2022 I-RIM Conference
October 7-9, Rome, Italy
ISBN: 9788894580532
DOI:10.5281/zenodo.7531280

Exploiting Robot Redundancy for Online Learning and Control

Marco Ficorilli, Marco Capotondi*, Valerio Modugno, Alessandro De Luca
Dipartimento di Ingegneria Informatica Automatica e Gestionale
Sapienza Università di Roma, Italy
Email: ficorilli.1797705@studenti.uniroma1.it, {capotondi,modugno,deluca}@diag.uniroma1.it

***Abstract*—Accurate trajectory tracking in the task space is critical in many robotics applications. Model-based robot controllers are able to ensure very good tracking but lose effectiveness in the presence of model uncertainties. On the other hand, online learning-based control laws can handle poor dynamic modeling, as long as prediction errors are kept small and decrease over time. However, in the case of redundant robots directly controlled in the task space, this condition is not usually met. We present an online learning-based control framework that exploits robot redundancy so as to increase the overall performance and shorten the learning transient. The validity of the proposed approach is shown through a comparative study conducted in simulation on a KUKA LWR4+ robot.**

***Index Terms*—model learning, redundant robot, variance optimization**

I. Introduction

Ensuring accurate trajectory tracking of robotic Cartesian tasks is relevant in many applications. In the literature, there exist control approaches that can guarantee very good tracking with perfect knowledge of the system model. In real application scenarios, however, this condition is hardly met, due to the presence of uncertain dynamic parameters and unmodeled dynamics.

In [1] the authors have presented an online learning scheme that compensates for model uncertainties in fully actuated robots, generating additional torque control actions to realize an accurate feedback linearization in the joint space. This approach was extended to underactuated robots in [2] by including an offline iterative planning phase for the zero dynamics of the system, which is updated from previous trials. The extension of [1] to the case of robots that are kinematically redundant with respect to the task raises some issues. If not properly managed by the control scheme, task redundancy usually prevents the generation of cyclic motions in the joint space in response to cyclic tasks [3], resulting in a continuous exploration of the input space. This behavior increases the prediction errors and thus the number of steps required for convergence by a learning algorithm (referred to as Learning Transient, LT in this work), drastically reducing control performance.

In this work, we propose to exploit robot redundancy so as to improve the performance of the learning control law. The main novelty consists in merging a quadratic approximation of Gaussian Process Regressors [4] with classical redundancy resolution schemes [5]. In particular, it is possible to mitigate the learning transient through optimal self-motions in the joint space. A suitable quadratic optimization problem is formulated whose efficient solution steers the redundant DoFs towards already explored joint space regions, thus increasing the regressor performance (similarly to [6], [7]). As a result, accurate trajectory tracking is obtained in the task space, as well as a more reliable motion in the joint space, with smoother velocity and torque profiles.

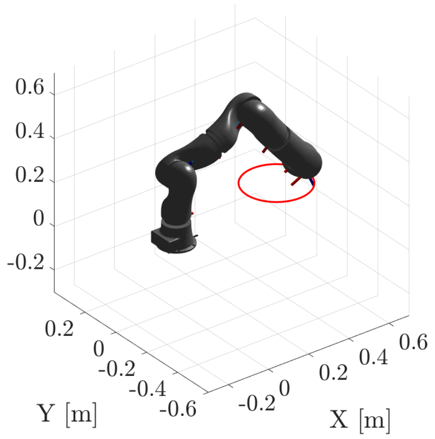


Fig. 1. The KUKA LWR4+ robot and the task Cartesian trajectory considered in the simulations.

The paper is organized as follows. The problem is formulated in Sec. II. Section III describes the learning control method with optimal redundancy resolution and its quadratic approximation. The method is validated through a comparative study in Sec. IV. Conclusions are drawn in Sec. V.

II. Problem Formulation

Given a fully actuated robot with n DoF, the system dynamics can be formulated as

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

where $\boldsymbol{q} \in \mathbb{R}^n$ is the configuration vector, $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ is the positive definite and symmetric inertia matrix, $\boldsymbol{n} \in \mathbb{R}^n$ is the sum of Coriolis and gravity vectors, and $\boldsymbol{\tau} \in \mathbb{R}^n$ is the joint torque vector. In presence of model uncertainties (uncertain parameters and/or unmodeled dynamics), we can write

$$\boldsymbol{M} = \hat{\boldsymbol{M}} + \Delta\boldsymbol{M} \qquad \boldsymbol{n} = \hat{\boldsymbol{n}} + \Delta\boldsymbol{n}. \qquad (2)$$

* Corresponding author.

in which $\hat{\boldsymbol{M}}$ and $\hat{\boldsymbol{n}}$ are the nominal terms. A preliminary nonlinear Feedback Linearization (FL) is performed on the nominal dynamics as follows:

$$\boldsymbol{\tau}_{\text{FL}} = \hat{\boldsymbol{M}}\boldsymbol{u} + \hat{\boldsymbol{n}}. \tag{3}$$

In the presence of unmodeled terms, from eqs. (1–3) we obtain the perturbed closed-loop dynamics

$$\ddot{\boldsymbol{q}} = \boldsymbol{u} + \boldsymbol{\delta}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u}), \tag{4}$$

where $\boldsymbol{\delta}$ represents the perturbation effects acting in the joint space.

Since the aim of this work is to realize accurate trajectory tracking at the task level, we characterize next the effects of the model mismatch in the task space. Consider the second-order kinematics

$$\ddot{\boldsymbol{x}} = \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}}, \tag{5}$$

where $\ddot{\boldsymbol{x}} \in \mathbb{R}^m$ is the task acceleration and $\boldsymbol{J} \in \mathbb{R}^{m\times n}$ is the task Jacobian, being $m < n$ the dimension of the task. Combining eq. (4) and eq. (5) yields the task dynamics

$$\ddot{\boldsymbol{x}} = \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{J}(\boldsymbol{q})(\boldsymbol{u} + \boldsymbol{\delta}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u})), \tag{6}$$

which can be rewritten as

$$\ddot{\boldsymbol{x}} = \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{J}(\boldsymbol{q})\boldsymbol{u} + \boldsymbol{\delta}_c(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u}), \tag{7}$$

where $\boldsymbol{\delta}_c = \boldsymbol{J}(\boldsymbol{q})\boldsymbol{\delta}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u})$ represents the perturbation acting in the task space.

III. Learning-Based Control of Redundant Robots

To improve task accuracy during the LT, we introduce the general forms of the proposed controller

$$\boldsymbol{u} = \boldsymbol{J}^\dagger(\boldsymbol{q})\left(\boldsymbol{a} - \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}}\right) + \boldsymbol{P}_J(\boldsymbol{q})\boldsymbol{u}_0 + \boldsymbol{u}_{gp}, \tag{8}$$

where $\boldsymbol{u} \in \mathbb{R}^n$ is the commanded joint acceleration,$\boldsymbol{J}^\dagger$ is the pseudoinverse of the task Jacobian, $\boldsymbol{a} \in \mathbb{R}^m$ is the task acceleration command, $\boldsymbol{P}_J = \boldsymbol{I} - \boldsymbol{J}^\dagger\boldsymbol{J} \in \mathbb{R}^{n\times n}$ is the null-space projector of the task, $\boldsymbol{u}_0 \in \mathbb{R}^n$ is an arbitrary null-space joint acceleration, and $\boldsymbol{u}_{gp} \in \mathbb{R}^n$ is the additional joint acceleration control action that should cancel the model mismatches by learning.

Since the model mismatch is unknown, it is approximated with $\boldsymbol{u}_{gp}$ using a vector of Gaussian Process (GP) regressors. To possibly obtain an asymptotically stable closed-loop error dynamics along the desired task trajectory $\boldsymbol{x}_d \in \mathbb{R}^m$, we set $\boldsymbol{a} = \boldsymbol{a}_{ff} + \boldsymbol{a}_{pd}$ where $\boldsymbol{a}_{ff} = \ddot{\boldsymbol{x}}_d$ is the task feedforward component and $\boldsymbol{a}_{pd}$ defines the feedback action

$$\boldsymbol{a}_{pd} = \boldsymbol{K}_P(\boldsymbol{x}_d - \boldsymbol{x}) + \boldsymbol{K}_D(\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}), \tag{9}$$

being $\boldsymbol{K}_D$ and $\boldsymbol{K}_P \in \mathbb{R}^{m\times m}$ positive definite (diagonal) gain matrices. Finally, redundancy is exploited by optimizing the term $\boldsymbol{u}_0$ in eq. (8), in order to mitigate the effect of the LT and improve the tracking performance. Accordingly, eq. (8) can be rewritten as

$$\boldsymbol{u} = -\boldsymbol{J}^\dagger(\boldsymbol{q})\dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{u}_{pd} + \boldsymbol{u}_{ff} + \boldsymbol{P}_J\boldsymbol{u}_0 + \boldsymbol{u}_{gp} \tag{10}$$

where

$$\boldsymbol{u}_{ff} = \boldsymbol{J}^\dagger(\boldsymbol{q})\boldsymbol{a}_{ff}, \qquad \boldsymbol{u}_{pd} = \boldsymbol{J}^\dagger(\boldsymbol{q})\boldsymbol{a}_{pd}.$$

A scheme of the proposed controller is shown in Fig. 2.

A. Model Learning for Control

Consider a dataset of input-output noisy observations $\mathcal{D} = \{(\boldsymbol{X}_i, Y_i = \phi(\boldsymbol{X}_i) + \boldsymbol{\omega}_i)\,|1 \le i \le n_d\}$, with $\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_\omega)$ and where ϕ is an unknown scalar function to be reconstructed. Given a kernel function $k(\cdot,\cdot)$, for a generic query point $\hat{\boldsymbol{X}}$ it is possible to compute the GP regressor predictive distribution

$$\varepsilon(\hat{\boldsymbol{X}}|\mathcal{D}) \sim \mathcal{N}\left(\mu(\hat{\boldsymbol{X}}, \mathcal{D}), \sigma^2(\hat{\boldsymbol{X}}, \mathcal{D})\right) \tag{11}$$

where μ and σ are respectively the regressor prediction and the epistemic error associated with each input [8]. In this work, we use a set of n GPs to approximate the vector perturbation $\boldsymbol{\delta}$. The data collection procedure, necessary for gathering the dataset to approximate the unknown function, is based on the difference between the commanded and the actual joint acceleration for each DoF. From eq. (4) we have for the k-th control step

$$\boldsymbol{\delta}_k = \ddot{\boldsymbol{q}}_k - \boldsymbol{u}_k. \tag{12}$$

A new data point is generated as

$$\boldsymbol{X}_k = (\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k, \boldsymbol{u}_k), \qquad \boldsymbol{Y}_k = \ddot{\boldsymbol{q}}_k - \boldsymbol{u}_k$$

with the acceleration $\ddot{\boldsymbol{q}}_k$ being reconstructed numerically. It is important to note that, through eq. (4), the actual acceleration of the system depends on the state $(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k)$ and on the commanded acceleration $\boldsymbol{u}_k$, i.e., on the input $\boldsymbol{X}_k$ of the regression scheme. Dealing with a persistent LT, we define the joint space prediction error as

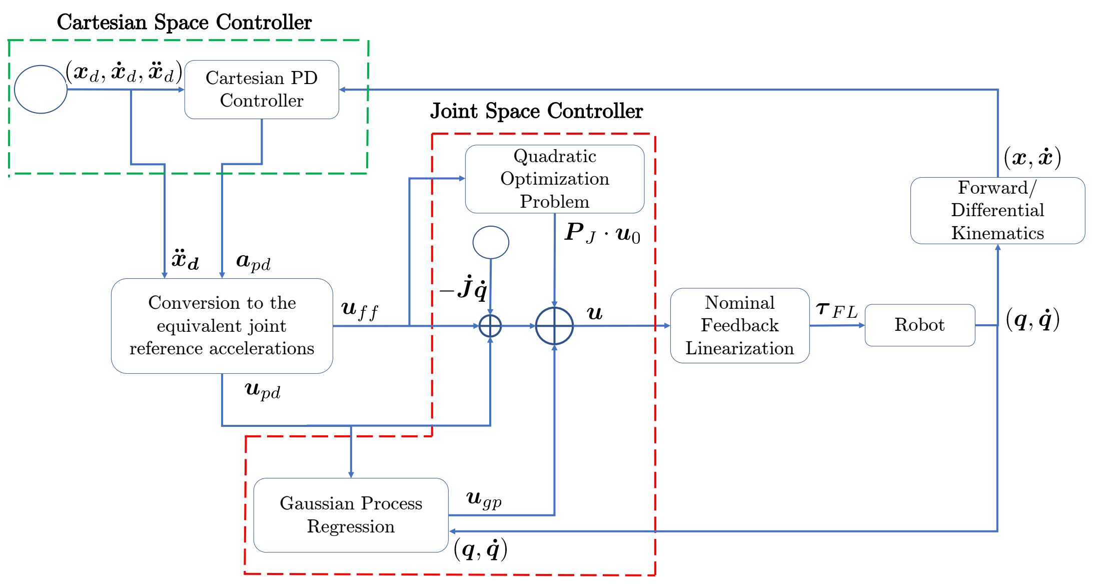
$$\boldsymbol{\varepsilon}(\hat{\boldsymbol{X}}) = \boldsymbol{\mu}(\hat{\boldsymbol{X}}) - \boldsymbol{\delta}(\hat{\boldsymbol{X}}) \tag{13}$$

which represents the vector of the GPs prediction errors.

B. Nonlinear Redundancy Resolution

In order to keep the system state evolution in regions that have already been explored, we formulate a nonlinear optimization problem to exploit the robot redundancy. This is very important in the context of Cartesian tasks for a redundant robot since, in general, redundancy introduces a drift in the joint motion at the acceleration level. Thus, the learning algorithm will continuously visit new regions of the input state, extending the LT and slowing the convergence. On the other hand, suitable joint space self-motions may reduce the prediction error and lead thus to higher tracking accuracy. Moreover, we would like to reduce the control effort and the joint velocities, obtaining overall smoother motions. To this end, we can choose the null-space component $\boldsymbol{u}_0$ of the commanded joint acceleration by minimizing a weighted combination of the actual and the one-step-ahead vector of the predictive variance of the GPs. In this way, a constrained nonlinear optimization problem is formulated, having a cost function equal to the sum of the norm of the vectors of predictive variances at the control steps k and $k+1$,

$$\boldsymbol{u}_0 = \min_{\boldsymbol{u}_{null}} \alpha\|\boldsymbol{\sigma}(\boldsymbol{x}_k)\|_2 + \beta\|\boldsymbol{\sigma}(\boldsymbol{x}_{k+1})\|_2, \tag{14}$$



Cartesian Space Controller
$(\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d, \ddot{\boldsymbol{x}}_d)$
Cartesian PD Controller
Joint Space Controller
Quadratic Optimization Problem
$\boldsymbol{P}_J \cdot \boldsymbol{u}_0$
$(\boldsymbol{x}, \dot{\boldsymbol{x}})$
Forward/ Differential Kinematics
$\ddot{\boldsymbol{x}}_d$
$\boldsymbol{a}_{pd}$
$-\dot{\boldsymbol{J}}\dot{\boldsymbol{q}}$
Conversion to the equivalent joint reference accelerations
$\boldsymbol{u}_{ff}$
$\boldsymbol{u}$
Nominal Feedback Linearization
$\boldsymbol{\tau}_{FL}$
Robot
$(\boldsymbol{q}, \dot{\boldsymbol{q}})$
$\boldsymbol{u}_{pd}$
Gaussian Process Regression
$\boldsymbol{u}_{gp}$
$(\boldsymbol{q}, \dot{\boldsymbol{q}})$

Fig. 2. Block scheme of the proposed framework

for suitable non-negative weights α and β, where $\boldsymbol{u}_{null}$ represents the null-space reference acceleration. The cost function is subject to state and input constraints and to the first-order hold discretization of the nominal feedback linearized state model.

C. Fast GPs and Approximate Redundancy Resolution

The use of exact Gaussian Process Regression for both the FL correction and the variance optimization may be impractical for real-time applications. In fact, GPs prediction computation time scales as $O(N^3)$, where N is the number of points in the dataset. Moreover, the prediction variance to be optimized is in general a very complex nonlinear and non-convex function. Nonetheless, through Fast GPs approximation [4] it is possible to formulate a simpler framework for both prediction and redundancy resolution, at the cost of an approximation error which is negligible under certain conditions.

Consider the unknown (latent) function ϕ to be reconstructed, which has been assumed to be differentiable and depends on the input X. The linearization of this latent function is

$$\phi(x + \Delta x) \simeq \tilde{\phi}_x(\Delta x) = \phi(x) + \Delta x^T \nabla_x \phi(x). \tag{15}$$

Defining $g(x) = \nabla_x \phi(x)$, $\hat{x} = \begin{pmatrix} 1 \\ \Delta x \end{pmatrix}$ and $\hat{\phi}(x) = \begin{pmatrix} \phi(x) \\ g(x) \end{pmatrix}$, we obtain $\tilde{\phi}_x(\Delta x) = \hat{x}^T \hat{\phi}(x)$. Since differentiation and the scalar product are linear operators, the result $\tilde{\phi}_x(\Delta x)$ is still a multivariate Gaussian Process. Hence, the posterior distribution of the approximated latent function will be a Gaussian distribution with mean and variance

$$\mu_\phi = \hat{m}^T \hat{x}(x), \qquad \sigma_\phi^2 = \hat{x}^T \hat{V} \hat{x}.$$

The vector $\hat{\boldsymbol{m}}$ and the positive definite matrix $\hat{\boldsymbol{V}}$ are defined as

$$\hat{\boldsymbol{m}} = \begin{pmatrix} \kappa(x, X) \\ K^{(1,0)}(x, X) \end{pmatrix} \left(K(X, X) + \sigma_n^2 I\right)^{-1} Y \tag{16}$$

$$\hat{\boldsymbol{V}} = \begin{pmatrix} \kappa(x, x) & K^{(0,1)}(x, x) \\ K^{(1,0)}(x, x) & K^{(1,1)}(x, x) \end{pmatrix} - \begin{pmatrix} \kappa(x, X) \\ K^{(1,0)}(x, X) \end{pmatrix} \left(\kappa(X, X) + \sigma_n^2 I\right)^{-1} \left(\kappa(X, x) \;\; K^{(0,1)}(X, x)\right), \tag{17}$$

where κ represents the kernel function associated to the covariance matrix $\boldsymbol{K}$, namely $\boldsymbol{K}_{i,j}(X, X) = \kappa(x_i, x_j)$ with

$$\boldsymbol{K}^{(1,0)} = \nabla_x \kappa(x, x'), \qquad \boldsymbol{K}^{(0,1)} = \nabla_{x'} \kappa(x, x'),$$

$$\boldsymbol{K}_{i,j}^{(1,1)}(x, x') = \frac{\partial^2 \kappa(x, x')}{\partial x_i \partial x_j'}.$$

As a result, the original nonlinear and non-convex optimization problem can be rewritten as a quadratic problem, whose solution is much easier and faster:

$$\boldsymbol{u}_0 = \min_{\boldsymbol{u}_{null}} \alpha \, \hat{x}_k^T \hat{\boldsymbol{V}} \hat{x}_k + \beta \, \hat{x}_{k+1}^T \hat{\boldsymbol{V}} \hat{x}_{k+1}. \tag{18}$$

IV. Simulation Results

The positional task has dimension $m = 3$ and thus the redundancy degree is $n - m = 4$. In particular, the nominal model presents a 30% mismatch in the mass and inertia of each link and it does not take into account the joint friction, which thus represents an unmodeled phenomenon. The proposed approach has been validated through numerical simulations on a KUKA LWR4+, a robot with $n = 7$ revolute joints. We addressed the problem of tracking a Cartesian horizontal circular trajectory at constant height (see Fig. 1) of radius $R = 15$ cm in $T = 2$ s, with uncertainty in the dynamic robot

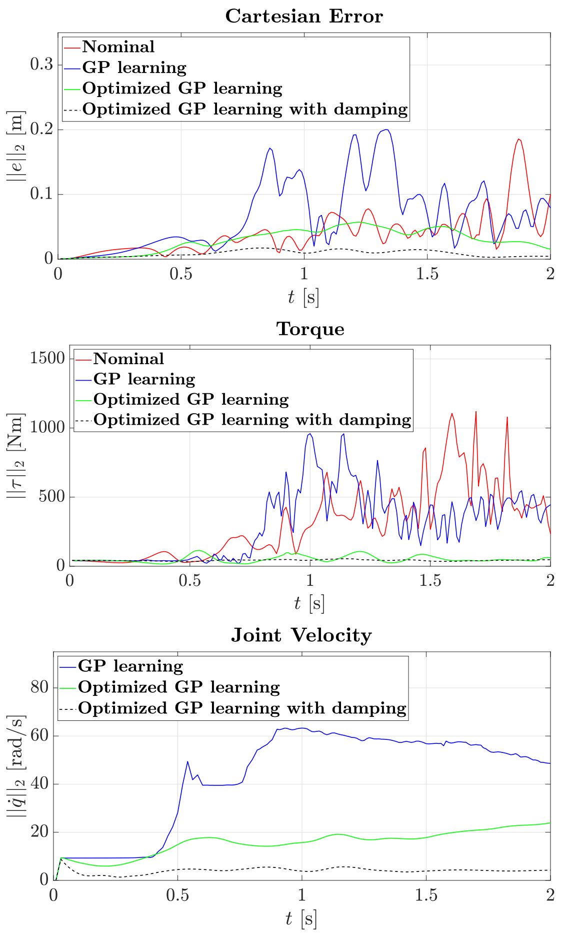


Fig. 3. Results for each of the considered controllers. From top to bottom: the norm of the cumulative Cartesian error, norm of the cumulative torque control effort, and norm of the cumulative joint velocity.

model. Also, the proportional and derivative gains have been set to $K_P = \text{diag}\{100, 100, 100\}$ and $K_D = \text{diag}\{20, 20, 20\}$ respectively. To analyze the performance of the learning framework, we conducted a comparative study between different versions of the proposed controller.

The first controller considered in the study is

$$\boldsymbol{u} = -\boldsymbol{J}^{\dagger}(\boldsymbol{q})\dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{u}_{pd} + \boldsymbol{u}_{ff}. \tag{19}$$

Due to the presence of model uncertainties, the tracking performance is drastically reduced, as shown in Fig. 3 (red line). On the other hand, the control effort and the joint velocities of the system are increased, since the controller tries to recover the cumulative tracking error by increasing the magnitude of the commanded torques. In the second controller, we added the learning correction $\boldsymbol{u}_{gp}$, with the aim of realizing the reference joint accelerations,

$$\boldsymbol{u} = -\boldsymbol{J}^{\dagger}(\boldsymbol{q})\dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{u}_{gp} + \boldsymbol{u}_{pd} + \boldsymbol{u}_{ff}. \tag{20}$$

Due to the presence of the learning transient in such a fast motion, the tracking performance w.r.t. the Cartesian PD controller is improved mainly toward the end of the trajectory execution (Fig. 3, blue line). Similarly, the control effort is reduced when the learning algorithm starts to converge.

The third controller includes the null-space contribution $\boldsymbol{u}_{uo}$:

$$\boldsymbol{u} = -\boldsymbol{J}^{\dagger}(\boldsymbol{q})\dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{u}_{gp} + \boldsymbol{u}_{pd} + \boldsymbol{u}_{ff} + \boldsymbol{u}_0. \tag{21}$$

This addition remarkably improves the tracking performance of the system (Fig. 3, green line). Finally, we tested the proposed framework by introducing, next to the optimization term $\boldsymbol{u}_0 = \boldsymbol{u}_{opt}$, also a damping term in the null space,

$$\boldsymbol{u}_0 = \boldsymbol{u}_{opt} - \boldsymbol{K}_v\dot{\boldsymbol{q}}. \tag{22}$$

This is known to be a convenient addition in any second-order redundancy resolution scheme, in order to ensure a stable joint motion [5]).

The best performance is achieved in this last case, obtaining very low Cartesian errors w.r.t. all the previous controllers and a significant reduction of control effort and joint velocity (Fig. 3, dashed black line). This is due to the fact that damping naturally limits the input space exploration, similarly to what self-motion optimization also does, though in a more limited way (its effect is just to reduce the joint velocity). As a result, the combination of both actions drastically reduces the LT duration while improving the learning process.

V. Conclusions and Future Work

We have proposed an online learning-based controller to realize accurate Cartesian trajectory tracking for redundant robots in presence of model uncertainties, guaranteeing high performance also during the learning transient through a self-motion optimization. We tested the approach by simulation on a KUKA LWR4+ robot through a comparative study that showed how the proposed control framework achieves the best behavior in terms of tracking error reduction, motion smoothness, and reduced control effort. We will test the approach through experiments on the same robot, and we plan to introduce a redundancy resolution strategy for active learning so as to reduce even further the Learning Transient.

References

[1] 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. Machine Learning Research (3rd Conf. on Robot Learning)*, vol. 100, 2020, pp. 1359–1368.

[2] G. Turrisi, M. Capotondi, C. Gaz, V. Modugno, G. Oriolo, and A. De Luca, "On-line learning for planning and control of underactuated robots with uncertain dynamics," *IEEE Robotics and Automation Lett.*, vol. 7, no. 1, pp. 358–365, 2022.

[3] A. De Luca, L. Lanari, and G. Oriolo, "Control of redundant robots on cyclic trajectories," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1992, pp. 500–506.

[4] T. X. Nghiem, "Linearized Gaussian processes for fast data-driven model predictive control," in *Proc. American Control Conf.*, 2019, pp. 1629–1634.

[5] A. De Luca, G. Oriolo, and B. Siciliano, "Robot redundancy resolution at the acceleration level," *Laboratory Robotics and Automation*, vol. 4, no. 2, p. 97–106, 1992.

[6] Y. Yun and A. D. Deshpande, "Control in the reliable region of a statistical model with Gaussian process regression," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2014, pp. 654–660.

[7] F. Cursi, V. Modugno, L. Lanari, G. Oriolo, and P. Kormushev, "Bayesian neural network modeling and hierarchical MPC for a tendon-driven surgical robot with uncertainty minimization," *IEEE Robotics and Automation Lett.*, vol. 6, no. 2, pp. 2642–2649, 2021.

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