MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Lagrangian Inspired Polynomial Estimator for black-box learning and control of underactuated systems

Giacomuzzo, Giulio; Cescon, Riccardo; Romeres, Diego; Carli, Ruggero; Dalla Libera, Alberto

TR2024-097 July 16, 2024

Abstract

The Lagrangian Inspired Polynomial (LIP) estimator [1] is a black-box estimator based on Gaussian Process Regression, recently presented for the inverse dynamics identification of Lagrangian systems. It relies on a novel multi-output kernel that embeds the structure of the Euler-Lagrange equation. In this work, we extend its analysis to the class of underactuated robots. First, we show that, despite being a black-box model, the LIP allows estimating kinetic and potential energies, as well as the inertial, Coriolis, and gravity components directly from the overall torque measures. Then we exploit these properties to derive a two-stage energy-based controller for the swing-up and stabilization of balancing robots. Experimental results on a simulated Pendubot confirm the feasibility of the proposed approach.

Learning for Dynamics & Control Conference (L4DC) 2024

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Lagrangian Inspired Polynomial Estimator for black-box learning and control of underactuated systems

Giacomuzzo Giulio[1], Riccardo Cescon[2], Diego Romeres[3], Ruggero Carli[1] and Alberto Dalla Libera[1]
[1]University of Padova, Padua, Italy, [2]École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
[3]Mitsubishi Electric Research Laboratories, Cambridge, MA, USA
<giacomuzzo,carlirug,dallaliber>@dei.unipd.it, riccardo.cescon@epfl.ch, romeres@merl.com

Abstract

The Lagrangian Inspired Polynomial (LIP) estimator [1] is a black-box estimator based on Gaussian Process Regression, recently presented for the inverse dynamics identification of Lagrangian systems. It relies on a novel multi-output kernel that embeds the structure of the Euler-Lagrange equation. In this work, we extend its analysis to the class of underactuated robots. First, we show that, despite being a black-box model, the LIP allows estimating kinetic and potential energies, as well as the inertial, Coriolis, and gravity components directly from the overall torque measures. Then we exploit these properties to derive a two-stage energy-based controller for the swing-up and stabilization of balancing robots. Experimental results on a simulated Pendubot confirm the feasibility of the proposed approach.

I. Introduction

In recent years, learning for control in complex robotics applications has gained increasing interest. Several machine learning methods have been presented, both for modeling and direct synthesis of the controller. In this context, underactuated robots (UR) are an important class of systems. UR are mechanical systems characterized by fewer control inputs than degrees of freedom (DOF). Such systems are ubiquitous in robotics: examples are manipulators with passive joints, autonomous bicycles and motorcycles, bipedal robots, and most of aerospace and marine vehicles. To control such systems, for instance, several Reinforcement Learning algorithms have been explored. These algorithms aim at automatically learning a control law, see for example [2] for a model-free approach or [3], [4] for model-based solutions. Although effective, generally these methods ignore previous contributions on UR from control theory. As a result, they typically require a huge amount of interactions with the system and do not provide any performance guarantees.

A viable alternative consists of combining model learning with classic model-based control methods. Within this approach, a model of the system dynamics is derived from experimental data using machine learning techniques without the need for manually deriving accurate models of the dynamics, which is the main drawback of classic model-based methods. Then, the learned model is used inside a model-based controller, exploiting results from control theory.

Typically, such control strategies rely on the inverse dynamics model, which relates torques to the robot trajectories. The problem of learning the inverse dynamics from data has been extensively explored in the literature [5], [6], [7], [8]. An interesting class of solutions is represented by black-box methods, which propose to learn the inverse dynamics map by means of universal approximators, without any knowledge about the underlying system kinematics and dynamics [9], [10], [11], [12]. However, learning inverse dynamics models for the control of UR is particularly challenging. Indeed, most of the strategies proposed for UR control require models that return the different inverse dynamics components and the system energy, see [13]. Typically, black-box models do not provide such information. Underactuation further exacerbates these problems: torques of the underactuated dimensions are constant signals equal to zero, leading to an ill-posed estimation problem.

Physics-Informed (PI) models are a class of black-box solutions that could mitigate the aforementioned problem [14], [15], [16]. PI methods have been recently proposed to improve the data efficiency and generalization of black-box estimators. Instead of learning the inverse dynamics in a completely unstructured manner, PI methods embed knowledge from physics as priors in the model structure. As a result, learned models not only show better accuracy but also provide insights about relevant physical properties. For instance, in [15], the authors proposed Deep Lagrangian Networks (DeLaNs), a neural networks model with structure inspired by Euler-Lagrange (EL) equations. Similar ideas have been explored in [17], [18], [19].

In the context of inverse dynamics identification of fully-actuated manipulators, we proposed a PI solution based on Gaussian Process Regression (GPR), named LIP estimator [1]. The LIP estimator derives a multioutput kernel of the torques by encoding EL equations. In this work, we extend the analysis of the LIP estimator to the class of UR and show its applicability in traditional model-based control strategies for UR.

Contribution: Our contribution is twofold. First, we review the recently proposed LIP estimator and extend its analysis in the context of UR, showing its ability to learn the inverse dynamics map, its different components, and the kinetic and potential energies. Second, we exploit the LIP to derive an energy-based controller for the stabilization of balancing UR. The controller is based on two modules: an energy-based controller for the swing-up, and an LQR which stabilizes the system

once it has reached the unstable equilibrium. The energy-based controller performs a partial feedback linearization on the actuated system and a regulation of the energy to steer the non-actuated system to a trajectory passing through the unstable equilibrium. Once the system is sufficiently close to the target, the control is switched to the LQR. We show that the LIP model is suitable to implement this kind of controller since it returns the inertia matrix, the Coriolis and gravity torques, energy estimates, and the linearization of the system dynamics required by the LQR. This allows performing the swing-up and stabilization of the balancing robot in a completely black-box fashion. The method is validated on a simulated setup involving a Pendubot [20]. The LIP method is compared with DeLaNs and with the true analytical model, which has been included as a baseline. Results confirm the feasibility of the proposed approach.

The paper unfolds as follows. In Section II, we review the modeling and control of UR and we define the inverse dynamics identification problem. Then, in Section III, we present the LIP estimator and show how to derive the balancing controller. Section IV reports the results of the performed experiments, while Section V draws the conclusions.

II. Background and problem statement

In this section, we first provide the inverse dynamics model of UR under the rigid body assumption. Then we review the class of energy-based controllers we consider in this work.

A. Inverse dynamics

Consider a mechanical system with nDOF and let $\boldsymbol{q} \in \mathbb{R}^n$ be the vector of generalized coordinates. We assume the system to have $m < n$ control inputs, each of which actuates a single DOF. We partition the vector $\boldsymbol{q} \in \mathbb{R}^n$ as $\boldsymbol{q}^T = [\boldsymbol{q}_1^T, \boldsymbol{q}_2^T]$, where $\boldsymbol{q}_1 \in \mathbb{R}^m$ and $\boldsymbol{q}_2 \in \mathbb{R}^{n-m}$ refer respectively to the actuated and the non-actuated DOFs.

Under the rigid body assumption, the inverse dynamics of the system can be derived from the Euler-Lagrange equations (see [13]) as

$$\underbrace{\begin{bmatrix} M_{11}(\boldsymbol{q}) & M_{12}(\boldsymbol{q}) \\ M_{21}(\boldsymbol{q}) & M_{22}(\boldsymbol{q}) \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}}_1 \\ \ddot{\boldsymbol{q}}_2 \end{bmatrix}}_{\boldsymbol{m}(\boldsymbol{q},\ddot{\boldsymbol{q}})} + \underbrace{\begin{bmatrix} c_1(\boldsymbol{q},\dot{\boldsymbol{q}}) \\ c_2(\boldsymbol{q},\dot{\boldsymbol{q}}) \end{bmatrix}}_{\boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}})} + \underbrace{\begin{bmatrix} g_1(\boldsymbol{q}) \\ g_2(\boldsymbol{q}) \end{bmatrix}}_{\boldsymbol{g}(\boldsymbol{q})} = \underbrace{\begin{bmatrix} \boldsymbol{\tau}_1 \\ 0 \end{bmatrix}}_{\boldsymbol{\tau}}, \tag{1}$$

where

$$M(\boldsymbol{q}) = \begin{bmatrix} M_{11}(\boldsymbol{q}) & M_{12}(\boldsymbol{q}) \\ M_{21}(\boldsymbol{q}) & M_{22}(\boldsymbol{q}) \end{bmatrix}$$

is the symmetric, positive definite inertia matrix, $\boldsymbol{m}(\boldsymbol{q}, \ddot{\boldsymbol{q}})$ represents the inertial torque, $\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ accounts for the Coriolis and centripetal torques, $\boldsymbol{g}(\boldsymbol{q})$ represents the gravity contribution while $\boldsymbol{\tau}_1 \in \mathbb{R}^m$ is the vector of generalized torques produced by the m actuators. For the sake of brevity, in the following, we will explicitly point out the dependency on $\boldsymbol{q}, \dot{\boldsymbol{q}}$ only when necessary.

The inverse dynamics identification problem consists of estimating the map in (1) that relates $\tilde{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ and the torques $\boldsymbol{\tau}$ from a set of noisy measures. Black-box solutions treat the inverse dynamics as an unknown function and, generally, rely on universal approximators to estimate the function from experimental data. Several solutions, including our approach, adopt GPR [21], which is a framework for Bayesian inference widely used in machine learning and robotics.

Remark 1: The standard approach when using GPR consists of considering the different torque components independently and solving n independent regression problems, one for each generalized coordinate. In UR, torques of the under-actuated dimensions are constant signals equal to zero, leading to an ill-posed estimation problem if the components are learned independently of each other, which prevents the possibility of deriving any inverse dynamics model useful for model-based control strategies.

B. Model-based balancing control of underactuated systems

In what follows, we focus our attention on a particular class of robots described by (1), known as balancing systems. Common examples of balancing systems are the Cartpole, the Furuta Pendulum, the Acrobot, and the Pendubot. For such systems, the typical control challenge requires swinging up and balancing the robot in the unstable equilibrium point, hereafter denoted by $x_\star = [\boldsymbol{q}_\star^T, \dot{\boldsymbol{q}}_\star^T]^T$ with $\dot{\boldsymbol{q}}_\star = 0$. In the remainder of this section, we review one of the most common approaches presented in the literature, which combines an energy-based controller for the swing-up and an LQR for the balancing.

1) Energy-based swing-up controller: We review the energy-based swing-up controller presented in [22], [23]. The first step consists of a partial feedback linearization. From (1) we isolate the dynamics of the actuated and non-actuated subsystems respectively as

$$M_{11}\ddot{\boldsymbol{q}}_1 + M_{12}\ddot{\boldsymbol{q}}_2 + c_1 + g_1 = \boldsymbol{\tau}_1, \tag{2}$$

$$M_{21}\ddot{\boldsymbol{q}}_1 + M_{22}\ddot{\boldsymbol{q}}_2 + c_2 + g_2 = 0. \tag{3}$$

We can solve (3) for $\ddot{\boldsymbol{q}}_2$ as $\ddot{\boldsymbol{q}}_2 = -M_{22}^{-1}(M_{21}\ddot{\boldsymbol{q}}_1 + c_2 + g_2)$ since M_{22} is invertible (given that $M > 0$). Substituting the resulting expression into (2) leads to

$$\bar{M}_{11}\ddot{\boldsymbol{q}}_1 + \bar{c}_1 + \bar{g}_1 = \boldsymbol{\tau}_1, \tag{4}$$

with $\bar{M}_{11} = M_{11} - M_{12}M_{22}^{-1}M_{21}$, $\bar{c}_1 = c_1 - M_{12}M_{22}^{-1}c_2$ and $\bar{g}_1 = g_1 - M_{12}M_{22}^{-1}g_2$.

A feedback linearizing controller for (4) can be defined as

$$\boldsymbol{\tau}_1 = \bar{M}_{11}u + \bar{c}_1 + \bar{g}_1, \tag{5}$$

where u is a design parameter. Choosing $\boldsymbol{\tau}_1$ as in (5) leads to the following closed-loop system

$$\ddot{\boldsymbol{q}}_1 = u \tag{6}$$

$$M_{22}\ddot{\boldsymbol{q}}_2 + c_2 + g_2 = -M_{21}u \tag{7}$$

We obtained a linear second-order dynamics for the actuated subsystem. Selecting u according to

$$u = -k_1\boldsymbol{q}_1 - k_2\dot{\boldsymbol{q}}_1 + \bar{u}, \quad k_1, k_2 > 0, \tag{8}$$

makes the linear subsystem in (6) asymptotically stable for $\bar{u} = 0$. The remaining design problem is the choice of $\bar{u}$, which can be used to stabilize the non-actuated dynamics in (7). Most of the existing literature proposes to design $\bar{u}$ based on energy concepts, penalizing the mismatch w.r.t. the system energy at the desired configuration. This leads to control laws of the form

$$\bar{u} = f_e(\mathcal{T}, \mathcal{V}), \tag{9}$$

where $\mathcal{T}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{1}{2}\dot{\boldsymbol{q}}^T M(\boldsymbol{q})\dot{\boldsymbol{q}}$ is the kinetic energy, while $\mathcal{V}(\boldsymbol{q})$ denotes the potential energy. However, the choice of f_e depends on the system of interest. We will clarify our choice in the experimental section after introducing our setup. Examples of energy-compensating terms for common balancing robots can be found in [22], [24], [25].

Remark 2: As highlighted in [22], the controller presented in this section is not stabilizing the system to a fixed point but only to a manifold. For this reason, in applications such as the swing-up of balancing robots, the control must switch to another controller achieving local asymptotic stability to the equilibrium.

2) Linear quadratic regulator: To stabilize the system at the equilibrium $\boldsymbol{x}_\star$ we resort to a Linear Quadratic Regulator (LQR). First, we provide a state space description for the system with dynamics expressed in (1). Let the system state $\boldsymbol{x}$ be such that $\boldsymbol{x}^T = [\boldsymbol{q}^T, \dot{\boldsymbol{q}}^T]$. The state evolution can be derived from (1) as

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ -M^{-1}(\boldsymbol{q})[\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})] \end{bmatrix} + \begin{bmatrix} 0_n \\ M^{-1}(\boldsymbol{q}) \end{bmatrix} \boldsymbol{\tau} = f(\boldsymbol{x}) + g(\boldsymbol{x}, \boldsymbol{\tau}) \tag{10}$$

Now we linearize the non-linear system in (10) around $\boldsymbol{x}_\star^T$. Moreover, let $\boldsymbol{\tau}_\star$ be the reference input at the equilibrium. Applying a first-order Taylor expansion, the system dynamics around $(\boldsymbol{x}_\star, \boldsymbol{\tau}_\star)$ can be approximated as

$$\dot{\boldsymbol{x}} \approx A(\boldsymbol{x} - \boldsymbol{x}_\star) + B(\boldsymbol{\tau} - \boldsymbol{\tau}_\star). \tag{11}$$

Recalling that at the equilibrium $\dot{\boldsymbol{q}}_\star = 0$, matrices A and B are

$$A = \frac{\partial f}{\partial \boldsymbol{x}}^T \Big|_{\boldsymbol{x}_\star, \boldsymbol{\tau}_\star} = \begin{bmatrix} 0 & I \\ -M^{-1}(\boldsymbol{q})\frac{\partial \boldsymbol{g}(\boldsymbol{q})}{\partial \boldsymbol{q}} & M^{-1}(\boldsymbol{q})C(\boldsymbol{q}, \dot{\boldsymbol{q}}) \end{bmatrix} \Big|_{\boldsymbol{x}_\star, \boldsymbol{\tau}_\star} = \begin{bmatrix} 0 & I \\ -M^{-1}(\boldsymbol{q}_\star)\frac{\partial \boldsymbol{g}(\boldsymbol{q}_\star)}{\partial \boldsymbol{q}} & 0 \end{bmatrix} \tag{12}$$

and

$$B = \frac{\partial g}{\partial \boldsymbol{\tau}}^T \Big|_{\boldsymbol{x}_\star, \boldsymbol{\tau}_\star} = \begin{bmatrix} 0 \\ M^{-1}(\boldsymbol{q}_\star) \end{bmatrix}, \tag{13}$$

where $C(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{n \times n}$ is the skew-symmetric Coriolis matrix, such that $\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$.

Then, we introduce the infinite horizon control problem on the linearized system in (11), namely,

$$\underset{\boldsymbol{\tau}}{\operatorname{argmin}} \int_0^\infty (\boldsymbol{x} - \boldsymbol{x}_\star)^T Q(\boldsymbol{x} - \boldsymbol{x}_\star) + (\boldsymbol{\tau} - \boldsymbol{\tau}_\star)^T R(\boldsymbol{\tau} - \boldsymbol{\tau}_\star) dt,$$

which leads to the control input $\boldsymbol{\tau} = -K\boldsymbol{x}$, with K being the optimal control gain, which can be computed by solving the *continuous time algebraic Riccati equation*.

III. Lagrangian Inspired Polynomial Estimator for modeling and control of under-actuated systems

In this section, we briefly review the proposed inverse dynamics LIP estimator for the control of UR. Then, we detail the application of the LIP model to the UR control described in Section II.

A. LIP estimator

The LIP estimator is based on GPR [21], which is a framework for Bayesian inference widely used in machine learning and robotics applications. Generally, GPR solutions for inverse dynamics identification model each torque component $\tau^i(\tilde{\boldsymbol{x}})$ as a Gaussian Process (GP) by assuming $\tau^i(\tilde{\boldsymbol{x}})$s are independent given $\tilde{\boldsymbol{x}}$, and then apply standard GPR inference. As discussed in the previous section, this approach is not effective for UR. The LIP estimator follows an alternative strategy and defines the kinetic and potential energies as two independent GPs. Then, it derives a multi-output kernel of the torques by exploiting EL equations. In this way, the inverse dynamics problem is well defined also in the UR setup.

Our framework models $\mathcal{T}$ and $\mathcal{V}$ as independent GPs, namely $\mathcal{T} \sim \mathcal{GP}\left(0, k^{\mathcal{T}}(\cdot,\cdot)\right)$ and $\mathcal{V} \sim \mathcal{GP}\left(0, k^{\mathcal{V}}(\cdot,\cdot)\right)$, where $k^{\mathcal{T}}$ and $k^{\mathcal{V}}$ are the kernels that defines the covariance of $\mathcal{T}$ and $\mathcal{V}$. For instance, let $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}'$ be two input locations, then the covariance between the values of $\mathcal{T}$ at $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}'$ is $E\left[\mathcal{T}(\tilde{\boldsymbol{x}})\mathcal{T}(\tilde{\boldsymbol{x}}')\right] = k^{\mathcal{T}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}')$. For convenience, we introduce also $K^{\mathcal{T}}_{XX'}$, that is the matrix that collects $k^{\mathcal{T}}$ evaluated at $X = \{\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_N\}$, $X' = \{\tilde{\boldsymbol{x}}'_1, \ldots, \tilde{\boldsymbol{x}}'_M\}$:

$$K^{\mathcal{T}}_{XX'} = \begin{bmatrix} k^{\mathcal{T}}(\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}'_1) & \ldots & k^{\mathcal{T}}(\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}'_M) \\ \vdots & \vdots & \vdots \\ k^{\mathcal{T}}(\tilde{\boldsymbol{x}}_N, \tilde{\boldsymbol{x}}'_1) & \ldots & k^{\mathcal{T}}(\tilde{\boldsymbol{x}}_N, \tilde{\boldsymbol{x}}'_M) \end{bmatrix}.$$

Similar definitions hold for $k^{\mathcal{V}}$ and all the other kernels we will introduce. The LIP estimator defines $k^{\mathcal{T}}$ and $k^{\mathcal{V}}$ relying on a polynomial formulation that extends [8]. Due to space constraint, we refer the interested reader to [1] for further details.

Notice that, (i) once we define $\mathcal{T}$ and $\mathcal{V}$ as zero-mean GPs, for the properties of GPs also the Lagrangian function $\mathcal{L} = \mathcal{T} - \mathcal{V}$ is a zero-mean GP with kernel $k^{\mathcal{L}}(\cdot,\cdot) = k^{\mathcal{T}}(\cdot,\cdot) + k^{\mathcal{V}}(\cdot,\cdot)$, namely $\mathcal{L} \sim \mathcal{GP}\left(0, k^{\mathcal{L}}(\cdot,\cdot)\right)$. Furthermore, (ii) under rigid body assumptions, each $\tau^i(\tilde{\boldsymbol{x}})$ is described by a linear differential equation of $\mathcal{L}$, namely, $\tau^i = \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}^i}\right) - \frac{\partial \mathcal{L}}{\partial q^i}$. Expanding explicit derivations w.r.t. time, we obtain

$$\tau^i = \sum_{j=1}^{n} \left(\frac{\partial^2 \mathcal{L}}{\partial \dot{q}^i \partial \dot{q}^j} \ddot{q}^j + \frac{\partial^2 \mathcal{L}}{\partial \dot{q}^i \partial q^j} \dot{q}^j \right) - \frac{\partial \mathcal{L}}{\partial q^i} =: \mathcal{G}_i \mathcal{L},$$

where $\mathcal{G}_i$ is the linear operator that maps $\mathcal{L}$ in the linear differential equation of τ^i. Finally, (iii) GPs are closed w.r.t. linear operators [26], namely, if f is a zero-mean GP with kernel $k^f(\cdot,\cdot)$ and $g(\tilde{\boldsymbol{x}}) = \mathcal{G} f(\tilde{\boldsymbol{x}})$, then also $g(\tilde{\boldsymbol{x}})$ is a zero-mean GP with kernel $k^g(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') = \mathcal{G}\mathcal{G}' k^f(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}')$. The last expression means that $k^g(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}')$ is obtained by applying two times the operator $\mathcal{G}$ to $k^f(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}')$, first w.r.t. the input $\tilde{\boldsymbol{x}}$ then w.r.t. $\tilde{\boldsymbol{x}}'$.

Based on (i), (ii), and (iii), we conclude that torques are a zero-mean GP, with covariance defined by a multi-output kernel $k^{\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') \in \mathbb{R}^{n \times n}$ that encodes the EL equations, and is expressed as

$$k^{\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') = \begin{bmatrix} \mathcal{G}_1 \mathcal{G}'_1 k^{\mathcal{L}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') & \ldots & \mathcal{G}_1 \mathcal{G}'_n k^{\mathcal{L}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') \\ \vdots & \ddots & \vdots \\ \mathcal{G}_n \mathcal{G}'_1 k^{\mathcal{L}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') & \ldots & \mathcal{G}_n \mathcal{G}'_n k^{\mathcal{L}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') \end{bmatrix}. \tag{14}$$

To derive (14) we applied the multi-output version of property (iii). For a more detailed derivation, we refer the interested reader to [1].

Once k^{τ} is defined, we can compute torque estimates following standard GPR. Let X be a set of N training input locations, and $\boldsymbol{y} = \left[\boldsymbol{y}_1^T, \ldots, \boldsymbol{y}_N^T\right]^T$ the respective torque measurements, with $\boldsymbol{y}_i \in \mathbb{R}^n$ equal to the torque measures at input $\tilde{\boldsymbol{x}}_i$. The LIP torque estimate in a general input location $\tilde{\boldsymbol{x}}$ is

$$\hat{\boldsymbol{\tau}}(\tilde{\boldsymbol{x}}) = K^{\tau}_{\tilde{\boldsymbol{x}}X} \left(K^{\tau}_{XX} + \Sigma_e\right)^{-1} \boldsymbol{y}, \tag{15}$$

where Σ_e is a regularization parameter that accounts for the additive Gaussian noise modeled by GPR. For each dimension, we assumed independent and identically distributed noise, thus obtaining a block diagonal matrix with equal diagonal blocks, namely $\Sigma_e = \text{diag}(\Sigma_{e_1}, \ldots, \Sigma_{e_N})$, with $\Sigma_{e_1} = \Sigma_{e_2} = \ldots = \Sigma_{e_N} = \text{diag}(\sigma^2_{e_1}, \ldots, \sigma^2_{e_n})$, where $\sigma^2_{e_i}$ is the variance of the τ^i measures.

B. LIP for control

In this section, we describe how to estimate the kinetic and potential energies, the inertial, Coriolis, and gravity vector as well as $\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}}$ required by the control laws presented in Section II.

1) Kinetic and potential energies: The kinetic energy $\mathcal{T}$ and potential energy $\mathcal{V}$ are required to implement the energy based control law described by (9). The LIP model provides a principled way to estimate them from the torque measurements $\boldsymbol{y}$. Indeed, within the LIP framework, $\mathcal{T}$, $\mathcal{V}$ and $\boldsymbol{\tau}$ are jointly Gaussian distributed, since the prior of $\boldsymbol{\tau}$ is derived by applying the linear operator $\mathcal{G}$ to the kinetic and potential GPs $\mathcal{T}$ and $\mathcal{V}$. The covariances between $\mathcal{T}$ and $\boldsymbol{\tau}$ and between $\mathcal{V}$ and $\boldsymbol{\tau}$ at general input locations $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}'$ are

$$\text{Cov}[\mathcal{T}(\tilde{\boldsymbol{x}}), \boldsymbol{\tau}(\tilde{\boldsymbol{x}}')] = \text{Cov}[\mathcal{T}(\tilde{\boldsymbol{x}}), \mathcal{G}\mathcal{L}(\tilde{\boldsymbol{x}}')] =: k^{\mathcal{T}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}'), \tag{16a}$$

$$\text{Cov}[\mathcal{V}(\tilde{\boldsymbol{x}}), \boldsymbol{\tau}(\tilde{\boldsymbol{x}}')] = \text{Cov}[\mathcal{V}(\tilde{\boldsymbol{x}}), \mathcal{G}\mathcal{L}(\tilde{\boldsymbol{x}}')] =: k^{\mathcal{V}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}'). \tag{16b}$$

Recalling that we model $\mathcal{T}$ and $\mathcal{V}$ as independent GPs, and in view of the properties of GPs under linear operators, we obtain

$$k^{\mathcal{T}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') = \text{Cov}[\mathcal{T}(\tilde{\boldsymbol{x}}), \mathcal{G}\mathcal{T}(\tilde{\boldsymbol{x}}')] = \left[\mathcal{G}' k^{\mathcal{T}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}')\right]^T = [\mathcal{G}_1' k^{\mathcal{T}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}'), \dots, \mathcal{G}_n' k^{\mathcal{T}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}')] \tag{17a}$$

$$k^{\mathcal{V}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') = \text{Cov}[\mathcal{V}(\tilde{\boldsymbol{x}}), \mathcal{G}\mathcal{V}(\tilde{\boldsymbol{x}}')] = \left[\mathcal{G}' k^{\mathcal{V}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}')\right]^T = [\mathcal{G}_1' k^{\mathcal{V}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}'), \dots, \mathcal{G}_n' k^{\mathcal{V}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}')] \tag{17b}$$

From the posterior distributions of $\mathcal{T}$ and $\mathcal{V}$ given $\boldsymbol{y}$ we can obtain an estimate of the energies at arbitrary input location $\tilde{\boldsymbol{x}}$ as

$$\hat{\mathcal{T}}(\tilde{\boldsymbol{x}}) = K_{\tilde{\boldsymbol{x}}X}^{\mathcal{T}\tau}(K_{XX}^{\tau} + \Sigma_e)^{-1}\boldsymbol{y}, \quad \hat{\mathcal{V}}(\tilde{\boldsymbol{x}}) = K_{\tilde{\boldsymbol{x}}X}^{\mathcal{V}\tau}(K_{XX}^{\tau} + \Sigma_e)^{-1}\boldsymbol{y}, \tag{18}$$

where the covariance matrices $K_{\tilde{\boldsymbol{x}}X}^{\mathcal{T}\tau} \in \mathbb{R}^{1\times nN}$ and $K_{\tilde{\boldsymbol{x}}X}^{\mathcal{V}\tau} \in \mathbb{R}^{1\times nN}$ are obtained as $K_{\tilde{\boldsymbol{x}}X}^{\mathcal{T}\tau} = \left[k^{\mathcal{T}\tau}(\tilde{\boldsymbol{x}}, \boldsymbol{x}_1), \dots, k^{\mathcal{T}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_N)\right]$ and $K_{\tilde{\boldsymbol{x}}X}^{\mathcal{V}\tau} = \left[k^{\mathcal{V}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_1), \dots, k^{\mathcal{V}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_N)\right]$.

2) Torque components: The energy based control law in (5) requires to estimate $\boldsymbol{m}$, $\boldsymbol{c}$ and $\boldsymbol{g}$, while the LQR in Sec. II-B.2 requires the inverse of the inertia matrix M as well as the term $\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}}$. These quantities are derived similarly to how we computed the energies in the previous section.

First, the inertia matrix is estimated component wise. The element in position ij of M is

$$M^{ij}(\boldsymbol{q}) = \frac{\partial^2 \mathcal{T}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial \dot{q}^i \partial \dot{q}^j} =: \mathcal{G}_{M^{ij}} \mathcal{T}(\tilde{\boldsymbol{x}}),$$

where we introduced the linear operator $\mathcal{G}_{M^{ij}}$. The covariance between $M_{i,j}$ and $\boldsymbol{\tau}$ at general input locations $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}'$ can be computed as

$$\text{Cov}\left[M^{ij}(\boldsymbol{x}), \boldsymbol{\tau}(\tilde{\boldsymbol{x}}')\right] = \text{Cov}\left[\mathcal{G}_{M^{ij}}\mathcal{T}(\tilde{\boldsymbol{x}}), \mathcal{G}\mathcal{T}(\tilde{\boldsymbol{x}}')\right] = \mathcal{G}_{M^{ij}}\mathcal{G}k^{\mathcal{T}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') =: k^{M^{ij}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}').$$

Accordingly, we can estimate M^{ij} at any input location $\tilde{\boldsymbol{x}}$ as $\hat{M}^{ij}(\tilde{\boldsymbol{x}}) = K_{\tilde{\boldsymbol{x}}X}^{M^{ij}\boldsymbol{\tau}}(K_{XX}^{\tau} + \Sigma_e)^{-1}\boldsymbol{y}$. with $K_{\tilde{\boldsymbol{x}}X}^{M^{ij}\tau} = [k^{M^{ij}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_1), \dots, k^{M^{ij}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_N)]$. Then, from the estimate of the inertia matrix we can derive an estimate of the inertial torque component as $\hat{\boldsymbol{m}}(\tilde{\boldsymbol{x}}) = \hat{M}(\tilde{\boldsymbol{x}})\ddot{\boldsymbol{q}}$.

Next, we can estimate the gravity contribution $\boldsymbol{g}$. Recall that the i-th component of the vector $\boldsymbol{g}$ is defined as $g^i(\boldsymbol{q}) = \frac{\partial \mathcal{V}(\boldsymbol{q})}{\partial q^i}$. The covariance between g^i and $\boldsymbol{\tau}$ at general input locations $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}'$ is

$$\text{Cov}\left[g^i(\boldsymbol{x}), \boldsymbol{\tau}(\tilde{\boldsymbol{x}}')\right] = \text{Cov}\left[\frac{\partial \mathcal{V}(\tilde{\boldsymbol{x}})}{\partial q^i}, \mathcal{G}\mathcal{V}(\tilde{\boldsymbol{x}}')\right] = \frac{\partial}{\partial q^i}\mathcal{G}' k^{\mathcal{V}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') =: k^{g^i\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}').$$

Accordingly g^i can be estimated at any input location $\tilde{\boldsymbol{x}}$ as $\hat{g}(\tilde{\boldsymbol{x}}) = K_{\tilde{\boldsymbol{x}}X}^{g^i\tau}(K_{XX}^{\tau} + \Sigma_e)^{-1}\boldsymbol{y}$, with $K_{\tilde{x}X}^{g^i\tau} = [k^{g^i\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_1), \dots, k^{g^i\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_N)]$.

Given the estimates of $\boldsymbol{m}$ and $\boldsymbol{g}$, it is possible to obtain an estimate of the Coriolis and centripetal contribution $\boldsymbol{c}$ as $\hat{\boldsymbol{c}}(\tilde{\boldsymbol{x}}) = \hat{\boldsymbol{\tau}}(\tilde{\boldsymbol{x}}) - \hat{\boldsymbol{m}}(\tilde{\boldsymbol{x}}) - \hat{\boldsymbol{g}}(\tilde{\boldsymbol{x}})$.

Finally, the matrix $\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}} \in \mathbb{R}^{n\times n}$ required in the computation of matrix A in (12), is estimated following the same procedure adopted for the inertia matrix. Its element in position ij is given by

$$\left[\frac{\partial \boldsymbol{g}(\boldsymbol{q})}{\partial \boldsymbol{q}}\right]_{ij} = \frac{\partial^2 \mathcal{V}(\boldsymbol{q})}{\partial q^i \partial q^j} =: \mathcal{G}_{G^{ij}}\mathcal{V}(\boldsymbol{q}) =: G^{ij}(\tilde{\boldsymbol{x}}).$$

Then, the covariance between $G^{ij}(\tilde{\boldsymbol{x}})$ and the torque vector $\boldsymbol{\tau}$ at general input locations $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}'$ is

$$\text{Cov}\left[G^{ij}(\tilde{\boldsymbol{x}}), \boldsymbol{\tau}(\tilde{\boldsymbol{x}}')\right] = \text{Cov}\left[\mathcal{G}_{G^{ij}}\mathcal{V}(\boldsymbol{q}), \mathcal{G}\mathcal{V}(\tilde{\boldsymbol{x}}')\right] = \mathcal{G}_{G^{ij}}\mathcal{G}' k^{\mathcal{V}}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') =: k^{G^{ij}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}').$$

Finally, the estimate at a general input location $\tilde{\boldsymbol{x}}$ is computed as $\hat{G}^{ij}(\tilde{\boldsymbol{x}}) = K_{\tilde{\boldsymbol{x}}X}^{G^{ij}\tau}(K_{XX}^{\tau} + \Sigma_e)^{-1}\boldsymbol{y}$, with $K_{\tilde{\boldsymbol{x}}X}^{G^{ij}\tau} = [k^{G^{ij}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_1), \dots, k^{G^{ij}\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_N)]$.

IV. Experimental results

We chose the Pendubot as a benchmark for testing the LIP estimator performance on the estimation and control of a UR. The Pendubot is a double-inverted pendulum composed of two links with only the first one actuated. In this example the links lengths are $l_1 = l_2 = 20\,cm$ and masses are $m_1 = m_2 = 0.3\,kg$ respectively. The task consists of swinging up and controlling the system in the unstable equilibrium point. To carry out the task, we first derive an inverse dynamics model using the LIP estimator, then we implement the controller presented in Section II with the learned model. We implemented all the simulations and controllers in Python and with the auxiliary library PyTorch, [27].

A. Estimation evaluation

First, we test the ability to estimate the torque $\boldsymbol{\tau}$ and the components $\mathbf{m}(\boldsymbol{q}, \ddot{\boldsymbol{q}}), \mathbf{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}), \mathbf{g}(\boldsymbol{q})$ and the potential and kinetic energies on the simulated Pendubot. The trajectories used for training and test were collected simulating (1). The input torques are different realizations of filtered white Gaussian noise with variance 0.5 and cutoff frequency of the filter 1 Hz. Test trajectories are significantly different from the training trajectories, because of high accelerations that are not present in the training set. The hyperparameters of the LIP estimator were optimized by marginal likelihood maximization. As a baseline, the LIP model is compared with the DeLaN model and we considered the same structure as in [15] and used the Adam optimizer. The accuracy when predicting torques and energies is evaluated by means of Mean Squared Error (MSE) in Fig. 1. We also show the estimated trajectories on the test dataset in Fig. 2. The proposed LIP model outperforms the

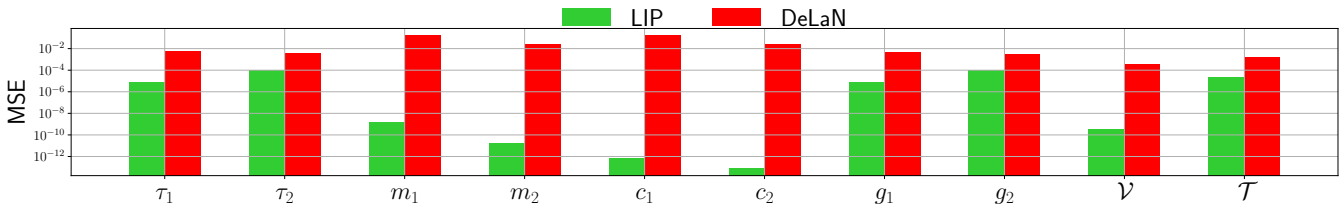

Fig. 1. MSE of torque components and energies of the LIP (green) and DeLaN (red).

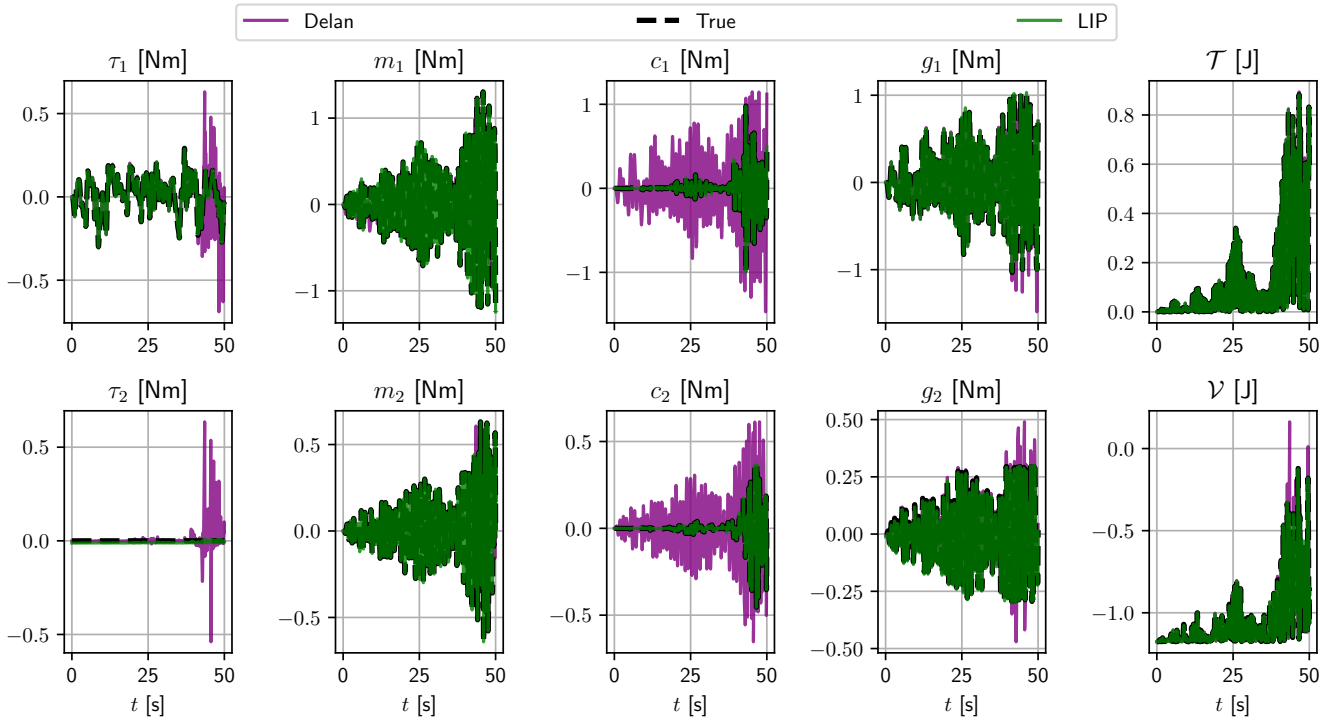

Fig. 2. Torques and energies and estimates. The dashed blue line represents the ground truth, LIP and DeLaN are in green and red, respectively.

NN-based torque estimator and can also well estimate all the components of the torque and the energies. Instead, even though the DeLaN produces reasonable estimates of torque and energies completely misses the decomposition of inertia

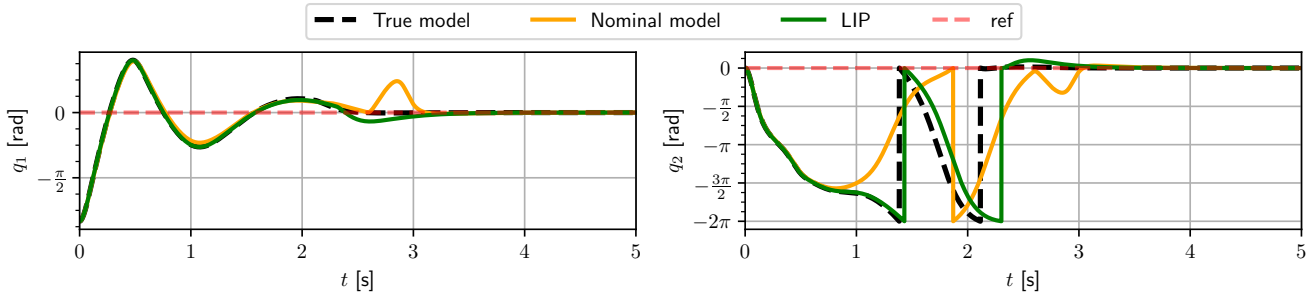

Fig. 3. Time evolution of the generalized coordinates q_1 and q_2. In dashed black is the real controller, in green the LIP-based one, and in orange the nominal one.

and Coriolis components. The ability of the LIP model to well capture all the different components is fundamental for model-based controllers like the energy-based one presented in Section II.

B. Control performance evaluation

The goal of this section is to show that using the black-box LIP model we can perform a complete swing up and balancing of the Pendubot without requiring the knowledge of the model. We implemented the general control scheme described in (5) and (8), with $k_1 := k_p/k_d$ and $k_2 := k_v/k_d$. Regarding the choice of $\bar{u}$, we implemented the same strategy as in [25], namely

$$\bar{u} = \frac{k_E(E - E^\star)\Xi}{k_d(k_d M_{22} + k_E(E - E^\star)|M|)},$$

with $\Xi = |M|(k_p q_1 + k_v \dot{q}_1) + k_d(M_{12}(c_2 + g_2) - M_{22}(c_1 + g_1))$. The values chosen for the controller's gains are $k_E = 0.8$, $k_p = 0.31$, $k_d = 0.029$ and $k_v = 0.064$. The energy-based controller is switched to the LQR when the Pendubot reaches a neighborhood of the vertical equilibrium, described by the condition $|q_1| + |q_2| + 0.1|\dot{q}_1| + 0.1|\dot{q}_2| < \epsilon$, with $\epsilon = 0.6$.

The feedback matrix K of the LQR controller is computed using the *lqr* routine of the *Python Control Systems Toolbox* with $Q = I$ and $R = 100$, obtaining $K = [8.2\ 8.2\ 1.8\ 1.2]$. The response of the LIP-based controller is compared against the response given by the true model and a nominal model. In the latter, we considered uncertain knowledge of the system's parameters, assuming that $m_1 = 0.315\,kg$, $m_2 = 0.285\,kg$, $l_1 = 20.4\,cm$, and $l_1 = 19.2\,cm$. The DeLaN estimator has not been considered for this task, since it does not provide the possibility to implement the LQR controller. Fig. 3 depicts the time evolution of the joints' variables. We can notice that the LIP based controller is capable of swinging up and balancing the system and that the trajectory produced is almost identical to the one we obtain using the true model. With the nominal model we are still able to achieve the same goal, however the performances of this controller are worse than the data-driven ones. The swing-up time needed to reach a neighborhood of the vertical position and switch to the LQR is higher. Also, the stabilizing controller takes more time, with a larger overshoot.

V. Conclusions and Future Works

In this paper, we extended our GP-based PI inverse dynamics estimator, named Lagrangian Inspired Polynomial estimator, to modeling and control of UR. Differently from standard GP-based black-box models, the LIP estimator, besides deriving an inverse dynamics model in a black-box fashion, can also estimate the different dynamics components and potential and kinetic energies. Thanks to these properties, the resulting model can rely on model-based controllers from control theory to solve control tasks for UR. The proposed strategy has been compared in simulation with state-of-art solutions, confirming the effectiveness of the proposed strategy.

Acknowledgements

Alberto Dalla Libera was supported by PNRR research activities of the consortium iNEST (Interconnected North-Est Innovation Ecosystem) funded by the European Union Next-GenerationEU (Piano Nazionale di Ripresa e Resilienza (PNRR) – Missione 4 Componente 2, Investimento 1.5 – D.D. 1058 23/06/2022, ECS_00000043). This manuscript reflects only the Authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

REFERENCES

[1] G. Giacomuzzo, A. D. Libera, D. Romeres, and R. Carli, "A black-box physics-informed estimator based on gaussian process regression for robot inverse dynamics identification," *arXiv preprint arXiv:2310.06585*, 2023.

[2] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[3] M. P. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *ICML*, 2011.

[4] F. Amadio, A. Dalla Libera, R. Antonello, D. Nikovski, R. Carli, and D. Romeres, "Model-based policy search using monte carlo gradient estimation with real systems application," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3879–3898, 2022.

[5] D. Nguyen-Tuong and J. Peters, "Using model knowledge for learning inverse dynamics," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 2677–2682.

[6] D. Romeres, M. Zorzi, R. Camoriano, and A. Chiuso, "Online semi-parametric learning for inverse dynamics modeling," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 2945–2950.

[7] R. Camoriano, S. Traversaro, L. Rosasco, G. Metta, and F. Nori, "Incremental semiparametric inverse dynamics learning," in *ICRA*, 2016, pp. 544–550.

[8] A. Dalla Libera and R. Carli, "A data-efficient geometrically inspired polynomial kernel for robot inverse dynamic," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 24–31, 2020.

[9] A. Gijsberts and G. Metta, "Incremental learning of robot dynamics using random features," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 951–956.

[10] A. S. Polydoros, L. Nalpantidis, and V. Krüger, "Real-time deep learning of robotic manipulator inverse dynamics," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 3442–3448.

[11] J. Schreiter, P. Englert, D. Nguyen-Tuong, and M. Toussaint, "Sparse gaussian process regression for compliant, real-time robot control," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2586–2591.

[12] E. Rueckert, M. Nakatenus, S. Tosatto, and J. Peters, "Learning inverse dynamics models in o(n) time with lstm networks," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, 2017, pp. 811–816.

[13] M. W. Spong and L. Praly, "Control of underactuated mechanical systems using switching and saturation," in *Control using logic-based switching*. Springer, 2005, pp. 162–172.

[14] S. Rezaei-Shoshtari, D. Meger, and I. Sharf, "Cascaded gaussian processes for data-efficient robot dynamics learning," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6871–6877.

[15] M. Lutter, C. Ritter, and J. Peters, "Deep lagrangian networks: Using physics as model prior for deep learning," *arXiv preprint arXiv:1907.04490*, 2019.

[16] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, 2021.

[17] I. Ayed, E. de Bézenac, A. Pajot, J. Brajard, and P. Gallinari, "Learning dynamical systems from partial observations," *arXiv preprint arXiv:1902.11136*, 2019.

[18] S. Greydanus, M. Dzamba, and J. Yosinski, "Hamiltonian neural networks," *Advances in neural information processing systems*, vol. 32, 2019.

[19] M. Cranmer, S. Greydanus, S. Hoyer, P. Battaglia, D. Spergel, and S. Ho, "Lagrangian neural networks," *arXiv preprint arXiv:2003.04630*, 2020.

[20] M. W. Spong and D. J. Block, "The pendubot: A mechatronic system for control research and education," in *Proceedings of 1995 34th IEEE conference on decision and control*, vol. 1. IEEE, 1995, pp. 555–556.

[21] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[22] M. W. Spong, "Energy based control of a class of underactuated mechanical systems," *IFAC Proceedings Volumes*, vol. 29, no. 1, pp. 2828–2832, 1996.

[23] O. Kolesnichenko and A. S. Shiriaev, "Partial stabilization of underactuated euler–lagrange systems via a class of feedback transformations," *Systems & Control Letters*, vol. 45, no. 2, pp. 121–132, 2002.

[24] K. Åström and K. Furuta, "Swinging up a pendulum by energy control," *Automatica*, vol. 36, no. 2, pp. 287–295, 2000.

[25] X. Xin, S. Tanaka, J. She, and T. Yamasaki, "New analytical results of energy-based swing-up control for the pendubot," *International Journal of Non-Linear Mechanics*, vol. 52, pp. 110–118, 2013.

[26] S. Särkkä, "Linear operators and stochastic partial differential equations in gaussian process regression," in *International Conference on Artificial Neural Networks*. Springer, 2011, pp. 151–158.

[27] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019.