An Online Learning Procedure for Feedback Linearization Control without Torque Measurements

M. Capotondi, G. Turrisi, C. Gaz, V. Modugno, G. Oriolo, A. De Luca
Dipartimento di Ingegneria Informatica, Automatica e Gestionale
Sapienza Università di Roma,
Via Ariosto 25, 00185 Roma, Italy
surname@diag.uniroma1.it

Abstract:
By exploiting an a-priori estimate of the dynamic model of a manipulator, it is possible to command joint torques which ideally realize a Feedback Linearization (FL) controller. The exact cancellation may nevertheless not be achieved due to model uncertainties and possible errors in the estimation of the dynamic coefficients. In this work, an online learning scheme for control based on FL is presented. By reading joint positions and joint velocities information only (without the use of any torque measurement), we are able to learn those model uncertainties and thus achieve perfect FL control. Simulations results on the popular KUKA LWR iiwa robot are reported to show the quality of the proposed approach.

Keywords: Robot Learning, Model Estimation, Model Predictive Control, Gaussian Process Regression

1 Introduction

The knowledge of accurate dynamic models is of paramount importance for several robotic applications. It is necessary, in fact, for designing control laws with superior performances [1], during robot interactions with the environment (for example when implementing 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] is required. In order to retrieve an estimation of the dynamic model, regression techniques are widely employed [4, 5]. These techniques are hinged on a well-known property: the linear dependence of the robot dynamic equations in terms of a set of ρ *dynamic coefficients* [6], (also denoted in the literature as *base parameters*) [7], which are linear combinations of the *dynamic parameters* of the links composing the robot (masses, centers of gravity and inertia tensors).

In order to retrieve a reliable estimation of the dynamic model, a series of exciting trajectories are typically commanded to the robot, and the joint positions and torques are recorded during motion. It is eventually possible to obtain a numerical estimation of the dynamic coefficients by exploiting the filtered joint torques and the filtered joint positions, velocities and accelerations obtained by numerical derivation [7]. Typically, this whole procedure, whose output is the estimation of the dynamic model of the robot under study, is performed offline. Therefore, in case of changes in the structural parameters of the robot, the identification procedure has to be carried out again from scratch. This is the case, for instance, when unknown payloads are attached to the end-effector of a manipulator: in this regard, a method to update the dynamic model has been presented recently [8], but it requires an initial phase of setup for updating the dynamic model parameters. Especially when collision detection and reaction strategies are adopted during motion, it is essential to have an adaptable and reliable estimation of the dynamic model, employed to properly implement those algorithms [2].

In the last years, to overcome the limitations imposed by the aforementioned approaches, a new set of techniques has emerged relying on the employment of regression techniques to face the problem

3rd Conference on Robot Learning (CoRL 2019), Osaka, Japan.

of robotics model learning. In literature there exist two different way to tackle this issue [9]: by learning the direct model [10], [11], [12], [13] or by reconstructing an inverse representation of the robot's dynamic [14],[15],[16]. In the first case, the aforementioned methods try to understand how the system, given its actual state, responds to a certain input, while the second approaches focus on estimating the input that needs to be given to the system in order to achieve a certain desired new state. In regards of the first class of methods in [10], the authors propose to learn the transition probability model, while in [12] the authors try to reconstruct the system nonlinear dynamics with a Gaussian Process (GP) in order to improve approximate linearization of the system around an operating point. In [13] the authors utilize a regressor as predictive model for a nonlinear Model Predictive Control (MPC). In our approach we make use of a linear MPC in the control scheme, but we design a procedure for learning the inverse dynamics of the system. In the context of inverse model learning, in [17] the authors describe a way to learn a computed torque controller, employing the torque measurements to build their training dataset. In our work we learn the unmodeled dynamics to improve the feedback linearization process without the use of any joint torque measurements, which are known to be noisy.

In the present work, we propose a method to reconstruct dynamic model uncertainties and parameters variations by means of an online algorithm based on Gaussian Process (GP) Regression. Having an *a priori* estimation of the dynamic model of a robot, we show that it is possible to improve the model by exploiting only joint position measures, without the need of any joint torque data. Indeed, torque measurements are usually affected by a high level of noise, typically higher than the noise added to the measures coming from the encoders which return joint positions [1]: thus, the employed algorithm is able to obtain a reliable estimate of dynamic uncertainties. This paper is organized as follows: in Sec. 2 the problem of not exact Feedback Linearization (FL) is stated; in Sec. 3 the learning procedure is presented along with the needed formalism; in Sec. 4 the complete control architecture is discussed and in Sec. 5 experimental results are shown; Finally, Sec. 6 concludes the paper.

2 Problem Formulation

For a n-DoFs robot, its dynamics can be described by the following equation:

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

in which $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \mathbb{R}^n$ are, respectively, the joint positions, velocities and accelerations; $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ is the inertia matrix and $\boldsymbol{\eta} \in \mathbb{R}^n$ is a vector obtained by the sum of the Coriolis and centrifugal forces and the gravity vector. Supposing that the robot is fully actuated, it is possible to design a FL controller [1]. This method provides a control input $\boldsymbol{\tau}_{\text{FL}}$ that cancels the nonlinear dynamics components of the model in eq. (1), as:

$$\boldsymbol{\tau}_{\text{FL}} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}_d + \boldsymbol{\eta}(\boldsymbol{q}, \dot{\boldsymbol{q}}), \tag{2}$$

where $\ddot{\boldsymbol{q}}_d \in \mathbb{R}^n$ represents the desired joint accelerations vector. In principle, given a perfect knowledge of the system dynamic model and applying the FL controller we would get:

$$\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}_d, \tag{3}$$

but in reality it is typically not true due to unmodeled dynamics and uncertainties in the system parameters. If we explicitly account for these uncertainties in the model, we have therefore:

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

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

where $\hat{\boldsymbol{M}}$ and $\hat{\boldsymbol{\eta}}$ are nominal quantities (for instance, previously estimated), $\Delta\boldsymbol{M}$ and $\Delta\boldsymbol{\eta}$ are increments characterizing the uncertainties, and $\boldsymbol{\tau}_f \in \mathbb{R}^n$ represents the joint friction torques. Thus, $\hat{\boldsymbol{M}}$ and $\hat{\boldsymbol{\eta}}$ incorporate our *a priori* knowledge of the system. If we apply the nominal FL control on the real system of eq. (1), that is

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

considering eqs. (4) and (5), we obtain:

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}(\boldsymbol{q})^{-1}\hat{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}}_d - \boldsymbol{M}(\boldsymbol{q})^{-1}(\Delta\boldsymbol{\eta}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\tau}_f(\dot{\boldsymbol{q}})) = \ddot{\boldsymbol{q}}_d + \boldsymbol{\epsilon}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}_d) \tag{7}$$

where $\boldsymbol{\epsilon}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}_d)$ accounts for all the unmodeled dynamic terms. One of the objectives of the present work is to retrieve an estimate of $\boldsymbol{\epsilon}$.

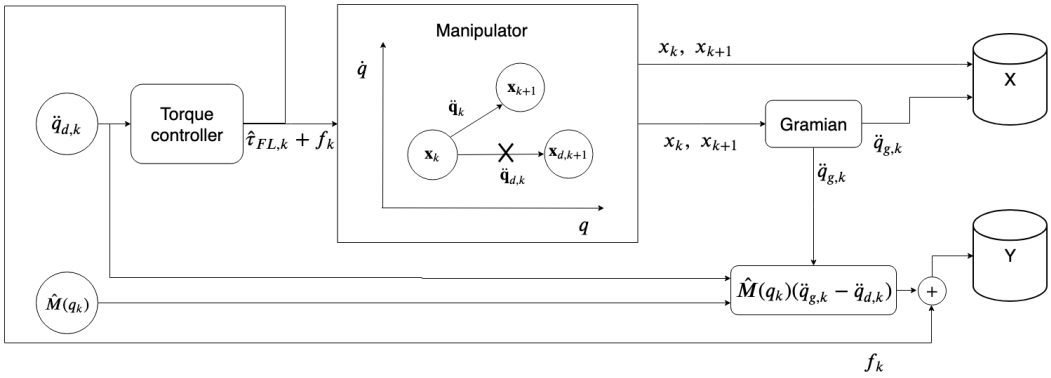

Figure 1: The scheme in the image describes the procedure used for the construction of the dataset for the online learning scheme.

3 Learning Algorithm

In this section we describe the entire pipeline for the dataset construction, as depicted in Fig. 1.

3.1 Dataset collection procedure

At first, we suppose that it is possible to drive the manipulator by commanding joint torques. Referring to Fig. 1, the current robot state is $\boldsymbol{x}_k = (\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k)^T$ and we want to reach a desired target state $\boldsymbol{x}_{d,k+1} = (\boldsymbol{q}_{d,k+1}, \dot{\boldsymbol{q}}_{d,k+1})^T$. We compute the input acceleration $\ddot{\boldsymbol{q}}_{d,k}$ that should drive the robot to the desired state supposing a perfect FL controller, i.e. by imposing the joint torque $\hat{\boldsymbol{\tau}}_{\text{FL}}$ from eq. (6):

$$\boldsymbol{M}(\boldsymbol{q}_k)\ddot{\boldsymbol{q}}_k + \boldsymbol{\eta}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k) = \hat{\boldsymbol{\tau}}_{\text{FL},k} = \hat{\boldsymbol{M}}(\boldsymbol{q}_k)\ddot{\boldsymbol{q}}_{d,k} + \hat{\boldsymbol{\eta}}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k). \tag{8}$$

Due to the effect of unmodeled dynamics, the robot reaches a different state $\boldsymbol{x}_{k+1} = (\boldsymbol{q}_{k+1}, \dot{\boldsymbol{q}}_{k+1})$. At this point, from eq. (8), it is possible to compute the unmodeled dynamics that depends only on the system state at time $\boldsymbol{x}_k$, the desired and actual system accelerations $\ddot{\boldsymbol{q}}_{d,k}, \ddot{\boldsymbol{q}}_k$:

$$\hat{\boldsymbol{M}}(\boldsymbol{q}_k)(\ddot{\boldsymbol{q}}_{d,k} - \ddot{\boldsymbol{q}}_k) = \Delta\boldsymbol{M}(\boldsymbol{q}_k)\ddot{\boldsymbol{q}}_k + \Delta\boldsymbol{\eta}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k). \tag{9}$$

In eq. (9) the term $\ddot{\boldsymbol{q}}_k$ cannot be known in advance and has to be computed *a posteriori* using the information about the states $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k+1}$. To this aim, we can utilize the notion of Controllability Gramian by calculating the true joints acceleration $\ddot{\boldsymbol{q}}_{g,k}$ that would have brought the perfectly feedback linearized system in the state $\boldsymbol{x}_{k+1}$ in the first place. In section 3.2 we will show that under certain conditions $\ddot{\boldsymbol{q}}_{g,k} = \ddot{\boldsymbol{q}}_k$.

Our learning framework is based on a torque controller to command the robot while collecting the data for estimating the model. By introducing a function regressor $\mathbf{f}(\cdot)$ (see Fig. 1) in the FL control law, we show that with our method it's possible to progressively (and thus online) improve the control performances exploiting all the data about the unknown dynamics acquired while commanding the robot. Therefore, we introduce the new FL control input $\boldsymbol{\tau}_{\text{FL},k}$ defined as the summation of the nominal FL torque and the regressor prediction $\mathbf{f}(\cdot)$, as:

$$\boldsymbol{\tau}_{\text{FL},k} = \hat{\boldsymbol{\tau}}_{\text{FL},k} + \mathbf{f}_k = \hat{\boldsymbol{M}}(\boldsymbol{q}_k)\ddot{\boldsymbol{q}}_{d,k} + \hat{\boldsymbol{\eta}}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k) + \mathbf{f}_k. \tag{10}$$

Considering the new FL torque $\boldsymbol{\tau}_{\text{FL},k}$, we can rewrite eq. (9) as follows:

$$\hat{\boldsymbol{M}}(\boldsymbol{q}_k)(\ddot{\boldsymbol{q}}_{d,k} - \ddot{\boldsymbol{q}}_{g,k}) + \mathbf{f}_k = \Delta\boldsymbol{M}(\boldsymbol{q}_k)\ddot{\boldsymbol{q}}_{g,k} + \Delta\boldsymbol{\eta}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k). \tag{11}$$

where $\ddot{\boldsymbol{q}}_{g,k}$ is used in place of $\ddot{\boldsymbol{q}}_k$ to ensure causality. If we repeat this procedure for several states, we can incrementally construct a dataset $\mathcal{D} = \{(\mathbf{X}_i, \mathbf{Y}_i) \,|i = 1, \ldots, n_d\}$, where n_d is the number of elements in our dataset. For each sample i, we have:

$$\mathbf{X}_i = (\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_{g,i}); \quad \mathbf{Y}_i = \hat{\boldsymbol{M}}(\ddot{\boldsymbol{q}}_{d,i} - \ddot{\boldsymbol{q}}_{g,i}) + \mathbf{f}_i \tag{12}$$

It is clear that this dataset fulfills the properties of uniqueness and causality, which are fundamental for learning inverse model for control [9]. At this stage it's important to point out that only the knowledge of state $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k+1}$ and of the desired acceleration $\ddot{\boldsymbol{q}}_{d,i}$ are needed while no torque information is necessary to build the dataset $\mathcal{D}$.

When, a time k, we want to predict the compensation torques that are required for cancelling the unmodeled dynamics, the regressor input will be:

$$\mathbf{f}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k, \ddot{\boldsymbol{q}}_{d,k}) \tag{13}$$

where $\ddot{\boldsymbol{q}}_{d,k}$ is the desired joint acceleration that we would be able to reach under the assumption of perfect FL. Unfortunately, due to the prediction error of our regressor, small errors will be always committed that will be eventually reduced over time by increasing the dataset size.

3.2 Controllability Gramian

The control algorithm presented in this manuscript requires the estimation of the real joint accelerations $\ddot{\boldsymbol{q}}_k$, in order to reconstruct the unknown part of the dynamic model of the robot.

For this purpose we employ the concept of the Controllability Gramian on the perfect feedback linearized system [18]. Given a generic continuous linear system

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) \end{cases} \tag{14}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the state vector, $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is the state matrix, $\boldsymbol{B} \in \mathbb{R}^{n\times p}$ is the input matrix, $\boldsymbol{u} \in \mathbb{R}^p$ is the input vector, $\boldsymbol{y} \in \mathbb{R}^q$ is the output vector and $\boldsymbol{C} \in \mathbb{R}^{q\times n}$ is the output matrix.

If we have only discrete observations (measurements) $\boldsymbol{y}_k$, it is necessary to modify the system (14) into the form (15):

$$\begin{cases} \boldsymbol{x}_{k+1} = \boldsymbol{\Phi}\boldsymbol{x}_k + \boldsymbol{\Gamma}\boldsymbol{u}_k \\ \boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k \end{cases}, \quad \text{with} \quad \begin{cases} \boldsymbol{\Phi}(T_s) = e^{\boldsymbol{A}T_s} \\ \boldsymbol{\Gamma}(T_s) = \int_0^{T_s} e^{\boldsymbol{A}s} ds \boldsymbol{B}, \end{cases} \tag{15}$$

where T_s is the sampling time, $\boldsymbol{y}_k$ is the k-th measure, $\boldsymbol{x}_{k+1}$ is the $(k+1)$-th state, and the k-th input $\boldsymbol{u}_k$ is kept constant in the time interval $t \in [t_k, t_{k+1}]$ by means of a Zero-Order Hold (ZOH).

In this work, the state consists of joint positions and joint velocities: in particular, for a n-DoFs robot, the state is $\boldsymbol{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}})^T \in \mathbb{R}^{2n}$. Under the hypothesis of a perfectly feedback linearized system, it is possible to represent it as chain of integrators, separating the whole system (14) into n independent subsystems of dimension 2. Therefore after FL, a single joint j, will have the following linear state-space representation:

$$\begin{cases} \dot{\boldsymbol{x}}^j(t) = \begin{bmatrix} \dot{\boldsymbol{q}}^j(t) \\ \ddot{\boldsymbol{q}}^j(t) \end{bmatrix} = \boldsymbol{A}\boldsymbol{x}^j(t) + \boldsymbol{B}u^j(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}^j(t) \\ \dot{\boldsymbol{q}}^j(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \ddot{q}_d(t) \\ y^j(t) = \boldsymbol{C}\boldsymbol{x}^j(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} q^j \\ \dot{q}^j \end{bmatrix}, \end{cases} \tag{16}$$

where $\boldsymbol{x}^j \in \mathbb{R}^2$ is the associate state for joint j. Thus, its discrete formulation (15) becomes:

$$\begin{aligned} \boldsymbol{x}^j_{(k+1)\cdot T_s} &= \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \boldsymbol{x}^j_{k\cdot T_s} + \begin{bmatrix} 0.5\, T_s^2 \\ T_s \end{bmatrix} \boldsymbol{u}^j_{k\cdot T_s} \\ y^j_{k\cdot T_s} &= [1 \quad 0]\, \boldsymbol{x}^j_{k\cdot Ts}. \end{aligned} \tag{17}$$

If the sampling time T_s is sufficiently small, the discrete system (17) approximates well its continuous counterpart (16). In simulation, using realistic robot dynamic parameters, exploiting the Controllability Gramian we obtained very good estimates of the actual joint accelerations, with an average error of 10^{-11} rad/s^2. The main advantage of using the Controllability Gramian is that we do not need any numerical differentiation to retrieve the joint accelerations.

The system (17) is controllable since:

$$\text{rank}(\boldsymbol{B}\ \boldsymbol{A}\boldsymbol{B}) = \text{rank}\left(\begin{bmatrix} 0.5Ts^2 & 1.5Ts^2 \\ Ts & Ts \end{bmatrix} \right) = 2, \tag{18}$$

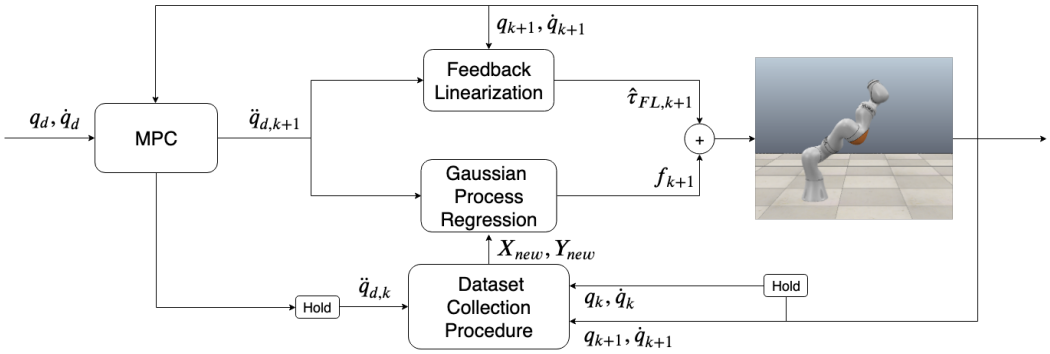

Figure 2: Control and Dataset Acquisition scheme. Note that to preserve the formalism introduced in the previous section, we consider as the actual state the one at time $k+1$. The *Hold* block is a Zero-Order Hold (ZOH) that keeps constant the input value for one sample interval. Therefore, it is possible to hold the measured state value at time t_k in order to make it available at t_{k+1}.

therefore it is possible to define the discrete Controllability Gramian as

$$\boldsymbol{W}(k-1) = \sum_{m=0}^{k-1} \boldsymbol{A}^m \boldsymbol{B}\boldsymbol{B}^T (\boldsymbol{A}^T)^m. \tag{19}$$

At this stage, it is possible to retrieve the input that drives the system from an initial state $\boldsymbol{x}_{\text{init}}$ to a final state $\boldsymbol{x}_{\text{goal}}$ in m steps, as:

$$u_k = -\boldsymbol{B}^T (\boldsymbol{A}^T)^{m-k} \boldsymbol{W}^{-1}(m)[(\boldsymbol{A}^T)^m \boldsymbol{x}_{\text{init}} - \boldsymbol{x}_{\text{goal}}] \qquad k = 0, \dots, m-1. \tag{20}$$

Since the systems consists in a chain of two integrators, only to two steps are required:

$$u = u_0 + u_1 = (\boldsymbol{B}^T \boldsymbol{A}^T + \boldsymbol{B}^T) \boldsymbol{W}^{-1}(1)[\boldsymbol{A}^T \boldsymbol{x}_{\text{init}} - \boldsymbol{x}_{\text{goal}}]. \tag{21}$$

Finally, to summarize the whole process: in case of perfect modeling, the FL would be exact and the system would act like a double integrator. Since the nominal model presents a mismatch with respect to the real one, the controlled system will behave in a different way, performing a diverse motion. This unexpected motion can be interpreted as if the feedback linearization was correct while the robot was driven by another unknown acceleration reference. Following this interpretation, it is always possible to use the Controllability Gramian on a double integrator in order to estimate this new reference acceleration, that will be used for the construction of the dataset, as explained in section 3. For this reason, the acceleration estimated by the Controllability Gramian is correct and it is independent from the current knowledge about the system's complete model.

4 Control Architecture

In this section we describe in detail the control algorithm (see Fig. 2). Our method allows for robot trajectory tracking even if we have only partial knowledge of its dynamic model. In order to achieve perfect FL, we estimate the unknown part of the model by means of a Gaussian Process (GP) Regression method, which is more effective for online estimation purposes [19]. GP balance poor generalization with the possibility to implement local algorithms [20] obtaining good results, comparable to the global case. In order to avoid unfeasible control actions or unfeasible reached states (i.e., out of known mechanical ranges), we generate the desired joint accelerations through a Model Predictive Control (MPC) algorithm, which is able to provide smooth joint acceleration signals while satisfying constraints.

4.1 Gaussian Process regression

Given a set of noisy observation $\mathcal{D} = \left\{ \left(\mathbf{X}^i, Y^i = f\left(\mathbf{X}^i\right) + \varepsilon^i \right) | 1 \leq i \leq n_d \right\}$ with $\varepsilon \sim \mathcal{N}(0,\, \Sigma_\varepsilon)$ the prior on the values of $\mathbf{f}$ is $Y \sim \mathcal{N}(0,\, K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I)$ with $K(\mathbf{X}, \mathbf{X})$ the covariance matrix.

Given a query point $\mathbf{X}_*$, the conditional probability of $\mathbf{f}(\mathbf{X}_*)$ is

$$\mathbf{f}\left(\mathbf{X}_*\right)|\mathcal{D} \sim \mathcal{N}\left(\boldsymbol{k}\left(\mathbf{X}_*, \mathbf{X}\right)\boldsymbol{\beta}, k\left(\mathbf{X}_*, \mathbf{X}_*\right) - \boldsymbol{k}\left(\mathbf{X}_*, \mathbf{X}\right)\left(K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I\right)^{-1}\boldsymbol{k}\left(\mathbf{X}, \mathbf{X}_*\right)\right) \quad (22)$$

Since no assumption has been made on the structure of the unmodeled dynamics, a squared exponential kernel is employed to define the covariance matrix.

In this work, $\mathbf{f}$ reconstruct the unmodeled dynamics of the system. Since $\mathbf{f}$ outputs is multivariate, we set conditionally independent GPs for each joint of the manipulator.

4.2 Model Predictive Control

MPC [21] is a closed loop model based control scheme which is formulated as an optimal control problem. The optimal solution is represented by the sequence of control inputs for the N_p steps ahead that satisfy the constraints while optimizing the corresponding cost function. Once a solution is computed only the first optimal input is then applied to the system. Under the assumption of perfect FL, the prediction model of the robot can be represented as set of n independent systems (one for each joint) as described in eq. (16). Each joint receives as input $u^j = \ddot{q}_d^j$ with $j = 1, \ldots, n$. In this work we define the cost function as the sum of a tracking cost J_t and a smoothness cost J_s:

$$J = \alpha J_t + \gamma J_s \quad (23)$$

$$J_t = \sum_{i=1}^{N_p} (\boldsymbol{q}_{d,i} - \boldsymbol{q}_i)^T (\boldsymbol{q}_{d,i} - \boldsymbol{q}_i) \quad (24)$$

$$J_s = \sum_{i=0}^{N_p} (\boldsymbol{u}_{i+1} - \boldsymbol{u}_i)^T (\boldsymbol{u}_{i+1} - \boldsymbol{u}_i) \quad (25)$$

with N_p the number of step for the prediction horizon, $\boldsymbol{q}_{d,i}$ the desired joint positions, $\boldsymbol{q}_i$ the predicted position of the joints, $\boldsymbol{u}_i$ the input with $i = 0, \ldots, N_p$ and α, γ weighting coefficients. The J_s term penalizes all the input signal $\boldsymbol{u}^j$ that change too rapidly from one step to the next to guarantee that the control input is suitable for the robot actuators. The feasibility of the commanded acceleration $\boldsymbol{q}_{d,k}$ is obtained by imposing a set of state and control constraints for our optimization problem:

$$\boldsymbol{q}_m \leq \boldsymbol{q} \leq \boldsymbol{q}_M \quad (26)$$

$$\dot{\boldsymbol{q}}_m \leq \dot{\boldsymbol{q}} \leq \dot{\boldsymbol{q}}_M \quad (27)$$

$$\ddot{\boldsymbol{q}}_m \leq \boldsymbol{u} \leq \ddot{\boldsymbol{q}}_m \quad (28)$$

where $\boldsymbol{q}_m, \dot{\boldsymbol{q}}_m, \ddot{\boldsymbol{q}}_m$ and $\boldsymbol{q}_M, \dot{\boldsymbol{q}}_M, \ddot{\boldsymbol{q}}_M$ are the lower and upper bounds for, respectively, joint positions, velocities and accelerations. The MPC constraints do not provide strict safety guarantee at the beginning of the learning process. Over time the online learning strategy improves the correction of the unmodeled dynamics resulting in a better correspondence between the MPC internal model and the real robot. Therefore, once the GP prediction error will became negligible, the optimal control input from the MPC will assure constraint satisfaction for the real manipulator as well.

5 Simulation results

In this section we report the results of a simulation performed using Matlab. We applied the proposed method on a KUKA LBR iiwa 7 R800 manipulator performing a trajectory tracking task. We analyze the tracking capabilities of the simulated robot, whose dynamic parameters are the ones reported in [22, 6] (originally identified for the academic version of the iiwa robot, namely the LightWeight Robot (LWR) 4+, which shares with the iiwa the same kinematics) with a deviation from the nominal parameters (around 20 % of their value), comparing the performances with and without the GP correction. In the experiment, for the simulated robot we employ a friction model with nonlinear and non-smooth behaviors (i.e., viscous and Coulomb friction with a Stribeck effect) [23]. The evolution of the system is discretized at 1 ms, the high level MPC runs at 200 Hz while the dataset acquisition is carried out at the same frequency of the FL controller. The simulations task consists in following a given reference trajectory in joint space, both for joint positions and velocities. In

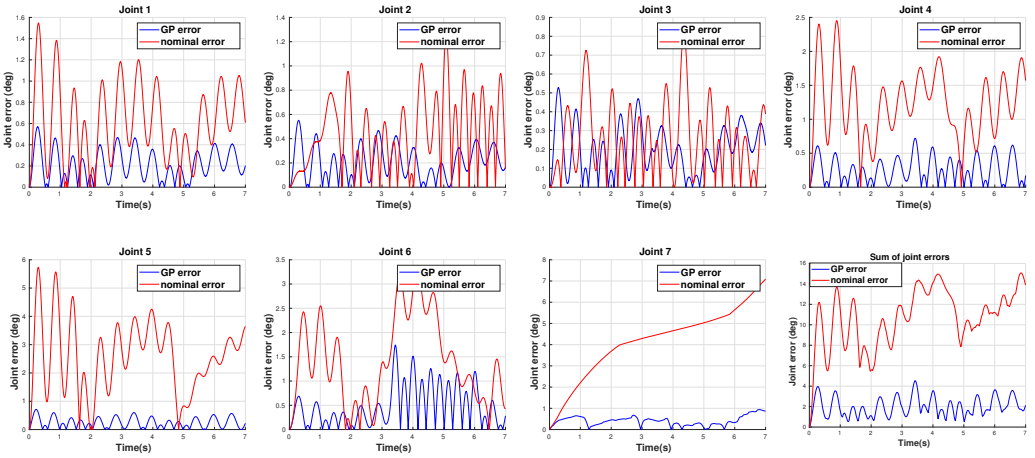

Figure 4: Joints angular position error comparison.

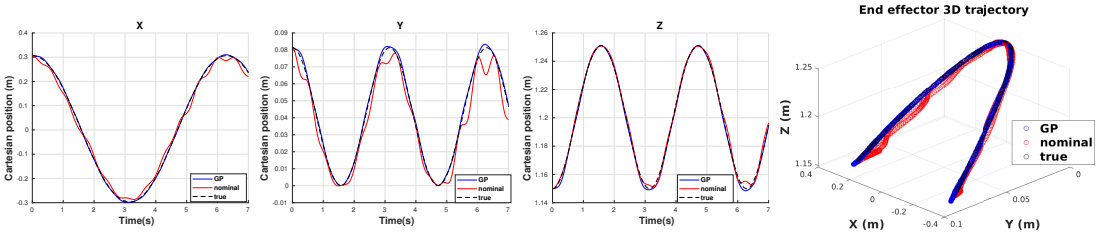

Figure 5: Comparison between two different control modalities: using the nominal dynamic model (red line) and improving it with the Gaussian Process Regression (blue line). The first three panels show the x, y and z coordinates of the end-effector during motion, while the fourth panel shows the end-effector trajectory in the Cartesian space.

particular, the first six joints should follow a sinusoidal path while the last one should remain at rest. The approach presented in this paper doesn't need Persistent Exciting Trajectories (PETs) because it is based on a non parametric regressor (PETs may be used, however, to identify the nominal model). Fig. 4 reports the joint angular position errors during the tracking task. The blue curves show the absolute value of the error when the learning scheme of Fig. 2 is adopted. The red curves show, instead, the absolute values of the errors when only the nominal model (*a priori* knowledge) is employed to compute the driving torque. It is evident that our learning scheme improves the quality of the tracking capabilities of the controller.

The tracking error in the joint space has a direct effect on the cartesian error as well. Indeed, even small errors of the angular positions of the joints may dramatically deviate the end-effector position from the desired path. Fig. 5 reports the cartesian error of the end-effector when the GP correction is active (blue curves) and when it is not (red curves). The first three panels report, respectively, the coordinates x, y and z of the end-effector, while the last panel shows its position in Cartesian space. Even in this case, the use of the learning routines improves the quality of the tracking controller. As shown in Fig. 1 of the supplementary material, we show that, with our framework, the reference accelerations computed by the MPC reach the measured ones on the controlled system. In fact, since the robot is torque-controlled, the accelerations convergence occurs only when the GP correctly reproduces the model mismatch, producing an exact FL.

6 Conclusion and Future Work

In this work we introduce a new approach for online learning the exact FL of a manipulator while executing a predefined task. The proposed method is composed by a dataset collection procedure ,designed to reconstruct the unmodeled dynamics, and by a controller that computes the commanding joint torques according to a desired trajectory. Assuming that the size of our dataset is small,

we employ Gaussian Process Regression to learn the unmodeled dynamics. In this work we showed that the proposed approach requires only the knowledge of joint positions and velocities without the need of any torque measurements. The Controllability Gramian is used for computing the joint's accelerations. Simulation results show that our framework has higher precision for the estimation of the model mismatch and a lower tracking error with respect to other approaches without online correction.

Possible extensions of this algorithm will be the implementation of the proposed method on a real manipulator, the analysis of the effect of the learning transient for the constraints satisfaction, the implementation of approximated methods to speed up the regressor prediction time and eventually the use of variance information for improving generalization. Moreover, since the framework presented is able to improve the knowledge of the robot dynamic model, it is possible to retrieve a more reliable estimation of the external torques – possibly, due to unforeseen contacts, for example using the methods presented in [2], which require an accurate knowledge of the dynamic model of the robot.

References

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

[2] S. Haddadin, A. De Luca, and A. Albu-Schäffer. Robot collisions: A survey on detection, isolation, and identification. *IEEE Transactions on Robotics*, 33(6):1292–1312, Dec 2017. ISSN 1552-3098. doi:10.1109/TRO.2017.2723903.

[3] E. Magrini and A. De Luca. Hybrid force/velocity control for physical human-robot collaboration tasks. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Oct. 2016.

[4] J. Hollerbach, W. Khalil, and M. Gautier. Model identification. In *Handbook of Robotics*, pages 321–344. Springer, 2008.

[5] A. Janot, P. Vandanjon, and M. Gautier. A generic instrumental variable approach for industrial robot identification. *IEEE Transactions on Control Systems Technology*, 22(1):132–145, 2014.

[6] 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*, pages 2075–2081, 2016.

[7] W. Khalil and E. Dombre. *Modeling, Identification and Control of Robots*. Hermes Penton London, 2002.

[8] C. Gaz and A. De Luca. Payload estimation based on identified coefficients of robot dynamics with an application to collision detection. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3033–3040, Sep. 2017.

[9] D. Nguyen-Tuong and J. Peters. Model learning for robot control: A survey. *Cognitive processing*, 12:319–40, 04 2011. doi:10.1007/s10339-011-0404-1.

[10] M. Deisenroth, D. Fox, and C. Edward Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37:408–423, 02 2015. doi:10.1109/TPAMI.2013.218.

[11] S. Kamthe and M. Deisenroth. Data-efficient reinforcement learning with probabilistic model predictive control. In *Proc. of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

[12] F. Berkenkamp and A. P. Schoellig. Safe and robust learning control with gaussian processes. In *2015 European Control Conference (ECC)*, pages 2496–2501, July 2015. doi:10.1109/ECC.2015.7330913.

[13] C. Ostafew, A. Schoellig, and T. D. Barfoot. Robust constrained learning-based NMPC enabling reliable mobile robot path tracking. *The International Journal of Robotics Research*, 35, 05 2016. doi:10.1177/0278364916645661.

[14] Z. Shareef, P. Mohammadi, and J. Steil. Improving the inverse dynamics model of the KUKA LWR IV+ using independent joint learning. *IFAC-PapersOnLine*, 49(21), 09 2016.

[15] T. Waegeman, F. Wyffels, and B. Schrauwen. Feedback control by online learning an inverse model. *Neural Networks and Learning Systems, IEEE Transactions on*, 23:1637–1648, 10 2012. doi:10.1109/TNNLS.2012.2208655.

[16] J. Umlauft, T. Beckers, M. Kimmel, and S. Hirche. Feedback linearization using gaussian processes. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 5249–5255, Dec 2017. doi:10.1109/CDC.2017.8264435.

[17] D. Nguyen-Tuong, M. Seeger, and J. Peters. Computed torque control with nonparametric regression models. In *2008 American Control Conference*, pages 212–217, June 2008. doi:10.1109/ACC.2008.4586493.

[18] C.-T. Chen. *Linear System Theory and Design*. Oxford University Press, 2012.

[19] R. C. Grande, G. Chowdhary, and J. How. Experimental validation of bayesian nonparametric adaptive control using gaussian processes. *Journal of Aerospace Information Systems*, 11: 565–578, 09 2014. doi:10.2514/1.I010190.

[20] D. Nguyen-Tuong, M. Seeger, J. Peters, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou. Local gaussian process regression for real time online model learning and control. *Advances in Neural Information Processing Systems 21: Proceedings of the 2008 Conference, 1193-1200 (2009)*, 01 2008.

[21] F. Borrelli, A. Bemporad, and M. Morari. *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017. doi:10.1017/9781139061759.

[22] C. Gaz, F. Flacco, and A. De Luca. Identifying the dynamic model used by the KUKA LWR: A reverse engineering approach. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1386–1392, 09 2014. doi:10.1109/ICRA.2014.6907033.

[23] F. Al-Bender, V. Lampaert, and J. Swevers. Modeling of dry sliding friction dynamics: From heuristic models to physically motivated models and back. *Chaos (Woodbury, N.Y.)*, 14:446–60, 07 2004. doi:10.1063/1.1741752.