2020 I-RIM Conference
December 10-12, 2020, Rome, Italy (Virtual)
ISBN: 9788894580518
DOI:10.5281/zenodo.4781489

Learning Feedback Linearization Control Without Torque Measurements

Marco Capotondi*, Giulio Turrisi*, Claudio Gaz, Valerio Modugno, Giuseppe Oriolo, Alessandro De Luca
Dipartimento di Informatica Automatica e Gestionale
Sapienza Università di Roma
Email: lastname@diag.uniroma1.it

***Abstract*—Feedback Linearization (FL) allows the best control performance in executing a desired motion task when an accurate dynamic model of a fully actuated robot is available. However, due to residual parametric uncertainties and unmodeled dynamic effects, a complete cancellation of the nonlinear dynamics by feedback is hardly achieved in practice. In this paper, we summarize a novel learning framework aimed at improving online the torque correction necessary for obtaining perfect cancellation with a FL controller, using only joint position measurements. We extend then this framework to the class of underactuated robots controlled by Partial Feedback Linearization (PFL), where we simultaneously learn a feasible trajectory satisfying the boundary conditions on the desired motion while improving the associated tracking performance.**

***Index Terms*—Robot Learning, Feedback Linearization, Gaussian Process Regression, Underactuated Robots, Optimal Control**

I. Introduction

Feedback Linearization (FL) is a widely used control design technique in fully actuated robots that allows to reach a precise execution of the desired motion tasks. Thanks to the perfect cancellation of all nonlinearities in nominal conditions, the trajectory tracking problem can be easily addressed in the resulting linear and input-output decoupled domain, achieving thus arbitrarily fast, exponential convergence of the tracking error. Indeed, a very accurate dynamic model of the robot is strictly required to obtain such a control performance.

Similarly, for the case of underactuated robots, it is possible to design a Partial Feedback Linearization (PFL) law so as to exactly linearize a suitable part of the model, either the actuated dynamics (collocated PFL) or the passive one (non-collocated PFL) [1], [2]. For such systems, one needs to define also a dynamically feasible trajectory for the uncontrolled variables, satisfying boundary conditions and additional constraints of the desired motion task. Optimization techniques can be used to address this problem off line, but their success is based again on the availability of an accurate dynamic model.

In this framework, one can improve performance in execution of arbitrary motion trajectories by working on a better parametric identification of the robot dynamic model [3], by resorting to non-parametric regression techniques [4], or by addressing the inaccuracy issues directly at the control level [5]. In general, these approaches require an initial offline phase which, in case of structural changes of operative conditions (e.g., handling of an unknown payload or a permanent effect due to mechanical wearing), must be repeated from scratch.

In [6], we have introduced an online learning strategy that greatly improves the trajectory tracking accuracy of FL control of fully actuated robots under large model mismatches, using only joint position measurements and avoiding the resort to noisy torque sensors. The aim of this paper is to review the main idea of our work and to present an extension of the method that deals with the problem of underactuation [7].

The paper is organized as follows. In Sec. II, we recap how model mismatches influence the FL control law and the resulting closed-loop dynamics in fully actuated robots, while the proposed learning strategy is summarized in Sec. III. In Sec. IV and V we formulate the problem and, respectively, illustrate the extension of our learning strategy for underactuated robots under collocated PFL control. Simulation results are reported in Sec. VI for a 7R KUKA iiwa robot and for the Pendubot, a 2R robot moving in the vertical plane with passive second joint. Conclusions and ongoing activities are briefly discussed in Sec. VII.

II. Fully Actuated Robots

The dynamics of a n-dof fully actuated robot is given by

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

with $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}} \in \mathbb{R}^n$ being, respectively, the joint position, velocity and acceleration vectors, $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ is the positive definite inertia matrix, and $\boldsymbol{n} \in \mathbb{R}^n$ is the Coriolis, centrifugal, and gravity vector (possibly including also friction terms). For such system, the input torque $\boldsymbol{\tau} \in \mathbb{R}^n$ can be chosen as the nonlinear feedback law $\boldsymbol{\tau}_{FL}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ that cancels all dynamic terms in (1)

$$\boldsymbol{\tau}_{\text{FL}} = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{u} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}), \tag{2}$$

where $\boldsymbol{u} \in \mathbb{R}^n$ is a desired joint acceleration. In principle, with a perfect knowledge of the robot dynamic model, applying the FL law (2) yields a linear system of n decoupled chains of double integrators, i.e.,

$$\ddot{\boldsymbol{q}} = \boldsymbol{u}. \tag{3}$$

To track a desired smooth trajectory $\boldsymbol{q}_d(t)$, the control design is completed by choosing $\boldsymbol{u} = \ddot{\boldsymbol{q}}_d + \boldsymbol{K}_D(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}}) + \boldsymbol{K}_P(\boldsymbol{q}_d - \boldsymbol{q})$, with diagonal PD gains $\boldsymbol{K}_P$, $\boldsymbol{K}_D > 0$.

* These authors have equally contributed.

In practice, because of uncertainties in the nominal model parameters used by the FL law (2), and due to the additional presence unmodeled dynamics, the obtained closed-loop system will not be described by (3). Taking explicitly into account model uncertainties, we can write the actual inertia matrix $\boldsymbol{M}$ and the vector $\boldsymbol{n}$ as

$$\boldsymbol{M}(\boldsymbol{q}) = \hat{\boldsymbol{M}}(\boldsymbol{q}) + \Delta\boldsymbol{M}(\boldsymbol{q}) \tag{4}$$

$$\boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \hat{\boldsymbol{n}}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \Delta\boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{5}$$

where $\hat{\boldsymbol{M}}$ and $\hat{\boldsymbol{n}}$ are the nominal quantities (those that can be used in (2), expressing our *a priori* knowledge of the system), while $\Delta\boldsymbol{M}$ and $\Delta\boldsymbol{n}$ are perturbation terms characterizing the uncertainties. Note that $\Delta\boldsymbol{n}$ can incorporate also unmodeled dynamic terms.

If we apply the FL control law (2) with the nominal values

$$\hat{\boldsymbol{\tau}}_{\text{FL}} = \hat{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{u} + \hat{\boldsymbol{n}}(\boldsymbol{q},\dot{\boldsymbol{q}}), \tag{6}$$

to the real system (1), considering eqs. (4) and (5), we obtain

$$\ddot{\boldsymbol{q}} = \boldsymbol{u} + \left(\boldsymbol{M}^{-1}(\boldsymbol{q})\hat{\boldsymbol{M}}(\boldsymbol{q}) - \boldsymbol{I}\right)u + \boldsymbol{M}^{-1}(\boldsymbol{q})\Delta\boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{7}$$

$$= \boldsymbol{u} + \boldsymbol{\delta}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{u}). \tag{8}$$

Thus, $\boldsymbol{\delta}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{u})$ is a nonlinear function that accounts for all modeling mismatches as well as any unmodeled dynamics.

III. Learning for Fully Actuated robots

The learning framework presented in [6] can be applied to torque-controlled robots and is based on the idea that, by comparing the actual robot motion with the FL prediction, it is possible to reconstruct online the needed dynamic model correction without the use of torque measurements.

In a digital implementation with sampling time T_c, suppose that $\boldsymbol{x}_k = (\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k)^T$ is the current robot state at $t = t_k = kT_c$ and we want to reach a desired state $\boldsymbol{x}_{d,k+1} = (\boldsymbol{q}_{d,k+1}, \dot{\boldsymbol{q}}_{d,k+1})^T$ at $t = t_{k+1}$. Using our preferred linear controller (e.g., a PD with feedforward term), we compute the acceleration command $\boldsymbol{u}_k$ that should do the task if the FL controller was the perfect one. We impose thus the joint torque $\hat{\boldsymbol{\tau}}_{\text{FL}}$ in (6) obtaining the closed-loop dynamics

$$\boldsymbol{M}(\boldsymbol{q}_k)\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}_k,\dot{\boldsymbol{q}}_k) = \hat{\boldsymbol{\tau}}_{\text{FL},k} = \hat{\boldsymbol{M}}(\boldsymbol{q}_k)\boldsymbol{u}_k + \hat{\boldsymbol{n}}(\boldsymbol{q}_k,\dot{\boldsymbol{q}}_k). \tag{9}$$

However, due to the uncertain dynamics, the robot will reach a different state $\boldsymbol{x}_{k+1} = (\boldsymbol{q}_{k+1}, \dot{\boldsymbol{q}}_{k+1})$. At this point, given the performed robot motion, one can reconstruct (by numerical differentiation) the robot acceleration $\ddot{\boldsymbol{q}}_k$ and then estimate the missing torque that would have been needed in the nominal FL-based control law in order to compensate the unmodeled dynamics. Plugging $\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}_k$ in (9), we obtain after some manipulation

$$\hat{\boldsymbol{M}}(\boldsymbol{q}_k)(\boldsymbol{u}_k - \ddot{\boldsymbol{q}}_k) = \Delta\boldsymbol{M}(\boldsymbol{q}_k)\ddot{\boldsymbol{q}}_k + \Delta\boldsymbol{n}(\boldsymbol{q}_k,\dot{\boldsymbol{q}}_k). \tag{10}$$

Using the *a posteriori* computable left-hand side of eq. (10), a regressor function $\boldsymbol{\varepsilon}(\cdot)$ can be introduced in the FL control law, through which it is possible to progressively compensate the unknown perturbations on the right-hand side. In order to exploit the data acquired so far, a moving dataset $\mathcal{D} = \{(\mathbf{X}_i, \mathbf{Y}_i)\,|i = 1,\dots,n_d\}$ can be constructed, where n_d is the number of its elements, with input $\mathbf{X}_i = (\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_i)$, and output $\mathbf{Y}_i = \hat{\boldsymbol{M}}(\boldsymbol{u}_i - \ddot{\boldsymbol{q}}_i) + \boldsymbol{\varepsilon}_i$. In $\mathbf{Y}_i$, the torque correction $\boldsymbol{\varepsilon}_i$ applied to the robot is also taken into account, in order to preserve dataset consistency. Moreover, when we predict the compensating torque $\boldsymbol{\varepsilon}_k$ at time t_k, the regressor input will be composed by the current robot state $\boldsymbol{x}_k$ and by the last commanded acceleration $\boldsymbol{u}_k$ generated by the chosen closed-loop controller.

As a result, the new FL-based control input $\boldsymbol{\tau}_{\text{FL},k}$ will be the sum of the nominal FL torque and of the online prediction by the regressor $\boldsymbol{\varepsilon}(\cdot)$, i.e.,

$$\boldsymbol{\tau}_{\text{FL},k} = \hat{\boldsymbol{\tau}}_{\text{FL},k} + \boldsymbol{\varepsilon}_k = \hat{\boldsymbol{M}}(\boldsymbol{q}_k)\boldsymbol{u}_k + \hat{\boldsymbol{n}}(\boldsymbol{q}_k,\dot{\boldsymbol{q}}_k) + \boldsymbol{\varepsilon}_k.$$

IV. Underactuated Robots

Consider a n-dof robot with generalized coordinates $\boldsymbol{q} \in \mathbb{R}^n$ and $m < n$ actuators. The robot dynamics can be partitioned as (see, e.g., [8])

$$\boldsymbol{M}_{aa}(\boldsymbol{q})\ddot{\boldsymbol{q}}_a + \boldsymbol{M}_{au}(\boldsymbol{q})\ddot{\boldsymbol{q}}_u + \boldsymbol{n}_a(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{\tau}$$
$$\boldsymbol{M}_{ua}(\boldsymbol{q})\ddot{\boldsymbol{q}}_a + \boldsymbol{M}_{uu}(\boldsymbol{q})\ddot{\boldsymbol{q}}_u + \boldsymbol{n}_u(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{0},$$

where $\boldsymbol{q} = (\boldsymbol{q}_a, \boldsymbol{q}_u)$, $\boldsymbol{q}_a$ are the m dofs actuated by $\boldsymbol{\tau} \in \mathbb{R}^m$, and $\boldsymbol{q}_u$ are the $n - m$ passive joints. The inertia matrix $\boldsymbol{M}$ and the nonlinear term $\boldsymbol{n}$ are partitioned accordingly.

Given an accurate knowledge of the robot model, it is possible to use nonlinear feedback to exactly linearize part of the system dynamics, namely the actuated one. Such a collocated PFL controller is always well defined and takes the form [1]

$$\boldsymbol{\tau}_{\text{PFL}} = \left(\boldsymbol{M}_{aa} - \boldsymbol{M}_{au}\boldsymbol{M}_{uu}^{-1}\boldsymbol{M}_{ua}\right)\boldsymbol{u} + \boldsymbol{n}_a - \boldsymbol{M}_{au}\boldsymbol{M}_{uu}^{-1}\boldsymbol{n}_u, \tag{11}$$

where $\boldsymbol{u} \in \mathbb{R}^m$ is the desired acceleration of the actuated joints. The control law (11) results in m decoupled chains of double integrators for the linear part, leaving unchanged (and still nonlinear) the passive one:

$$\ddot{\boldsymbol{q}}_a = \boldsymbol{u}$$
$$\ddot{\boldsymbol{q}}_u = -\boldsymbol{M}_{uu}^{-1}(\boldsymbol{n}_u + \boldsymbol{M}_{ua}\boldsymbol{u}).$$

Taking now into account model uncertainties, an imperfect cancellation of dynamic terms will result when applying the PFL law (11) based on the nominal system dynamics. The perturbed closed-loop system becomes

$$\ddot{\boldsymbol{q}}_a = \boldsymbol{u} + \boldsymbol{\delta}_a(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{u})$$
$$\ddot{\boldsymbol{q}}_u = -\hat{\boldsymbol{M}}_{uu}^{-1}(\hat{\boldsymbol{n}}_u + \hat{\boldsymbol{M}}_{ua}\boldsymbol{u}) + \boldsymbol{\delta}_u(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{u}),$$

where $\boldsymbol{\delta}_a$ and $\boldsymbol{\delta}_u$ represent the effect of modeling mismatches, respectively on the actuated and on the passive subsystems. In the latter, we have explicitly isolated the known nominal terms.

Indeed, we can correct $\boldsymbol{\delta}_a$ similarly to the fully actuated case in Sec. III, using a regressor $\boldsymbol{\varepsilon}_a$. However, because of the presence of the perturbation $\boldsymbol{\delta}_u$ on the passive dynamics,

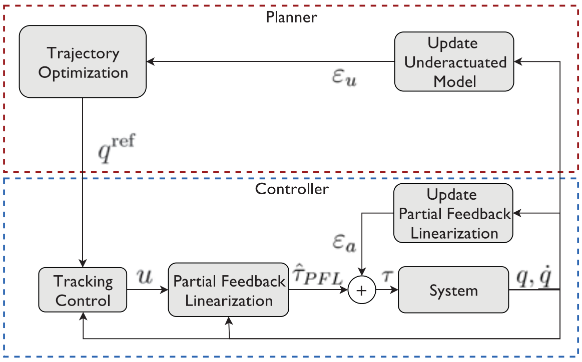

Fig. 1. Block diagram of the control framework for underactuated robots.

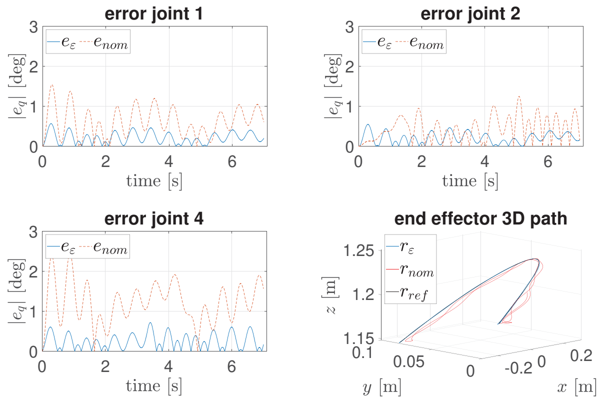

Fig. 2. Comparison between the two different control modalities: using the nominal dynamic model (red line) and using our learning method (blue line). From top-left to down-right: the absolute values of joint position error of the first, second, fourth joint and the 3D path realized by the end effector.

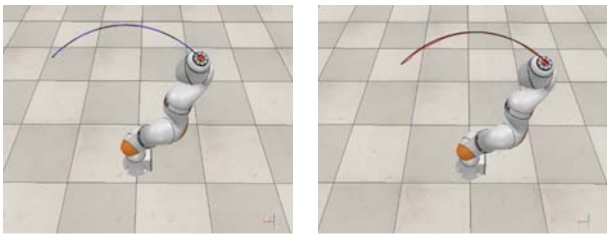

Fig. 3. A KUKA LBR iiwa realizing a trajectory tracking task with the nominal controller (right, red) and with the proposed controller (left, blue).

the accurate tracking of a desired trajectory for the actuated dofs will not be sufficient for complete realization of a desired motion task. A dynamically consistent trajectory should be planned also for the passive joints, typically solving a constrained optimization problem. The presence of uncertainty in the model will prevent producing such a feasible solution.

V. Learning for Underactuated Robots

To compensate for the model mismatch $\boldsymbol{\delta}_u$, we developed an iterative learning procedure [7], similarly to what is done in [9], [10]. However, in these latter works the reference trajectory is never adjourned using an updated model; thus, when working with inaccurate nominal dynamics, the computed trajectory will typically be unfeasible even in the ideal case of a correct partial feedback linearization law.

Instead, the perturbation $\boldsymbol{\delta}_u$ on the passive dynamics can be approximated using a second regressor $\boldsymbol{\varepsilon}_\mathbf{u}$, comparing the acceleration of the passive part of the system with the acceleration predicted by the nominal model. Given an approximation of the actual acceleration $\ddot{\boldsymbol{q}}_{a,k}$, one can compute the acceleration associated with the nominal model of the passive joints as

$$\ddot{\boldsymbol{q}}_{u,k}^{\text{pred}} = -\hat{\boldsymbol{M}}_{uu,k}^{-1}(\hat{\boldsymbol{n}}_{u,k} + \hat{\boldsymbol{M}}_{ua,k}\ddot{\boldsymbol{q}}_{a,k}).$$

A new datapoint can be generated at time t_k by comparing the acceleration estimated numerically with the one predicted by the nominal model, i.e.,

$$\mathbf{X}_{u,k} = (\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k, \ddot{\boldsymbol{q}}_{a,k}); \quad \mathbf{Y}_{u,k} = \ddot{\boldsymbol{q}}_{u,k} - \ddot{\boldsymbol{q}}_{u,k}^{\text{pred}}.$$

The dataset is incrementally built during each trial, and the regressor $\boldsymbol{\varepsilon}_\mathbf{u}$ is employed during each trajectory planning phase so as to work a successively more accurate estimation of the real system dynamics.

A block diagram of the complete learning control framework for the underactuated case is shown in Fig. 1. Initially, for a desired motion task, a reference trajectory is generated for the system using the best available nominal model. Then, the first trial is executed, during which the PFL-based law is corrected online to compensate for the presence of $\boldsymbol{\delta}_a$. At the end, a new reference trajectory is computed taking into account the approximation of the residual term $\boldsymbol{\delta}_u$, and the motion is performed again. These steps are iterated until model errors converge to zero and the task is correctly executed.

VI. Numerical Results

In order to validate the presented framework, simulations have been performed for a fully actuated ans and underactuated robotic platform.

A. Fully actuated case

Our learning method has been tested first on a simulated 7R KUKA LBR iiwa performing a trajectory tracking task. In particular, the first six joints should follow a sinusoidal path, while the last one should remain at rest. We have analyzed the tracking performance of this robot, whose dynamic parameters are reported in [3], [11], for a deviation of about 20% from their nominal values. We compared the results with and without the use of the proposed method. The FL torque correction $\boldsymbol{\varepsilon}$ is approximated using Gaussian Process regression [12]. The resulting joint errors and the end-effector Cartesian path with and without the use of our online learning method are reported in Fig. 2. Snapshots of the obtained robot motion are shown Fig. 3: our method sensibly reduces the tracking error.

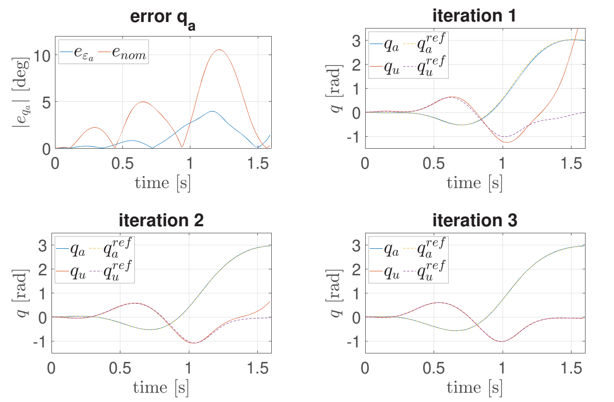

Fig. 4. Pendubot reference joint trajectories (dashed lines) and actual positions (solid lines) during each iteration. At the top left, it is reported the position error of the actuated joint at Iteration 1: in blue, when the correction is applied, while in red when only the PD correction is employed.

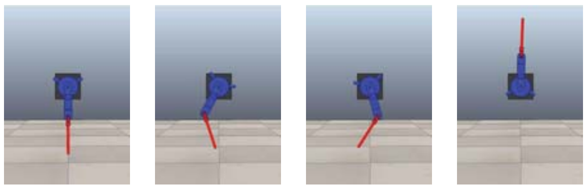

Fig. 5. The Pendubot performing the swing-up task. After three iterations, the robot is capable of reaching the up-up configuration.

B. Underactuated case

The proposed control method has been tested by simulation on the Pendubot, a 2R robot with a passive second joint and moving in the vertical plane. The Pendubot should perform a rest-to-rest swing-up manouver, from the initial downward equilibrium configuration $\boldsymbol{x}_0 = (0,0,0,0)^T$ to the up-up configuration $\boldsymbol{x}_{\mathrm{g}} = (q_a, q_u, \dot{q}_a, \dot{q}_u) = (\pi, 0, 0, 0)^T$. We considered a mismatch between nominal and actual model parameters of about 30%. As for the fully actuated case, any closed-loop controller could be used in order to generate a reference acceleration for the PFL. The planning procedure is formulated as an optimal control problem and solved using a Sequential Quadratic Programming optimizer [13], including the correction for the residual term $\boldsymbol{\varepsilon}_{\mathbf{u}}$ in the dynamic constraint. Here, $\boldsymbol{\varepsilon}_{\mathbf{u}}$ is reconstructed through a Gaussian Process regression. Three iterations for the actual and the desired joint trajectories are shown in Fig. 4, together with a comparison of the actuated joint position error during the first iteration when only the PD is applied (top left, red curve) and when ε_a is employed (blue curve). It is worth noticing that the tracking performance improves even for the actuated joint at the first iteration. Moreover, our method perfectly recovers the tracking error with an handful of trials. Few frames of the final successful Pendubot motion obtained wirh our learning method are shown in Fig. 5.

VII. Conclusions

We have presented a method for online learning of feedback linearization control without torque measurements for fully actuated robots, and its extension to underactuated platforms. In both cases, the correction of the linearizing part of the torque input converges quite fast to the correct value. This feature depends mostly on the locality properties of the used GP regressors, in particular for repetitive tasks in which the input space is already partially explored.

For fully actuated robots, the method displays large performance improvements which allow to eventually execute very high precision motion tasks. The presented extension to underactuated robots is able to solve a complex swing-up task in just an handful of trials, even with an extremely bad nominal model.

Despite of the obtained effective results, a limit of the presented approach concerns the generalization problem. In particular, it is not straightforward to adapt the actual model knowledge to different tasks without restarting the learning procedure from scratch. In order to overcome this problem, we are currently focusing on the design of novel data-collection strategies, which consider different trajectories and leverage a heuristics, such as information maximization, while exploiting robot redundancy for exploration.

References

[1] M. Spong, "Partial feedback linearization of underactuated mechanical systems," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1994, pp. 314–321.

[2] ——, "Underactuated mechanical systems," in *Control Problems in Robotics and Automation*, B. Siciliano and K. Valavanis, Eds. Springer, 1998, pp. 135–150.

[3] 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. on Robotics and Automation*, 2014, pp. 1386–1392.

[4] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Computed torque control with nonparametric regression models," in *Proc. American Control Conf.*, 2008, pp. 212–217.

[5] T. Westenbroek, D. Fridovich-Keil, E. Mazumdar, S. Arora, V. Prabhu, S. S. Sastry, and C. J. Tomlin, "Feedback linearization for unknown systems via reinforcement learning," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2020, pp. 1364–1371.

[6] M. Capotondi, G. Turrisi, C. Gaz, V. Modugno, G. Oriolo, and A. De Luca, "An online learning procedure for feedback linearization control without torque measurements," in *Proc. of Machine Learning Research (3rd Conf. on Robot Learning)*, vol. 100, 2020, pp. 1359–1368.

[7] G. Turrisi, M. Capotondi, C. Gaz, V. Modugno, G. Oriolo, and A. De Luca, "Iterative learning control for underactuated robots," *IEEE Robotics and Automation Lett.*, (submitted in November 2020).

[8] A. De Luca, S. Iannitti, R. Mattone, and G. Oriolo, "Underactuated manipulators: Control properties and techniques," *Machine Intelligence and Robotic Control*, vol. 4, no. 3, pp. 113–125, 2002.

[9] A. Schoellig and R. D'Andrea, "Optimization-based iterative learning control for trajectory tracking," in *Proc. 10th European Control Conf.*, 2009, pp. 1505–1510.

[10] F. Mueller, A. Schoellig, and R. D'Andrea, "Iterative learning of feed-forward corrections for high-performance tracking," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2012, pp. 3276–3281.

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

[12] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[13] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.