Automatically Deployable Robust Control of Modular Reconfigurable Robot Manipulators

Carlo Nainer and Andrea Giusti, *Member, IEEE*

***Abstract*—We propose an automatically deployable robust control scheme for modular reconfigurable robot manipulators, which accounts for noisy velocity measurements, yet it maximizes the tracking performance while avoiding chattering effects. Our proposed control approach automatically adapts to any of the possible robot compositions of given sets of modules. The robust stability is guaranteed by exploiting the use of a recursive Newton-Euler scheme with interval arithmetic computations. Moreover, the robust performance is maximized via an online regulation of the control parameters by analyzing the power spectral density of the commanded torque signal. Being fully automatic, the proposed approach allows for a quick deployment and reconfiguration of modular reconfigurable manipulators. The algorithm is validated via simulations and experiments on a commercially available robot.**

***Index Terms*—Cellular and modular robots, robust/adaptive control, industrial robots.**

I. Introduction

MODULAR reconfigurable robot (MRR) manipulators are mechatronic systems composed of multiple rearrangeable modules. This modular nature allows them to be configured and adapted for different applications. This added flexibility, with respect to classical fixed-structure robots, gives a clear advantage to these systems. MRRs are especially useful in flexible environments and applications, such as space operation and exploration [1], search and rescue [2], and human-robot collaboration in manufacturing [3]. The modularity also enables optimization of MRR compositions [4].

In the last decades there have been several examples of MRR systems, such as RMMS [5], TOMMS [6], IRIS [7], PolyBot [8], and the spring-assisted MRR of [9]. The advantages of these systems, and in particular their versatility, come with challenges, especially related to the modeling and control design. In fact, given the large number of possible robot configurations of MRRs, the design of the controllers becomes challenging due to the different kinematics and dynamics to be considered. This topic has been a longstanding problem in research (see e.g., [10], [11]). The majority of the works considered decentralized control approaches. Adaptive [12], [13] and robust [14] control approaches have been the most common control techniques presented in the literature for MRR systems. Hybrid approaches, that use centralized and decentralized operations, are also present [15].

These methods are mainly motivated by the intrinsic difficulty of designing full centralized model-based controllers given the large number of module combinations in MRR systems. Although decentralized control techniques can achieve satisfying results, in principle, it is always possible to design centralized controllers with performance equal or superior to decentralized approaches. In fact, contrary to decentralized controllers, centralized control schemes can directly compensate the subsystem couplings instead of treating them as disturbances [16]. Moreover, centralized architectures present advantages for optimal control techniques, compliance control and failure detection [17]. For these reasons, in the last decade, the design of centralized controllers, also for MRRs, has been investigated [18], [19]. These techniques, being model based, typically require automatic modeling algorithms [20]–[22].

Although the above mentioned results allow the automatic design of controllers for MRRs, practical deployment can be difficult when high tracking performance is desired. In particular, when robust performance is desired, measurement noise amplification on the control commands can lead to chattering phenomena in practice [23]. Commonly, the control gains should be limited in order to avoid excessive amplification of the high-frequency noise and disturbance. This often requires manual tuning of the control parameters, action that is typically performed by the user, thus limiting the swift deployability of MRR systems.

In this letter we present an automatically deployable robust control approach for MRR manipulators. Our proposed solution exploits a previous work of the authors [22] to automatically obtain kinematics and dynamics models for any of the possible robot configurations, and extends it for automatically obtaining model uncertainties using interval arithmetic. Furthermore, inspired by [23], a centralized control scheme that guarantees robust stability is derived by considering the automatically obtained parametric model uncertainties. Finally, we also propose a novel scheme to automatically adapt, online, the robust control parameters for maximizing the tracking performance, while keeping robust stability as well as limited noise amplification on the control commands. This allows us to avoid chattering

Manuscript received November 17, 2021; accepted February 18, 2022. Date of publication March 3, 2022; date of current version March 14, 2022. This letter was recommended for publication by Associate Editor T. Bruckmann and Editor C. Gosselin upon evaluation of the reviewers' comments. This work was supported in part by the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement Number 101016007 CONCERT and in part by the Research Südtirol/Alto Adige grant for the project Reconfigurable Collaborative Agri-Robots (RECOARO) with CUP I52F20000300005. *(Corresponding author: Carlo Nainer.)*

The authors are with the Unit of Robotics and Intelligent Systems Engineering, Fraunhofer Italia Research, 39100 Bolzano, Italy (e-mail: carlo.nainer@fraunhofer.it; andrea.giusti@fraunhofer.it).

Digital Object Identifier 10.1109/LRA.2022.3155826

effects that can be present at the control deployment, with no need for user intervention.

The remainder of this letter is structured as follows. Section II illustrates the control problem we address, after a brief preliminary on interval arithmetic. The proposed robust control approach and its capability to automatically regulate its parameters are described in Section III. Finally, simulation results and an experiment on a 7 degrees-of-freedom robot arm are presented in Section IV and Section V, respectively. The conclusions follow in Section VI.

II. Problem Statement

A. Preliminary on Interval Arithmetic

The proposed control design (Section III) is based on interval arithmetic. For clarity of the subsequent description, we recall the following definitions.

Definition 1: A multidimensional interval is a set of real numbers defined as

$$[\boldsymbol{x}] := [\underline{\boldsymbol{x}}\,, \overline{\boldsymbol{x}}]\,, \quad \underline{\boldsymbol{x}} \in \mathbb{R}^{n\times 1}\,, \overline{\boldsymbol{x}} \in \mathbb{R}^{n\times 1}\,, \tag{1}$$

where $\underline{\boldsymbol{x}}$ and $\overline{\boldsymbol{x}}$ denote the infimum and supremum of the interval, respectively.

Definition 2: Given a function $\boldsymbol{z} : \mathbb{R}^{n\times 1} \to \mathbb{R}^{m\times 1}$, its interval arithmetic evaluation of a set $[\boldsymbol{x}]$ is

$$\boldsymbol{z}([\boldsymbol{x}]) := \{\boldsymbol{z} \,|\, \boldsymbol{x} \in [\boldsymbol{x}]\}\,. \tag{2}$$

Definition 3: Given $[x] \in \mathbb{R}$ and $[y] \in \mathbb{R}$ (sets of scalar intervals), the result of the binary operations $* \in \{+, -, \cdot, /\}$ is defined as

$$[x] \circledast [y] := \{x * y \,|\, x \in [x]\,,\, y \in [y]\}\,. \tag{3}$$

For further details, the reader may refer to [24], [25].

B. Robust Control Problem

We consider a modular reconfigurable robot manipulator composed of N rigid links from a set of robot modules. The kinematics and dynamics data of these modules are assumed to be known (up to a certain accuracy), allowing for the computation of the full robot kinematics and dynamics parameters [22]. An illustration of the considered robot modules, and how they form the robot links is shown in Fig. 1. The overall model of the manipulator dynamics can be expressed as [16]

$$\boldsymbol{M}(\boldsymbol{q}, \boldsymbol{\Delta})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\Delta}) + \boldsymbol{f}(\dot{\boldsymbol{q}}, \boldsymbol{\Delta}) + \boldsymbol{g}(\boldsymbol{q}, \boldsymbol{\Delta}) = \boldsymbol{u} + \boldsymbol{d}\,, \tag{4}$$

where $\boldsymbol{q} \in \mathbb{R}^{N\times 1}$ is the vector of the joint variables, $\boldsymbol{M}(\boldsymbol{q}, \boldsymbol{\Delta}) \in \mathbb{R}^{N\times N}$ is the symmetric positive definite inertia matrix, $\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\Delta}) \in \mathbb{R}^{N\times 1}$, $\boldsymbol{f}(\dot{\boldsymbol{q}}, \boldsymbol{\Delta}) \in \mathbb{R}^{N\times 1}$, and $\boldsymbol{g}(\boldsymbol{q}, \boldsymbol{\Delta}) \in \mathbb{R}^{N\times 1}$, are the vectors containing the centrifugal and Coriolis, the friction, and the gravity model terms, respectively. Finally, $\boldsymbol{u} \in \mathbb{R}^{N\times 1}$ and $\boldsymbol{d} \in \mathbb{R}^{N\times 1}$ are the control input and the disturbance vectors. The term $\boldsymbol{\Delta}$ is the set of robot dynamics parameters.

Similarly to [23], the following assumptions are considered. The robot model is assumed to be uncertain, with the model parameters bounded by the multidimensional interval $[\boldsymbol{\Delta}]$ ($\boldsymbol{\Delta} \in [\boldsymbol{\Delta}]$), and the disturbance term $\boldsymbol{d}$ is bounded in 2-norm. The vector of nominal parameters $\boldsymbol{\Delta_0}$ is available and it is contained within the interval $[\boldsymbol{\Delta}]$ ($\boldsymbol{\Delta_0} \in [\boldsymbol{\Delta}]$). The model terms in (4) are continuous for each joint state and their derivatives ($\boldsymbol{q}$, $\dot{\boldsymbol{q}}$), and Lipschitz continuous for each $\boldsymbol{\Delta} \in [\boldsymbol{\Delta}]$. In addition, the joint velocity measurements are assumed to be affected by a white noise whose standard deviation is known.

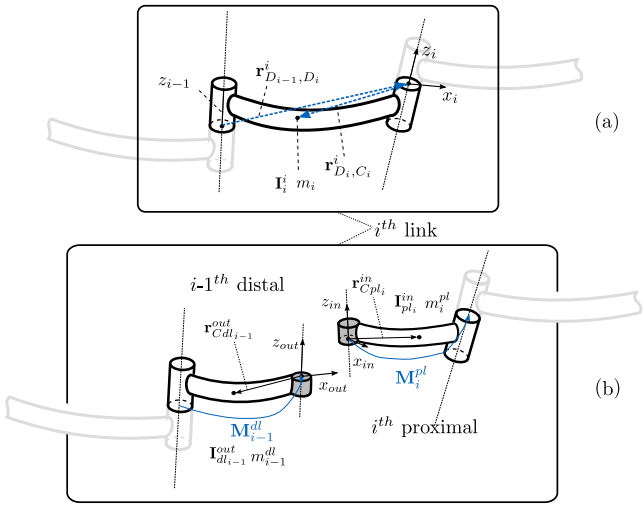


Fig. 1. Robot link (a) from robot modules (b).

For sake of compactness, the model terms computed with the nominal parameters $\boldsymbol{\Delta_0}$ are denoted by adding a 0 as subscript (e.g., $\boldsymbol{M}(\boldsymbol{q}, \boldsymbol{\Delta_0})$ is denoted as $\boldsymbol{M_0}(\boldsymbol{q})$). Moreover, the mismatch between nominal and real model is denoted by the tilde symbol (e.g., $\tilde{\boldsymbol{M}}(\boldsymbol{q}, \boldsymbol{\Delta}) = \boldsymbol{M}(\boldsymbol{q}, \boldsymbol{\Delta}) - \boldsymbol{M_0}(\boldsymbol{q})$).

For MRRs the dynamics and kinematics parameters are initially available only for the individual modules. The kinematics of each module is described by two roto-translations, one, $\boldsymbol{M^{pl}}$, for its proximal part and one, $\boldsymbol{M^{dl}}$, for its distal part (see Fig. 1). The dynamics is described by the mass (m^{pl}, m^{dl}), center of mass position ($\boldsymbol{r^{in}_{C_{pl}}}$, $\boldsymbol{r^{out}_{C_{dl}}}$), inertia tensors ($\boldsymbol{I^{in}_{pl}}$, $\boldsymbol{I^{out}_{dl}}$), for the proximal and distal part, respectively. Additional parameters, such as joint friction coefficients, gear ratio and rotor inertia, are also included into the module data. Each distal part should be merged with the proximal part of the subsequent module in order to obtain the link parameters. Since lower and upper bounds of the dynamics parameters are required for the robust control design, all the operations required to compute $\boldsymbol{\Delta}$ are performed by using interval arithmetic. The main interval link parameters, such as the mass $[m_i]$, center of mass $[\boldsymbol{r_C}]$ and inertia $[\boldsymbol{I_i}]$, are computed by evaluating their corresponding relations in [22] with set based operations as in (3). For example, for the mass and center of mass of the i-th link we have the following

$$\begin{aligned} [m_i] &= [m^{dl}_{i-1}] \oplus [m^{pl}_i]\,, \\ [\boldsymbol{r^{io}_{C_i}}] &= ([m^{dl}_{i-1}] \odot [\boldsymbol{r^{out}_{Cdl_{i-1}}}] \oplus [m^{pl}_i] \odot [\boldsymbol{r^{in}_{Cpl_i}}]) \oslash [m_i]. \end{aligned} \tag{5}$$

For the kinematics parameters, classical operations are used since no uncertainty is considered (the reader may refer to [22]). This full set of uncertain parameters forms the $[\boldsymbol{\Delta}]$ term.

Under the previously stated assumptions, and given a reference joint trajectory $\boldsymbol{q_d(t)}$ (at least twice differentiable), we face the problem of designing a robust controller which guarantees robust stability and maximizes tracking performance for any of the possible modular robot configuration by adapting itself to different robot kinematics and dynamics, and by requiring no user intervention at its deployment. In particular, the tracking performance is automatically maximized, avoiding undesirable effects such as chattering from excessive noise amplification.

III. Proposed Approach

In this section we describe the proposed robust control approach. First, in Section III.A, the robust control design and structure is presented. Afterwards, in Section III.B, a conservative initialization of the control parameters is proposed in order to enable an automatic control deployment. Finally, in Section III.C we introduce a technique to automatically adapt the robust control parameters in order to maximize the tracking performance while keeping under control the chattering effects.

A. Robust Control Design

A proportional-integrative-derivative (PID) controller is used as a base for the control design,[1] with the standard inverse-dynamics control law [16] as

$$\boldsymbol{u} = \boldsymbol{M_0(q)y} + \boldsymbol{c_0(q,\dot{q})} + \boldsymbol{f_0(\dot{q})} + \boldsymbol{g_0(q)} - \boldsymbol{\nu_{ID}}\,, \quad (6)$$

with

$$\boldsymbol{y} = \boldsymbol{\ddot{q}_d} + \boldsymbol{K_D}\dot{\boldsymbol{e}} + \boldsymbol{K_P e} + \boldsymbol{K_I}\int \boldsymbol{e}\,dt\,, \quad (7)$$

where $\boldsymbol{e} = \boldsymbol{q_d} - \boldsymbol{q}$, $\boldsymbol{K_P}$, $\boldsymbol{K_I}$ and $\boldsymbol{K_D}$ are the positive definite matrices of proper dimensions defining the proportional, integral, and derivative gains. Furthermore, $\boldsymbol{\nu_{ID}}$ is an additional term allowing for the introduction of the robust feedback control term.

For sake of compactness, we define $\boldsymbol{n(q,\dot{q},\Delta)} = \boldsymbol{c(q,\dot{q},\Delta)} + \boldsymbol{f(\dot{q},\Delta)} + \boldsymbol{g(q,\Delta)}$. By applying (6) into (4) we obtain

$$\begin{aligned}\boldsymbol{M(q,\Delta)\ddot{q}} = \boldsymbol{M_0(q)y} + \boldsymbol{n_0(q,\dot{q})} - \boldsymbol{n(q,\dot{q},\Delta)} \\ - \boldsymbol{\nu_{ID}} + \boldsymbol{d}\,, \end{aligned} \quad (8)$$

and by subtracting $\boldsymbol{M(q,\Delta)}(\boldsymbol{\ddot{e}} + \boldsymbol{K_D}\dot{\boldsymbol{e}} + \boldsymbol{K_P e} + \boldsymbol{K_I}\int \boldsymbol{e}\,dt)$ from both sides we have

$$\begin{aligned}\boldsymbol{M(q,\Delta)}\left(\boldsymbol{\ddot{e}} + \boldsymbol{K_D}\dot{\boldsymbol{e}} + \boldsymbol{K_P e} + \boldsymbol{K_I}\int \boldsymbol{e}\,dt\right) \\ = \boldsymbol{w_{ID}}\left(\boldsymbol{q,\dot{q},y,\Delta_0,\Delta,d}\right) + \boldsymbol{\nu_{ID}}\,, \end{aligned} \quad (9)$$

with

$$\boldsymbol{w_{ID}}\left(\boldsymbol{q,\dot{q},y,\Delta_0,\Delta,d}\right) = \boldsymbol{\tilde{M}(q,\Delta)y} + \boldsymbol{\tilde{n}(q,\dot{q},\Delta)} - \boldsymbol{d}\,, \quad (10)$$

which represents the overall perturbation due to external disturbances and the model mismatch.

[1] A simpler PD controller can be also chosen.

Using interval arithmetic, similarly to [23], we can compute the worst-case perturbation

$$\boldsymbol{\rho([\Phi_{ID}])} = \max(|\underline{\boldsymbol{\Phi}}_{\boldsymbol{ID}}|, |\overline{\boldsymbol{\Phi}}_{\boldsymbol{ID}}|)\,, \quad (11)$$

where the max operator is applied element-wise and

$$\begin{aligned}\boldsymbol{[\Phi_{ID}]} &= \boldsymbol{w_{ID}(q,\dot{q},y,\Delta_0,[\Delta],[d])}\,, \\ &= \text{IANEA}(\boldsymbol{q,\dot{q},y,[\Delta]}) \ominus \text{IANEA}(\boldsymbol{q,\dot{q},y,\Delta_0})\,. \end{aligned} \quad (12)$$

IANEA, in (12), denotes the interval-arithmetic-based Newton-Euler algorithm [26] and it is used to compute $\boldsymbol{[\Phi_{ID}]}$ with linear computational complexity in the number of assembled robot degrees of freedom. From the set-based operations we have $\boldsymbol{w_{ID}(q,\dot{q},y,\Delta_0,\Delta,d)} \in \boldsymbol{[\Phi_{ID}]}$, and consequently

$$\boldsymbol{\rho_i([\Phi_{ID}])} \geq |\boldsymbol{w_{ID,i}(q,\dot{q},y,\Delta_0,\Delta,d)}| \quad (13)$$

$\forall\, \boldsymbol{q}\,,\, \boldsymbol{\dot{q}}\,,\, \boldsymbol{y}\,, \in \mathbb{R}^{N\times 1}\,, \boldsymbol{\Delta} \in \boldsymbol{[\Delta]}\,,\, \boldsymbol{\Delta_0} \in \boldsymbol{[\Delta]}\,,\, \boldsymbol{d} \in \boldsymbol{[d]}\,.$
From (9), the closed-loop error dynamics can be written as

$$\boldsymbol{\dot{\xi}} = \boldsymbol{A\,\xi} + \boldsymbol{BM^{-1}\nu_{ID}} + \boldsymbol{BM^{-1}w_{ID}}\,, \quad (14)$$

where

$$\boldsymbol{\xi} = \begin{pmatrix}\int \boldsymbol{e}\,dt \\ \boldsymbol{e} \\ \boldsymbol{\dot{e}}\end{pmatrix}, \boldsymbol{A} = \begin{pmatrix}\boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} \\ -\boldsymbol{K_I} & -\boldsymbol{K_P} & -\boldsymbol{K_D}\end{pmatrix}, \boldsymbol{B} = \begin{pmatrix}\boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{I}\end{pmatrix}. \quad (15)$$

By considering a symmetric positive-definite matrix $\boldsymbol{P}$, such that

$$\boldsymbol{A^T P} + \boldsymbol{P\,A} = -\boldsymbol{Q}\,, \quad (16)$$

where $\boldsymbol{Q}$ is a symmetric positive definite matrix, we can use the following continuously differentiable, positive definite function for the proposed controller

$$V = \boldsymbol{\xi^T P \xi}\,. \quad (17)$$

Taking the derivative over time of (17), along the trajectories of the system (14), the following equation is obtained

$$\begin{aligned}\dot{V} &= -\boldsymbol{\xi^T Q\,\xi} + 2\boldsymbol{\xi^T P\,B\,M^{-1}}(\boldsymbol{\nu_{ID}} + \boldsymbol{w_{ID}}) \\ &\leq -\boldsymbol{\xi^T Q\,\xi} + 2\boldsymbol{\xi^T P\,B\,M^{-1}\nu_{ID}} \\ &\quad + 2\|\boldsymbol{\xi^T P\,B}\|\|\,\boldsymbol{M^{-1}}\|\|\boldsymbol{\rho([\Phi_{ID}])}\|\,. \end{aligned} \quad (18)$$

By setting the robustifying term $\boldsymbol{\nu_{ID}}$ as

$$\boldsymbol{\nu_{ID}} = -(\kappa_P\|\boldsymbol{\rho([\Phi])}\| + \varphi_P)\,\boldsymbol{z}\,, \quad (19)$$

with $\boldsymbol{z} = \boldsymbol{B^T\,P\,\xi}$ and $\kappa_P\,,\, \varphi_P \geq 1$, (18) becomes

$$\begin{aligned}\dot{V} &\leq -2\boldsymbol{z^T M^{-1}}\varphi_P\boldsymbol{z} - 2\boldsymbol{z^T M^{-1}}\kappa_P\|\boldsymbol{\rho([\Phi_{ID}])}\|\boldsymbol{z} \\ &\quad + 2\|\boldsymbol{\xi^T P\,B}\|\|\,\boldsymbol{M^{-1}}\|\|\boldsymbol{\rho([\Phi_{ID}])}\| - \boldsymbol{\xi^T Q\,\xi} \\ &\leq -\frac{2}{\lambda_M}\varphi_P\|\boldsymbol{z}\|^2 + 2h_3(\boldsymbol{z}) - \boldsymbol{\xi^T Q\,\xi}\,, \end{aligned} \quad (20)$$

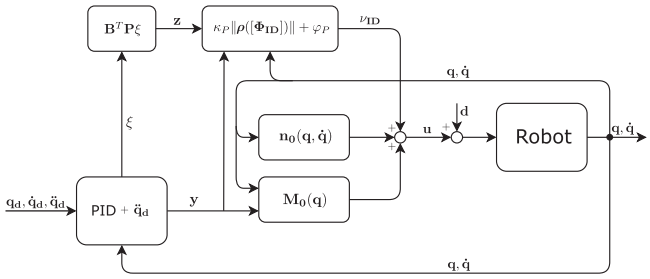


Fig. 2. Block diagram of the robust controller.

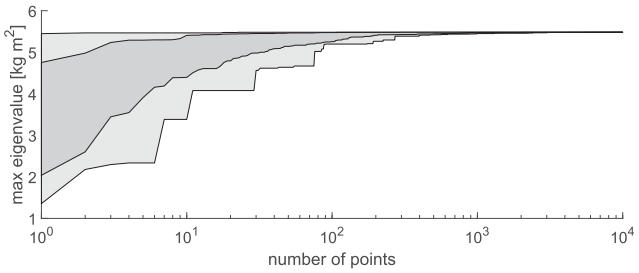


Fig. 3. Overview of 100 Monte Carlo approaches (of 10^4 random vectors each) for the estimation of the maximum eigenvalue of the robot inertia matrix $M_0(q)$. The light gray area contains the top and bottom 10th percentile.

where

$$h_3 = -\frac{1}{\lambda_M}\kappa_P\|\boldsymbol{\rho}([\boldsymbol{\Phi_{ID}}])\|\|\boldsymbol{z}\|^2 + \frac{1}{\lambda_m}\|\boldsymbol{\rho}([\boldsymbol{\Phi_{ID}}])\|\|\boldsymbol{z}\|, \tag{21}$$

and where λ_M and λ_m are the largest and smallest eigenvalues of $\boldsymbol{M}$, respectively. It follows that, for $\|\boldsymbol{z}\| \geq \frac{\lambda_M}{\kappa_P \lambda_m}$, $\dot{V} \leq -\boldsymbol{\xi}^T \boldsymbol{Q}\, \boldsymbol{\xi}$ since $h_3(\boldsymbol{z}) \leq 0$.

Then, since[2] $\|\boldsymbol{\xi}\| \leq \frac{1}{\sigma_{min}(\boldsymbol{B^T P})}\|\boldsymbol{z}\|$ and considering the result in [27, Chapter 4], we can conclude that robust stability is achieved with the trajectories $\boldsymbol{\xi}$ ultimately bounded by

$$\|\boldsymbol{\xi}\| \leq \frac{\lambda_M}{\kappa_P \lambda_m \sigma_{min}(\boldsymbol{B^T P})}\sqrt{\frac{\lambda_{max}(\boldsymbol{P})}{\lambda_{min}(\boldsymbol{P})}}\,, \tag{22}$$

where $\lambda_{max}(\cdot)$ and $\lambda_{min}(\cdot)$ represent the maximum and minimum eigenvalue of the corresponding matrix.

The block diagram of this control scheme is shown in Fig. 2. It should be noted that, contrary to the PID gains, the robust term is not scaled with respect to the link inertia values. Having the same gain on all the joints could not be convenient in practice, especially when there could be a substantial difference among the inertia values. This could either lead to a degraded tracking performance or to an excessive commanded torque at joints with smaller inertia values. An effective solution to this problem consists in choosing the diagonal terms of $\mathbf{Q}$ in (16) proportional to the diagonal values of $\mathbf{M_0}$. This has a direct effect on the $\mathbf{P}$ matrix to scale the robust term differently among each joint.

B. Initialization of Control Parameters

To enable the automatic deployment, we propose to compute first the parameters of the PID (or PD) inverse-dynamics (ID) controller in a conservative way, and then use an automatic time-varying tuning of the parameters governing the robust control term $\boldsymbol{\nu_{ID}}$ (19).

The parameters of the ID controller need to be conservative with respect to the possible model errors and to the sampling rate of the system. To choose the parameters of the PID part of the controller, we aim to avoid high-frequency components in the motor current signal (and thus in the generated torque) due to the noise amplification that may lead to chattering.

From (6) and (7), the effect of $\dot{\boldsymbol{e}}$ on $\boldsymbol{u}$ is only influenced by the terms $\boldsymbol{M_0}(\boldsymbol{q})$ and $\boldsymbol{K_D}$. By setting a maximum allowed noise on the commanded torque, we can find a value for the $\boldsymbol{K_D}$ gain that respects that limit. If $\boldsymbol{M_0}$ were a constant value, the maximum eigenvalue λ_M could be considered for the design of $\boldsymbol{K_D}$. In fact, given a vector $\boldsymbol{x}$, the following holds [28, Chapter 2]

$$\lambda_m\|\boldsymbol{x}\|^2 \leq \boldsymbol{x}^T \boldsymbol{M_0}\, \boldsymbol{x} \leq \lambda_M\|\boldsymbol{x}\|^2\,, \forall \boldsymbol{x} \in \mathbb{R}^{N\times 1}\,. \tag{23}$$

However, being $\boldsymbol{M_0}(\boldsymbol{q})$ function of the joint positions, in order to be conservative, the worst case scenario among the possible robot positions must be considered. Given that $\boldsymbol{M_0}(\boldsymbol{q})$ is nonlinear with respect to $\boldsymbol{q}$, a Monte Carlo approach can be used. This consists in generating a series of random values for $\boldsymbol{q}$ and computing their corresponding λ_M values. Each time a new set of joint states is generated, the maximum value of λ_M is kept. Fig. 3 shows the case for a 6 degrees of freedom robot, where 100 Monte Carlo simulations of 10000 runs each are used to highlight the converge rate towards the maximum value of λ_M. It can be noted that already after 1000 random samples a reasonable convergence for the maximum value of λ_M has been reached for all the 100 Monte Carlo. Given a maximum acceptable noise standard deviation $\sigma_{u_{max}}$ on the generated torque, and given the standard deviation of the noise $\sigma_{\dot{e}}$ on the velocity measurements, the value of $\boldsymbol{K_D}$ is automatically computed as follows

$$\boldsymbol{K_D} = \frac{\sigma_{u_{max}}}{\sigma_{\dot{e}} \cdot \eta \cdot \lambda_{M_{max}}} \cdot \boldsymbol{I}\,, \tag{24}$$

where $\eta > 1$ is a constant value included to add conservativeness on the obtained maximum value of λ_M. This operation is fast (it requires a few seconds on a common computer for a 6 degrees-of-freedom robot assembly) and can be easily performed each time a new robot assembly is configured.

C. Automatic Regulation of the Robust Control Parameters

Once the ID-controller parameters are obtained, it remains the problem of tuning the parameters κ_P, φ_P of the robust control term $\boldsymbol{\nu_{ID}}$ (19), which is also the main practical limitation of the approach in [23] when considering automatic deployment. While robust stability must always be guaranteed, the term $\boldsymbol{\nu_{ID}}$ can also be exploited to improve the overall control tracking performance. The basic idea behind our approach consists in looking at the power spectral density (PSD) of the noise on the requested torque by the robust controller. In practice, if the noise effect on the commanded torque is too high, then the robust control term parameters should be reduced, whereas, if

[2] $\sigma_{min}(B^T P)$ denotes the smallest singular value of $B^T P$.

there is still significant margin available, these parameters can be increased to improve the tracking performance. In order to discard the effect of the noise-free signal in the noise analysis, only the high-frequency part of the PSD has been considered.

For the tuning of the robust $\boldsymbol{\nu_{ID}}$ term one could try to optimize both the κ_P and φ_P parameters. However, it should be noted that we have guaranteed robust stability for any $\kappa_P \geq 1$ and $\varphi_P \geq 1$. Therefore, to make the optimization simpler and more effective, it is possible to optimize the overall gain effect of the two terms, that is

$$\kappa_P \|\boldsymbol{\rho}([\boldsymbol{\Phi}])\| + \varphi_P = \kappa_C \,, \tag{25}$$

thus a single parameter κ_C can be optimized, and the overall robustifying term $\boldsymbol{\nu_{IC}}$ can be substituted by

$$\boldsymbol{\nu_{IC}} = \kappa_C \, \boldsymbol{z} \,, \tag{26}$$

where, in order to maintain the robust stability property, the following constraint must be imposed

$$\kappa_C \geq 1 + \|\boldsymbol{\rho}([\boldsymbol{\Phi}])\| \,, \tag{27}$$

that is

$$\kappa_C \geq (\varphi_P + \kappa_P \|\boldsymbol{\rho}([\boldsymbol{\Phi}])\|) \,, \quad \text{with} \quad \kappa_P = \varphi_P = 1 \,. \tag{28}$$

The first step for the analysis, and subsequent tuning of the parameters, consists in creating a batch of data of the commanded torques by the robust control terms $\boldsymbol{u_r} = \kappa_C \boldsymbol{z}$. The size of this batch should be large enough to enable an accurate computation of the PSD, while also small enough to guarantee a responsive tuning of the parameters due to the undesirable delay inherent in the creation of the batch. An Hamming window is applied on the obtained batch, $\boldsymbol{u_r}$. The Hamming window coefficients have been generated as follows

$$w_{hamm}(n) = 0.54 - 0.46 \cos\left(2\,\pi \frac{n}{N}\right) \quad \forall\, 0 \leq n \leq N \,, \tag{29}$$

where n, N, $\in \mathbb{N}$ and $N = L - 1$ where L is the window length. The PSD is then computed via a fast Fourier transform (FFT) [29], $\boldsymbol{S_{xx}} = \frac{\boldsymbol{s_f} \cdot \overline{\boldsymbol{s_f}}}{\Delta f}$, where $\boldsymbol{S_{xx}}$ is the PSD vector, $\boldsymbol{s_f}$ is the FFT vector, $\overline{\boldsymbol{s_f}}$ is its complex conjugate, and Δf is the frequency bin width of the FFT. If the window is large enough to contain low-frequencies components that include the noise-free signal, $\boldsymbol{S_{xx}}$ is truncated and only the frequencies above a threshold are kept.

With the considered settings, the mean of the PSD (PSD_m) can be directed linked to the noise standard deviation

$$\text{PSD}_m = 2\sigma_u^2 \cdot T_s \,, \tag{30}$$

where T_s is the sampling time and σ_u is the noise standard deviation on the generated torque. We can therefore impose a desired maximum noise on the generated torque (or equivalently on the actuator current), and aim at reaching that amplification. In fact, if the noise amplification is lower it means that we can still allow for higher gain and further improve the tracking performance, meanwhile, if the noise is higher we may face reduced performance due to chattering.

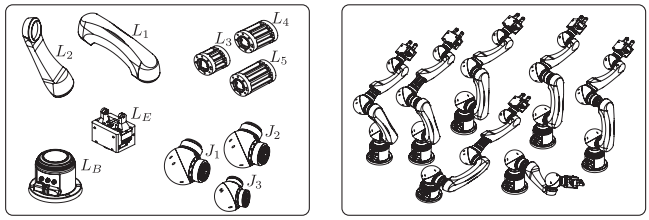
The following algorithm is used for the controller parameter update, where a hysteresis is introduced by setting two values for the desired σ_u: a lower and an upper value $\sigma_{u,min}$, $\sigma_{u,max}$ (and equivalently, from (30), a PSD_{min} and a PSD_{max}):

$$\kappa_{C_j} = \begin{cases} \kappa_{C_{j-1}} \,, & \text{if } \text{PSD}_{min} < \max(\text{PSD}_m) < \text{PSD}_{max}, \\ \kappa_{C_{j-1}} \left(1 + \lambda \frac{\text{PSD}_t - \max(\text{PSD}_m)}{\text{PSD}_t}\right) \,, & \text{otherwise}, \end{cases} \tag{31}$$

with λ being a gain parameter controlling the amount of change per iteration, j is the number of iteration step, $\text{PSD}_t = (\text{PSD}_{max} + \text{PSD}_{min})/2$, and where the "max" operator is used to select the highest mean PSD among the different joints. A minimum bound to the value of κ_C, (27), is imposed to guarantee the robust stability.

Given that the function $\boldsymbol{\rho}([\boldsymbol{\Phi}])$ is changing with respect to the joint states, and it may assume very low values at low velocities, the tuning operation is temporarily halted when the robot is moving very slowly or stopped.

In order to avoid issues due to the limited sampling frequency, an upper bound for the gains is also imposed. A rule of thumb is to have at least 10 sampling periods per rise time (t_{rise}) of the closed-loop step response, or alternatively, have the bandwidth of the system $1/30$ of the sampling frequency [30, Chapter 3]. Given that the sampling rate is fixed by the controller hardware, we can compute the maximum natural frequency of the decoupled closed-loop error dynamics (and consequently the maximum control gains).



(a) (b)

Fig. 4. Overview of the main modules (a) and a subset of the possible robot configurations (b).

IV. Simulation Results

The proposed robust control approach has been tested via simulations. The MRR manipulators of [31, suppl. materials] are considered as the main system. A total of 6 different robot configurations are analyzed, ranging from 4 to 6 joints (see overview in Fig. 4).

Using the modules described in Fig. 4(a) as a reference, the 6 assemblies have been configured as follows

assembly 1: $[L_B \,,\, J_1 \,,\, L_1 \,,\, J_2]$,

assembly 2: $[L_B \,,\, J_1 \,,\, L_1 \,,\, J_2 \,,\, L_2 \,,\, J_3]$,

assembly 3: $[L_B \,,\, J_1 \,,\, L_1 \,,\, J_2 \,,\, L_2 \,,\, J_3 \,,\, L_E]$,

assembly 4: $[L_B \,,\, J_1 \,,\, L_1 \,,\, J_2 \,,\, L_4 \,,\, L_2 \,,\, J_3 \,,\, L_E]$,

assembly 5: $[L_B \,,\, J_1 \,,\, L_1 \,,\, J_2 \,,\, L_5 \,,\, L_2 \,,\, J_3 \,,\, L_E]$,

assembly 6: $[L_B \,,\, J_1 \,,\, L_1 \,,\, L_3 \,,\, J_2 \,,\, L_5 \,,\, L_2 \,,\, J_3 \,,\, L_E]$.

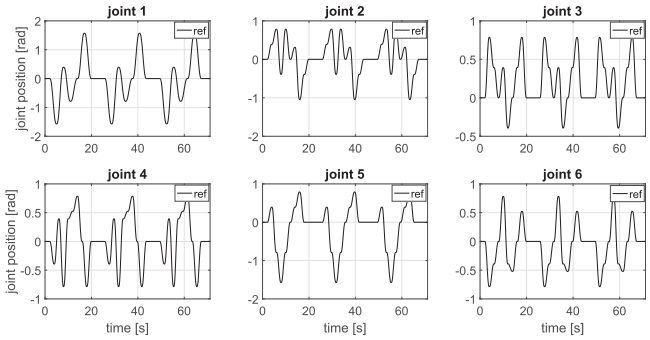


Fig. 5. Reference trajectory for the joint positions (6-degrees of freedom configuration case).

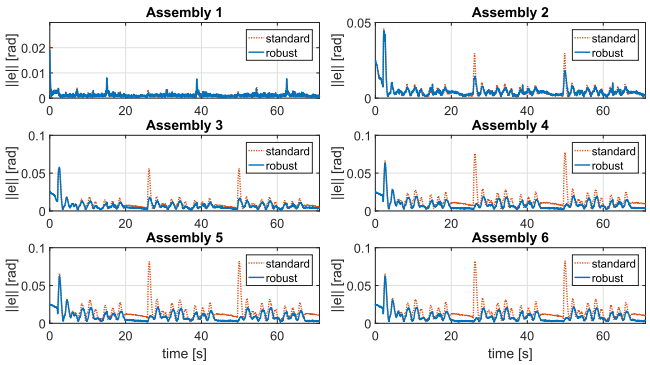


Fig. 6. Comparison of the tracking errors for each of the 6 assemblies.

The joint sensors and controller sampling frequencies were set at $500\,Hz$. The noise standard deviation on the joint position and velocity were set to $\sigma_q = 4 \cdot 10^{-4}\,rad$ and $\sigma_{\dot{q}} = 0.03\,rad/s$, respectively. For the robot, a model error of 4% was introduced for the masses and inertias, of 2% for the centers of mass, and of 8% for the joint friction coefficients. For the interval arithmetic, the model used uncertainties of $\pm 5\%$, $\pm 2.5\%$ and $\pm 10\%$ for the same quantities, respectively. A reference trajectory (see e.g., Fig. 5) has been used as input to the system. For the automatic tuning of the robust control parameter κ_C, the standard deviations on the torque (from which the corresponding PSD value can be computed) have been set as $\sigma_{u,min} = 2.4\,Nm$ and $\sigma_{u,max} = 2.8\,Nm$. The window size for the FFT was set to 128 samples, and $\lambda = 0.02$ was used for the control gain parameter update. The $\boldsymbol{K_D}$ parameter has been computed automatically from the conservative approach described in Section III, where a single Monte Carlo with 1000 random vectors has been used to compute the maximum λ_M, $\eta = 1.1$ is used for (24), whereas $\boldsymbol{K_P}$ and $\boldsymbol{K_I}$ have been set in order to have 3 overlapping poles for the decoupled closed-loop error dynamics. The robust term $\boldsymbol{\nu_{IC}}$ has been initialized with $\kappa_C = 40$, a relatively small value that allows to show the automatic tuning process.

Firstly, the robust controller has been tested in standard conditions for the 6 different assemblies. The tracking performance is compared with the one obtained from the inverse-dynamics controller (without the robust term). These results are shown in Fig. 6, where for sake of compactness the norm among the several joint errors is shown. The automatic design was able to guarantee a good tracking performance for all the assemblies, and the proposed robust controller outperformed the standard approach.

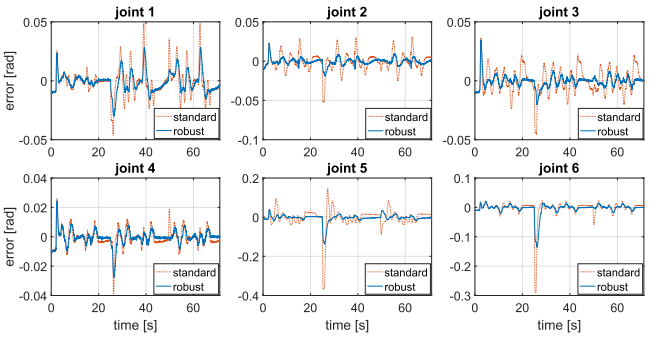


Fig. 7. Tracking error of the joints state for the standard PID and robust controller. External disturbance added at $25\,s$.

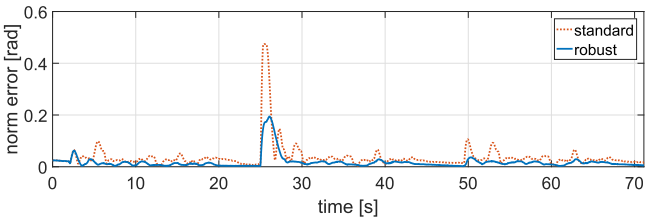


Fig. 8. Norm of the tracking errors for the standard and proposed robust controllers.

An assembly with 6 degrees of freedom (assembly n°5 of [31, suppl. materials]) has been further used to test the robust controller in more complex scenarios, however these results can be extended for any of the robot configurations. An additional mass of $2\,kg$ was added at the end effector, without updating the controller model, in order to simulate the presence of an unknown payload. Moreover, after $25\,s$ in the simulation a constant disturbance torque of $[14,\ 14,\ 10,\ 10,\ 8,\ 8]\,Nm$ was added to the joints (from 1-st to 6-th joint, respectively).

The proposed approach has been compared again to the inverse-dynamics controller. The tracking performance of the two different approaches is highlighted in Fig. 7, where the joint tracking errors are shown. The robust controller presents a significantly better tracking, especially for joint 5 and 6. It can also be noted how the robust controller, initially, behaves similar to the PID controller since κ_C has been initialized with a very small value. However, in a few seconds the algorithm is able to improve the performance. Fig. 8 shows the overall norm of the tracking error for the two controllers, and it highlights how the robust control term is also able to quickly compensate for the external disturbance torque introduced at $25\,s$.

The value of $\max(\text{PSD}_m)$ and of κ_C are shown in Fig. 9, where the automatic tuning allows for maintaining the maximum of the PSD_m between the desired maximum and minimum bounds (black dashed lines). The second plot also highlights the minimum robust term to guarantee stability for the current model uncertainties $(1 + \|\rho\|)$ and when the model errors are increased by 10 times $(1 + \|\rho_{\times 10}\|)$. In the case under consideration, as shown by the values of κ_C and $\kappa_{C\times 10}$ (the latter for the larger

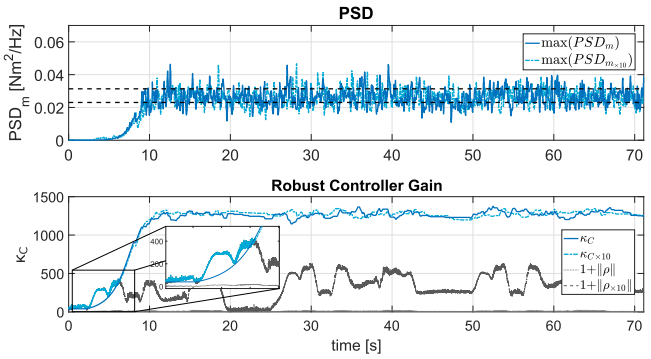


Fig. 9. Max of the average PSD (PSD_m) through the experiment (first plot). Value of κ_C computed and used by the robust controller (second plot).

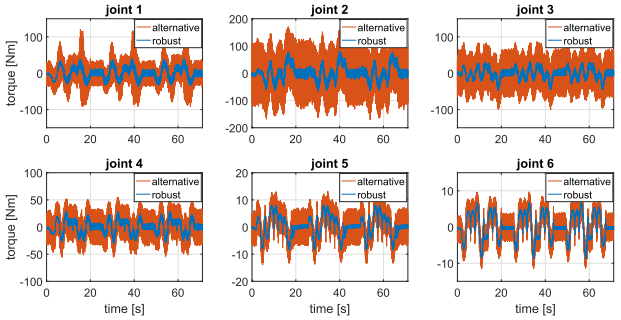


Fig. 10. Torque comparison between the proposed robust controller (*"robust"*) and the controller presented in [32] (*"alternative"*).

model bounds case), there is still a significant margin in the interval arithmetic bounds, thus negligibly affecting the robust gain.

Finally, the proposed algorithm has been compared to an alternative robust controller for robot manipulators presented in the literature [32] which contains a revision of a classical robust control approach for robot manipulators. The same simulation settings used in assembly n° 5 of Fig. 6 have been considered. The results are shown in Fig. 10. The robust parameters for the alternative controller have been chosen as described in [32]. However, the method is much more conservative compared to our proposed solution, thus its commanded torques are significantly higher. This can make the alternative approach less practical for real implementation given the much higher chance of chattering behavior and the need for user intervention to ensure proper functionalities at the deployment.

V. Experiments

Our proposed automatically deployable robust control approach has been tested on the Franka Emika robot manipulator in order to validate it via real experiments. The robot is treated as a MRR system with 7 degrees-of-freedom, where the parameters have been derived from [33]. In particular, we decompose each link into virtual joint modules in order to have a representation of a fully modular manipulator, similarly to what has been done in [4] for other robots. Without loss of generality, even if the commercially available robot under consideration is a fixed-structure manipulator, the scenario we considered reflects

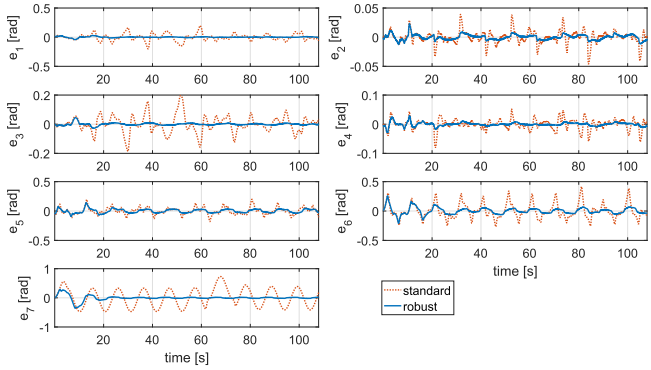


Fig. 11. Comparison of the tracking error between the proposed robust controller and the standard inverse-dynamics control law from the Franka Emika experiment.

a composition of a MRR manipulator setup. The model errors have been set to $\pm 5\%$, $\pm 2.5\%$, $\pm 8\%$ for inertias/masses, centers of mass, and friction coefficients, respectively. The controller sampling frequency is $1\,kHz$, while for the automatic tuning of the robust control gain we set $\lambda = 0.0025$, $\sigma_{u,min} = 0.7\,Nm$, $\sigma_{u,max} = 0.8\,Nm$, and the window size for the FFT has been kept to 128 samples. The following reference trajectory was given to each joint

$$q_r = C_r \cdot \sin^2(\omega_r t) + q(t_0)\,, \tag{32}$$

where each joint axis had different trajectory parameters: $C_r = [0.7,\ 0.7,\ 0.7,\ 0.7,\ 0.8,\ 0.8,\ 0.8]$, $\omega_r = [0.22,\ 0.15,\ 0.18,\ 0.2,\ 0.28,\ 0.32,\ 0.3]$.

The tracking performance compared to the standard inverse-dynamics controller (without the robust control term) is shown in Fig. 11. The proposed approach, after a similar start, visibly outperforms the standard controller through the automatic tuning of the robust gain that allows for a reduction of the tracking error throughout the experiment. In the standard method, the scaling with $\boldsymbol{M}$ (6) significantly reduces the effect of the PID gains for joint 6 and 7, making it less robust with respect to the friction effects (thus explaining the poor tracking performance for those joints). On the other hand, for the proposed approach, the robust term is able to compensate for the low PID values and a satisfying tracking performance is obtained for all the joints.

The behavior of the automatic parameter regulation technique is shown in Fig. 12. For sake of clarity, only the torques of joint 1 and 3 are shown, since they were the ones that experienced higher frequency components. As highlighted by the zoomed frames, when the high-frequency components increased over a certain threshold the algorithm promptly reduced κ_C, thus avoiding the risk of chattering.

Overall, these results show the feasibility and high versatility of the proposed approach, making it viable for a wide range of robot manipulators (modular or not) where robust control deployment is required to be completely automatic.

VI. Conclusion

An automatically deployable robust control approach for MRR manipulators has been presented and successfully applied

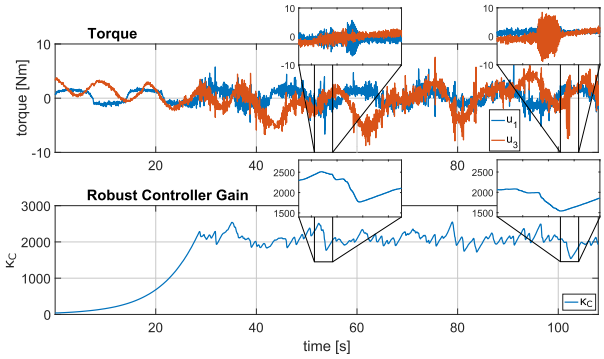


Fig. 12. Commanded torque for joint 1 and 3, and gain κ_C of the robust controller during the experiment.

both in simulations and on a real robot arm. The proposed algorithm includes an automatic control design based on MRR manipulator models with the advantage of not requiring any user intervention with respect to other classical techniques. This allows for a full exploitation of the flexibility and reconfigurability properties of MRR systems. Our approach includes the online regulation of the control parameters from a PSD analysis of the control signal. As shown in the simulation results and the experiments, the online regulation is able to promptly modify the gains in order to avoid chattering, yet it maximizes the tracking performance.

REFERENCES

[1] M. Yim, K. Roufas, D. Duff, Y. Zhang, C. Eldershaw, and S. Homans, "Modular reconfigurable robots in space applications," *Auton. Robots*, vol. 14, no. 2, pp. 225–237, 2003.

[2] C. Wright *et al.*, "Design and architecture of the unified modular snake robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 4347–4354.

[3] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi, "An atlas of physical human-robot interaction," *Mechanism Mach. Theory*, vol. 43, no. 3, pp. 253–270, 2008.

[4] S. B. Liu and M. Althoff, "Optimizing performance in automation through modular robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 4044–4050.

[5] C. Paredis, H.B. Brown, and P. Khosla, "A rapidly deployable manipulator system," in *Proc. Int. Conf. Robot. Automat.*, vol. 2, Apr., 1996, pp. 1434–1439.

[6] T. Matsumaru, "Design and control of the modular robot system: TOMMS," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1995, vol. 2, pp. 2125–2131.

[7] R. Hui *et al.*, "Design of the IRIS facility-A modular, reconfigurable and expandable robot test bed," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1993, pp. 155–160.

[8] M. Yim, D. G. Duff, and K. D. Roufas, "Polybot: A modular reconfigurable robot," in *Proc. IEEE Int. Conf. Robot. Automat. Symp. Proc.*, 2000, vol. 1, pp. 514–520.

[9] G. Liu, Y. Liu, and A. A. Goldenberg, "Design, analysis, and control of a spring-assisted modular and reconfigurable robot," *IEEE/ASME Trans. Mechatronics*, vol. 16, no. 4, pp. 695–706, Aug. 2011.

[10] G. Liu, S. Abdul, and A. A. Goldenberg, "Distributed modular and reconfigurable robot control with torque sensing," in *Proc. IEEE Int. Conf. Mechatronics Automat.*, 2006, pp. 384–389.

[11] W.-H. Zhu, T. Lamarche, E. Dupuis, D. Jameux, P. Barnard, and G. Liu, "Precision control of modular robot manipulators: The VDC approach with embedded FPGA," *IEEE Trans. Robot.*, vol. 29, no. 5, pp. 1162–1179, Oct. 2013.

[12] M. Zhu and Y. Li, "Decentralized adaptive fuzzy control for reconfigurable manipulators," in *Proc. IEEE Conf. Robot., Automat. Mechatronics*, 2008, pp. 404–409.

[13] W. W. Melek and A. A. Goldenberg, "Neurofuzzy control of modular and reconfigurable robots," *IEEE/ASME Trans. Mechatronics*, vol. 8, no. 3, pp. 381–389, Sep. 2003.

[14] Z. Li, W. W. Melek, and C. M. Clark, "Decentralized robust control of robot manipulators with harmonic drive transmission and application to modular and reconfigurable serial arms," *Robotica*, vol. 27, no. 2, pp. 291–302, 2009.

[15] D. Wang, A. A. Goldenberg, and G. Liu, "Development of control system architecture for modular and re-configurable robot manipulators," in *Proc. IEEE Int. Conf. Mechatronics Automat.*, 2007, pp. 20–25.

[16] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. New York, NY, USA: Springer Science & Business Media, 2010.

[17] A. De Luca and R. Mattone, "Actuator failure detection and isolation using generalized momenta," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2003, vol. 1, pp. 634–639.

[18] A. Giusti and M. Althoff, "Automatic centralized controller design for modular and reconfigurable robot manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 3268–3275.

[19] A. Giusti and M. Althoff, "On-the-fly control design of modular robot manipulators," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 4, pp. 1484–1491, Jul. 2018.

[20] I.-M. Chen and G. Yang, "Automatic model generation for modular reconfigurable robot dynamics," *J. Dyn. Syst., Meas., Control*, vol. 120, pp. 346–352, 1998.

[21] E. Meister, A. Gutenkunst, and P. Levi, "Dynamics and control of modular and self-reconfigurable robotic systems," *Int. J. Adv. Intell. Syst.*, vol. 6, no. 1, pp. 66–78, 2013.

[22] C. Nainer, M. Feder, and A. Giusti, "Automatic generation of kinematics and dynamics model descriptions for modular reconfigurable robot manipulators," in *Proc. IEEE 17th Int. Conf. Automat. Sci. Eng.*, 2021, pp. 45–52.

[23] A. Giusti, S. B. Liu, and M. Althoff, "Interval-arithmetic-based robust control of fully actuated mechanical systems," *IEEE Trans. Control Syst. Technol.*, to be published, doi: 10.1109/TCST.2021.3118488.

[24] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to Interval Analysis*. Philadelphia, PA, USA: SIAM, 2009.

[25] M. Althoff and D. Grebenyuk, "Implementation of interval arithmetic in CORA 2016," in *Proc. 3rd Int. Workshop Appl. Verification Continuous Hybrid Syst.*, 2016, pp. 91–105.

[26] A. Giusti and M. Althoff, "Efficient computation of interval-arithmetic-based robust controllers for rigid robots," in *Proc. 1st IEEE Int. Conf. Robotic Comput.*, 2017, pp. 129–135.

[27] H. K. Khalil, *Nonlinear Systems*. Englewood Cliffs, NJ, USA: Prentice Hall, 2002.

[28] R. Ortega, J. A. L. Perez, P. J. Nicklasson, and H. J. Sira-Ramirez, *Passivity-Based Control of Euler-Lagrange Systems: Mechanical, Electrical and Electromechanical Applications*. New York, NY, USA: Springer Science & Business Media, 2013.

[29] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Englewood Cliffs, NJ, USA: Prentice Hall, 1999.

[30] K. J. Åström and B. Wittenmark, *Computer-Controlled Systems: Theory and Design*. Mineola, New York, USA: Courier Corporation, 2013.

[31] M. Althoff, A. Giusti, S. B. Liu, and A. Pereira, "Effortless creation of safe robots from modules through self-programming and self-verification," *Sci. Robot.*, vol. 4, no. 31, 2019.

[32] L. Bascetta and P. Rocco, "Revising the robust-control design for rigid robot manipulators," *IEEE Trans. Robot.*, vol. 26, no. 1, pp. 180–187, Feb. 2010.

[33] C. Gaz, M. Cognetti, A. Oliva, P. R. Giordano, and A. De Luca, "Dynamic identification of the franka emika panda robot with retrieval of feasible parameters using penalty-based optimization," *IEEE Robot. Automat. Lett.*, vol. 4, no. 4, pp. 4147–4154, Oct. 2019.