT$ (left column) and OPT_a (right column). The nominal trajectories are in red while the perturbed ones are shown in green. $N_{sim} = 30$ simulations and $N_{exp} = 15$ real-world experiments are reported. Red dots at $t = t_f$ mark the closest points of each trajectory to the desired target $\boldsymbol{r}_d(t_f)$. Parameters $\boldsymbol{p}$ were uniformly drawn in our simulation framework as in (11) and in experiments by modifying the box package as in Fig. 2.

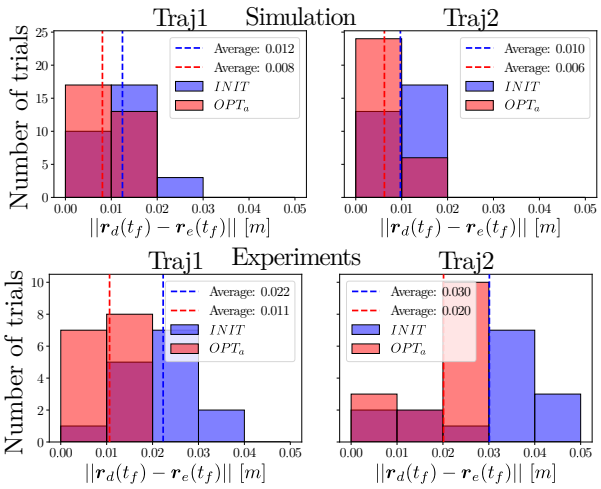

Fig. 4. Histograms of two trajectory types (left: Traj1, right: Traj2) for both cases ($INIT$ in blue, OPT_a in red. The "dark purple" is the overlap of the two colors) showing the distances to the desired target ($||\boldsymbol{r}_d(t_f) - \boldsymbol{r}_e(t_f)||$) in [m] and its average (dashed thick lines) across $N_{sim} = 30$ runs in simulation (top) and $N_{exp} = 15$ runs in experiments (bottom).

0.02 m for Traj2. Finally, it is worth mentioning that the variance is much smaller for the OPT_a case, confirming the improvements provided by our approach.

To further validate our results, we report in Fig. 5 the uncertainty tubes for the EE position over time for Traj1 in case of real-world experiments (a similar behavior is experienced for Traj2, omitted for the sake of space). In particular, the red solid line represents the nominal case, while the black solid lines correspond to the upper and lower bounds of the tubes.

From Fig. 5, it is clear that the tubes' size decreases notably at t_f. Specifically, OPT_a reduced – w.r.t. $INIT$ – the tube radius of 33% for x, 5% for y, and 17% for z of the robot EE position at t_f. This confirms the importance of minimizing the sensitivity to ensure robustness and enhance accuracy.

Finally, it is worth mentioning that the trajectory obtained by OPT_a often involves an initial maneuver followed by a vertical approach to the target location, either from above or below (see video and Figs. 1,3) while minimizing the sensitivity at $\bar{t} = t_f$. This behavior can be seen for both Traj1 and Traj2 and it has been empirically verified in most of the trajectories obtained from (16). One possible explanation is that a final vertical motion helps to better minimize the effects of uncertainties in the considered parameters $\boldsymbol{p}$.

B. Sensitivity Optimization at $\bar{t} = t_f$ with Integral Action

In this section, we aim to highlight the impact of the integral action in (7). In particular, we compare the $INIT$ and OPT_a approaches described so far with their counterparts including an integral action. These new variants – obtained considering $\boldsymbol{K}_I \neq 0$ in (7) – will be referred to as $INIT_i$ and OPT_{ai}, respectively. Furthermore, we explored two scenarios for the controller incorporating integral action: (a) the robot stops at t_f (as in $INIT$ and OPT_a); and (b) an additional time budget of 5 seconds is allocated, which significantly extends the total trajectory duration (2.2 s). This latter case is included to allow more time for the integral action to recover from uncertainties.

As shown in Figs. 6 and 7, OPT_a not only outperforms $INIT$ but also $INIT_i$ across both scenarios (without and with an additional time budget). The final positions achieved by OPT_a are consistently closer to the desired target $\boldsymbol{r}_d(t_f)$ and demonstrate less variance. Specifically, for the $INIT$ case, the average distance to $\boldsymbol{r}_d(t_f)$ is 0.022 m, which reduces to 0.017 m and further to 0.015 m for $INIT_i$ in scenarios (a) and (b) (without and with an additional time budget, as mentioned above), respectively. However, the performance of OPT_a is even better, achieving an average error of 0.011 m, while OPT_{ai} reduces this further to 0.007 m in scenario (a) and to 0.006 m in scenario (b). These results highlight the positive impact of the integral action and, more importantly, validate our approach. Even with the additional time budget, which provides the best outcome for $INIT_i$, OPT_a (without integral action and post-delay) still outperforms the best $INIT_i$ outcome. Furthermore, to provide a broader perspective on the results, Fig. 8 presents the outcomes of a simulation campaign involving 10 distinct trajectories with randomized target locations, covering both the $INIT_i$ and OPT_{ai} cases. For each case, we conducted $N_{sim} = 30$ perturbed runs. This

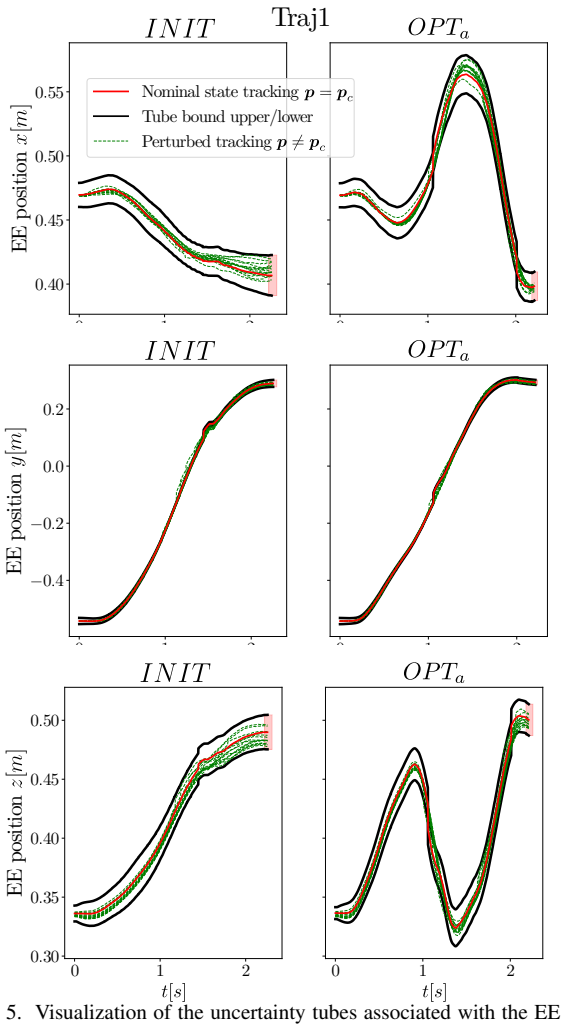

Fig. 5. Visualization of the uncertainty tubes associated with the EE position over time for $INIT$ (left) and OPT_a (right). The solid red lines denote the nominal case, while the solid black lines represent the upper/lower bounds for the tubes. The dashed green lines depict the N_{exp} experimental runs. As expected, the tubes are much smaller for OPT_a w.r.t. $INIT$ at $\bar{t} = t_f$.

simulation campaign supports the experimental results shown in Fig. 7, highlighting statistically similar improvements in terms of target reach. Specifically, the latter improved from 0.013 m for $INIT_i$ to 0.008 m for OPT_{ai} in Fig. 8, which is consistent with scenario (a) in Fig. 7, based on $N_{exp} = 15$.

C. Sensitivity Optimization at $\bar{t} = t_w$ (robot in motion)

In this section, we validate our framework in case the robot is passing over a point of interest while moving (e.g., for dropping or tossing a package). It is worth noting that the additional time budget discussed in Sect. IV-B would not make any difference for this scenario since the point of interest is in the middle of the trajectory and the robot should pass over it while moving. For this reason, for the cases that include an integral term ($INIT_i$ and OPT_{ai}), we will focus on the case where no additional time budget is given.

As it can be seen in Fig. 9, the average distance error is 0.014 m for $INIT_i$ and 0.012 m for OPT_{ai} in simulation. Moreover, the deviations from the desired velocity are 0.031 m/s for the $INIT_i$ and 0.023 m/s for OPT_{ai}.

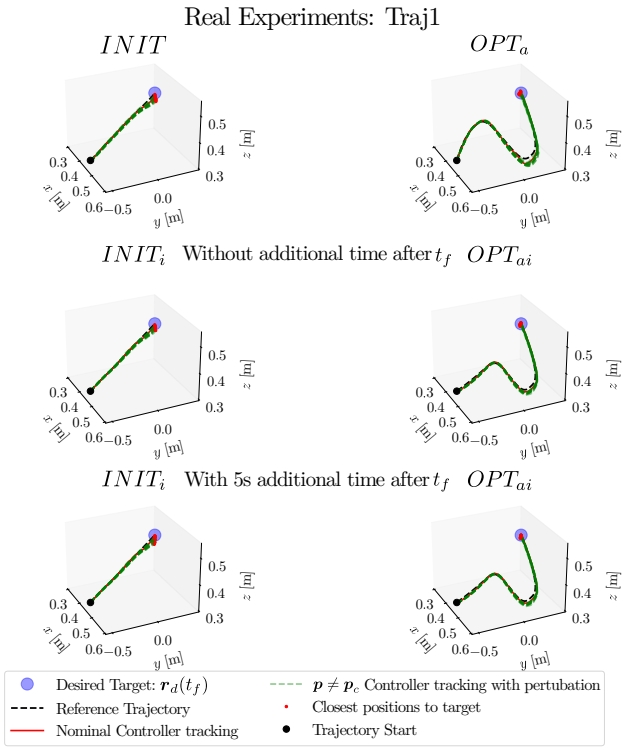

Fig. 6. Resulting trajectories for $N_{exp} = 15$ real-world experiments. First row: no integral term in (6): $INIT$ and OPT_a. Second row: integral term in (6) and robot stopping at t_f: $INIT_i$ and OPT_{ai}. Third row: integral term in (6), giving additional 5 s after t_f to the controller: $INIT_i$ and OPT_{ai}. Nominal trajectories are in red while the perturbed ones are in green. The red spheres at $\bar{t} = t_f$ depict the closest points to $\boldsymbol{r}_d(t_f)$ reached by the robot.

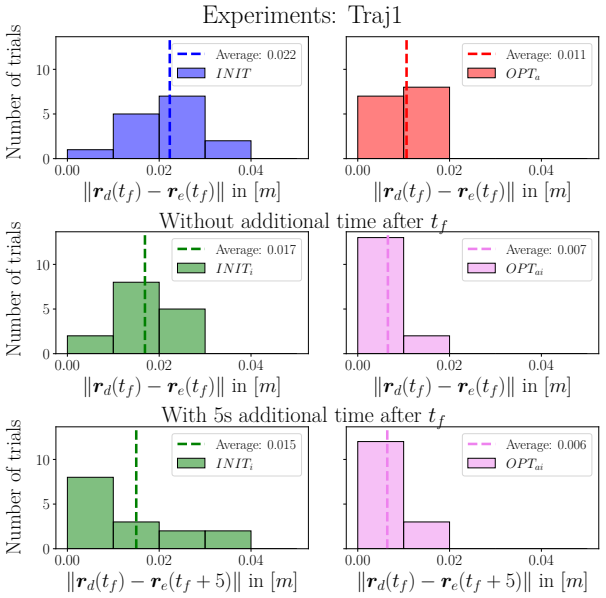

Fig. 7. Histograms of Traj1 for the four cases ($INIT$ in blue, $INIT_i$ in green, OPT_a in red, and OPT_{ai} in violet) showing the distances to the desired target $||\boldsymbol{r}_d(t_f) - \boldsymbol{r}_e(t_f)||$ in [m] and its average (dashed thick lines) across $N_{exp} = 15$ experiments. For the cases including an integral term ($INIT_i$ and OPT_{ai}), two scenarios are considered: (a) stopping the robot at t_f (as for $INIT$ and OPT_a) - second row; and (b) providing an additional time budget of 5 s - third row. As it is evident, OPT_{ai} obtains the best performance in terms of average error and variance, in particular when we donated an additional time budget to the integral term.

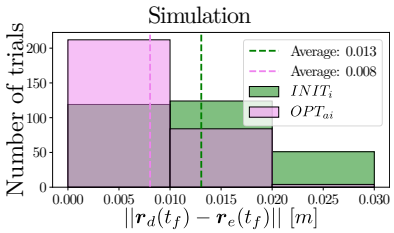

Fig. 8. Histogram illustrating the results for 10 distinct trajectories with randomized target locations for $INIT_i$ and OPT_{ai}. For each trajectory, $N_{sim} = 30$ perturbed runs were performed, showing the variation in the distance $\|\boldsymbol{r}_d(t_f) - \boldsymbol{r}_e(t_f)\|$ in [m] at the final time t_f. The vertical dashed lines indicate the average distance for $INIT_i$ and OPT_{ai}, offering a comparative insight into their respective performance.

Thus, OPT_{ai} shows slight improvements in terms of average error with less variance in velocity. However, in experimental results, the differences between the two approaches become more prominent, particularly in terms of deviations from the desired target velocity $\boldsymbol{v}_d(t_w)$. The average velocity error is 0.13 m/s for $INIT_i$, while it reduces to 0.07 m/s for OPT_{ai}, representing about a 46% improvement. In terms of deviation from the target reach, OPT_{ai} shows a deviation of 0.019 m, compared to 0.023 m for $INIT_i$. Thus, our approach produced more robust and precise trajectories, leading to reduced deviations in both EE position and velocity.

To appreciate the effect of the perturbations over the control inputs, we report in Fig. 10 the input tubes. For the sake of space, we report only the input torque τ_4 but similar results were obtained for the other inputs. In Fig. 10, the black solid lines represent the upper/lower tube bounds obtained from simulations, while the green dashed lines correspond to the perturbed torques (for $N_{exp} = 15$ cases) and the red solid line represents the nominal torque from experiments. The perturbed torques generally remain within these bounds, demonstrating that the input tubes effectively capture the system's input behavior under parameter perturbations. The violations in Fig. 10 of the tubes are due to the first-order approximation used for computing the tubes [23], the difficulty in modeling all the parameters that might be uncertain (e.g., joint friction or backlash), and the impossibility of exactly knowing the nominal parameters $\boldsymbol{p}_c$ in real-world experiments which is probably the main factor affecting the tube behavior.

V. CONCLUSIONS

In this paper, we experimentally validated the *closed-loop state and input sensitivity* for a robotic arm that has to track a trajectory while grasping a payload whose dynamic parameters are uncertain. We performed simulations and real-world experiments where these parameters were altered, comparing the tracking of a minimum snap trajectory with the one obtained with our sensitivity framework using the same controller. The results confirm that our approach effectively mitigates the effects of the uncertain parameters, allowing for more precise tracking in terms of precision of the EE position and velocity. Future works will explore scenarios such as tossing an uncertain payload into a desired location, aiming to increase the tossing and landing accuracy.

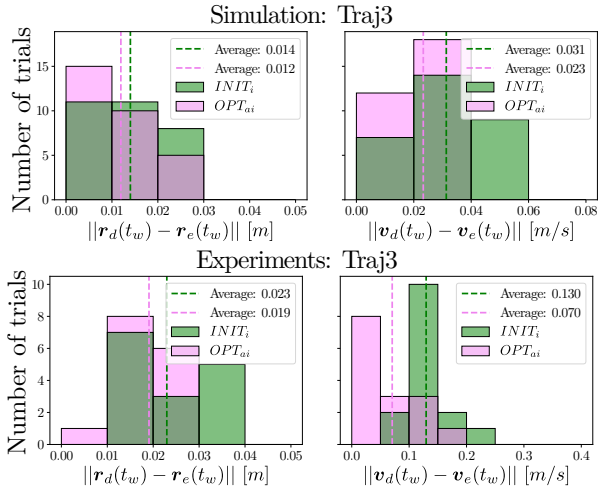

Fig. 9. Histograms of Traj3 in simulation (top) and experiments (bottom) for the $INIT_i$ (green) and the OPT_{ai} (violet) cases. EE position distance ($||\boldsymbol{r}_d(t_w) - \boldsymbol{r}_e(t_w)||$) in [m] and EE velocity distance ($||\boldsymbol{v}_d(t_w) - \boldsymbol{v}_e(t_w)||$) in [m/s] w.r.t. the desired trajectory at $\bar{t} = t_w$.

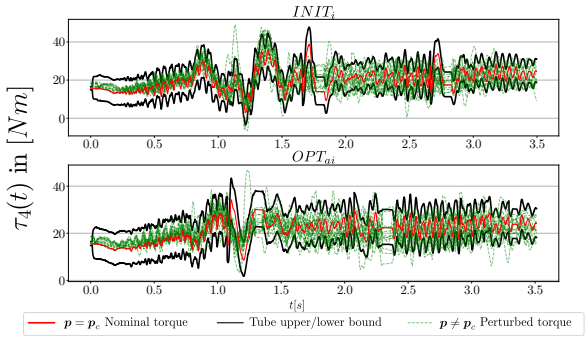

Fig. 10. Experimental actuator torque comparison for Traj3: $INIT_i$ (top) vs. OPT_{ai} (bottom). The solid red line denotes nominal actuator torque, while the solid black lines represent the input tube upper/lower bounds (15). The dashed green lines depict N_p perturbation runs.

REFERENCES

[1] C. Gaz and A. De Luca, "Payload estimation based on identified coefficients of robot dynamics—with an application to collision detection," in *2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 3033–3040.

[2] M. Hamad, N. Mansfeld, S. Abdolshah, and S. Haddadin, "The role of robot payload in the safety map framework," in *2019 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. IEEE, 2019, pp. 195–200.

[3] A. Bahloul, S. Tliba, and Y. Chitour, "Dynamic parameters identification of an industrial robot with and without payload," *Ifac-Papersonline*, vol. 51, no. 15, pp. 443–448, 2018.

[4] A. Kurdas, M. Hamad, J. Vorndamme, N. Mansfeld, S. Abdolshah, and S. Haddadin, "Online payload identification for tactile robots using the momentum observer," in *2022 Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5953–5959.

[5] K. N. Kumar, I. Essa, S. Ha, and C. K. Liu, "Estimating mass distribution of articulated objects using non-prehensile manipulation," *arXiv preprint arXiv:1907.03964*, 2019.

[6] I. Carlucho, D. W. Stephens, and C. Barbalata, "An adaptive data-driven controller for underwater manipulators with variable payload," *Applied Ocean Research*, vol. 113, p. 102726, 2021.

[7] Y. Chen, W. Zhan, B. He, L. Lin, Z. Miao, X. Yuan, and Y. Wang, "Robust control for unmanned aerial manipulator under disturbances," *Ieee Access*, vol. 8, pp. 129 869–129 877, 2020.

[8] Y. Dai, H. Gao, S. Yu, and Z. Ju, "A fast tube model predictive control scheme based on sliding mode control for underwater vehicle-manipulator system," *Ocean Engineering*, vol. 254, p. 111259, 2022.

[9] Z. Brei, J. Michaux, B. Zhang, P. Holmes, and R. Vasudevan, "Serving time: Real-time, safe motion planning and control for manipulation of unsecured objects," *IEEE Robotics and Automation Letters*, vol. 9, no. 3, pp. 2383–2390, 2024.

[10] J. Luo and K. Hauser, "Robust trajectory optimization under frictional contact with iterative learning," *Autonomous Robots*, vol. 41, pp. 1447–1461, 2017.

[11] P. Robuffo Giordano, Q. Delamare, and A. Franchi, "Trajectory generation for minimum closed-loop state sensitivity," in *2018 IEEE Int. Conf. on Robotics and Automation*. IEEE, 2018, pp. 286–293.

[12] P. Brault, Q. Delamare, and P. Robuffo Giordano, "Robust trajectory planning with parametric uncertainties," in *2021 IEEE Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 095–11 101.

[13] S. Wasiela, P. Robuffo Giordano, J. Cortes, and T. Simeon, "A Sensitivity-Aware Motion Planner (SAMP) to Generate Intrinsically-Robust Trajectories," in *2023 IEEE Int. Conf. on Robotics and Automation*, 2023, pp. 12 707–12 713.

[14] S. Wasiela, M. Cognetti, P. Robuffo Giordano, J. Cortés, and T. Siméon, "Robust motion planning with accuracy optimization based on learned sensitivity metrics," *IEEE Robotics and Automation Letters*, vol. 9, no. 11, pp. 10 113–10 120, 2024.

[15] A. Pupa, P. Robuffo Giordano, and C. Secchi, "Optimal energy tank initialization for minimum sensitivity to model uncertainties," in *2023 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2023.

[16] A. Srour, A. Franchi, and P. Robuffo Giordano, "Controller and trajectory optimization for a quadrotor uav with parametric uncertainty," in *2023 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2023)*, Detroit, Michigan, United States, October 2023.

[17] A. Srour, S. Marcellini, T. Belvedere, M. Cognetti, A. Franchi, and P. Robuffo Giordano, "Experimental validation of sensitivity-aware trajectory planning for a quadrotor uav under parametric uncertainty," in *2024 Int. Conf. on Unmanned Aircraft Systems*, 2024, pp. 572–578.

[18] M. Hafezipour and S. Khodaygan, "An uncertainty analysis method for error reduction in end-effector of spatial robots with joint clearances and link dimension deviations," *International Journal of Computer Integrated Manufacturing*, vol. 30, no. 6, pp. 653–663, 2017.

[19] M. Okada, A. Pekarovskiy, and M. Buss, "Robust trajectory design for object throwing based on sensitivity for model uncertainties," in *2015 IEEE Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 3089–3094.

[20] M. Okada, S. Oniwa, and W. Hijikata, "Robust throwing design based on dynamic sensitivity analysis," *Mechanical Engineering Journal*, vol. 5, no. 1, pp. 17–00 442, 2018.

[21] M. Okada and T. Sekiguchi, "Throwing motion design based on minimum sensitivity with respect to error covariance of robot dynamic parameters," *Mechanical Engineering Journal*, vol. 8, no. 1, pp. 20–00 299, 2021.

[22] C. Böhm, P. Brault, Q. Delamare, P. Robuffo Giordano, and S. Weiss, "Cop: Control & observability-aware planning," in *2022 International Conference on Robotics and Automation*, 2022, pp. 3364–3370.

[23] A. Afifi, T. Belvedere, A. Pupa, P. Robuffo Giordano, and A. Franchi, "Safe and robust planning for uncertain robots: A closed-loop state sensitivity approach," *IEEE Robotics and Automation Letters*, vol. 9, no. 11, pp. 9962–9969, 2024.

[24] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[25] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, "Tossingbot: Learning to throw arbitrary objects with residual physics," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1307–1319, 2020.

[26] M. J. Jongeneel, L. Poort, N. van de Wouw, and A. Saccon, "Experimental Validation of Nonsmooth Dynamics Simulations for Robotic Tossing involving Friction and Impacts," Feb. 2023, working paper or preprint. [Online]. Available: https://hal.science/hal-03974604

[27] M. Lubbers, J. van Voorst, M. Jongeneel, and A. Saccon, "Learning suction cup dynamics from motion capture: Accurate prediction of an object's vertical motion during release," in *2022 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. IEEE, 2022, pp. 1541–1547.

[28] A. R. Conn, K. Scheinberg, and P. L. Toint, "On the convergence of derivative-free methods for unconstrained optimization," *Approximation theory and optimization: tributes to MJD Powell*, pp. 83–108, 1997.

[29] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, pp. 1–36, 2019.