Dynamic Identification of the Franka Emika Panda Robot With Retrieval of Feasible Parameters Using Penalty-Based Optimization

Claudio Gaz, Marco Cognetti, Alexander Oliva, Paolo Robuffo Giordano, and Alessandro De Luca

***Abstract*—In this letter, we address the problem of extracting a feasible set of dynamic parameters characterizing the dynamics of a robot manipulator. We start by identifying through an ordinary least squares approach the dynamic coefficients that linearly parametrize the model. From these, we retrieve a set of feasible link parameters (mass, position of center of mass, inertia) that is fundamental for more realistic dynamic simulations or when implementing in real time robot control laws using recursive Newton-Euler algorithms. The resulting problem is solved by means of an optimization method that incorporates constraints on the physical consistency of the dynamic parameters, including the triangle inequality of the link inertia tensors as well as other user-defined, possibly nonlinear constraints. The approach is developed for the increasingly popular Panda robot by Franka Emika, identifying for the first time its dynamic coefficients, an accurate joint friction model, and a set of feasible dynamic parameters. Validation of the identified dynamic model and of the retrieved feasible parameters is presented for the inverse dynamics problem using, respectively, a Lagrangian approach and Newton-Euler computations.**

***Index Terms*—Franka Emika Panda, dynamic identification, friction model, feasible physical parameters, nonlinear global optimization, penalty methods.**

I. Introduction

THE knowledge of accurate dynamic models is of fundamental importance for many robotic applications. It is necessary, in fact, for designing control laws with superior performance, in free motion or when interacting with the environment [1], e.g., in strategies for the sensorless detection, isolation and reaction to unexpected collisions [2] or when regulating force or imposing a desired impedance control at the contact [3].

In order to obtain an estimation of the dynamic model, regression techniques are widely employed for industrial [4], [5] or humanoid robots [6], [7]. These techniques are hinged on a fundamental property: the linear dependence of the robot dynamic equations in terms of a set of ρ *dynamic coefficients* $\boldsymbol{\pi}_R \in \mathbb{R}^\rho$ [8], also known in the literature as *base parameters* [9], which are linear combinations of the *dynamic parameters* of each link composing the robot. In particular, each link has 10 parameters, specifying the mass, the position of the center of mass (CoM), and the 6 elements of the symmetric inertia tensor. Then, a robot with l links has a total $10\,l$ of such parameters, denoted as $\boldsymbol{p} \in \mathbb{R}^{10\,l}$. In addition, one may also include a number of parameters for modeling joint friction.

The regrouping of dynamic parameters, i.e., the dynamic coefficients, occurs because some parameters are not excitable during motion (they have no influence on the robot dynamics), while some others have an effect on the dynamics only in combinations (they are not separately identifiable).

The identification of the dynamic coefficients $\boldsymbol{\pi}_R$ is often sufficient for many robotic applications, such as dynamic motion robot control and motion planning, since knowledge of $\boldsymbol{\pi}_R$ allows for a numerical evaluation of the robot dynamic model in the Euler-Lagrange (E-L) form. However, the retrieval of a set of feasible numerical values for the dynamic parameters $\boldsymbol{p}$ is also relevant. This is the case, for instance, when performing dynamic simulations via a CAD-based robotic simulator – like V-REP [10] – or when implementing torque-level control laws (such as the feedback linearization) under hard real-time constraints. In this case, a widely adopted solution is to use the recursive numerical Newton-Euler (N-E) algorithm, which is preferred to the evaluation of the symbolic computationally more expensive E-L approach (which relies on dynamic coefficients). However, usual N-E routines require the knowledge of the dynamic parameters $\boldsymbol{p}$ of each link in the kinematic chain, and not just of the dynamic coefficients $\boldsymbol{\pi}_R$. In [8], we addressed the problem of recovering a complete set of values for the original robot parameters starting from the identified dynamic coefficients. In general, this is a nonlinear problem admitting an infinite number of solutions. However, not all solutions are physically consistent (as example, negative masses may appear). In order to discard unfeasible solutions, we considered upper and lower bounds on each component of $\boldsymbol{p}$ by solving a constrained nonlinear optimization problem. Because of the ill-conditioned nature of the solution space, we used global optimization methods, such as simulated annealing.

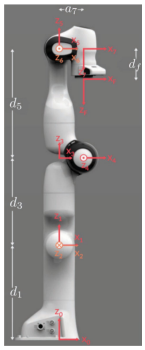
Manuscript received February 24, 2019; accepted June 30, 2019. Date of publication July 25, 2019; date of current version August 8, 2019. This letter was recommended for publication by Associate Editor Y. Amirat and Editor D. Song upon evaluation of the reviewers' comments. *(Corresponding author: Claudio Gaz.)*

C. Gaz and A. De Luca are with the Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, 00185 Roma, Italy (e-mail: gaz@diag.uniroma1.it; deluca@diag.uniroma1.it).

M. Cognetti, A. Oliva, and P. Robuffo Giordano are with the CNRS, University of Rennes, Inria, IRISA, 35000 Rennes, France (e-mail: marco.cognetti@irisa.fr; alexander.oliva@inria.fr; prg@irisa.fr).

This letter has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/LRA.2019.2931248



i	a_i	α_i	d_i	θ_i
1	0	0	d_1	q_1
2	0	$-\pi/2$	0	q_2
3	0	$\pi/2$	d_3	q_3
4	a_4	$\pi/2$	0	q_4
5	a_5	$-\pi/2$	d_5	q_5
6	0	$\pi/2$	0	q_6
7	a_7	$\pi/2$	0	q_7
8	0	0	d_f	0

Fig. 1. Denavit-Hartenberg frames and table of parameters for the Franka Emika Panda. The reference frames follow the modified Denavit-Hartenberg convention. In the figure, $d_1 = 0.333$ m, $d_3 = 0.316$ m, $d_5 = 0.384$ m, $d_f = 0.107$ m, $a_4 = 0.0825$ m, $a_5 = -0.0825$ m, $a_7 = 0.088$ m.

The physical consistency of the identified dynamic parameters, as introduced in [11], is currently attracting more and more attention. Researchers have treated the problem of physical feasibility within the framework of linear matrix inequalities (LMIs), solving the problem of the identification of a physically consistent set of parameters by means of semi-definite programming (SDP) techniques [12]. Recently, this framework has been enriched by the addition of the triangle inequality of the inertia tensors [13], [14], a constraint which was originally mentioned in [15]. The approach presented in this letter can be considered as an alternative to the LMI-SDP framework.

All these approaches act on the parameter identification phase, obtaining from this dynamic coefficients that are typically different from the classic ordinary least squares (OLS) solution [4], [5]. In any event, the approach presented in [12], [14] requires to express constraints as linear matrix inequalities, while the proposed optimization algorithm manages both linear and nonlinear (e.g., *if-else*) constraints, without any mathematical manipulation. This additional flexibility allows to handle directly nonlinear constraints coming from the geometric shape of cylindrical or spherical links (like for Universal Robots manipulators or for the sixth link of the KUKA LWR IV+), or when the use of approximate box constraints may generate solutions which are even unfeasible (e.g., a center of mass outside a convex link of the robot). On the other hand, a drawback of the proposed algorithm is that convergence to a global optimum in a finite number of steps cannot be guaranteed.

The proposed approach is general and can be applied to a large class of robot manipulators. In this work, as case study, we apply it to the Franka Emika Panda robot (see Fig. 1), a manipulator that is attracting a large interest in the robotics and industrial communities due to its high usability and relatively low price among the torque-controlled manipulators. For this robot we obtain a complete identification of a set of feasible dynamic parameters.

Summarizing, the main contributions of this letter are: (i) presentation of a framework for the robot dynamic parameters retrieval that deals with linear, nonlinear and conditional constraints; (ii) identification of the dynamic model of the Franka Emika Panda robot;[1] (iii) retrieval of a feasible set of dynamic parameters.

The letter is organized as follows. In Sec. II, we briefly present the main features of the Franka Emika Panda robot. In Sec. III we recall the general procedure for identifying the dynamic coefficients, and we present the problem of physical consistency of the parameters. The identification procedure used to retrieve the dynamic model for this robot is described in Sec. IV, together with an estimation procedure of the joint friction. Retrieval of the feasible parameters from the dynamic coefficients is presented in Sec. V. Finally, validation results are reported in Sec. VI and conclusions are drawn in Sec. VII.

II. The Franka Emika Panda robot

Fig. 1 shows the Franka Emika Panda robot and its kinematic parameters according to the modified Denavit-Hartenberg convention. This robot is equipped with $n = 7$ revolute joints, each mounting a torque sensor, and it has a total weight of approximately 18 kg, having the possibility to handle payloads up to 3 kg. It is possible to control the robot through the Franka Control Interface (FCI), that is able to provide, via the `libfranka` interface (at 1 kHz), the joint positions $\boldsymbol{q}$ and velocities $\dot{\boldsymbol{q}}$, as well as link side torque vector $\boldsymbol{\tau}$. Moreover, it returns the *numerical* values of the inertia matrix $\overline{\boldsymbol{M}}(\boldsymbol{q})$, of the gravity vector $\overline{\boldsymbol{g}}(\boldsymbol{q})$, as well as the Jacobian $\overline{\boldsymbol{J}}(\boldsymbol{q})$ and the Coriolis term $\overline{\boldsymbol{c}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ at a given joint position $\boldsymbol{q}$ and velocity $\dot{\boldsymbol{q}}$. These data will be of fundamental importance for the identification of the dynamic model of the robot (see Section IV-A for details).

The robot can be controlled in different modalities, according to the user requirements: torque-mode (by providing a vector $\boldsymbol{\tau}_d$ to the robot motors), position-mode (by giving a desired joint position $\boldsymbol{q}_d$), velocity-mode (by sending a desired joint velocity vector $\dot{\boldsymbol{q}}_d$) are the most common modalities. The FCI controller is designed in such a way that the command inputs given by the user are appropriately manipulated so that the motors generate the proper torque $\boldsymbol{\tau}_c$ for the commanded task. Fig. 2 depicts all the control signals above-mentioned.

III. Preliminaries

A. Building the Inverse Dynamic Model

In order to derive the symbolic dynamic model of a robot with elastic joints, such as the Franka Emika Panda, one may follow the procedure presented in [16], separating the motor torques from the link-side torques. Nevertheless, the particular features offered by the robot controller allow us to simplify the modeling: in fact, since the FCI controller is able to return the estimations of the link-side torques (exploiting the motor position measures read from the encoders), we are able to adopt the classical model structure as for a rigid joints robot, neglecting the elasticity [17], and henceforth $\boldsymbol{\tau} \in \mathbb{R}^n$ is the vector of the link-side torques.

[1]The dynamic identification procedure has been performed in this letter according to a reverse engineering approach. In the Supplementary material accompanying this manuscript, however, the reader can find also the results of the classical identification approach.

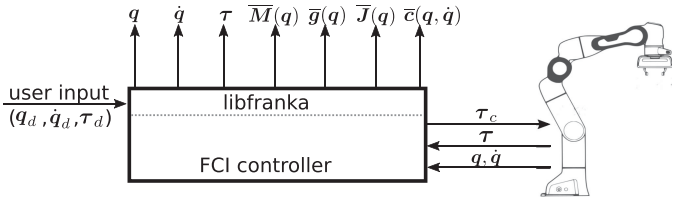


Fig. 2. Signal flows from and to the controller. The user sends a command to the `libfranka` interface that communicates with the FCI controller. This input is then converted to a commanded torque $\boldsymbol{\tau}_c$ to the robot that returns the measured joint torque $\boldsymbol{\tau}$, as well as the joint positions $\boldsymbol{q}$ and velocities $\dot{\boldsymbol{q}}$. The FCI controller computes the numerical values for the inertia matrix $\overline{\boldsymbol{M}}(\boldsymbol{q})$, as well for the gravity vector $\overline{\boldsymbol{g}}(\boldsymbol{q})$, the Jacobian $\overline{\boldsymbol{J}}(\boldsymbol{q})$, and the Coriolis term $\overline{\boldsymbol{c}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$. These data are sent back to the user through the `libfranka` interface. A more detailed description of the FCI can be found at: https://frankaemika.github.io/docs/index.html.

From the E-L equations [9], we can obtain the dynamic model of a n-dof robot as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{S}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{1}$$

where $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}} \in \mathbb{R}^n$ are, respectively, the joint positions, velocities and accelerations, $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n \times n}$ is the inertia matrix, $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^n$ is the gravity vector and $\boldsymbol{S}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^n$ is the vector of the Coriolis and centrifugal forces. The dynamic model in the form (1) includes typically nonlinear functions of $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$ and the dynamic parameters described in detail further. For each link ℓ_i, $i = 1, \ldots, n$, let m_i be the mass and let

$${}^i\boldsymbol{r}_{i,ci} = \begin{pmatrix} c_{ix} \\ c_{iy} \\ c_{iz} \end{pmatrix}, \quad {}^i\boldsymbol{J}_{\ell_i} = \begin{pmatrix} J_{ixx} & J_{ixy} & J_{ixz} \\ J_{ixy} & J_{iyy} & J_{iyz} \\ J_{ixz} & J_{iyz} & J_{izz} \end{pmatrix}, \tag{2}$$

be the position of the center of mass and the symmetric inertia tensor with respect to the i-th link frame, respectively.

At this stage, if we collect the dynamic parameters of all the robot links in the three vectors

$$\begin{aligned} \boldsymbol{p}_1 &= \begin{pmatrix} m_1 & \ldots & m_n \end{pmatrix}^T, \\ \boldsymbol{p}_2 &= \begin{pmatrix} c_{1x}m_1 & c_{1y}m_1 & c_{1z}m_1 & \ldots & c_{nx}m_n & c_{ny}m_n & c_{nz}m_n \end{pmatrix}^T, \\ \boldsymbol{p}_3 &= \begin{pmatrix} \mathcal{J}_1^T & \ldots & \mathcal{J}_n^T \end{pmatrix}^T, \end{aligned} \tag{3}$$

with $\boldsymbol{p}_1 \in \mathbb{R}^n$, $\boldsymbol{p}_2 \in \mathbb{R}^{3n}$, $\boldsymbol{p}_3 \in \mathbb{R}^{6n}$ and

$$\mathcal{J}_i = \begin{pmatrix} J_{ixx} & J_{ixy} & J_{ixz} & J_{iyy} & J_{iyz} & J_{izz} \end{pmatrix}^T, \tag{4}$$

it is possible to rearrange (1) as

$$\boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\ \boldsymbol{\pi}(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3) = \boldsymbol{\tau}, \tag{5}$$

where the vector $\boldsymbol{\pi}(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3) = \begin{pmatrix} \boldsymbol{p}_1^T & \boldsymbol{p}_2^T & \boldsymbol{p}_3^T \end{pmatrix}^T \in \mathbb{R}^p$. Moreover, $\boldsymbol{\pi}$ appears linearly in the dynamic model (5), multiplied by the regressor matrix $\boldsymbol{Y}$ of known time-varying functions.

The dynamic identification procedure is performed by collecting $M \gg n$ p joint torque samples as well as M joint position samples, while the joint velocity and the acceleration are computed by off-line differentiation. For each numerical sample $(\boldsymbol{\tau}_k, \boldsymbol{q}_k, \dot{\boldsymbol{q}}_k, \ddot{\boldsymbol{q}}_k)$, with $k = 1, \ldots, M$, we have

$$\boldsymbol{Y}_k(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k, \ddot{\boldsymbol{q}}_k)\boldsymbol{\pi} = \boldsymbol{\tau}_k. \tag{6}$$

By stacking these quantities in vectors and matrices, one has

$$\overline{\boldsymbol{Y}}\boldsymbol{\pi} = \overline{\boldsymbol{\tau}}, \tag{7}$$

with $\overline{\boldsymbol{\tau}} \in \mathbb{R}^{Mn}$ and $\overline{\boldsymbol{Y}} \in \mathbb{R}^{Mn \times p}$. According to [9], we can prune the stacked regressor $\overline{\boldsymbol{Y}}$ so as to obtain a matrix with full column rank $\overline{\boldsymbol{Y}}_R$, and then identify the dynamic coefficients by solving an ordinary least-squares (OLS) problem via pseudoinversion

$$\hat{\boldsymbol{\pi}}_R = \overline{\boldsymbol{Y}}_R^{\#}\overline{\boldsymbol{\tau}}. \tag{8}$$

With the solution $\hat{\boldsymbol{\pi}}_R \in \mathbb{R}^\rho$ of regrouped dynamic parameters, i.e. the dynamic coefficients, we can provide a joint torque estimate as

$$\hat{\boldsymbol{\tau}} = \boldsymbol{Y}_R(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\hat{\boldsymbol{\pi}}_R \tag{9}$$

for validation on any new motion $\boldsymbol{q}(t)$. Finally, following [8], one can extract from the identified vector $\hat{\boldsymbol{\pi}}_R$ a feasible set of dynamic parameters $\hat{\boldsymbol{p}} = (\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2, \hat{\boldsymbol{p}}_3)$ — not necessarily the true ones — such that $\boldsymbol{\pi}_R(\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2, \hat{\boldsymbol{p}}_3) = \hat{\boldsymbol{\pi}}_R$ and the upper/lower bounds on the components of $\boldsymbol{p}_i$, $i = 1, 2, 3$, are also satisfied. However, the triangular inequality constraint of inertia tensors is not taken into account in [8], as instead done in Sec. V.

B. Physical Consistency of the Dynamic Parameters

The obtained estimation of the dynamic coefficients vector $\hat{\boldsymbol{\pi}}_R$ might be possibly physically inconsistent (e.g., a negative link mass), and this can be caused, for instance, by modeling errors or by noisy measurements. Recent works [13]–[15] highlighted these physical constraints and provided frameworks to consider them during the identification phase, by solving linear constrained optimization problems using the following cost function:

$$\min_{\boldsymbol{\pi}} f(\boldsymbol{\pi}) = \|\overline{\boldsymbol{Y}}\boldsymbol{\pi} - \overline{\boldsymbol{\tau}}\|. \tag{10}$$

Physical constraints regard the mass of each link, which has to be positive, and the barycentric inertia tensor of each link, which has to be positive definite. That is, for each link ℓ_i of the manipulator, one has:

$$m_i > 0 \tag{11}$$

and

$$\boldsymbol{I}_{\ell_i} = \begin{pmatrix} I_{ixx} & I_{ixy} & I_{ixz} \\ I_{ixy} & I_{iyy} & I_{iyz} \\ I_{ixz} & I_{iyz} & I_{izz} \end{pmatrix} \succ 0, \tag{12}$$

where $\boldsymbol{I}_{\ell_i}$ is the inertia tensor of link ℓ_i with respect to its center of mass. Moreover, it is always possible to express the barycentric inertia tensor in a diagonal form, exploiting a particular rotation matrix $\bar{\boldsymbol{R}}_i$, such as

$$\boldsymbol{I}_{\ell_i} = \bar{\boldsymbol{R}}_i \bar{\boldsymbol{I}}_{\ell_i} \bar{\boldsymbol{R}}_i^T, \tag{13}$$

where $\bar{\boldsymbol{I}}_{\ell_i}$ is the diagonal inertia tensor. Since the diagonal elements ($\bar{I}_{i,x}$, $\bar{I}_{i,y}$, $\bar{I}_{i,z}$) of $\bar{\boldsymbol{I}}_{\ell_i}$ are also the eigenvalues of $\boldsymbol{I}_{\ell_i}$, condition (12) can be rewritten as

$$\bar{I}_{i,x} > 0 \quad , \quad \bar{I}_{i,y} > 0 \quad , \quad \bar{I}_{i,z} > 0\ . \tag{14}$$

These three inequalities are however included in the following triangle inequality:

$$\begin{cases} \bar{I}_{i,x} + \bar{I}_{i,y} > \bar{I}_{i,z} \\ \bar{I}_{i,y} + \bar{I}_{i,z} > \bar{I}_{i,x} \\ \bar{I}_{i,z} + \bar{I}_{i,x} > \bar{I}_{i,y} \end{cases} \tag{15}$$

which, with simple manipulations [14], leads to the following condition on $\boldsymbol{I}_{\ell_i}$:

$$\frac{\mathrm{tr}(\boldsymbol{I}_{\ell_i})}{2} - \lambda_{\max}(\boldsymbol{I}_{\ell_i}) > 0, \tag{16}$$

since $\bar{I}_{i,x} + \bar{I}_{i,y} + \bar{I}_{i,z} = \mathrm{tr}(\boldsymbol{I}_{\ell_i})$ and denoting with $\mathrm{tr}(\boldsymbol{I}_{\ell_i})$ and $\lambda_{\max}(\boldsymbol{I}_{\ell_i})$, respectively, the trace and the maximum eigenvalue of the inertia tensor $\boldsymbol{I}_{\ell_i}$.

Therefore, in order to have physical consistency, conditions (11) and (16) must be satisfied for each link.

IV. Identification Procedure

A. Identifying the Model Used by the Controller

Exploiting the features offered by the controller of the Panda robot (see Sec. II), it is possible to avoid the classical procedure involving exciting trajectories [18], and obtain the estimates of the gravitational and inertial coefficients by collecting a set of static positions only by means of a reverse engineering procedure.[2] The same procedure had been used to retrieve the dynamic coefficients of the KUKA LWR robot [17], [19]: in that case, the gravitational and the inertial coefficients have been estimated separately, while now a slightly different approach has been used, due to the fact that many coefficients can be retrieved both from the inertia matrix and from the gravity vector. As a first operation, one has to rearrange the symbolic inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ in a vector form (the *inertia stack*), by exploiting its symmetry. Having for the Panda robot $n = 7$ joints, we obtain a vector $\tilde{\boldsymbol{m}}(\boldsymbol{q}) \in \mathbb{R}^m$, with $m = n(n+1)/2 = 28$ components, containing all the lower triangular elements of $\boldsymbol{M}(\boldsymbol{q})$. Now, it is possible to obtain – as described in Sec. III – the symbolic regressor $\boldsymbol{Y}_s(\boldsymbol{q})$ from the column vector $\boldsymbol{s}(\boldsymbol{q}) \in \mathbb{R}^{m+n}$ in such a way that

$$\boldsymbol{s}(\boldsymbol{q}) = \begin{pmatrix} \tilde{\boldsymbol{m}}(q) \\ \boldsymbol{g}(q) \end{pmatrix} = \boldsymbol{Y}_s(\boldsymbol{q})\boldsymbol{\pi}_s, \tag{17}$$

where $\boldsymbol{\pi}_s \in \mathbb{R}^{10n}$ is the vector containing both the gravitational and inertial dynamic parameters, which are the same – excluding joint friction and motor inertias – as in vector $\boldsymbol{\pi}$ of eq. (7).

In order to obtain a numerical estimation of the dynamic coefficients vector, a data acquisition procedure should be carried out: in the general case, performing exciting trajectories is required in order to span all the admissible joint positions, velocities and accelerations (see eq. (8)). Since $\boldsymbol{s}(\boldsymbol{q})$ depends only on the joint positions $\boldsymbol{q}$, it is just sufficient to retrieve data by imposing static positions.

The `libfranka` software provides the numerical evaluation of the gravity vector and the inertia matrix of the Panda robot at the current link position. Therefore, it is possible to collect a fair amount of data even only in a static way, i.e., bringing the manipulator to a desired configuration and then retrieving and storing the numerical values of the gravity vector and the inertia matrix. This acquisition procedure can be performed during a motion as well. The main advantage of imposing static joint positions is that this procedure avoids any influence of friction and uncertainty (e.g., due to measure noise or to any unmodeled phenomenon).

The data is acquired and collected in a list of M different (special and/or random) configurations, under the weak condition

$$Mn \gg p. \tag{18}$$

For a generic configuration $\boldsymbol{q}_k$, with $k = 1 \dots M$, we have

$$\overline{\boldsymbol{s}}_k = \begin{pmatrix} \overline{\boldsymbol{m}}_k \\ \overline{\boldsymbol{g}}_k \end{pmatrix} = \overline{\boldsymbol{Y}}_{sk}\boldsymbol{\pi}_s, \tag{19}$$

where $\overline{\boldsymbol{m}}_k$ and $\overline{\boldsymbol{g}}_k$ represent, respectively, the numerical inertia stack vector and the numerical gravity vector, as they are retrieved from the `libfranka` interface at a given configuration $\boldsymbol{q}_k$, and $\overline{\boldsymbol{Y}}_{sk} = \boldsymbol{Y}_s(\boldsymbol{q}_k)$ is the evaluated regressor.

When all data are collected, they can be stacked into a vector $\overline{\boldsymbol{s}}$ such as:

$$\overline{\boldsymbol{s}} = \begin{pmatrix} \overline{\boldsymbol{s}}_1 \\ \overline{\boldsymbol{s}}_2 \\ \vdots \\ \overline{\boldsymbol{s}}_N \end{pmatrix} = \begin{pmatrix} \overline{\boldsymbol{Y}}_{s1} \\ \overline{\boldsymbol{Y}}_{s2} \\ \vdots \\ \overline{\boldsymbol{Y}}_{sN} \end{pmatrix} \boldsymbol{\pi}_s = \overline{\overline{\boldsymbol{Y}}}_s \boldsymbol{\pi}_s. \tag{20}$$

Nevertheless, the regressor $\overline{\overline{\boldsymbol{Y}}}_s$ is typically rank-deficient: this implies that the elements of the vector $\boldsymbol{\pi}_s$ are not fully identifiable. Therefore, it is necessary to drop linear dependent columns of the regressor in order to reach a full (column) rank condition (i.e., by means of the Gauss-Jordan elimination technique). As a consequence, some dynamic parameters will be grouped together accordingly, in the form of dynamic coefficients [9]. Exploiting condition (18) on the minimal number M of samples to retrieve, the ill-conditioning of the matrix is avoided. Denoting as $\hat{\boldsymbol{\pi}}_{s,R}$ the vector containing the regrouped parameters (a.k.a. dynamic coefficients), and as $\overline{\overline{\boldsymbol{Y}}}_{s,R}$ the full rank numerical regressor, eq. (20) is solved using a least squares technique as

$$\hat{\boldsymbol{\pi}}_{s,R} = \left(\overline{\overline{\boldsymbol{Y}}}_{s,R}^T \overline{\overline{\boldsymbol{Y}}}_{s,R} \right)^{-1} \overline{\overline{\boldsymbol{Y}}}_{s,R}^T \overline{\boldsymbol{s}} = \overline{\overline{\boldsymbol{Y}}}_{s,R}^{\#} \overline{\boldsymbol{s}}, \tag{21}$$

where $^{\#}$ denotes pseudoinversion.

Once one has the dynamic coefficients estimation $\hat{\boldsymbol{\pi}}_s$, from eq. (17), it is possible to obtain the estimates $\hat{\boldsymbol{g}}(\boldsymbol{q})$ and $\hat{\boldsymbol{M}}(\boldsymbol{q})$ as:

$$\hat{\boldsymbol{s}}(\boldsymbol{q}) = \begin{pmatrix} \hat{\tilde{\boldsymbol{m}}}(q) \\ \hat{\boldsymbol{g}}(q) \end{pmatrix} = \boldsymbol{Y}_{s,R}(\boldsymbol{q})\hat{\boldsymbol{\pi}}_{s,R}, \tag{22}$$

where $\boldsymbol{Y}_{s,R}(\boldsymbol{q})$ is the symbolic regressor pruned of the dependent columns (according to the full-rank matrix $\overline{\overline{\boldsymbol{Y}}}_{s,R}$) and $\hat{\boldsymbol{M}}(\boldsymbol{q})$ is built from the estimated inertia stack $\hat{\tilde{\boldsymbol{m}}}(q)$. Finally, the

[2] See the Supplementary material accompanying this letter for a comparison of the dynamic coefficients estimated through the reverse engineering process with those obtained from the classical approach exploiting exciting trajectories.

estimation of the Coriolis and centrifugal forces vector $\hat{\boldsymbol{c}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is derived from $\hat{\boldsymbol{M}}(\boldsymbol{q})$ using the Christoffel's symbols according to [1]. The form of the inverse dynamics formula, providing an estimation of the joint torques $\hat{\boldsymbol{\tau}}$ needed to accomplish a given trajectory $(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$, is therefore:

$$\hat{\boldsymbol{\tau}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \hat{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \hat{\boldsymbol{c}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \hat{\boldsymbol{g}}(\boldsymbol{q}). \tag{23}$$

B. Friction Estimation

If a validation trajectory is executed and the measured torques $\boldsymbol{\tau}$ are compared (after a filtering procedure) with the estimated torques $\hat{\boldsymbol{\tau}}$ generated by eq. (23), a difference in the two signals may be observed. This discrepancy is due to estimation errors (e.g., due to the noise affecting the measurements) or to unmodeled effects, such as joint friction. Typically, the latter effect, assumed to act separately on each joint, is expressed as an additional torque $\tau_{f,j},\ j \in \{1, \ldots, n\}$:

$$\tau_{f,j}(\dot{q}_j) = f_{v,j}\dot{q}_j + f_{c,j}\,\mathrm{sign}(\dot{q}_j) + f_{o,j}, \tag{24}$$

where $f_{v,j}$ and $f_{c,j}$ represent, respectively, viscous and Coulomb friction, while $f_{o,j}$ is the Coulomb friction offset. Model (24), although being simple and effective, has the main drawback of exhibiting sudden discontinuities in the neighborhood of $\dot{q}_j = 0$. In order to attenuate this chattering, a sigmoidal friction model can be used for avoiding discontinuities for low joint velocities. In case the viscous effects are negligible and a symmetric behavior for positive and negative joint velocities is observed (as for the Panda robot), $\tau_{f,j}(\dot{q}_j)$ can be expressed as the following function:

$$\tau_{f,j}(\dot{q}_j) = \frac{\varphi_{1,j}}{1 + e^{-\varphi_{2,j}(\dot{q}_j + \varphi_{3,j})}} - \frac{\varphi_{1,j}}{1 + e^{-\varphi_{2,j}\varphi_{3,j}}}, \tag{25}$$

which is characterized by 3 parameters for each joint ($\varphi_{1,j}, \varphi_{2,j}$, $\varphi_{3,j}$). In order to estimate the $3n = 21$ friction parameters for the Panda robot, trajectories spanning all possible joint velocities can be executed for each joint (possibly, keeping the others at rest). Computing the inverse dynamics for the previous trajectories according to (23), the difference $\Delta\tau_j = \tau_j - \hat{\tau}_j$ (where τ_j and $\hat{\tau}_j$ are, respectively, the measured and the estimated joint torques) can be interpreted as a measure of the friction effects. Solving a least squares problem, it is possible to find the parameters that make the curve (25) to fit data at best – e.g., by means of a Nelder-Mead routine.

The presented reverse engineering approach has the main advantage of allowing an estimation of the dynamic parameters by exploiting only static measures (which are affected by low noise) without the need of numerical derivations, which can dramatically affect the results of the identification process. Moreover, having a separate step for joint friction identification allows to repeat only this step (instead of the whole identification process) when lubrication changes or when moving to a new instance of the same robot. Conversely, we must rely on the numerical values of $\overline{\boldsymbol{s}}_k$ provided by the manufacturer.

V. Retrieval of Feasible Parameters

In [8] a framework has been presented for retrieving a feasible set of dynamic parameters from the previously identified dynamic coefficients, by solving a nonlinear global optimization problem.

For some purposes, in fact, the estimated dynamic coefficients (linear combination of dynamic parameters, such as masses, position of the centers of mass and inertia tensors) are not sufficient, and an estimation of the dynamic parameters themselves is needed. This occurs, for instance, if the dynamic parameters values are needed for simulating the robot behavior in a CAD software (like V-REP), or if a Newton-Euler routine is required to compute the joint torques estimation during the robot motion, with strict time constraints (in this case, a Lagrangian approach may not fit the real-time requirements), or in case one is interested in computing the wrenches acting on the robot joints.

The approach presented in [8] is able to return a possible and feasible set of the dynamic parameters: in general, infinite solutions exist (among which the real one) and the more constraints are given, the closer to the real solution is the returned one.

Starting from the identified vector of dynamic coefficients $\hat{\boldsymbol{\pi}}_R$, the following transformations may be applied to the corresponding symbolic vector $\boldsymbol{\pi}_R(\boldsymbol{p})$ (parallel axis theorem, see [1]):

$$\begin{aligned}
&J_{ixx} \to I_{ixx} + m_i c_{iy}^2 + m_i c_{iz}^2 \quad J_{ixy} \to I_{ixy} - c_{ix}c_{iy}m_i \\
&J_{iyy} \to I_{iyy} + m_i c_{ix}^2 + m_i c_{iz}^2 \quad J_{ixz} \to I_{ixz} - c_{ix}c_{iz}m_i \\
&J_{izz} \to I_{izz} + m_i c_{ix}^2 + m_i c_{iy}^2 \quad J_{iyz} \to I_{iyz} - c_{iy}c_{iz}m_i
\end{aligned} \tag{26}$$

for each link $\ell_i,\ i = 1 \ldots n$. We can now rearrange the parameters vector $\boldsymbol{p} = \begin{pmatrix} \boldsymbol{p}_1^T & \boldsymbol{p}_2^T & \boldsymbol{p}_3^T \end{pmatrix}^T \in \mathbb{R}^{10n}$ as:

$$\begin{aligned}
\boldsymbol{p}_1 &= \begin{pmatrix} m_1 & \ldots & m_n \end{pmatrix}^T \in \mathbb{R}^n, \\
\boldsymbol{p}_2 &= \begin{pmatrix} c_{1x} & c_{1y} & c_{1z} & \ldots & c_{nx} & c_{ny} & c_{nz} \end{pmatrix}^T \in \mathbb{R}^{3n}, \\
\boldsymbol{p}_3 &= \begin{pmatrix} \mathcal{I}_1^T & \ldots & \mathcal{I}_n^T \end{pmatrix}^T \in \mathbb{R}^{6n},
\end{aligned} \tag{27}$$

where

$$\mathcal{I}_i = \begin{pmatrix} I_{ixx} & I_{ixy} & I_{ixz} & I_{iyy} & I_{iyz} & I_{izz} \end{pmatrix}^T. \tag{28}$$

It is possible to provide lower bounds (LB) and upper bounds (UB) to $\boldsymbol{p}$ based on *a priori* information. In particular, condition (11) is managed by assigning a lower bound of zero to each link mass. Upper bounds for the masses are assigned exploiting, for instance, data retrieved from the datasheets of the robot. Moreover, for each link, one can easily infer that the center of mass is located inside the smallest parallel box which includes the link geometry, in the most general case. The lower and upper bounds are then set in order to guarantee a physical meaning to the obtained solution. In order to retrieve a possible set of dynamic parameter $\hat{\boldsymbol{p}}$, we propose to solve the nonlinear optimization problem depicted in Algorithm 1.

The first two lines of Algorithm 1 are the initialization step of the algorithm: the starting point is randomly selected between the lower and the upper bounds using a uniform distribution. Moreover, ξ_1 is set to zero. Lines 3–6 of Algorithm 1 consist in solving the constrained nonlinear optimization problem κ times: at a given step $k = 1 \ldots \kappa$, the initial state is the optimal solution found at step $k - 1$. The objective function presents also an exterior penalty function, in which ξ_k is the – progres-

Algorithm 1: Parameters retrieval

$\boldsymbol{p}_0 \leftarrow LB + (UB - LB)\boldsymbol{u}, \quad \text{with} \quad \boldsymbol{u} \sim \mathcal{U}(0,1)$;
$\xi_1 \leftarrow 0$;
for $k = 1, \ldots, \kappa$ **do**
  // Start the optimization from the previous step solution
  $p_{k,\text{init}} \leftarrow p_{k-1}$;
  // Solve the following optimization problem

$$\begin{cases} \min\limits_{\boldsymbol{p}_k} f(\boldsymbol{p}_k) &= \phi(\boldsymbol{p}_k) + \xi_k \gamma(\boldsymbol{p}_k) \\ &= \|\boldsymbol{\pi}(\boldsymbol{p}_k) - \hat{\boldsymbol{\pi}}\|^2 + \xi_k \sum_\iota g(h_\iota(\boldsymbol{p}_k)) \\ \text{s.t.} \quad LB \leq \boldsymbol{p}_k \leq UB \end{cases}$$

  $\xi_{k+1} \leftarrow 10k$ (29)
end

sively increasing – penalty coefficient: this function provides a penalty in case one of any additive constraint $h_\iota(\boldsymbol{p}_k)$ is violated; function $g(\cdot)$ is chosen to return a measure of the constraint violation. In particular, two kinds of constraints have been included:

- for each link ℓ_i, the triangle inequality (16) must be satisfied;
- the total sum of the link masses must be in a given range, that is:

$$m_{\text{rob,min}} \leq \sum_i m_i \leq m_{\text{rob,max}}. \tag{30}$$

Note that the presented framework is extremely flexible, and further external constraints can be easily included. The manifold generated by the cost function $f(\boldsymbol{p}_k)$ contains multiple local minima, and therefore a global optimization method, like genetic algorithms [20] or simulated annealing (SA) [21] is mandatory to address the problem (29). In the present case, we have used SA, applying a more sophisticated interior-point (IP) Nelder-Mead local optimization algorithm at the end of each SA iteration k. Moreover, Q runs have been launched having a different random initial point $\boldsymbol{p}_0$, in order to span as much as possible the cost function manifold.

The improvement of the parameters retrieval framework (with respect to the one presented in [8]) has been proven necessary since the introduction of some constraints – as for instance the triangle inequality of the inertia tensors – eventually led the algorithm to get stuck in local minima. This was caused by recurring abrupt changes in the cost function given by the constant penalty function adopted before. Therefore, a violation-dependent penalty [22] has been implemented for the present algorithm, solving this problem. Moreover, the sequence of successive runs of the algorithm could in practice help in improving the solution. The term $\phi(\boldsymbol{p}_k)$ of the parameters retrieval algorithm in eq. (29) requires, to be computed, a previous identification step returning the coefficients estimation $\hat{\boldsymbol{\pi}}$. Another possible $\phi(\boldsymbol{p}_k)$ function, yielding to a single-step procedure, is described in the Supplementary material accompanying this letter.

VI. Results

In this section, results from the dynamic coefficients and joint friction estimations for the Panda robot are reported, in addition to results from the parameters retrieval procedure.

In order to obtain a numerical estimation of the dynamic coefficients $\boldsymbol{\pi}_{s,R}$ (see eq. (21)), the numerical values of the inertia matrix and the gravity vector are retrieved from a set of $M = 1010$ static positions, spanning the whole joint space, according to the robot documentation.[3] The symbolic vector $\boldsymbol{s}(\boldsymbol{q})$ (see eq. (17)) has been computed according to [9], using the modified Denavit-Hartenberg convention.

Since our robot is mounted on a table parallel to the ground, the first element of the gravity vector (relative to joint 1) is not informative, and therefore it is discarded.

Stacking all the numerical quantities of the data acquisition phase and after evaluating the corresponding regressor (see eq. (20)), we obtained that $\text{rank}(\overline{\overline{\boldsymbol{Y}}}_s) = 43$: using the Matlab function `rref`, which implements Gauss-Jordan elimination technique, we finally obtained the dynamic coefficients estimations $\hat{\boldsymbol{\pi}}_{s,R}$ according to eq. (21), from a regressor $\overline{\overline{\boldsymbol{Y}}}_{s,R}$ whose condition number is 49. Moreover, the relative error percentage of predictions (defined in eq. (84) of [12]) for the identification set is 0.031%. Computing their standard deviations (see [5], [12]), we found that two coefficients exhibit a standard deviation greater than 30%, and therefore they were discarded (since their estimations are not reliable). All the estimated dynamic coefficients are reported with their standard deviations in the Supplementary material, together with a comparison with the corresponding coefficients obtained from the classical identification procedure (see eq. (8)).

From the identified dynamic coefficients $\hat{\boldsymbol{\pi}}_{s,R}$, we were able to reconstruct the inertia matrix $\hat{\boldsymbol{M}}(\boldsymbol{q})$, the gravity vector $\hat{\boldsymbol{g}}(\boldsymbol{q})$ and the Coriolis and centrifugal force vector $\hat{\boldsymbol{c}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ following a Lagrangian approach (see Sec. IV).

After deriving the inverse dynamic model (23), we validate it by comparing the measured joint torques with the estimated ones during several motions. In particular, the robot was commanded in velocity-mode by means of sinusoidal trajectories in the joint space. In other words, each joint is commanded according to the following equation:

$$\dot{q}_{i,\text{des}}(t) = A_i \ \sin\left(\frac{2\,\pi}{T_i} t\right), \quad i \in [1, \ldots, n] \tag{31}$$

where A_i is the amplitude of the velocity profile and T_i is the period of the sinusoidal signal for the i-th joint. The numerical values for A_i and T_i, $i \in [1, \ldots, n]$ for a typical experiment are reported in Table I, where 5458 samples were collected. The joint torque signals were recorded (and filtered through a 4-th order zero-phase digital Butterworth filter with a cutoff frequency of 1 Hz) during this motion, and compared with our Lagrangian inverse dynamics estimations $\hat{\boldsymbol{\tau}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ (from eq. (23)), feeding that model with the measured joint positions and velocities, and

[3]The joint position bounds of the Franka Emika Panda robot can be found at: https://frankaemika.github.io/docs/control_parameters.html

TABLE I
AMPLITUDES A_i AND PERIODS T_i, $i \in [1, \dots, n]$ OF THE SINUSOIDAL TRAJECTORIES IN EQ. (31) USED FOR VALIDATING THE FRICTION ACTING ON THE ROBOT JOINTS

	1	2	3	4	5	6	7
A_i	2.21	-2.21	1.2	-2.1	-2.3	2.1	-2.5
T_i	3.68	2.04	2.98	1.75	4.43	2.749	1.06

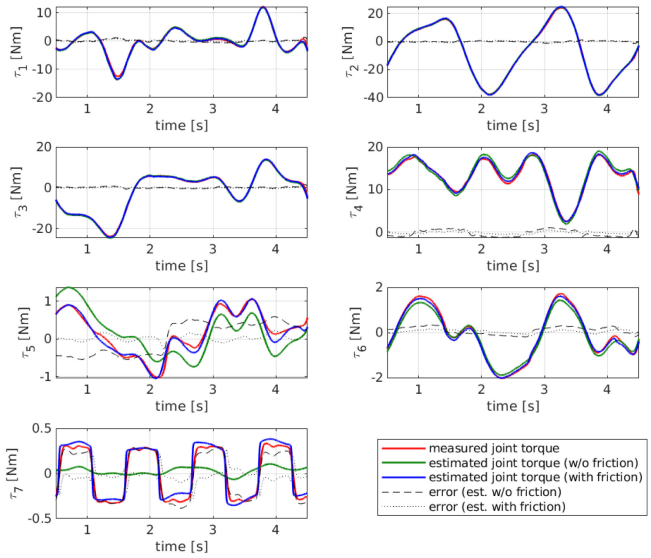


Fig. 3. Comparison between the torques of the derived E-L dynamic model. The red lines represent the measured joint torques during the validation experiment reported in Table I, while the green lines are the torque estimations $\hat{\boldsymbol{\tau}}$ computed according to eq. (23)). The blue lines represent the joint torque estimates comprehensive of the joint friction term (25). The dashed and the dotted lines are the errors between the torque sensors readings and, respectively, the torques estimates without and with the friction component.

with the joint accelerations obtained by numerical differentiation of the filtered velocities. The joint torque comparison is reported in Fig. 3: the torque estimations are almost perfectly superimposing the measured ones for joints $1 \dots 4$, while the last joints show some discrepancies. One can notice, though, that the difference between the two signals strongly depends on joint velocities (it is more evident, e.g., for joint 7): this behavior is typical of joint friction.

Therefore, we performed the estimation of the joint friction according to the procedure reported in Sec. IV-B: we collected more than 10k samples of joint velocities and torques during sinusoidal motions in (31); eventually, we found that the best friction model that fits the data was given by a sigmoidal function (25), which is characterized by 3 parameters per joint. These were estimated by solving a nonlinear least squares problem by means of a Nelder–Mead routine (Matlab function `fminsearch`), using as fitting data the differences $\Delta\tau_i$ between the measured joint torques and the estimated torques $\hat{\tau}_i$ for each joint i separately. The results of the fitting procedures are reported in Fig. 4, while the numerical values are reported in the Supplementary material. Finally, adding the newly estimated joint friction components $\hat{\boldsymbol{\tau}}_f(\dot{\boldsymbol{q}}) = \left(\hat{\tau}_{f,1}(\dot{q}_1)\ \hat{\tau}_{f,2}(\dot{q}_2)\ \dots\ \hat{\tau}_{f,7}(\dot{q}_7)\right)^T$ to the

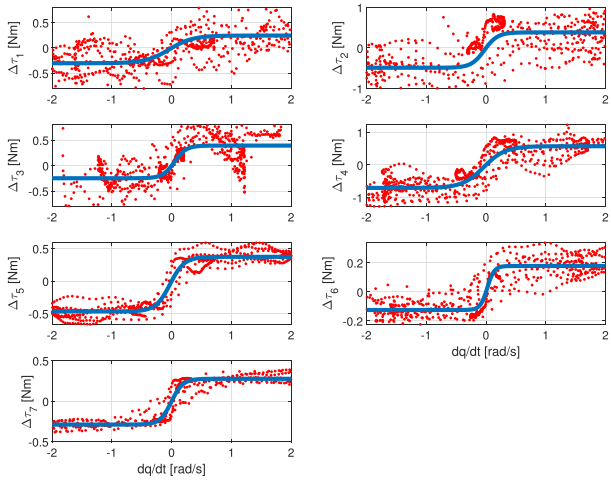


Fig. 4. Joint friction estimates. The red dots are the differences $\Delta\tau_j = \tau_j - \hat{\tau}_j$ for each joint j, while the blue lines are the sigmoidal fitting functions. The effect of friction is clear for joints $4 \dots 7$, while its contribution to the total joint torque is slight for the other joints.

previous estimations $\hat{\boldsymbol{\tau}}$, we obtained a satisfactory compensation, as shown with the blue solid lines of Fig. 3.

The estimated inverse dynamic model in Lagrangian form described so far, though, is very cumbersome for a 7 DoF robot, and computationally intensive, such that it would not be reliable under real-time constraints. For this reason, a Newton-Euler (N-E) approach would be more appropriate and effective to quickly return joint torque estimates, due to its recursive form. Nevertheless, a N-E routine requires the dynamic parameters (masses, inertia tensors and centers of mass of each link) of the robot. Exploiting the parameters retrieval algorithm (29), though, we are able to extract a *feasible* set of dynamic parameters, which provides the same dynamics (although there is no guarantee that the estimated robot parameters set is coincident with the real one).

In order to implement Algorithm 1, the (bounded) simulated annealing Matlab function `simulannealbnd` has been used, together with the IP hybrid function `fmincon`; the problem parameters were $Q = 100$ (total number of independent runs, providing Q solutions – possibly coincident, since there might exist multiple minima) and $\kappa = 10$ (number of successive runs). The penalty functions $g(h_\iota(\boldsymbol{p}_k))$ are chosen as the distance functions of the external constraints, that is:

$$g_{1,i} = -\min\left\{0, \mathrm{tr}(\boldsymbol{I}_{\ell_i})/2 - \lambda_{\max}(\boldsymbol{I}_{\ell_i})\right\}, \quad i \in [1, \dots 7]$$
$$g_2 = -\min\left\{0, m_{\mathrm{rob,max}} - \sum_{i=1}^{7} m_i, \sum_{i=1}^{7} m_i - m_{\mathrm{rob,min}}\right\} \tag{32}$$

where $g_{1,i}$ regards the triangle inequality on each inertia tensor (eq. 16), while g_2 regards the constraint on the total mass of the robot (we chose $m_{\mathrm{rob,min}} = 16$ kg and $m_{\mathrm{rob,max}} = 20$ kg).

A feasible set of dynamic parameters $\hat{\boldsymbol{p}}$ has been then retrieved and its numerical values are included in the supplementary

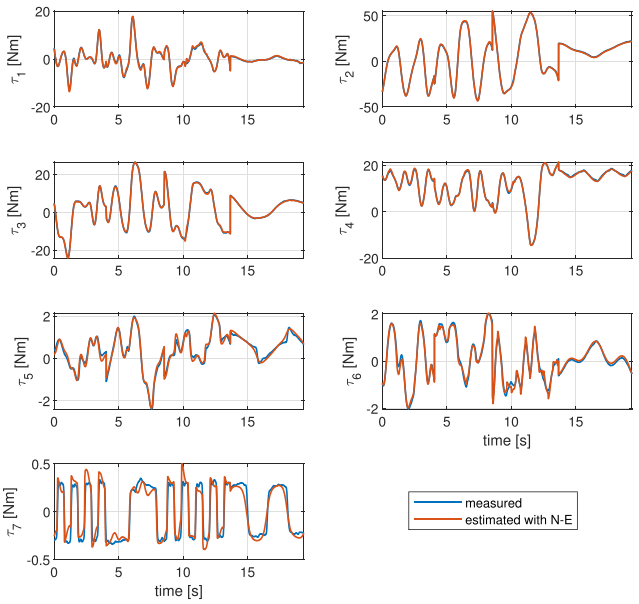


Fig. 5. Comparison between the torques $\hat{\boldsymbol{\tau}}_{ne}$ (including friction $\hat{\boldsymbol{\tau}}_f$) from the N-E dynamic model and the ones retrieved from the the joint torque sensors. The overlapping of two signals shows the quality of the obtained parameters estimation $\hat{\boldsymbol{p}}$.

material. In order to validate this set, we inserted the retrieved dynamic parameters in a N-E routine in order to compute the joint torques $\hat{\boldsymbol{\tau}}_{ne}$ necessary to perform a given validation trajectory in the joint space. In particular, we commanded a sequence of sinusoidal trajectories (with different amplitudes and periods) to the joints according to eq. (31). We then compared the measured joint torques $\boldsymbol{\tau}$ with the estimations $\hat{\boldsymbol{\tau}}_{ne} + \hat{\boldsymbol{\tau}}_f$: the result of this comparison is reported in Fig. 5, showing good results.

VII. Conclusions

In this letter, we addressed the problem of extracting a feasible set of parameters that characterizes the dynamics of the robot. We identified the dynamic coefficients through a standard least squares algorithm by means of a reverse engineering approach. Thanks to an improved version of the algorithm proposed in [8], we retrieved a set of feasible parameters by solving a nonlinear optimization problem, taking into account several constraints including the physical bounds on the dynamic parameters (such as the total mass of the robot) and the triangle inequalities of the link inertia tensors. The proposed framework was validated by deriving the Lagrangian model of the Franka Emika Panda robot and testing it through experiments. We also validated the extracted feasible set of dynamic parameters using a Newton-Euler routine. To the best of the authors' knowledge, this is the first work that retrieves the dynamic coefficients and a feasible parameter set for the Panda, a robot that is being more and more used in the robotics community. A major feature of the proposed framework is that it can be easily modified to include further (possibly) nonlinear constraints (e.g., based on *a priori* information on the robot).

We are releasing both the dynamic model and the parameters retrieval framework as open-source code, such that they will be available to the robotics community at the following website: http://diag.uniroma1.it/gaz/panda2019.html.

As a future work, the authors would like to consider other possible nonlinear constraints and use the derived dynamic model for detecting, isolating and reacting to collisions between the robot and the environment.

References

[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modeling, Planning and Control*, 3rd ed. London, U.K.: Springer, 2008.

[2] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Trans. Robot.*, vol. 33, no. 6, pp. 1292–1312, Dec. 2017.

[3] E. Magrini and A. De Luca, "Hybrid force/velocity control for physical human-robot collaboration tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2016, pp. 857–863.

[4] J. Hollerbach, W. Khalil, and M. Gautier, "Model identification," in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Berlin, Germany: Springer, 2008, pp. 321–344.

[5] A. Janot, P. Vandanjon, and M. Gautier, "A generic instrumental variable approach for industrial robot identification," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 1, pp. 132–145, Jan. 2014.

[6] J. Jovic, A. Escande, K. Ayusawa, E. Yoshida, A. Kheddar, and G. Venture, "Humanoid and human inertia parameter identification using hierarchical optimization," *IEEE Trans. Robot.*, vol. 32, no. 3, pp. 726–735, Jun. 2016.

[7] V. Bonnet, P. Fraisse, A. Crosnier, M. Gautier, A. González, and G. Venture, "Optimal exciting dance for identifying inertial parameters of an anthropomorphic structure," *IEEE Trans. Robot.*, vol. 32, no. 4, pp. 823–836, Aug. 2016.

[8] C. Gaz, F. Flacco, and A. De Luca, "Extracting feasible robot parameters from dynamic coefficients using nonlinear optimization methods," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 2075–2081.

[9] W. Khalil and E. Dombre, *Modeling, Identification and Control of Robots*. London, U.K.: Hermes Penton, 2002.

[10] Coppelia Robotics, "V-REP virtual robot experimentation platform," 2015. [Online]. Available: http://www.coppeliarobotics.com

[11] V. Mata, F. Benimeli, N. Farhat, and A. Valera, "Dynamic parameter identification in industrial robots considering physical feasibility," *Adv. Robot.*, vol. 19, no. 1, pp. 101–119, 2005.

[12] C. Sousa and R. Cortesão, "Physical feasibility of robot base inertial parameter identification: A linear matrix inequality approach," *Int. J. Robot. Res.*, vol. 33, no. 6, pp. 931–944, 2014. [Online]. Available: https://doi.org/10.1177/0278364913514870

[13] P. Wensing, S. Kim, and J.-J. E. Slotine, "Linear matrix inequalities for physically-consistent inertial parameter identification: A statistical perspective on the mass distribution," *IEEE Robot. Autom. Lett.*, vol. 3, no. 1, pp. 60–67, Jan. 2018.

[14] C. Sousa and R. Cortesão, "Inertia tensor properties in robot dynamics identification: A linear matrix inequality approach," *IEEE/ASME Trans. Mechatronics*, vol. 24, no. 1, pp. 406–411, Feb. 2019.

[15] S. Traversaro, S. Brossette, A. Escande, and F. Nori, "Identification of fully physical consistent inertial parameters using optimization on manifolds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2016, pp. 5446–5451.

[16] M. Spong, "Modeling and control of elastic joint robots," *ASME J. Dyn. Syst. Meas. Control*, vol. 109, no. 4, pp. 310–319, 1987.

[17] C. Gaz, F. Flacco, and A. De Luca, "Identifying the dynamic model used by the KUKA LWR: A reverse engineering approach," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2014, pp. 1386–1392.

[18] J. Swevers, W. Verdonck, and J. De Schutter, "Dynamic model identification for industrial robots," *IEEE Control Syst. Mag.*, vol. 27, no. 5, pp. 58–71, Oct. 2007.

[19] A. Jubien, M. Gautier, and A. Janot, "Dynamic identification of the Kuka LightWeight robot: Comparison between actual and confidential Kuka's parameters," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics*, Jul. 2014, pp. 483–488.

[20] T. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997.

[21] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2009.

[22] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. Boston, MA, USA: Athena Scientific, 1996.