Reduced-order model based dynamic tracking for soft manipulators: Data-driven LPV modeling, control design and experimental results

Shijie Li, Tran Anh-Tu Nguyen, Thierry-Marie Guerra, Alexandre Kruszewski

Reduced-Order Model Based Dynamic Tracking for Soft Manipulators: Data-Driven LPV Modeling, Control Design and Experimental Results

Article *in* Control Engineering Practice · July 2023

DOI: 10.1016/j.conengprac.2023.105618

4 authors, including:

Anh-Tu Nguyen
Université Polytechnique Hauts-de-France
140 PUBLICATIONS **2,084** CITATIONS

SEE PROFILE

Thierry-Marie Guerra
Université Polytechnique Hauts-de-France
349 PUBLICATIONS **9,873** CITATIONS

SEE PROFILE

Alexandre Kruszewski
École Centrale de Lille
99 PUBLICATIONS **2,492** CITATIONS

SEE PROFILE

Accepted to IFAC Control Engineering Practice
Published in Emerging Leader in CEP series

Reduced-Order Model Based Dynamic Tracking for Soft Manipulators: Data-Driven LPV Modeling, Control Design and Experimental Results

Shijie Li[a,c], Anh-Tu Nguyen[a,b,*], Thierry-Marie Guerra[a], Alexandre Kruszewski[c]

[a] *LAMIH laboratory, UMR CNRS 8201, Université Polytechnique Hauts-de-France, Valenciennes, France*
[b] *INSA Hauts-de-France, Valenciennes, France*
[c] *DEFROST team, INRIA, University of Lille, Centrale Lille, CRIStAL laboratory, UMR CNRS 9189, Villeneuve d'Ascq, France*

Abstract

We propose a *generic* nonlinear reduced-order tracking control method for elastic soft robots. To this end, a new linear parameter varying (LPV) control framework is developed using the data collected from the soft robots. Specifically, for LPV modeling we first derive a nonlinear robot model, which is *large-scale* by nature, using finite element methods (FEM). Then, a proper orthogonal decomposition (POD) algorithm is used to generate a set of linearized reduced-order models, representing the *local* behaviors of the soft robot at different operating points within the workspace. Via a unified POD projector, not only the order of these linearized models can be significantly reduced but also their mechanical structure and stability properties can be preserved for LPV modeling and control design. Next, using radial basis function (RBF) networks, we propose an iterative training method to build the LPV robot model by interpolating a subset of selected linearized models with a specified interpolation error. For LPV control design, the equivalent-input-disturbance (EID) concept is exploited to develop a dynamic tracking control scheme, which is composed of three core components: feedforward control, disturbance-estimator control and feedback control. The feedforward control is designed to account for the effects of the trajectory reference and the time-varying affine term, issued from the FEM-based model linearization. The disturbance-estimator control is obtained from a generalized proportional integral LPV observer, which also provides the estimated reduced-order states for feedback control. The observer-based feedback control design is reformulated as a convex optimization problem under linear matrix inequality (LMI) constraints. The globally uniformly ℓ_∞ stability of the closed-loop LPV robot model is guaranteed by means of Lyapunov stability theory. Experimental tests are conducted with a soft Trunk robot, inspired by the elephant trunk, under several scenarios with small and large deformations to show the effectiveness of the proposed LPV tracking control framework. A comparative study is also performed with a recent linear EID-based controller and an iterative learning controller to emphasize the interests this nonlinear control method for soft elastic robots. This paper is complemented with a series of demonstration videos: https://bit.ly/3D8C4Vd.

Keywords: Soft robots, LPV control, motion control, data-driven modeling, model order reduction, RBF networks.

1. Introduction

Inspired by natural organisms, soft robots have capabilities in terms of large-scale flexibility, dexterity, compliance and adaptability [1]. These robots have been designed to perform complex tasks in interaction with humans, which involve a high degree of uncertainty, *e.g.*, surgery, assistive medical devices, search and rescue. Up to now, a considerable effort has been devoted to technological developments [2, 3]. As a result, various hardware solutions have been developed to push the boundaries of robot abilities [4]. Based on caterpillar locomotion, soft inchworm robotic platforms have been designed such that they can crawl, inch or roll [5–7]. Using silicone rubbers, soft fish-like robots being able to swim underwater have been fabricated [8–11]. Soft manipulators have also been received a particular attention in soft robotics with the aim to achieve smooth and flexible deformations such as elephant trunks [12–15], octopus tentacles [16, 17], or skeletal spines of vertebrate animals [18, 19]. Soft manipulators can be designed and fabricated with various structures, materials, and actuation technologies, which range from rigid robots with multiple joints [20] to continuum robots actuated by steel cables [21], by shape-memory alloys [22], by pneumatic artificial muscles [13, 23], or silicone soft robots [15, 24].

Despite the technological advance, how to best model and control soft robots is still an open-ended question [25–27]. This is due to the following main factors. First, due to their infinite degrees-of-freedom (DoF) motions, soft robots are formulated as infinite-dimensional systems. Second, such robots exhibit highly nonlinear uncertain dynamics caused by the nonlinear characteristics from material properties and geometrical structures, which are further complicated by imperfect fabrication and actuation-sensing techniques [28]. Hence, in contrast to the rigid case, developing reliable models in soft robotics remains

*Corresponding author.
Email addresses: shijie.li@uphf.fr (Shijie Li), tnguyen@uphf.fr (Anh-Tu Nguyen), guerra@uphf.fr (Thierry-Marie Guerra), alexandre.kruszewski@univ-lille.fr (Alexandre Kruszewski)

challenging for both simulation and control design purposes [25]. With appropriate assumptions and approximations, simulation models can be derived to reproduce accurately the nonlinear robot behaviors, for instance using finite element methods (FEM). However, they are generally not suitable for control design because of the involved computational burden and especially their *large-scale* nature [29, 30]. Therefore, a generic methodology to model and control soft robots, which can account for a large variability of technological solutions, is one of the most exciting research challenges in soft robotics.

Static input-output relationship has been exploited to build piecewise constant curvature (PCC) models at early stage of soft robots control [21]. For PCC modeling, the soft robot is represented as a set of circular arcs, then only bending behaviors can be taken into account. Since the relationship between constant curvature sections can be described by a homogeneous transformation matrix [31], PCC-based control methods are mostly suitable for multi-section soft manipulators. Similar to the case of rigid manipulators, inverse kinematics can be obtained from PCC models to perform a closed-loop control using the curvature measurements given by internal sensors or cameras [32]. To enlarge the application scope of PCC-based formulation, a dynamical model obtained by connecting the soft robot to an equivalent augmented rigid robot has been proposed in [31] for dynamic tracking control of planar soft robots. Some other PCC-based extensions, *e.g.*, polynomial curvature modeling, have been discussed in [33]. Note that PCC-based modeling method is inherently limited to beam-like robots, which seems difficult to be adapted to a wide range of soft robot structures [34]. To overcome the difficulties in obtaining high-quality models for soft robots, data-driven and machine learning approaches have been also developed [25]. Using experimental data, these approaches directly identify the robot input-output relationship to derive accurate kinematic models for open-loop control of soft robots [26, 32]. However, open-loop control suffers a lack of performance and robustness in presence of modeling uncertainty or disturbances. With the recent success of deep learning, reinforcement learning (RL) has also been applied to soft robots for task-oriented control, *e.g.*, [35] with a deep-Q learning algorithm, [36] with a trust region policy optimization (TRPO) algorithm, [37] with an episode-based RL algorithm. For RL algorithms, the policy function is learnt during the training process, which is used for both challenging objectives in soft robotics, *i.e.*, motion planning and control. Although the structures of these RL controllers are relatively simple, the parameter convergence is still slow as for rigid robots [36]. It is emphasized that these RL-based methods do not account the mechanical properties, *e.g.*, stiffness and deformation, and the load influence of soft robots, which may limit the high-control performance in terms of rapidity, robustness, and precision, especially for dexterous control tasks.

Model-based control has been long considered unfeasible for soft robots due to the large variability of hardware solutions and the related overcomplex modeling [27]. However, the relevance of model-based strategies has been recently proved thanks to various finite-dimensional modeling techniques such as finite-element methods (FEM), which can provide accurate and tractable models [38]. Moreover, although the design of feedback schemes is essentially based on simplified soft robot dynamics, closed-loop feedback control can already outperform model-free counterparts, especially in terms of robustness. Two notable modeling approaches to obtain dynamical models of soft robots under actuation and deformation can be mentioned: i) discretized Cosserat modeling, ii) finite element modeling. Cosserat rod theories take into account possible rod-deformation modes, *e.g.*, bending, twisting, shearing, stretching, under a wide range of boundary conditions [39]. The continuous Cosserat method represents the dynamics of soft robots by continuously stacking an infinite number of microsolids, leading to infinite-dimensional robot models formulated as partial differential equations (PDEs). Although this method can accurately describe continuum structures in the presence of external loads and different actuation methods [40–42], it is hard to directly leverage PDEs formulations for control. Discretization of Cosserat models has been proposed in [43] for multisection soft manipulator dynamics under the assumption of piecewise constant strain (PCS) along the soft arm. Inheriting the advantages of the continuous Cosserat modeling, the discretized Cosserat method can preserve the geometrical and mechanical properties such as intrinsic parameterization and greater generality. However, Cosserat-rod modeling is mainly relevant to describe soft robots with rod-like structures. Moreover, a fine discretization of Cosserat-rod models results in a drastic increase of computational burden. Alternatively, the finite element method is a remarkable solution for soft robot modeling, which is based on the discretization of a complex geometric robot shape into a finite number of smaller and simpler elements. Since the first FEM-based modeling and control result in [44], various extensions have been successfully developed for soft robots, *e.g.*, collision handling [30], inflatable deformations [45], force sensing [46], etc. FEM-based methods can provide not only high-quality models but also a great flexibility to deal with a large variability of soft robot structures. Moreover, while simulating robot deformations, FEM modeling allows introducing some physical effects so that actuation methods, including magnetic field force and thermal strain, piezoelectric effect, and frictional force, can be integrated [47]. Note that FEM-based modeling has laid a solid foundation for recent theoretical and technical developments related to the SOFA open-source simulation platform [45], which has proved the feasibility and the relevance of model-based strategies for soft robots.

Based on FEM models, both *quasi-static* and *dynamic* control methods can be designed for soft robots. Quasi-static control exploits the Jacobian matrix, established under static equilibrium, between the control input and the displacement of the robot end-effector. An extra synchronization step is required during each sampling step to update the soft robot model, making it quasi-static. Based on quadratic programming (QP) optimizations, a control input corresponding to desired displacements is computed under possible constraints and contact conditions [45]. Since the soft robot dynamics are ignored, quasi-static control performance can be limited, especially for highly dynamic tasks. FEM-based dynamic feedback control is substantially more challenging. Indeed, the quality of FEM models

highly depends on the density of the mesh to be performed for the geometric robot shape. Hence, a reliable soft robot model requires a huge number of finite elements, *i.e.*, states, which makes classical control results unapplicable. Therefore, model-order reduction is essential for dynamic feedback control design of such large-scale robot models [29]. Note that the design of FEM-based dynamic controllers is still mainly relied on linearized models, obtained from a linearization around either a local equilibrium [15, 48] or a desired trajectory [49]. Then, the effectiveness of these dynamic feedback controllers is generally limited to a small range of the workspace due to the large nonlinearity of soft robots. The primary goal of this paper is to propose a *generic* solution for this critical issue in soft robotics via a model-based nonlinear dynamic control scheme.

The proposed nonlinear control framework for soft robots is sequentially performed in two steps: i) linear parameter varying (LPV) modeling, and ii) tracking LPV control design. For LPV modeling, we first derive the nonlinear FEM model of the soft robot, which is not relevant for control design due to its large-scale and nonlinear nature. Then, linearization is performed at several equilibrium points, densely selected to cover the whole workspace of the soft robot. From the resulting large-scale models, a proper orthogonal decomposition (POD) algorithm is used to generate a set of linearized reduced-order models, which represent the *local* behaviors of the soft robot corresponding to the selected equilibrium points. Using a unified POD projector, not only the order of these linearized models can be significantly reduced but also their mechanical structure and stability properties can be preserved. With such a property preservation, the *coherence* between the linearized models is naturally ensured, which is essential for interpolation-based LPV modeling [50, 51]. For LPV interpolation, due to the large number of linearized models and their correlation with each other, we propose a *decorrelation* algorithm to limit the numerical complexity of the LPV model, *i.e.*, number of local linear submodels, while guaranteeing a high-quality LPV modeling. To this end, we inspire the covariance projection regression (CPR) algorithm [52] to develop an iterative training method, which builds the LPV model of the soft robot by interpolating a subset of selected linearized models with help of radial basis function (RBF) networks. Concerning the LPV tracking control scheme, its design is based on the equivalent-input-disturbance (EID) control concept [53], whose core components are feedforward control, disturbance-estimator control and feedback control. The feedforward control is used to deal with the effects of the trajectory reference and the time-varying affine term, stemmed from the FEM-based model linearization. A generalized proportional integral LPV observer is designed to provide not only the estimate of reduced-order states for feedback control but also the disturbance-estimator control action to compensate *matched* disturbances, *e.g.*, modeling errors, external loads. The closed-loop stability of the soft robot system is guaranteed by the feedback control, whose design is reformulated as a convex optimization problem under linear matrix inequality (LMI) constraints. Using Lyapunov stability theory, an $\ell_\infty-$gain performance is incorporated in the feedback control design to improve the tracking performance under the presence of unknown and unmatched disturbances. The feasibility of LMI-based design conditions can be easily checked using a suitable semidefinite programming software [54]. The main contributions of this paper can be summarized as follows.

- A constructive LPV modeling method, including FEM-based modeling and discretization, reduced-order reduction and iterative RBF-based interpolation, which is relevant to represent and to control the nonlinear dynamics of soft robots within the whole workspace.
- An ℓ_∞ LPV control framework with formal proofs of closed-loop stability, which requires only the output information to achieve an effective tracking performance under unknown disturbances.
- Due to its generic feature, the proposed LPV control methodology can be applied to a large variability of elastic soft robots as well as other uncertain LPV engineering systems.
- Experimental results and suitable comparative studies are conducted with a soft Trunk robot to fully validate the proposed LPV control framework in terms of nonlinear modeling and tracking performance.

Notation. $\mathbb{N}$ is the set of non-negative integers, and we denote $\mathcal{I}_p = \{1, 2, ..., p\} \subset \mathbb{N}$. For a matrix X, $X^\top$ denotes its transpose, $X \succ 0$ means that X is positive definite, $\mathrm{He}X = X + X^\top$, and $X_{(i)}$ denotes its ith column. $\mathrm{diag}\{X_1, X_2\}$ denotes a block-diagonal matrix composed of X_1, X_2. I is the identity matrix of appropriate dimension. For a vector $x \in \mathbb{R}^n$, we denote its 2-norm as $\|x\| = \sqrt{x^\top x}$. For a sequence of vectors $\{x_k\}_{k\in\mathbb{N}}$, we denote $\|x\|_{\ell_\infty} = \sup_{k\geq 0} \|x_k\|$. Then, $\{x_k\}_{k\in\mathbb{N}} \in \ell_\infty$ if $\|x\|_{\ell_\infty} < \infty$. The symbol "$\star$" stands for the terms deduced by symmetry.

2. LPV Modeling of Soft Robots

This section presents a new LPV modeling method for large-scale soft robots based on their FEM models and the well-known RBF interpolation.

2.1. FEM-Based Reduced-Order Models

FEM technique has been used for soft robots modeling by discretizing the infinite-dimensional model into a finite number of tiny elements. Depending on the choice of FEM methods, each element can have several degrees of freedom [45], yielding large-scale models. A higher element density leads to a higher modeling accuracy with a larger computation burden.

2.1.1. Linearized State-Space Representation

After the FEM discretization, the dynamics of a deformable soft robot can be described as follows [45]:

$$\mathcal{M}(\boldsymbol{q}(t))\dot{\boldsymbol{v}}(t) = \mathcal{P}(\boldsymbol{q}(t)) - \mathcal{F}(\boldsymbol{q}(t), \boldsymbol{v}(t)) + \mathcal{H}(\boldsymbol{q}(t))\boldsymbol{u}(t) \quad (1)$$

where $\boldsymbol{q}(t) \in \mathbb{R}^m$ is the displacement vector, $\boldsymbol{v}(t) = \dot{\boldsymbol{q}}(t)$ is the velocity vector, $\boldsymbol{u}(t) \in \mathbb{R}^p$ is the actuation input, $\mathcal{M}(\boldsymbol{q}) \in$

$\mathbb{R}^{m\times m}$ is the inertia matrix, $\mathcal{F}(\boldsymbol{q},\boldsymbol{v})$ represents the internal forces of the robot described by the constitutive law, $\mathcal{P}(\boldsymbol{q}) \in \mathbb{R}^m$ represents the gravity force, and $\mathcal{H}(\boldsymbol{q}) \in \mathbb{R}^{m\times p}$ is the control input matrix. Assuming that the gravity and the mass distribution does not change over time, then around the equilibrium the matrices $\mathcal{P}(\boldsymbol{q}) = P$ and $\mathcal{M}(\boldsymbol{q}) = M$ are constant.

To simplify the presentation and without loss of generality, we consider the equilibrium point $(\boldsymbol{q}_0, \boldsymbol{v}_0, \boldsymbol{u}_0) \equiv (0,0,0)$ to linearize system (1), *i.e.*,

$$P - \mathcal{F}(0,0) = 0. \tag{2}$$

Moreover, using the Taylor-Young formula, the internal force $\mathcal{F}(\boldsymbol{q},\boldsymbol{v})$ can be approximated around the equilibrium as

$$\mathcal{F}(\boldsymbol{q}(t),\boldsymbol{v}(t)) \approx \mathcal{F}(0,0) + \mathcal{K}(0,0)\boldsymbol{q}(t) + \mathcal{D}(0,0)\boldsymbol{v}(t). \tag{3}$$

The compliance matrix $\mathcal{K}(\boldsymbol{q},\boldsymbol{v})$ and the damping matrix $\mathcal{D}(\boldsymbol{q},\boldsymbol{v})$ are respectively defined as

$$\mathcal{K}(\boldsymbol{q},\boldsymbol{v}) = \frac{\partial \mathcal{F}(\boldsymbol{q},\boldsymbol{v})}{\partial \boldsymbol{q}}, \quad \mathcal{D}(\boldsymbol{q},\boldsymbol{v}) = \frac{\partial \mathcal{F}(\boldsymbol{q},\boldsymbol{v})}{\partial \boldsymbol{v}}. \tag{4}$$

For brevity, we denote $K = \mathcal{K}(0,0)$, $D = \mathcal{D}(0,0), F = \mathcal{F}(0,0)$ and $H = \mathcal{H}(0)$. From (2), (3) and (4), the linearized version of the FEM robot model (1) around the equilibrium point $(\boldsymbol{q}_0, \boldsymbol{v}_0, \boldsymbol{u}_0)$ can be given by [15]

$$M\dot{\boldsymbol{v}}(t) = -K\boldsymbol{q}(t) - D\boldsymbol{v}(t) + H\boldsymbol{u}(t). \tag{5}$$

For real-time control implementation, we perform the control design in the discrete-time domain. For the large-scale model (5), to improve the numerical efficiency at least in terms of numerical methods, the implicit Euler-discretization method can be used [55]. Considering a sampling time step h and a constant control input $\boldsymbol{u}$ during the sampling interval, the implicit Euler-discretization of model (5) is given by

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + h\boldsymbol{v}_{k+1} \tag{6}$$
$$M\boldsymbol{v}_{k+1} = M\boldsymbol{v}_k + h(-K\boldsymbol{q}_{k+1} - D\boldsymbol{v}_{k+1} + H\boldsymbol{u}_k). \tag{7}$$

It follows from (6) and (7) that

$$S\boldsymbol{v}_{k+1} = -hK\boldsymbol{q}_k + M\boldsymbol{v}_k + hH\boldsymbol{u}_k \tag{8}$$

with $S = M + hD + h^2K$. From (6) and (8), the discrete-time robot model can be reformulated as

$$E\begin{bmatrix}\boldsymbol{q}_{k+1}\\ \boldsymbol{v}_{k+1}\end{bmatrix} = \begin{bmatrix} I & 0\\ -hS^{-1}K & S^{-1}M\end{bmatrix}\begin{bmatrix}\boldsymbol{q}_k\\ \boldsymbol{v}_k\end{bmatrix} + \begin{bmatrix}0\\ hS^{-1}H\end{bmatrix}\boldsymbol{u}_k \tag{9}$$

with $E = \begin{bmatrix} I & -hI\\ 0 & I\end{bmatrix}$. Left-multiplying (9) with E^{-1}, the following state-space robot model can be obtained:

$$\begin{bmatrix}\boldsymbol{q}_{k+1}\\ \boldsymbol{v}_{k+1}\end{bmatrix} = \begin{bmatrix} I - h^2S^{-1}K & hI - h^2S^{-1}(D+hK)\\ -hS^{-1}K & I - hS^{-1}(D+hK)\end{bmatrix}\begin{bmatrix}\boldsymbol{q}_k\\ \boldsymbol{v}_k\end{bmatrix} + \begin{bmatrix}h^2S^{-1}H\\ hS^{-1}H\end{bmatrix}\boldsymbol{u}_k. \tag{10}$$

With a small sampling time step h, the impact of the terms related to h^2, *i.e.*, $h^2S^{-1}K$, $h^2S^{-1}H$ and $h^2S^{-1}(D+hK)$, on the modeling quality can be considered as small. Therefore, these terms are removed from (10), which leads to the simplified model

$$\boldsymbol{x}_{L,k+1} = A_L\boldsymbol{x}_{L,k} + B_L\boldsymbol{u}_k \tag{11}$$

where $\boldsymbol{x}_{L,k} = \begin{bmatrix}\boldsymbol{q}_k^\top & \boldsymbol{v}_k^\top\end{bmatrix}^\top \in \mathbb{R}^{2m}$ is the state vector, and the matrices $A_L \in \mathbb{R}^{2m\times 2m}$ and $B_L \in \mathbb{R}^{2m\times p}$ are defined as

$$A_L = \begin{bmatrix} I & hI\\ -hS^{-1}K & I - hS^{-1}D\end{bmatrix}, \quad B_L = \begin{bmatrix}0\\ hS^{-1}H\end{bmatrix}. \tag{12}$$

The system output $\boldsymbol{y}_k \in \mathbb{R}^q$ represents the coordinates of the robot end-effectors, defined as

$$\boldsymbol{y}_k = C_L\boldsymbol{x}_{L,k} \tag{13}$$

where the non-zero elements of the output matrix C_L corresponds to the states of the end-effectors.

Remark 1. Performing the simplification of model (10) allows preserving the mechanical model structure of soft robots, *i.e.*, the matrices A_L and B_L in (12) have specific structures with constant upper-half block-elements. This feature plays a key role not only for the use of EID-based control concept to effectively compensate the unknown uncertainty/disturbance [15], but also for RBF-based LPV modeling, see also Remarks 6 and 7. Moreover, despite this simplification, the dominant nonlinear dynamics of soft robots can be well captured by the proposed modeling method as illustrated in Section 5.2.

Remark 2. Model-based control design using the linearized robot model (11) is challenging due to its large-scale nature. For instance, a high-fidelity FEM modeling of the Trunk robot, studied in Section 5, results in a state-space representation with 4452 states, which makes any optimization-based analysis and control design unrealistic without performing preliminary model reductions [15, 48]. Note also that due to the scattered connection between local neighbor elements in FEM modeling, the state-space matrices A_L, B_L and C_L are also very sparse.

2.1.2. POD-Based Model Order Reduction

Model order reduction (MOR) allows to drastically reduce the dimensions of large-scale systems while keeping their dominant dynamics. Note that for the considered class of elastic soft robots, the analytical description (4) is only used to compute local matrices for each element in the FEM method. The FEM soft robot model is an iterative computational model, which requires to be assembled at each step of the computation. This non-analytical model cannot be used directly to derive a reduced-order model for a soft robot. POD is one of the most popular MOR methods for nonlinear systems due to several advantages [56]. First, this data-driven method does not require any assumption on the size of the original system. Second, this easy-to-implement method enables an effective MOR with *a priori* error bound. Third and more importantly, the POD

method is well suited for complex applications, where extensive high-fidelity simulations are available for data collection, as the case of soft robots simulated under SOFA platform [45]. Indeed, in contrast to other MOR methods, the projectors of POD algorithms are obtained from data collected in the form of snapshots, *i.e.*, responses of the system states with respect to the input excitation signals. Hence, POD-based methods can be directly applied to large-scale systems without an iterative reduction computation. In soft robotics, POD has been shown as an effective MOR method, *e.g.*, linear control design [15, 29] or contact computation [57] for soft robots.

For POD-based model reduction, we first collect the response data of soft robots in the snapshot matrix $M_s \in R^{m\times l}$, where m and l are respectively the number of the robot states and of the time steps. This offline process is the most computational step since we must intensively generate the robot snapshots with various test scenarios. Then, a singular value decomposition is performed to compute the singular values and the corresponding singular vectors of M_s as

$$M_s = U\Sigma V \tag{14}$$

where the matrices $U \in R^{m\times m}$ and $V \in R^{l\times l}$ are orthonormal. The diagonal entries of Σ are the singular values of M_s in a decreasing order. The projector T of the POD method is constructed via the truncation of the matrix U in (14) according to a desired number r of singular values. Let us denote $T = U_r$, where $T \in R^{r\times m}$ contains the first r columns, with $r \ll m$, of the matrix U. Then, the reduced state vector is obtained as [15]

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{q}_r \\ \boldsymbol{v}_r \end{bmatrix} = \begin{bmatrix} T & 0 \\ 0 & T \end{bmatrix} \begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{v} \end{bmatrix} = T_r \boldsymbol{x}_L. \tag{15}$$

Applying the projector (15) to the large-scale linearized system (11)–(13), we obtain the reduced-order model of the form

$$\begin{aligned} \boldsymbol{x}_{k+1} &= A_r \boldsymbol{x}_k + B_r \boldsymbol{u}_k + \boldsymbol{w}_k \\ \boldsymbol{y}_k &= C\boldsymbol{x}_k \end{aligned} \tag{16}$$

where the disturbance $\boldsymbol{w}_k$ represents the MOR error, and

$$\begin{aligned} A_r &= T_r A_L T_r^\top = \begin{bmatrix} I & hI \\ -hTS^{-1}KT^\top & I - hTS^{-1}DT^\top \end{bmatrix} \\ B_r &= T_r B_L = \begin{bmatrix} 0 \\ hTS^{-1}H \end{bmatrix}, \quad C = C_L T_r^\top. \end{aligned} \tag{17}$$

Remark 3. The accuracy of the POD reduced-order models (16) highly depends on how to perform the truncation of the singular values of the snapshots according to the number of reduced states [29]. For illustrations, Figure 1 shows the evolution of the singular values of displacement snapshots of the Trunk robot studied in Section 5. We can observe a fast decay of singular values for the first states, and no significant change of singular values, *i.e.*, accuracy improvement, is achieved with more than 4 states. Hence, with the POD method (14)–(15), we can significantly reduce the number of states from 4452 to 4, which is suitable for the control design of this Trunk robot.

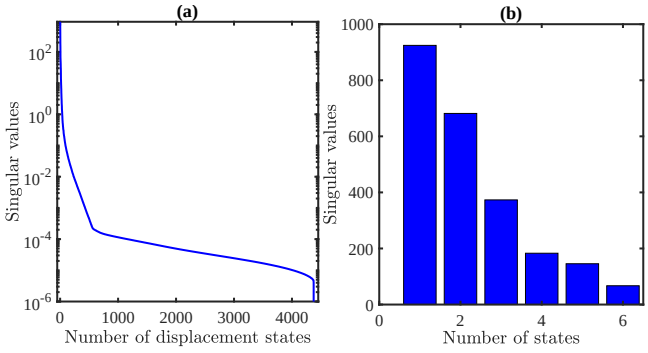


Figure 1: Singular values of snapshots of the Trunk robot studied in Section 5. (a) Evolution of all displacement singular values. (b) Evolution of the first six singular values.

2.2. RBF-Based LPV Modeling

Due to the large nonlinearity of soft robots, a control design based on a linearized robot model (16) may not be effective for different robot configurations. For instance, Figure 2 illustrates the nonlinear behavior of the Trunk robot during a bending process. We can see that increasing the actuation force can lead to different directions of displacement along the $x-$axis. Indeed, the $x-$axis displacement first increases till a certain "singular" position, then it starts to decrease if the force continues to increase. Moreover, the variation of the curve slope shows that the stiffness of the Trunk robot increases with respect to the increment of actuation forces. To deal with such position-dependent nonlinear dynamics of a deformable robot, we propose an LPV modeling method based on a family of local reduced-order models as in (16).

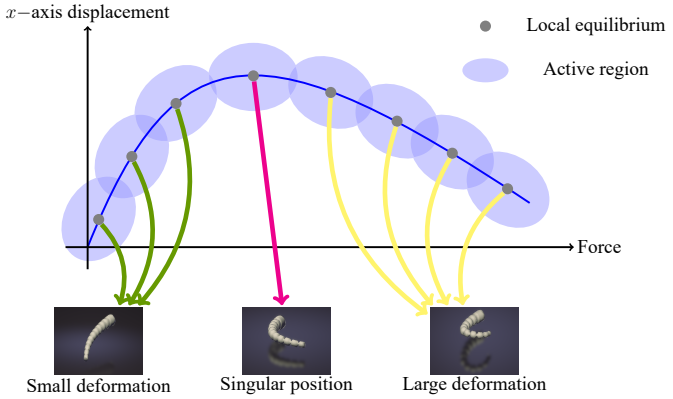


Figure 2: Illustration of nonlinear behaviors of the Trunk robot during bending.

2.2.1. Generating LPV Local Linear Models

To capture the complex dynamics of a soft robot, we select an appropriate number of equilibrium points as illustrated in Figure 3 to cover as large as possible its whole workspace. For each equilibrium point $(\boldsymbol{x}_i^*, \boldsymbol{u}_i^*)$, with $\boldsymbol{x}_i^* = \begin{bmatrix} \boldsymbol{q}_{ri}^{*\top} & \boldsymbol{v}_{ri}^{*\top} \end{bmatrix}^\top$, we first perform the linearization of the FEM model (1) and the implicit Euler-discretization as in (6)–(7). Then, the POD-based order reduction (14)–(15) is applied to obtain the reduced-order

model as in (16):

$$\begin{aligned}\delta \boldsymbol{x}_{i,k+1} &= A_{ri}\delta \boldsymbol{x}_{i,k} + B_{ri}\delta \boldsymbol{u}_{i,k} + \boldsymbol{w}_k \\ \boldsymbol{y}_k &= C(\delta \boldsymbol{x}_{i,k} + \boldsymbol{x}_i^*)\end{aligned} \tag{18}$$

where $\delta \boldsymbol{x}_{i,k} = \boldsymbol{x}_k - \boldsymbol{x}_i^* \in \mathbb{R}^n$, with $n = 2r$, and $\delta \boldsymbol{u}_{i,k} = \boldsymbol{u}_k - \boldsymbol{u}_i^* \in \mathbb{R}^p$. Note that the state $\delta \boldsymbol{x}_{i,k}$ of the ith linearized model (18) is defined as a relative distance between the global coordinate $\boldsymbol{x}_k$ and its local equilibrium $\boldsymbol{x}_i^*$. For simplicity of notation, we use the same disturbance vector $\boldsymbol{w}_k$ to represent the MOR errors in different reduced-order models. Similar to (17), the state-space matrices $A_{ri} \in \mathbb{R}^{n\times n}$ and $B_{ri} \in \mathbb{R}^{n\times m}$ are given by

$$A_{ri} = \begin{bmatrix} I & hI \\ \mathcal{K}_{ri} & \mathcal{D}_{ri} \end{bmatrix}, \quad B_{ri} = \begin{bmatrix} 0 \\ \mathcal{H}_{ri} \end{bmatrix} \tag{19}$$

with

$$\begin{aligned}\mathcal{K}_{ri} &= -hTS_i^{-1}K_iT^\top \\ \mathcal{D}_{ri} &= I - hTS_i^{-1}D_iT^\top \\ \mathcal{H}_{ri} &= hTS_i^{-1}H_i.\end{aligned} \tag{20}$$

The matrices S_i, K_i, D_i and H_i in (20) corresponding to the equilibrium point $(\boldsymbol{x}_i^*, \boldsymbol{u}_i^*)$ are obtained in a similar way as S, K, D and H in (12), respectively. It is important to note that since the same projector T_r in (15) is applied to all linearized submodels, the corresponding reduced models have the same order. Note also that the state $\delta \boldsymbol{x}_{i,k}$ of each linearized model (18) only has a local meaning, *i.e.*, relative distance between $\boldsymbol{x}_k$ and $\boldsymbol{x}_i^*$. Then, we cannot directly interpolate these linearized local models. To deal with this misfit, we reformulate the local model (18) in function of the global state $\boldsymbol{x}_k$ of the robot as

$$\begin{aligned}\boldsymbol{x}_{k+1} &= A_{ri}\boldsymbol{x}_k + B_{ri}\boldsymbol{u}_k + \boldsymbol{\zeta}_{ri} + \boldsymbol{w}_k \\ \boldsymbol{y}_k &= C\boldsymbol{x}_k.\end{aligned} \tag{21}$$

The constant affine term $\boldsymbol{\zeta}_{ri}$ of the ith local model can be defined from the equilibrium point $(\boldsymbol{x}_i^*, \boldsymbol{u}_i^*)$ as

$$\boldsymbol{\zeta}_{ri} = (I - A_{ri})\boldsymbol{x}_i^* - B_{ri}\boldsymbol{u}_i^*. \tag{22}$$

At the equilibrium, we have $\boldsymbol{v}_{ri}^* = 0$. Then, substituting (19) into (22), the term $\boldsymbol{\zeta}_{ri}$ can be reexpressed as

$$\boldsymbol{\zeta}_{ri} = \begin{bmatrix} 0 \\ -\mathcal{K}_{ri}\boldsymbol{q}_{ri}^* - \mathcal{H}_{ri}\boldsymbol{u}_i^* \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{\zeta}_{ri}^* \end{bmatrix} \tag{23}$$

with $\boldsymbol{\zeta}_{ri}^* \in \mathbb{R}^r$.

2.2.2. *RBF-Based Interpolation*

To represent the position-dependent nonlinear dynamics of soft robots, we select the end-effector position as the scheduling vector $\boldsymbol{\theta}$ for LPV modeling, *i.e.*, $\boldsymbol{\theta} = \boldsymbol{y}$. RBF networks allow to construct accurate interpolants of unstructured data, possibly in high-dimensional spaces, with a small number of radial basis functions, which is particularly suitable for nonlinear modeling

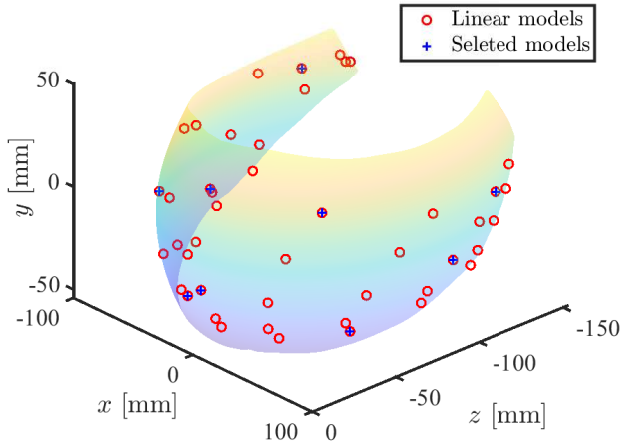


Figure 3: Workspace of the soft Trunk robot studied in Section 5 and selected linearized models for its RBF-based LPV modeling.

of soft robots. The output $\psi(\boldsymbol{\theta})$ of a normalized RBF network is defined as [58]

$$\psi(\boldsymbol{\theta}) = \sum_{i=1}^{N} \boldsymbol{a}_i \eta_i(\boldsymbol{\theta}) \tag{24}$$

where the scheduling variable $\boldsymbol{\theta}$ is the input vector of the RBF network, N is the number of RBFs, and $\boldsymbol{a}_i$ is the output parameter of the ith RBF. The normalized radial basis function $\eta_i(\boldsymbol{\theta})$ is defined as

$$\eta_i(\boldsymbol{\theta}) = \frac{\varphi_i\left(\|\boldsymbol{\theta} - \boldsymbol{c}_i\|\right)}{\sum_{j=1}^{N} \varphi_j\left(\|\boldsymbol{\theta} - \boldsymbol{c}_j\|\right)}, \quad i \in \mathcal{I}_N \tag{25}$$

where $\boldsymbol{c}_i$ is the center vector for the ith RBF, which represents the equilibrium point $(\boldsymbol{x}_i^*, \boldsymbol{u}_i^*)$ corresponding to the ith linearized model (21). As usual in practice, we select the Gaussian function as the radial basis function $\varphi_i\left(\|\boldsymbol{\theta} - \boldsymbol{c}_i\|\right)$ in (25)

$$\varphi_i\left(\|\boldsymbol{\theta} - \boldsymbol{c}_i\|\right) = e^{-(\varepsilon\|\boldsymbol{\theta} - \boldsymbol{c}_i\|)^2} \tag{26}$$

where the parameter ε is the width of the receptive field.

Remark 4. The Gaussian radial basis function (26) is used as a similarity measure, *i.e.*, the similarity reaches the maximum at the center $\boldsymbol{c}_i$ and decreases when the scheduling variable $\boldsymbol{\theta}$ is far away from $\boldsymbol{c}_i$. Then, the key idea of the proposed LPV modeling is to represent the dynamics of a soft robot at any given point in the workspace by interpolating similar ones of neighbor linearized models.

Remark 5. RBF networks are universal approximators, which can approximate any continuous function on a compact set with an arbitrary precision [59]. However, the radial basis functions of conventional RBF networks are constructed according to each data sample, *i.e.*, each data sample requires its own RBF [60]. Then, to represent accurately a high nonlinear dynamics, a large number of RBFs N, *i.e.*, a large number of linearized models, may be required, which induces difficulties for

the control design and real-time implementation of soft robots. To avoid this major issue, based on the idea of the CPR algorithm [52], we propose an interpolator being able to effectively represent a large number of local linearized models with as less as possible number of RBFs N, as described below.

RBF networks are typically trained from pairs of input and target values. To proceed the training process of the proposed RBF network, we first collect a dataset of linearized models (21) of M equilibrium points $(\boldsymbol{x}_i^*, \boldsymbol{u}_i^*)$, corresponding to the input vectors $\boldsymbol{\theta}_i^* = \boldsymbol{y}_i^*$, for $i \in \mathcal{I}_M$. Then, for each collected data sample, we reshape the data into a column vector $Y_{(i)}$, for $i \in \mathcal{I}_M$, containing the parameters of $\mathcal{K}_{ri}$, $\mathcal{D}_{ri}$ and $\mathcal{H}_{ri}$ involved in the matrices A_{ri} and B_{ri} in (19), and in the affine term $\boldsymbol{\zeta}_{ri}^*$ defined in (22). Note that since the linearized robot models are highly correlated, a sparse representation of these submodels can be performed. To this end, similar to the CPR algorithm [52], we *iteratively* define the centers of the proposed RBF network. For each step, a new RBF is added in the center position corresponding to the largest squared error, until a given number of functions or a desired fitting performance is reached. The iterative training procedure is summarized in Algorithm 1 and illustrated in Figure 4. The network weights are obtained from the Orthogonal Least Square Learning algorithm with respect to all training data as described in [61].

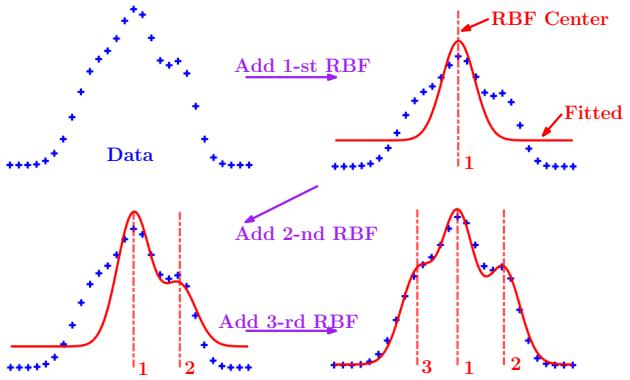


Figure 4: Illustration of the iterative RBF training procedure in Algorithm 1. Here, the sample data are fitted using three radial basis functions.

After the training, the dataset of M linearized models can be interpolated by an $N-$ element RBF network, with $N \leq M$. Hence, the resulting interpolated LPV model has N local linear submodels, defined as

$$\begin{aligned} \boldsymbol{x}_{k+1} &= A(\eta)\boldsymbol{x}_k + B(\eta)\boldsymbol{u}_k + \boldsymbol{\zeta}(\eta) + \boldsymbol{w}_k \\ \boldsymbol{y}_k &= C\boldsymbol{x}_k \end{aligned} \tag{27}$$

with

$$\begin{bmatrix} A(\eta) & B(\eta) & \boldsymbol{\zeta}(\eta) \end{bmatrix} = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k) \begin{bmatrix} A_i & B_i & \boldsymbol{\zeta}_i \end{bmatrix}. \tag{28}$$

The radial basis functions $\eta_i(\boldsymbol{\theta}_k)$, for $i \in \mathcal{I}_N$, are used as the membership functions of the LPV model (27), *i.e.*, to weight local linear submodels, which satisfy the convex sum property

$$0 \leq \eta_i(\boldsymbol{\theta}_k) \leq 1, \quad \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k) = 1, \quad i \in \mathcal{I}_N. \tag{29}$$

Let Ω be the set of membership functions satisfying (29), *i.e.*, $\eta = \begin{bmatrix} \eta_1(\boldsymbol{\theta}_k) & \dots & \eta_N(\boldsymbol{\theta}_k) \end{bmatrix}^\top \in \Omega$, for $\forall k \in \mathbb{N}$. We also denote $\eta_+ = \begin{bmatrix} \eta_1(\boldsymbol{\theta}_{k+1}), \eta_2(\boldsymbol{\theta}_{k+1}), \dots, \eta_N(\boldsymbol{\theta}_{k+1}) \end{bmatrix}^\top \in \Omega$. With the proposed LPV modeling method, the parameter structures of A_i, B_i and $\boldsymbol{\zeta}_i$ in (28) are preserved as the same as those of A_{ri}, B_{ri} and $\boldsymbol{\zeta}_{ri}$ in (19) and (23), which are given by

$$\begin{aligned} A_i &= \begin{bmatrix} I & hI \\ \mathcal{K}_i & \mathcal{D}_i \end{bmatrix}, \quad B_i = \begin{bmatrix} 0 \\ \mathcal{H}_i \end{bmatrix} \\ \boldsymbol{\zeta}_i &= \begin{bmatrix} 0 \\ -\mathcal{K}_i \boldsymbol{q}_{ri}^* - \mathcal{H}_i \boldsymbol{u}_i^* \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{\zeta}_i^* \end{bmatrix}. \end{aligned} \tag{30}$$

The parameter matrices $\mathcal{K}_i$, $\mathcal{D}_i$, $\mathcal{H}_i$, and the affine term $\boldsymbol{\zeta}_i^*$ in (30) are constructed from the reshaped data in the output column vector $\boldsymbol{a}_i$, for $i \in \mathcal{I}_N$, defined in (24). The validation of the proposed LPV modeling with a soft Trunk robot is reported in Section 5.

Remark 6. The proposed LPV modeling method has several advantages compared to classical identification-based LPV modeling methods. From system data, these latter directly identify local linear submodels, which are generally independent from each other, also called *non-coherent* [51]. Interpolating non-coherent submodels requires some specific canonical state-space representations, which may not only lead to a loss of the mechanical model structure of soft robots but also alter the system behaviors [62]. For the proposed LPV modeling, the coherence between local linearized submodels can be naturally guaranteed by the unified POD projector T in (15). Moreover, since the mechanical structure is preserved, the upper-half block-elements of A_i, B_i and $\boldsymbol{\zeta}_i$ given in (30) are constant. Hence, the number of parameters to be interpolated with the proposed LPV modeling is reduced by 50%.

2.3. Modeling Error Analysis

Modeling errors are unavoidable when representing the nonlinear dynamics of soft robots. To this end, we assume that the parameter matrices in (30) are subject to some unknown-but-bounded uncertainties as

$$\begin{aligned} \hat{\mathcal{K}}_i &= \mathcal{K}_i + \Delta\mathcal{K}_i, \quad \hat{\mathcal{D}}_i = \mathcal{D}_i + \Delta\mathcal{D}_i \\ \hat{\mathcal{H}}_i &= \mathcal{H}_i + \Delta\mathcal{H}_i, \quad \hat{\boldsymbol{\zeta}}_i^* = \boldsymbol{\zeta}_i^* + \Delta\boldsymbol{\zeta}_i^* \end{aligned} \tag{31}$$

where $\hat{\mathcal{K}}_i$, $\hat{\mathcal{D}}_i$, $\hat{\mathcal{H}}_i$, $\hat{\boldsymbol{\zeta}}_i^*$ are the estimations, $\mathcal{K}_i$, $\mathcal{D}_i$, $\mathcal{H}_i$, $\boldsymbol{\zeta}_i^*$ are the nominal values, and $\Delta\mathcal{K}_i$, $\Delta\mathcal{D}_i$, $\Delta\mathcal{H}_i$, $\Delta\boldsymbol{\zeta}_i^*$ correspond to the uncertainties, for $i \in \mathcal{I}_N$. From (27), (30) and (31), the uncertain LPV robot model can be described as

$$\boldsymbol{x}_{k+1} = \hat{A}(\eta)\boldsymbol{x}_k + \hat{B}(\eta)\boldsymbol{u}_k + \hat{\boldsymbol{\zeta}}(\eta) + \boldsymbol{w}_k \tag{32}$$

with

$$\begin{bmatrix} \hat{A}(\eta) & \hat{B}(\eta) & \hat{\boldsymbol{\zeta}}(\eta) \end{bmatrix} = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k) \begin{bmatrix} \hat{A}_i & \hat{B}_i & \hat{\boldsymbol{\zeta}}_i \end{bmatrix}$$

Algorithm 1 Iterative Training of RBF Network

Inputs:
Data set $\mathscr{D} = \left\{\left(\theta_1^*, Y_{(1)}\right), \ldots, \left(\theta_M^*, Y_{(M)}\right)\right\}$
Outputs:
Weights $\boldsymbol{a}_i$ and centers $\boldsymbol{c}_i$ of RBFs, for $i \in \mathcal{I}_M$
Required parameters:
Maximum number N of the RBF network
Width of the receptive field ε of the RBF network
Desired interpolation error σ
Initialization:
· Set the estimates of all $Y_{(i)} = 0$ as $\hat{Y}_{(i)} = 0$, for $i \in \mathcal{I}_M$
· Initialize an empty set $\mathscr{S}$ of weights $\boldsymbol{a}_i$ and centers $\boldsymbol{c}_i$ for the RBF network
Begin
for $i = 1$ **to** N **do**
  for $j = 1$ **to** i **do**
    · Compute the activation vector of RBFs
      $\Phi_j = [\eta_j(\theta_1^*), \eta_j(\theta_2^*), \ldots, \eta_j(\theta_M^*)]^\top$
    · Build the activation matrix $\Psi = [\Phi_1, \ldots, \Phi_i]^\top$
    · Build the coefficient matrix $\mathbf{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_i]$
  end for
  · Compute the estimation output $\hat{Y} = \mathbf{A}\Psi$
  · Find the index k, for $k \in \mathcal{I}_M$, corresponding to the largest prediction error $\|\hat{Y}_{(k)} - Y_{(k)}\|^2$
  · Add a new RBF with the center $\boldsymbol{c}_k = \theta_k^*$ into the set $\mathscr{S}$
  · Update the activation matrix Ψ with the new RBF
  · Recompute the coefficients $\boldsymbol{a}_k$ from the least square solution of $Y = \mathbf{A}\Psi$ with respect to $\mathbf{A}$
  if $\sum_{k=1}^{N} \|\hat{Y}_{(k)} - Y_{(k)}\|^2 < \sigma$ **then**
    `break`
  end if
end for
End

where the local uncertain matrices are defined as

$$\hat{A}_i = A_i + \Delta A_i, \qquad \hat{B}_i = B_i + \Delta B_i, \quad \hat{\boldsymbol{\zeta}}_i = \boldsymbol{\zeta}_i + \Delta\boldsymbol{\zeta}_i$$
$$\Delta A_i = \begin{bmatrix} 0 & 0 \\ \Delta\mathcal{K}_i & \Delta\mathcal{D}_i \end{bmatrix}, \quad \Delta B_i = \begin{bmatrix} 0 \\ \Delta\mathcal{H}_i \end{bmatrix}, \quad \Delta\boldsymbol{\zeta}_i = \begin{bmatrix} 0 \\ \Delta\boldsymbol{\zeta}_i^* \end{bmatrix}.$$

Exploiting the specific structures of the state-space matrices in (30), it follows from (32) that

$$\boldsymbol{v}_{r,k+1} = \hat{\mathcal{K}}(\eta)\boldsymbol{q}_{r,k} + \hat{\mathcal{D}}(\eta)\boldsymbol{v}_{r,k} + \hat{\mathcal{H}}(\eta)\boldsymbol{u}_k + \hat{\boldsymbol{\zeta}}^*(\eta) + \boldsymbol{w}_{v,k} \quad (33)$$

with

$$\begin{bmatrix} \hat{\mathcal{K}}(\eta) & \hat{\mathcal{D}}(\eta) \\ \hat{\mathcal{H}}(\eta) & \hat{\boldsymbol{\zeta}}^*(\eta) \end{bmatrix} = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k) \begin{bmatrix} \mathcal{K}_i & \mathcal{D}_i \\ \mathcal{H}_i & \boldsymbol{\zeta}_i^* \end{bmatrix}.$$

Note that for (33) the disturbance vector $\boldsymbol{w}_k$ is partitioned as $\boldsymbol{w}_k = \begin{bmatrix} \boldsymbol{w}_{q,k}^\top & \boldsymbol{w}_{v,k}^\top \end{bmatrix}^\top$, where the disturbance $\boldsymbol{w}_{q,k} \in \mathbb{R}^r$ (respectively $\boldsymbol{w}_{v,k} \in \mathbb{R}^r$) affects the dynamics of the generalized displacements $\boldsymbol{q}_{r,k}$ (respectively generalized velocities $\boldsymbol{v}_{r,k}$). To represent the parametric uncertainties involved in soft robots modeling, let us define a *lumped* disturbance

$$\boldsymbol{d}_{e,k} = \mathcal{H}(\eta)^\dagger \Delta\Sigma_k \quad (34)$$

where $\Delta\Sigma_k = \mathcal{K}(\eta)\boldsymbol{q}_k + \Delta\mathcal{D}(\eta)\boldsymbol{v}_k + \Delta\mathcal{H}(\eta)\boldsymbol{u}_k + \Delta\boldsymbol{\zeta}^*(\eta)$, and $\mathcal{H}(\eta)^\dagger$ is the pseudo-inverse of $\mathcal{H}(\eta)$. Using the disturbance expression (34) and considering (33), the uncertain LPV model (32) can be reformulated as

$$\begin{aligned} \boldsymbol{x}_{k+1} &= A(\eta)\boldsymbol{x}_k + B(\eta)(\boldsymbol{u}_k + \boldsymbol{d}_{e,k}) + \boldsymbol{\zeta}(\eta) + \boldsymbol{w}_k \\ \boldsymbol{y}_k &= C\boldsymbol{x}_k. \end{aligned} \quad (35)$$

Hence, model (35) can be used for LPV control design of soft robots under both disturbances and parametric uncertainties.

Remark 7. Using the projector T_r in (15) allows to reduce significantly the model order while preserving the mechanical structure of the original robot model (1), *i.e.*, the reduced-order states $\boldsymbol{q}_{r,k}$ and $\boldsymbol{v}_{r,k}$ still respectively play the role of the generalized displacements and velocities with

$$\boldsymbol{q}_{r,k+1} = \boldsymbol{q}_{r,k} + h\boldsymbol{v}_{r,k+1}.$$

With such a structure preservation, the robot parameters are only involved in the lower halves of the state-space matrices A_i, B_i and the affine term $\boldsymbol{\zeta}_i$. This leads to two major advantages for LPV modeling of soft robots. First, for RBF network training, the number of parameters to be interpolated is reduced by 50% since the upper halves of the matrices A_{ri}, B_{ri} in (19) and the affine term $\boldsymbol{\zeta}_{ri}$ in (23) are *fixed* constants. Second, the modeling uncertainties are only present in the lower halves of ΔA_i, ΔB_i and $\Delta\boldsymbol{\zeta}_i$, for $i \in \mathcal{I}_N$. This allows to represent the parametric uncertainty and a part of MOR errors in the LPV robot model (35) via the lumped disturbance $\boldsymbol{d}_{e,k}$, defined in (34). Note that $\boldsymbol{d}_{e,k}$ shares the same distribution matrix $B(\eta)$ with the control input $\boldsymbol{u}_k$, *i.e.*, $\boldsymbol{d}_{e,k}$ is a *matching* disturbance. This key feature enables an effective EID-based framework for LPV tracking control of soft robots under modeling uncertainty as recently shown in [15]. It is important to note that these advantages of the proposed LPV modeling cannot be achieved with existing POD-based modeling results [14, 29, 48] and related references therein, which cannot directly guarantee the structure preservation.

3. Tracking Control Problem Formulation

This section formulates the tracking control problem of soft robots. To this end, we define the tracking error as

$$\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x}_{r,k} \quad (36)$$

with $\boldsymbol{x}_{r,k} = \begin{bmatrix} \boldsymbol{x}_{rq,k}^\top & \boldsymbol{x}_{rv,k}^\top \end{bmatrix}^\top$, where $\boldsymbol{x}_{rq,k} \in \mathbb{R}^r$ (respectively $\boldsymbol{x}_{rv,k} \in \mathbb{R}^r$) is the reference trajectory corresponding to the generalized displacements (respectively velocities). Then, the tracking error dynamics can be defined from (35) and (36) as

$$\begin{aligned} \boldsymbol{e}_{k+1} = {} & A(\eta)\boldsymbol{e}_k + B(\eta)(\boldsymbol{u}_k + \boldsymbol{d}_{e,k}) + \boldsymbol{w}_k \\ & + A(\eta)\boldsymbol{x}_{r,k} - \boldsymbol{x}_{r,k+1} + \boldsymbol{\zeta}(\eta). \end{aligned} \quad (37)$$

To improve the tracking performance of soft robots under modeling uncertainties, we extend the EID-based linear control scheme

in [15] to the LPV model (37). This feedback-feedforward control scheme is composed of three components as

$$\boldsymbol{u}_k = \boldsymbol{u}_k^{\text{ff}} + \boldsymbol{u}_k^{\text{dc}} + \boldsymbol{u}_k^{\text{fb}} \tag{38}$$

where $\boldsymbol{u}_k^{\text{ff}}$ is the feedforward control, $\boldsymbol{u}_k^{\text{dc}}$ is the disturbance-estimator based control, and $\boldsymbol{u}_k^{\text{fb}}$ is the feedback control. The proposed control scheme is illustrated in Figure 5.

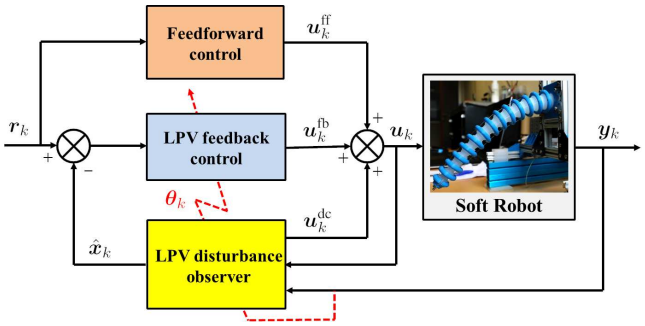


Figure 5: EID-based LPV tracking control scheme for elastic soft robots.

3.1. Feedforward Control

The approximation capabilities of LPV modeling are substantially improved with the offset terms [63, 64]. However, the presence of these affine terms leads to more involved stability analysis and control synthesis procedures [65]. In addition, for tracking control, the information of the reference signal can be obtained online. Hence, dealing with the reference trajectory as an *unknown* input or disturbance may lead to conservative control results. To avoid these issues, we can design a feedforward control $\boldsymbol{u}_k^{\text{ff}}$ to account for the affects of $\boldsymbol{x}_{r,k}$ and $\boldsymbol{\zeta}(\eta)$ on the closed-loop system by exploiting the mechanical structure of the proposed LPV model (35) for soft robots, see Remark 7.

The feedforward control is determined under a perfect tracking condition, *i.e.*, $\boldsymbol{e}_k \to 0$ for $k \to \infty$, that is

$$\boldsymbol{x}_k^r = \boldsymbol{x}_{r,k}, \quad \boldsymbol{\theta}_k^r = \boldsymbol{y}_k^r = C\boldsymbol{x}_{r,k}, \quad k \to \infty \tag{39}$$

where $\boldsymbol{x}_k^r$, $\boldsymbol{y}_k^r$ and $\boldsymbol{\theta}_k^r$ respectively represent the reduced robot state, the robot output and the scheduling variable corresponding to the perfect tracking. To guarantee a smooth control action for soft robots, especially in large deformation scenarios, the feedforward control is designed following the RBF-based interpolation similar to the construction of the LPV robot model (27). From the *local* viewpoint, the error dynamics (37) around the ith equilibrium point is given by

$$\begin{aligned}\boldsymbol{e}_{k+1} = {} & A_i\boldsymbol{e}_k + B_i(\boldsymbol{u}_k + \boldsymbol{d}_{e,k}) + \boldsymbol{w}_k \\ & + A_i\boldsymbol{x}_{r,k} - \boldsymbol{x}_{r,k+1} + \boldsymbol{\zeta}_i.\end{aligned} \tag{40}$$

Then, the local feedforward control action can be selected from (40) as

$$B_i\boldsymbol{u}_{i,k}^{\text{ff}} + A_i\boldsymbol{x}_{r,k} - \boldsymbol{x}_{r,k+1} + \boldsymbol{\zeta}_i = 0. \tag{41}$$

Substituting (30) into (41), it follows that

$$\mathcal{H}_i\boldsymbol{u}_{i,k}^{\text{ff}} + \mathcal{K}_i\boldsymbol{x}_{rq,k} + \mathcal{D}_i\boldsymbol{x}_{rv,k} - \boldsymbol{x}_{rv,k+1} - \mathcal{K}_i\boldsymbol{q}_{ri}^* - \mathcal{H}_i\boldsymbol{u}_i^* = 0. \tag{42}$$

Under condition (39), it follows that

$$\mathcal{K}_i\boldsymbol{x}_{rq,k} \simeq \mathcal{K}_i\boldsymbol{q}_{ri}^*. \tag{43}$$

Moreover, with a small sampling time value h and a smooth reference trajectory $\boldsymbol{x}_{r,k}$, it follows that

$$\boldsymbol{x}_{r,k+1} \simeq \boldsymbol{x}_{r,k}, \quad \mathcal{D}_i = I - hTS_i^{-1}D_iT^\top \simeq I. \tag{44}$$

From (42), (43) and (44), it is reasonable to select a local feedforward control as $\boldsymbol{u}_{i,k}^{\text{ff}} = \boldsymbol{u}_i^*$. Then, the feedforward control $\boldsymbol{u}_k^{\text{ff}}$ is computed using the RBF-based interpolation as

$$\boldsymbol{u}_k^{\text{ff}} = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k^r)\boldsymbol{u}_{i,k}^{\text{ff}} = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k^r)\boldsymbol{u}_i^*. \tag{45}$$

Remark 8. A "natural" solution to obtain the feedforward control $\boldsymbol{u}_k^{\text{ff}}$ is directly based on the tracking error model (37) as

$$B(\eta)\boldsymbol{u}_k^{\text{ff}} + A(\eta)\boldsymbol{x}_{r,k} - \boldsymbol{x}_{r,k+1} + \boldsymbol{\zeta}(\eta) = 0. \tag{46}$$

Substituting the explicit expressions of $A(\eta)$, $B(\eta)$ and $\boldsymbol{\zeta}(\eta)$ into (46), it follows that

$$\mathcal{H}(\eta)\boldsymbol{u}_k^{\text{ff}} + \mathcal{K}(\eta)\boldsymbol{x}_{rq,k} + \mathcal{D}(\eta)\boldsymbol{x}_{rv,k} - \boldsymbol{x}_{rv,k+1} + \boldsymbol{\zeta}^*(\eta) = 0. \tag{47}$$

Then, the feedforward control can be deduced from (47) as

$$\boldsymbol{u}_k^{\text{ff}} = -\mathcal{H}(\eta)^\dagger \left(\mathcal{K}(\eta)\boldsymbol{x}_{rq,k} + \mathcal{D}(\eta)\boldsymbol{x}_{rv,k} - \boldsymbol{x}_{rv,k+1} + \boldsymbol{\zeta}^*(\eta)\right). \tag{48}$$

However, the expression of $\boldsymbol{u}_k^{\text{ff}}$ in (48) requires an online pseudo-inverse of the parameter-dependent matrix $\mathcal{H}(\eta)$, *i.e.*, control direction matrix, which could yield large control oscillations or even unstable control behaviors in case of large deformations. On the contrary, the feedforward control $\boldsymbol{u}_k^{\text{ff}}$ in (45) is a convex combination of the control input values corresponding to different selected equilibriums, which allows a smooth shifting between operating points of soft robots. Note also that the approximation errors caused by (43) and (44) when computing $\boldsymbol{u}_k^{\text{ff}}$ in (45) can be viewed as a matching disturbance, which can be effectively dealt with using the EID-based control concept as shown in the design of disturbance-estimator based control.

3.2. Disturbance-Estimator Based Control

The lumped disturbance $\boldsymbol{d}_{e,k}$, defined in (34), enters in the tracking error dynamics (37) via the same channel as the control input $\boldsymbol{u}_k$. Then, the disturbance-estimator based control can be designed as follows:

$$\boldsymbol{u}_k^{\text{dc}} = -\hat{\boldsymbol{d}}_k^{\text{ef}} \tag{49}$$

where $\hat{\boldsymbol{d}}_k^{\text{ef}}$ is a filtered estimate of the lumped disturbance $\boldsymbol{d}_{e,k}$. Due to the low-frequency behaviors of soft robots, it has been shown that $\boldsymbol{d}_{e,k}$ can be considered of low-frequency [15]. After several trials, the dynamics of $\boldsymbol{d}_{e,k}$ can be sufficiently described with a piecewise second-order polynomial signal. Moreover, based on the concept of EID control approach [53], the following low-pass filter is integrated to limit the angular-frequency band of the disturbance estimation:

$$\mathbf{F}(s) = \frac{1}{1 + T_f s} I \tag{50}$$

where s is the Laplace variable. The filter time constant T_f in (50) can be chosen such that

$$T_f \in \left[\frac{1}{10\omega_r}, \frac{1}{5\omega_r}\right],$$

where ω_r is the highest angular frequency selected for disturbance rejection [53]. Then, the continuous-time model of the lumped disturbance is given by [15]

$$\dot{\boldsymbol{d}}(t) = J_c \boldsymbol{d}(t) \tag{51}$$

with

$$\boldsymbol{d}(t) = \begin{bmatrix} \boldsymbol{d}^{\text{ef}}(t) \\ \boldsymbol{d}^{\text{e}}(t) \\ \dot{\boldsymbol{d}}^{\text{e}}(t) \end{bmatrix}, \quad J_c = \begin{bmatrix} -\frac{1}{T_f} I & \frac{1}{T_f} I & 0 \\ 0 & 0 & I \\ 0 & 0 & 0 \end{bmatrix}.$$

Using the explicit Euler-discretization, the discrete-time counterpart of model (51) can be obtained as

$$\boldsymbol{d}_{k+1} = J \boldsymbol{d}_k \tag{52}$$

with $J = I + hJ_c$. From the LPV robot model (35) and the disturbance model (52), we propose the following Luenberger-like observer to estimate *simultaneously* the state $\boldsymbol{x}_k$ and the unknown disturbance $\boldsymbol{d}_{e,k}$:

$$\begin{aligned} \hat{\boldsymbol{x}}_{k+1} &= A(\eta)\hat{\boldsymbol{x}}_k + B(\eta)(\boldsymbol{u}_k + \hat{\boldsymbol{d}}_{e,k}) + L_x(\eta)(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k) \\ \hat{\boldsymbol{d}}_{k+1} &= J\hat{\boldsymbol{d}}_k + L_d(\eta)(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k) \\ \hat{\boldsymbol{y}}_k &= C\hat{\boldsymbol{x}}_k \end{aligned} \tag{53}$$

where $\hat{\boldsymbol{x}}_k$ is the estimate of $\boldsymbol{x}_k$, and $\hat{\boldsymbol{d}}_k$ is the estimate of $\boldsymbol{d}_k$. The parameter-dependent observer gains $L_x(\eta) \in \mathbb{R}^{n\times q}$ and $L_d(\eta) \in \mathbb{R}^{3p\times q}$ are to be designed. The estimation error dynamics can be defined from (35), (52) and (53) as

$$\boldsymbol{\varepsilon}_{k+1} = (A_o(\eta) - L(\eta)C_o)\boldsymbol{\varepsilon}_k + B_{ow}\boldsymbol{w}_k \tag{54}$$

where $\boldsymbol{\varepsilon}_k = \begin{bmatrix} \boldsymbol{\varepsilon}_{x,k}^\top & \boldsymbol{\varepsilon}_{d,k}^\top \end{bmatrix}^\top$, with $\boldsymbol{\varepsilon}_{x,k} = \boldsymbol{x}_k - \hat{\boldsymbol{x}}_k$ and $\boldsymbol{\varepsilon}_{d,k} = \boldsymbol{d}_k - \hat{\boldsymbol{d}}_k$. The system matrices in (54) are defined as

$$\begin{bmatrix} A_o(\eta) & L(\eta) \end{bmatrix} = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k) \begin{bmatrix} A_{oi} & L_i \end{bmatrix}$$

with

$$A_{oi} = \begin{bmatrix} A_i & B_{oi} \\ 0 & J \end{bmatrix}, \quad B_{ow} = \begin{bmatrix} I \\ 0 \end{bmatrix}, \quad L_i = \begin{bmatrix} L_{xi} \\ L_{di} \end{bmatrix}$$
$$C_o = \begin{bmatrix} C & 0 \end{bmatrix}, \quad B_{oi} = \begin{bmatrix} B_i & 0 \end{bmatrix}.$$

Remark 9. Disturbance observers have been successfully applied to compensate unknown disturbances/uncertainties for tracking control of various engineering systems [66], including robotics applications [67–70]. However, related works generally imply complex analysis and design methods or require restrictive assumptions and/or additional measurements, *e.g.*, both system state and its derivatives must be available [66]. Using the generalized Luenberger observer (53), we only require the output information to compute directly the disturbance-estimator based control (49) with a simple assumption of low-frequency matching disturbance, which is reasonable for soft robots control [15].

3.3. Feedback Control

The feedback control is used to guarantee the closed-loop stability under the effects of modeling uncertainties and disturbances. For tracking control purposes, we consider the following LPV proportional-integral control structure:

$$\boldsymbol{u}_k^{\text{fb}} = K_P(\eta)(\hat{\boldsymbol{x}}_k - \boldsymbol{x}_{r,k}) + K_I(\eta)\boldsymbol{e}_{I,k} \tag{55}$$

where the parameter-dependent feedback gains $K_P(\eta) \in \mathbb{R}^{p\times n}$ and $K_I(\eta) \in \mathbb{R}^{p\times q}$ are to be determined, and

$$\boldsymbol{e}_{I,k+1} = \boldsymbol{e}_{I,k} + hC(\boldsymbol{x}_k - \boldsymbol{x}_{r,k}).$$

With $\boldsymbol{u}_k^{\text{ff}}$, $\boldsymbol{u}_k^{\text{dc}}$ and $\boldsymbol{u}_k^{\text{fb}}$ respectively defined in (45), (49) and (55), substituting the control expression (38) into (37), the tracking error dynamics can be represented by

$$\boldsymbol{\xi}_{k+1} = (A_c(\eta) + B_c(\eta)K(\eta))\boldsymbol{\xi}_k + B_e(\eta)\boldsymbol{\varepsilon}_k + B_{cw}\boldsymbol{w}_k \tag{56}$$

with $\boldsymbol{\xi}_k = \begin{bmatrix} \boldsymbol{e}_k^\top & \boldsymbol{e}_{I,k}^\top \end{bmatrix}^\top$, and

$$\Phi(\eta) = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k)\Phi_i, \quad \Phi \in \{A_c, B_c, K\}$$
$$B_e(\eta) = \sum_{i=1}^{N}\sum_{j=1}^{N} \eta_i(\boldsymbol{\theta}_k)\eta_j(\boldsymbol{\theta}_k)B_{eij}.$$

The system matrices in (56) are given by

$$A_{ci} = \begin{bmatrix} A_i & 0 \\ C & I \end{bmatrix}, \quad B_{ci} = \begin{bmatrix} B_i \\ 0 \end{bmatrix}, \quad B_{cw} = \begin{bmatrix} I \\ 0 \end{bmatrix}$$
$$B_{eij} = \begin{bmatrix} -B_i K_{Pj} & 0 \\ 0 & 0 \end{bmatrix}, \quad K_i = \begin{bmatrix} K_{Pi} & K_{Ii} \end{bmatrix}.$$

Then, the extended closed-loop dynamics can be defined from (54) and (56) as

$$\bar{\boldsymbol{x}}_{k+1} = \begin{bmatrix} \bar{A}_c(\eta) & B_e(\eta) \\ 0 & \bar{A}_o(\eta) \end{bmatrix} \bar{\boldsymbol{x}}_k + \begin{bmatrix} B_{cw} \\ B_{ow} \end{bmatrix} \boldsymbol{w}_k \tag{57}$$

with $\bar{\boldsymbol{x}}_k = \begin{bmatrix} \boldsymbol{\xi}_k^\top & \boldsymbol{\varepsilon}_k^\top \end{bmatrix}^\top$, and

$$\bar{A}_{oi} = A_{oi} - L_i C_o, \quad \bar{A}_o(\eta) = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k)\bar{A}_{oi}$$
$$\bar{A}_{cij} = A_{ci} + B_{ci}K_j, \quad \bar{A}_c(\eta) = \sum_{i=1}^{N}\sum_{j=1}^{N} \eta_i(\boldsymbol{\theta}_k)\eta_j(\boldsymbol{\theta}_k)\bar{A}_{cij}.$$

To study the stability of system (57), we consider the following parameter-dependent Lyapunov candidate function:

$$\mathcal{V}(\bar{\boldsymbol{x}}) = \bar{\boldsymbol{x}}^\top \text{diag}\{\lambda Q^{-\top} P_c(\eta) Q^{-1}, P_o(\eta)\}\bar{\boldsymbol{x}} \tag{58}$$

where $\lambda > 0$, the matrix $Q \in \mathbb{R}^{(n+q)\times(n+q)}$ is nonsingular, and the parameter-dependent matrices $P_c(\eta) \in \mathbb{R}^{(n+q)\times(n+q)}$ and $P_o(\eta) \in \mathbb{R}^{(n+3p)\times(n+3p)}$ are defined as

$$P_c(\eta) = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k)P_{ci}, \quad P_o(\eta) = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_k)P_{oi}$$

with $P_{ci} \succ 0$ and $P_{oi} \succ 0$, for $\forall i \in \mathcal{I}_N$. For tracking control purposes, we also introduce the performance output $\boldsymbol{z}_k$ associated to system (57) as the position tracking error $\boldsymbol{z}_k = \boldsymbol{e}_k = C_z \boldsymbol{\xi}_k$, with $C_z = \begin{bmatrix} C & 0 \end{bmatrix}$, or

$$\boldsymbol{z}_k = F \bar{\boldsymbol{x}}_k, \quad F = \begin{bmatrix} C_z & 0 \end{bmatrix}. \tag{59}$$

We are ready to formulate the following observer-based output tracking control problem for soft robots.

Problem 1. Consider the LPV robot model (35) with the control law (38). Determine the parameter-dependent observer gain $L(\boldsymbol{\theta})$ and controller gain $K(\boldsymbol{\theta})$ such that the extended error dynamics (57) verifies the following properties.

(P1) If $\boldsymbol{w}_k = 0$, for $\forall k \in \mathbb{N}$, the closed-loop dynamics (57) is exponentially stable with a decay rate $\alpha \in (0, 1)$.

(P2) The closed-loop state $\bar{\boldsymbol{x}}_k$ is uniformly bounded for any initial condition $\bar{\boldsymbol{x}}_0$ and any sequence $\{\boldsymbol{w}_k\}_{k\in\mathbb{N}} \in \ell_\infty$. That is, there exists a bound $\varphi(\bar{\boldsymbol{x}}_0, \|\boldsymbol{w}\|_{\ell_\infty})$ such that $\|\bar{\boldsymbol{x}}_k\| \leq \varphi(\bar{\boldsymbol{x}}_0, \|\boldsymbol{w}\|_{\ell_\infty})$, for $\forall k \geq 0$. Moreover, the performance output verifies

$$\lim_{k\to\infty} \sup \|\boldsymbol{z}_k\| < \gamma \|\boldsymbol{w}\|_{\ell_\infty} \tag{60}$$

where the $\ell_\infty-$gain γ is specified in Theorem 1. We also deduce from (60) that if $\bar{\boldsymbol{x}}_0 = 0$, then $\|\boldsymbol{z}_k\| < \gamma \|\boldsymbol{w}\|_{\ell_\infty}$, for $\forall k \in \mathbb{N}$.

System (57) verifying properties (P1)–(P2) is said to be globally uniformly $\ell_\infty-$stable with a performance level γ, see [71, Chapter 4]. It follows from (59) and (60) that a smaller value of the $\ell_\infty-$gain γ leads to a better tracking control performance. Note also that a larger value of the decay rate α leads to a faster closed-loop response of the soft robot, which may induce aggressive control behaviors.

4. LPV Output Feedback Tracking Control with $\ell_\infty-$Gain Performance Guarantee

This section presents LMI-based conditions to simultaneously design an LPV observer (53) and an LPV feedback controller (55) such that system (57) verifies the closed-loop properties specified in Problem 1.

Theorem 1. Consider the LPV robot model (35) with the control law (38) and a decay rate $\alpha \in (0, 1)$. If there exist parameter-dependent positive definite matrices $P_o(\eta) \in \mathbb{R}^{(n+3p)\times(n+3p)}$, $P_c(\eta) \in \mathbb{R}^{(n+q)\times(n+q)}$, parameter-dependent matrices $M(\eta) \in \mathbb{R}^{p\times(n+q)}$, $N(\eta) \in \mathbb{R}^{(n+3p)\times q}$, matrices $G \in \mathbb{R}^{(n+3p)\times(n+3p)}$, $Q \in \mathbb{R}^{(n+q)\times(n+q)}$, and positive scalars μ, ν such that

$$\begin{bmatrix} (1-\alpha)P_o(\eta) & 0 & A_o^\top(\eta)G^\top - C_o^\top N^\top(\eta) \\ \star & \alpha\nu I & B_{ow}^\top G^\top \\ \star & \star & G + G^\top - P_o(\eta_+) \end{bmatrix} \succ 0 \tag{61}$$

$$\begin{bmatrix} (1-\alpha)P_c(\eta) & A_c(\eta)Q + B_c(\eta)M(\eta) \\ \star & Q + Q^\top - P_c(\eta_+) \end{bmatrix} \succ 0 \tag{62}$$

$$\begin{bmatrix} P_c(\eta) & 0 & Q^\top C_z^\top \\ \star & P_o(\eta) & 0 \\ \star & \star & \mu I \end{bmatrix} \succeq 0 \tag{63}$$

for all $\eta(\boldsymbol{\theta}_k), \eta(\boldsymbol{\theta}_{k+1}) \in \Omega$, with

$$P_o(\eta_+) = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_{k+1}) P_{oi}, \quad P_c(\eta_+) = \sum_{i=1}^{N} \eta_i(\boldsymbol{\theta}_{k+1}) P_{ci}.$$

Then, the closed-loop system (57) verifies the properties defined in Problem 1 with a guaranteed $\ell_\infty-$gain $\gamma = \sqrt{\nu\mu}$. Moreover, the feedback control gain and the observer gain are respectively given by

$$K(\eta) = M(\eta)Q^{-1}, \quad L(\eta) = G^{-1}N(\eta). \tag{64}$$

Proof. To begin with, it follows from (61) and (62) that

$$G + G^\top \succ P_o(\eta_+), \quad Q + Q^\top \succ P_c(\eta_+). \tag{65}$$

Since $P_o(\eta_+) \succ 0$ and $P_c(\eta_+) \succ 0$, for $\forall \eta(\boldsymbol{\theta}_{k+1}) \in \Omega$, it follows from (65) that the matrices G and Q are nonsingular, *i.e.*, invertible, thus the feedback control gain $K(\eta)$ and the observer gain matrix $L(\eta)$ in (64) are well-defined. For brevity, we denote

$$\begin{aligned} \bar{\mathbf{A}}_o(\eta) &= \begin{bmatrix} \bar{A}_o(\eta) & B_{ow} \end{bmatrix} \\ \mathbf{P}_o(\eta) &= \operatorname{diag}\{(1-\alpha)P_o(\eta), \alpha\nu I\} \\ \mathbf{A}_c(\eta) &= A_c(\eta)Q + B_c(\eta)M(\eta) \\ \mathbf{Q}_c(\eta) &= Q^{-\top}P_c(\eta)Q^{-1}, \ \mathbf{Q}_c(\eta_+) = Q^{-\top}P_c(\eta_+)Q^{-1}. \end{aligned}$$

Inspired by the congruence transformation proposed in [72], we multiply inequality (61) with

$$\begin{bmatrix} I & 0 & -\bar{A}_o^\top(\eta) \\ 0 & I & -B_{ow}^\top \end{bmatrix}$$

on the left and its transpose on the right while considering $N(\eta) = GL(\eta)$, it follows that

$$\Gamma(\eta) = \bar{\mathbf{A}}_o^\top(\eta) P_o(\eta_+) \bar{\mathbf{A}}_o(\eta) - \mathbf{P}_o(\eta) \prec 0. \tag{66}$$

Then, multiplying inequality (62) with

$$\begin{bmatrix} I & -\mathbf{A}_c^\top(\eta)Q^{-\top} \end{bmatrix}$$

on the left and its transpose on the right, we have

$$\mathbf{A}_c^\top(\eta)\mathbf{Q}_c(\eta_+)\mathbf{A}_c(\eta) - (1-\alpha)P_c(\eta) \prec 0. \tag{67}$$

Pre- and post-multiplying (67) with $Q^{-\top}$ and its transpose while considering $M(\eta) = K(\eta)Q$, it follows that

$$\Pi(\eta) = \bar{A}_c^\top(\eta)\mathbf{Q}_c(\eta_+)\bar{A}_c(\eta) - (1-\alpha)\mathbf{Q}_c(\eta) \prec 0. \tag{68}$$

Since $\Gamma(\eta) \prec 0$ and $\Pi(\eta) \prec 0$, then for $\Psi(\eta) \succeq 0$, there always exists a positive scalar λ such that [73]

$$\Gamma(\eta) + \lambda\Psi(\eta) \prec \lambda\Upsilon^\top(\eta)\Pi^{-1}(\eta)\Upsilon(\eta) \tag{69}$$

with

$$\begin{aligned} \Psi(\eta) &= \begin{bmatrix} B_e^\top(\eta) \\ B_c^\top(\eta) \end{bmatrix} \mathbf{Q}_c(\eta_+) \begin{bmatrix} B_e(\eta) & B_c(\eta) \end{bmatrix} \\ \Upsilon(\eta) &= \begin{bmatrix} \bar{A}_c^\top(\eta)\mathbf{Q}_c(\eta_+)B_e(\eta) & \bar{A}_c^\top(\eta)\mathbf{Q}_c(\eta_+)B_{cw} \end{bmatrix}. \end{aligned} \tag{70}$$

Applying the Schur complement lemma [54], we can prove that condition (69) is equivalent to

$$\begin{bmatrix} \lambda\Pi(\eta) & \lambda\Upsilon(\eta) \\ \star & \Gamma(\eta)+\lambda\Psi(\eta) \end{bmatrix} \prec 0. \tag{71}$$

Substituting the expressions of $\Gamma(\eta)$ in (66), $\Pi(\eta)$ in (68), $\Psi(\eta)$ and $\Upsilon(\eta)$ in (70) into condition (71), we obtain

$$\begin{bmatrix} \Sigma_{11}(\eta) & \Sigma_{12}(\eta) & \Sigma_{13}(\eta) \\ \star & \Sigma_{22}(\eta) & \Sigma_{23}(\eta) \\ \star & \star & \Sigma_{33}(\eta) \end{bmatrix} \prec 0 \tag{72}$$

with

$$\begin{aligned}
\Sigma_{11}(\eta) &= \lambda\bar{A}_c^\top(\eta)\mathbf{Q}_c(\eta_+)\bar{A}_c(\eta) + \lambda(\alpha-1)\mathbf{Q}_c(\eta) \\
\Sigma_{12}(\eta) &= \lambda\bar{A}_c^\top(\eta)\mathbf{Q}_c(\eta_+)B_e(\eta) \\
\Sigma_{13}(\eta) &= \lambda\bar{A}_c^\top(\eta)\mathbf{Q}_c(\eta_+)B_{cw} \\
\Sigma_{22}(\eta) &= \lambda B_e^\top(\eta)\mathbf{Q}_c(\eta_+)B_e(\eta) + \Xi(\eta) \\
\Xi(\eta) &= \bar{A}_o^\top(\eta)P_o(\eta_+)\bar{A}_o(\eta) - (1-\alpha)P_o(\eta) \\
\Sigma_{23}(\eta) &= \lambda B_e(\eta)^\top\mathbf{Q}_c(\eta_+)B_{cw} + \bar{A}_o^\top(\eta)P_o(\eta_+)B_{ow} \\
\Sigma_{33}(\eta) &= -\alpha\nu I + \lambda B_{cw}^\top\mathbf{Q}_c(\eta_+)B_{cw} + B_{ow}^\top P_o(\eta_+)B_{ow}.
\end{aligned}$$

Since $\lambda B_e^\top(\eta)\mathbf{Q}_c(\eta_+)B_e(\eta) \succeq 0$, it follows from (72) that

$$\begin{bmatrix} \Sigma_{11}(\eta) & \Sigma_{12}(\eta) & \Sigma_{13}(\eta) \\ \star & \Xi(\eta) & \Sigma_{23}(\eta) \\ \star & \star & \Sigma_{33}(\eta) \end{bmatrix} \prec 0. \tag{73}$$

Multiplying inequality (73) with $\begin{bmatrix} \bar{\boldsymbol{x}}_k^\top & \boldsymbol{w}_k^\top \end{bmatrix}^\top$ on the left and its transpose on the right, the following condition can be obtained after some algebraic manipulations:

$$\Delta\mathcal{V}(\bar{\boldsymbol{x}}_k) + \alpha\left(\mathcal{V}(\bar{\boldsymbol{x}}_k) - \nu\boldsymbol{w}_k^\top\boldsymbol{w}_k\right) < 0, \quad \forall k \in \mathbb{N} \tag{74}$$

where the parameter-dependent Lyapunov function $\mathcal{V}(\bar{\boldsymbol{x}}_k)$ is defined in (58), and $\Delta\mathcal{V}(\bar{\boldsymbol{x}}_k) = \mathcal{V}(\bar{\boldsymbol{x}}_{k+1}) - \mathcal{V}(\bar{\boldsymbol{x}}_k)$ is its difference along the trajectories of the extended closed-loop error dynamics (57). We distinguish the two following cases.

First, if $\boldsymbol{w}_k = 0$, for $\forall k \in \mathbb{N}$, it follows from (74) that

$$\Delta\mathcal{V}(\bar{\boldsymbol{x}}_k) + \alpha\mathcal{V}(\bar{\boldsymbol{x}}_k) < 0, \quad \forall k \in \mathbb{N}$$

which proves Property (P1) on the exponential stability with a decay rate α of system (57).

Second, if $\boldsymbol{w}_k \neq 0$ and $\{\boldsymbol{w}_k\}_{k\in\mathbb{N}} \in \ell_\infty$, it follows from (74) that

$$\mathcal{V}(\bar{\boldsymbol{x}}_k) < (1-\alpha)\mathcal{V}(\bar{\boldsymbol{x}}_{k-1}) + \alpha\nu\left\|\boldsymbol{w}_{k-1}\right\|^2, \ \ \forall k \geq 1. \tag{75}$$

By recursivity and since $\alpha \in (0,1)$, it follows from (75) that

$$\begin{aligned}
\mathcal{V}(\bar{\boldsymbol{x}}_k) &< (1-\alpha)^k\mathcal{V}(\bar{\boldsymbol{x}}_0) + \alpha\nu\sum_{i=0}^{k-1}(1-\alpha)^i\left\|\boldsymbol{w}_{k-1-i}\right\|^2 \\
&< (1-\alpha)^k\mathcal{V}(\bar{\boldsymbol{x}}_0) + \alpha\nu\|\boldsymbol{w}\|_{\ell_\infty}^2\sum_{i=0}^{k-1}(1-\alpha)^i \\
&< (1-\alpha)^k\mathcal{V}(\bar{\boldsymbol{x}}_0) + \nu\|\boldsymbol{w}\|_{\ell_\infty}^2, \ \ \forall k \geq 1
\end{aligned} \tag{76}$$

which guarantees that $\bar{\boldsymbol{x}}_k$ is uniformly bounded for any initial condition $\bar{\boldsymbol{x}}_0$ and any sequence $\{\boldsymbol{w}_k\}_{k\in\mathbb{N}} \in \ell_\infty$.

Applying the congruence transformation to inequality (63) with $\mathrm{diag}\{Q^{-\top}, I, I\}$, it follows that

$$\begin{bmatrix} \mathbf{Q}_c(\eta) & 0 & C_z^\top \\ \star & P_o(\eta) & 0 \\ \star & \star & \mu I \end{bmatrix} \succeq 0. \tag{77}$$

Using the Schur complement lemma, we can prove that condition (77) is equivalent to

$$\mu\mathrm{diag}\{\mathbf{Q}_c(\eta), P_o(\eta)\} - F^\top F \succeq 0. \tag{78}$$

Pre- and post-multiplying condition (78) with $\bar{\boldsymbol{x}}_k^\top$ and its transpose yields

$$\|\boldsymbol{z}_k\|^2 \leq \mu\mathcal{V}(\bar{\boldsymbol{x}}_k). \tag{79}$$

It follows from (76) and (79) that

$$\|\boldsymbol{z}_k\| \leq \sqrt{\mu\mathcal{V}(\bar{\boldsymbol{x}}_0)}(1-\alpha)^{k/2} + \gamma\|\boldsymbol{w}\|_{\ell_\infty}, \ \ \forall k \geq 1 \tag{80}$$

with $\gamma = \sqrt{\nu\mu}$. For any initial condition $\bar{\boldsymbol{x}}_0$, it follows from (80) that

$$\lim_{k\to\infty}\sup\|\boldsymbol{z}_k\| \leq \gamma\|\boldsymbol{w}\|_{\ell_\infty}. \tag{81}$$

Conditions (76), (80) and (81) guarantee Property (P2), which concludes the proof. □

Remark 10. Using some congruence matrix transformations, the LPV observer-based control design in Theorem 1 offers an extra degree of freedom to enable less conservative results. Specifically, slack variables Q and G are introduced such that the parameter-dependent matrices $P_c(\eta)$ and $P_o(\eta)$ of the Lyapunov function $\mathcal{V}(\bar{\boldsymbol{x}})$ defined in (58) can be decoupled from any products with the state-space matrices. Moreover, the feedback control gain $K(\eta)$ and the observer gain $L(\eta)$, given in (64), do not *explicitly* depend on $P_c(\eta)$ and $P_o(\eta)$, respectively. It is important to note that the design conservatism of the control results in Theorem 1 can be further reduced using parameter-dependent slack variables as $Q(\eta)$ and $G(\eta)$. However, in this case the expressions of the control and observer gains require online matrix inversions, *i.e.*, $K(\eta) = M(\eta)Q^{-1}(\eta)$ and $L(\eta) = G^{-1}(\eta)N(\eta)$, which could induce some difficulties in control tuning for practical uses [74, 75].

Theorem 1 cannot be directly used for control design due to the presence of $\eta(\boldsymbol{\theta}_k), \eta(\boldsymbol{\theta}_{k+1}) \in \Omega$ in the LMI-based matrix inequalities (61)–(63). It is important to note that the Gaussian membership functions $\eta_i(\boldsymbol{\theta}_k)$, for $i \in \mathcal{I}_N$, are *locally* defined. Hence, there is no overlap between two membership functions $\eta_i(\boldsymbol{\theta}_k)$ and $\eta_j(\boldsymbol{\theta}_k)$, which are not adjacent to each other as illustrated in Figure 2. As a result, the product of these two membership functions are identically zero, *i.e.*, $\eta_i(\boldsymbol{\theta}_k)\eta_j(\boldsymbol{\theta}_k) = 0$, for $\forall k \in \mathbb{N}$. Exploiting this fact and the LMI relaxation result proposed in [76], the following theorem provides a tractable solution for Problem 1 while reducing the design conservatism and numerical complexity.

Theorem 2. Consider the LPV robot model (35) with the control law (38) and a decay rate $\alpha \in (0,1)$. Assume that the maximum number of membership functions that are activated for all $k \in \mathbb{N}$ is less than or equal to s where $1 < s \leq N$. Then, the closed-loop system (57) verifies the properties defined in Problem 1 with a guaranteed minimum $\ell_\infty-$gain if there exist positive definite matrices $P_{oi} \in \mathbb{R}^{(n+3p)\times(n+3p)}$, $P_{ci} \in \mathbb{R}^{(n+q)\times(n+q)}$, matrices $M_i \in \mathbb{R}^{p\times(n+q)}$, $N_i \in \mathbb{R}^{(n+3p)\times q}$, for $i \in \mathcal{I}_N$, matrices $G \in \mathbb{R}^{(n+3p)\times(n+3p)}$, $Q \in \mathbb{R}^{(n+q)\times(n+q)}$, a positive semidefinite matrix $W \in \mathbb{R}^{(n+q)\times(n+q)}$, and positive scalars μ, ν, solution to the following optimization problem:

$$\text{minimize} \quad \mu + \nu \tag{82}$$

such that

$$\begin{bmatrix} (1-\alpha)P_{oi} & 0 & A_{oi}^\top G^\top - C_o^\top N_i^\top \\ \star & \alpha\nu I & B_{ow}^\top G^\top \\ \star & \star & G + G^\top - P_{ol} \end{bmatrix} \succ 0, \ i,l \in \mathcal{I}_N \tag{83}$$

$$\begin{bmatrix} P_{ci} & 0 & Q^\top C_z^\top \\ \star & P_{oi} & 0 \\ \star & \star & \mu I \end{bmatrix} \succeq 0, \ \ i \in \mathcal{I}_N \tag{84}$$

$$\Theta_{iil} \succ (s-1)\mathbf{W}, \ \ i \in \mathcal{I}_N \tag{85}$$

$$\Theta_{ijl} + \Theta_{jil} \succeq -2\mathbf{W}, \ \ i,j,l \in \mathcal{I}_N, \ i < j \tag{86}$$

for all i and j excepting the pairs (i,j) such that $\eta_i(\boldsymbol{\theta}_k)\eta_j(\boldsymbol{\theta}_k) = 0$, $\forall k \in \mathbb{N}$ and $s > 1$. The quantity Θ_{ijl} and the matrix $\mathbf{W}$ in (85)–(86) are defined as

$$\Theta_{ijl} = \begin{bmatrix} (1-\alpha)P_{ci} & A_{ci}Q + B_{ci}M_j \\ \star & Q + Q^\top - P_{cl} \end{bmatrix}, \ \ \mathbf{W} = \begin{bmatrix} W & 0 \\ \star & 0 \end{bmatrix}.$$

Proof. Since $\eta(\boldsymbol{\theta}_k), \eta(\boldsymbol{\theta}_{k+1}) \in \Omega$, for $\forall k \in \mathbb{N}$, multiplying condition (83) with $\eta_i(\boldsymbol{\theta}_k)\eta_l(\boldsymbol{\theta}_{k+1}) \geq 0$ and summing up for $i,l \in \mathcal{I}_N$, we obtain (61). Similarly, we can prove that condition (84) implies (63). Moreover, using the relaxation result in [76], it follows that conditions (85)–(86) guarantee (62). Note also that the $\ell_\infty-$gain $\gamma = \sqrt{\mu\nu}$ can be minimized via the optimization problem (82). Then, by the result of Theorem 1, we can conclude the proof. □

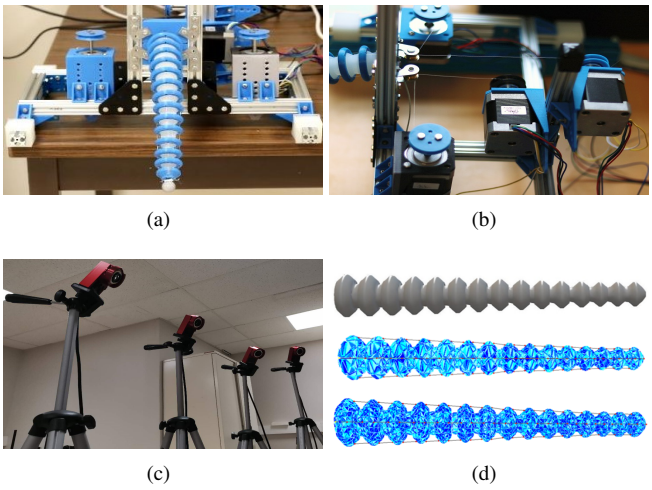
Remark 11. The LPV tracking control design of soft robots is reformulated as a convex optimization problem (82) under LMI constraints (83)–(86), which can be effectively solved offline with YALMIP toolbox and SDPT3 solver [77].

5. Experimental Results of LPV Tracking Control for a Soft Trunk Robot

This section presents the experimental results obtained with a Trunk robot to illustrate the effectiveness of the proposed RBF-based LPV modeling and the EID-based LPV output feedback control method for elastic soft robots.

5.1. Experimental Soft Robot Platform

The Trunk robot platform is depicted in Figure 6. This soft robot is made of silicone rubber with 14 segments to make it highly deformable. The weight of the Trunk robot is 40 [g] and its length is 195 [mm] in the initial position. The Young's modulus of this soft robot is 450 [kPa], whereas its Poisson's ratio is 0.45. The Trunk robot is driven by four stepper motors via four cables mounted on the robot body to guarantee the accessibility of each direction in the workspace. The two control inputs of the robot are realized by pulling these four cables as a pulley system. The position of the end-effector, *e.g.*, system output, is measured by an OptiTrack tracking system and a reflective marker is mounted on the endpoint as an end-effector. The data rate of the OptiTrack tracking system is 100 [Hz] with an accuracy of 0.1 [mm] after preliminary calibrations. The actuator driver code and the communication with the computer are implemented on the MegaPI embedded platform. The control and estimation algorithms are implemented in Matlab/Simulink software on an Intel i9 mobile workstation with 32 GB RAM. Since the robot modeling and the control design are performed offline before real-time experiments, the computational burden of the proposed method is not heavy for soft robots control.



(a) (b) (c) (d)

Figure 6: Soft Trunk robot. (a) Robot platform. (b) Stepper motors. (c) OptiTrack tracking system. (d) FEM modeling of the trunk with different meshes.

5.2. LPV Model Validation

The FEM simulation and modeling of the soft Trunk robot is implemented using the open-source SOFA framework[1]. Two main steps are carried out to achieve a high-fidelity FEM model that can accurately represent the soft Trunk robot dynamics. First, some preliminary investigations on the design, the structure, and the material are done for both the soft robot and its actuation method so that they can be built and modeled as accurately as possible with respect to the robot specifications. Second, after the soft robot is fabricated, several calibrations on the robot parameters and the mesh generation method are performed via FEM modeling and simulation under the SOFA platform to guarantee the best tradeoff between computational

[1]More details on the plugin SoftRobots of the SOFA framework with related publications and/or documentations can be found at the address: https://project.inria.fr/softrobot.

complexity, numerical stability and modeling accuracy. The Trunk robot FEM model has 1484 nodes with 8904 state variables. A four-order reduced model can be obtained via the POD-based order reduction method, see Remark 3. To construct the RBF-based LPV model of the soft Trunk robot, we collect the data of 45 different robot configurations, *i.e.*, equilibrium points, that can cover the whole workspace, see Figure 3. Then, POD-based model reduction method is applied to obtain the 45 corresponding reduced-order linear submodels of the form (16). After some preliminary validations, 9 of these submodels are selected to build the interpolated LPV model of the Trunk robot using Algorithm 1 as illustrated in Figure 3.

To illustrate the nonlinear phenomenon caused by the actuation forces, Figure 7 shows the evolution of the non-zero elements of the reduced-order input matrix

$$B(\eta) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ b_{31}(\eta) & b_{32}(\eta) \\ b_{41}(\eta) & b_{41}(\eta) \end{bmatrix}$$

corresponding to a trajectory with a large deformation as illustrated in Figure 2. First, we can see that the parameter values of the 9 local linearized models, used to establish the reduced-order LPV robot model, are well interpolated with the proposed RBF-based method. Second, the value of $b_{31}(\eta)$, corresponding to the action from the horizontal cable actuator to the x-axis of robot, varies from negative to positive. This illustrates the nonlinear behavior of elastic soft robots concerning the change of the actuation direction as previously discussed.

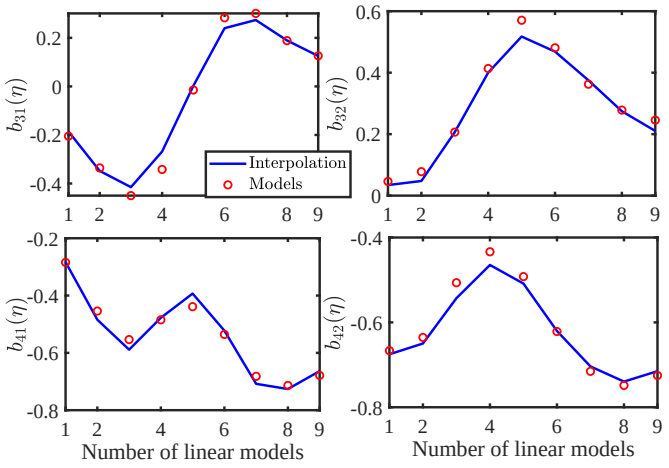


Figure 7: Evolution of the non-zero elements of the input matrix $B(\eta)$: RBF-based interpolation results (——), model parameters from collected data (○).

To further validate the proposed LPV modeling method, we compare the following robot models:

- the linear model in [15], corresponding to the equilibrium point $(\boldsymbol{q}_0, \boldsymbol{v}_0, \boldsymbol{u}_0) \equiv (0, 0, 0)$;
- the 1484-node nonlinear FEM model used to derive the proposed LPV robot model;
- the proposed LPV model.

For comparison purposes, we apply a ramp input signal to the Trunk robot to *gradually* increase the robot deformation, which can reproduce the nonlinear behavior illustrated in Figure 2. The displacement of the robot end-effector is measured. Besides, simulations are performed with the same input signal for the linear model, the FEM model and the LPV model. The comparison result is shown in Figure 8. Observe that the nonlinear FEM model and the LPV model provide similar behaviors, which capture well the nonlinear dynamics of the soft Trunk robot even if the input signal becomes large. However, the linear reduced-order model can only approximate the robot dynamics under small deformations with a small input signal. Note that the linear model is obtained from the configuration, for which the actuation does not have any impact on the $z-$axis displacement. The above validation results in Figures 7 and 8 confirm the validity of the proposed LPV modeling method. Hereafter, the relevance of this LPV modeling for dynamic tracking control design of soft robots under large deformations is demonstrated.

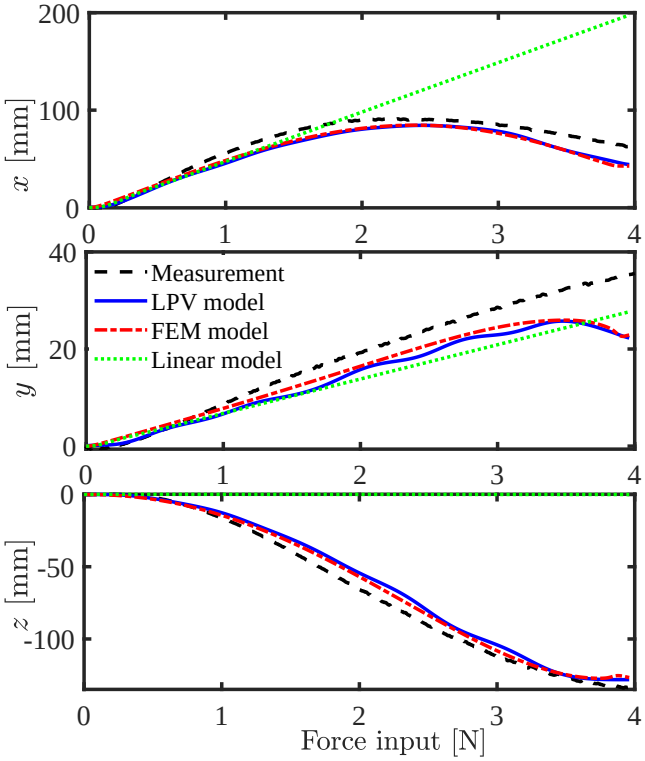


Figure 8: Comparison between different modeling methods for the Trunk robot.

5.3. Tracking Control Validation

To show the effectiveness of the proposed LPV tracking control framework for elastic soft robots, we perform several experimental tests with both large and small deformations. Note that this soft robot is *horizontally* positioned as an elephant trunk, which leads to a distorted sphere workspace due to the gravity effect as depicted in Figure 9. Since the center of mass of the soft Trunk robot changes with respect to its motions and deformations, the gravity effect plays a role of time-varying

disturbance to the robot control system. For comparison purposes, we have recently shown in [15] that the linear EID-based controller therein can outperform, in terms of tracking performance, some standard controllers for soft elastic robots, *e.g.*, inverse kinematics based QP control [18, 44] and Jacobian-based PID control [78, 79]. Hence, in the sequel we mainly focus on the comparisons between the proposed LPV control method, the linear EID-based control method in [15] and a linear iterative learning control (ILC) method. This latter is known as a powerful scheme for dynamic tracking control in the presence of repetitive disturbances [80].

5.3.1. *Test 1: Dynamic Tracking with a Predefined Trajectory*

For this test, we select a reference trajectory that can cover the entire workspace of the Trunk robot. Note that the workspace of the Trunk robot can be parameterized by a spherical coordinate system with two angular coordinates $(\vartheta(t), \varphi(t))$, which enable to easily define the trajectories in the workspace. For illustrations, we consider a reference trajectory with the following altitude and azimuth angles:

$$\vartheta(t) = 2.1\sin(0.3t)\ \text{[rad]}, \quad \varphi(t) = 0.5\cos(0.3t)\ \text{[rad]} \quad (87)$$

which corresponds to a large circular trajectory in the workspace. Figure 9 depicts the evolution of the end-effector positions of the Trunk robot within the shell-like workspace, which are obtained with the proposed LPV control method and the linear control method in [15]. We can see that the linear control is only valid a small operating region of the workspace. In contrast, LPV control can provide an effective tracking for the whole predefined trajectory. To examine the tracking control performance in more detail, Figures 10 and 11 present the tracking control results, projected on the $\vartheta\varphi-$plane. Remark that both LPV and linear control methods share a similar tracking performance around the initial robot configuration. However, after reaching the singular configuration of the Trunk robot, *i.e.*, with a change of the actuation direction, the linear control method is unable to perform the tracking task while the LPV control method still offers a satisfactory reference tracking result. The corresponding control inputs are shown in Figure 12. Note that the $\vartheta-$axis force control input has a sinus shape, which is synchronized with the $\vartheta-$axis trajectory. However, the $\varphi-$axis control input is similar to a square wave, which does not correspond to the form of the $\varphi-$axis trajectory. This is not the case of the dynamic tracking control with small deformations in [15], which also illustrates the effects of large nonlinearities, *i.e.*, large deformations, in soft robots control. The validation video for this test can be found at: https://bit.ly/3RIXlsF.

5.3.2. *Test 2: Robustness with respect to External Disturbances*

The EID-based control concept has been shown in [15] as an effective tool to deal with the parametric uncertainties of elastic soft robots, analyzed in Section 2.3, for linear dynamic tracking control. This test is used to demonstrate that this control concept is also useful for interpolated LPV control framework to deal with not only parametric uncertainties but also external disturbances. To this end, we add an extra load near the

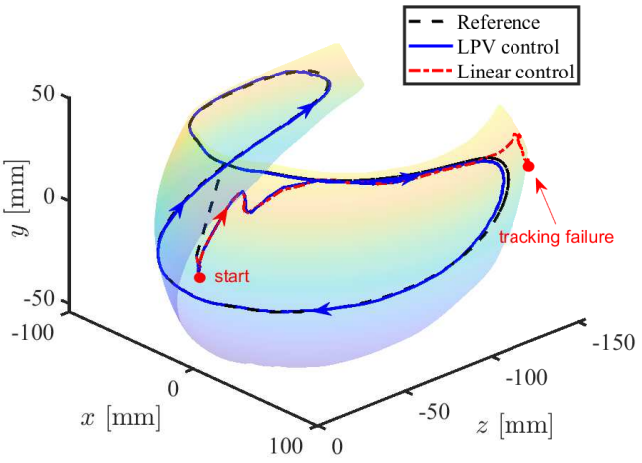


Figure 9: Experimental results of 3D dynamic tracking control with a circular reference trajectory (Test 1).

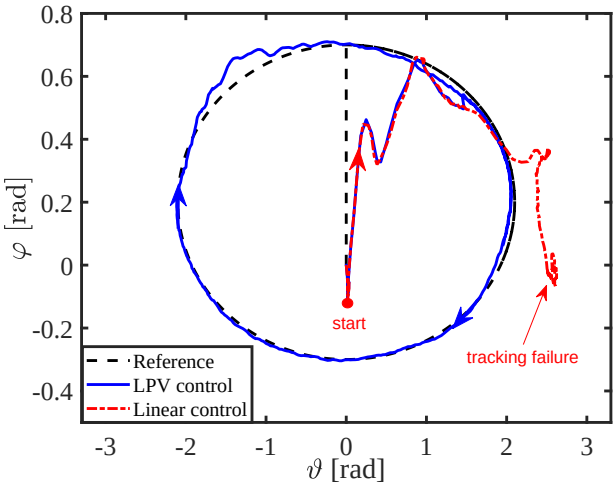


Figure 10: Experiment tracking control results with a circular reference trajectory projected on the spherical coordinate $\vartheta\varphi-$plane (Test 1).

end-effector of the Trunk robot as an external disturbance during the trajectory tracking as shown in Figure 13. The load contains 3 coins with a total mass of 24.8 [g], which is about 15% of the robot weight. The tracking task of this test is composed of two phases with the same reference trajectory as defined in (87), and the load is added in the second phase for comparison purposes. The corresponding tracking control result in 3D and its projection on each axis are presented in Figures 14 and 15, respectively. We can see that the disturbance effect, caused by the extra load added at around 27 [s], is quickly compensated. Indeed, there is no significant difference after about 1 [s] on the tracking control performance between Phase 1 and Phase 2. We can observe in Figure 15(c) that due to the presence of the additional load, the $\varphi-$axis force control input is also reduced accordingly. Since the extra load only affects to the $\varphi-$axis, there is no change for the $\vartheta-$axis force input between Phase 1 and Phase 2. The validation video for this test can be found at: https://bit.ly/3RMDgBO.

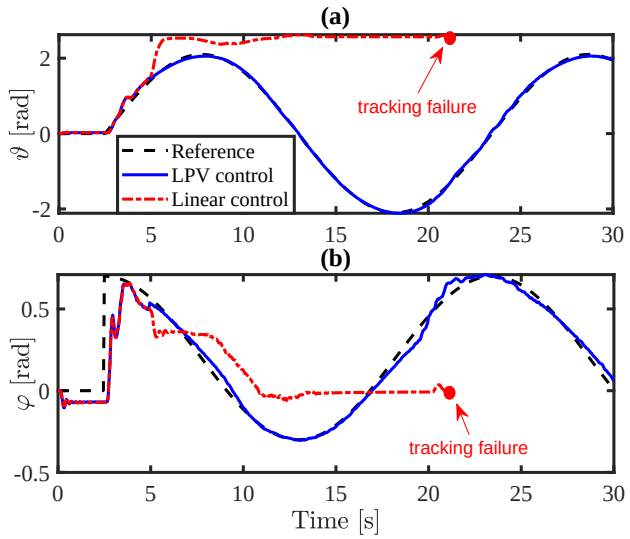


Figure 11: Tracking control task in the parametric space (Test 1). (a) Tracking performance along $\vartheta-$axis. (b) Tracking performance along $\varphi-$axis.

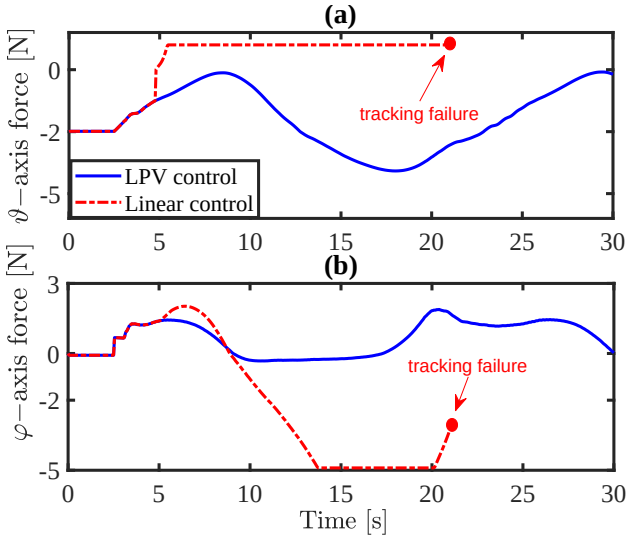


Figure 12: Force control inputs along the $\vartheta-$axis and the $\varphi-$axis (Test 1).

5.3.3. *Test 3: Comparisons with Iterative Learning Control*

ILC aims at generating a feedforward control to track a reference trajectory of repetitive processes on a finite time interval while rejecting real-time disturbances [81–83]. For comparisons, we perform a circular trajectory tracking with the norm-optimal ILC control [84], which shares the common model-based principle as iterative learning model predictive control [85]. The designed ILC control scheme is composed of two decentralized single-input single-output (SISO) ILC controllers corresponding to the $\vartheta-$axis and the $\varphi-$axis, respectively. The design of each norm-optimal ILC controller is formulated as an optimization problem, as described in Appendix A. We distinguish two test scenarios for trajectory tracking: i) with a small robot deformation, and ii) with a large robot deformation.

a. Scenario 1: Tracking with a Small Deformation. This test scenario is performed with the following small-range reference

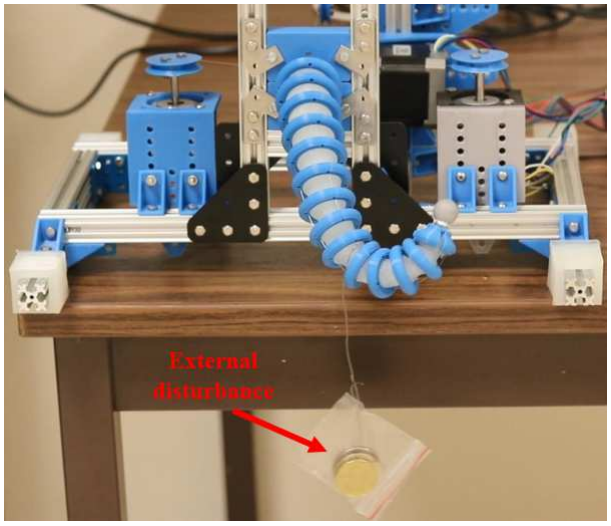


Figure 13: Soft Trunk robot with an additional load (Test 2).

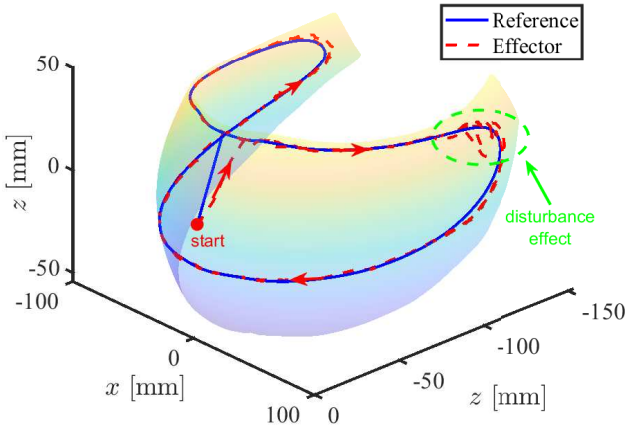


Figure 14: Experimental results of 3D dynamic tracking control in the presence of an external disturbance (Test 2).

such that linear control can be still effective:

$$\vartheta(t) = 0.4\sin(0.9t)\ [\text{rad}], \quad \varphi(t) = 0.4\cos(0.9t)\ [\text{rad}]. \quad (88)$$

The tracking control results obtained with both LPV control and ILC control are shown in Figure 16. We can see in Figures 16(a) and (c) that the tracking errors become smaller after each control iteration. After 15 iterations, the robot end-effector can track the desired reference (88) with the ILC control method. Moreover, the tracking control performance of both control methods is similar in this case.

b. Scenario 2: Tracking with a Large Deformation. For this test, the tracking control task is performed with the large-range reference defined in (87) to show that linear control is not effective anymore for large-deformation situations. The corresponding tracking control results are depicted in Figure 17. Observe that ILC control can improve loop-after-loop the tracking performance for some first iterations, *i.e.*, when the deformation is still small. However, when the deformation becomes

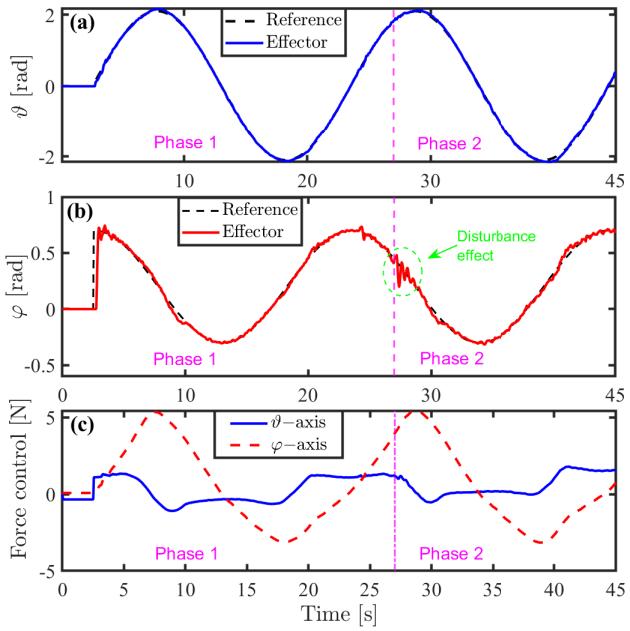


Figure 15: Tracking control task with an external disturbance in the parametric space (Test 2). (a) Tracking performance along $\vartheta-$axis. (b) Tracking performance along $\varphi-$axis. (c) Force control inputs.

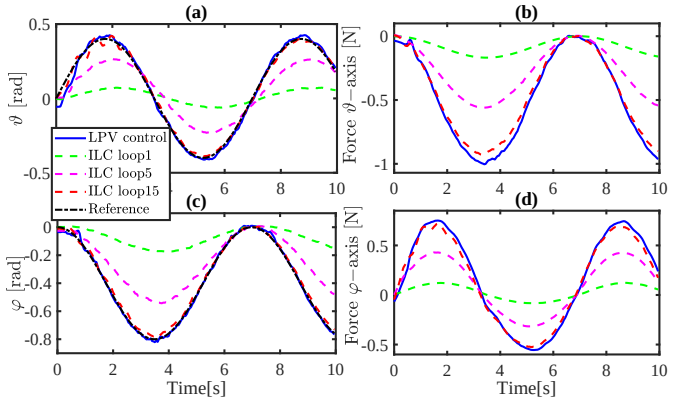


Figure 16: Tracking performance comparison between LPV control and different loops of ILC control for the small-deformation reference (88). (a) Tracking performance along $\vartheta-$axis. (b) Force control input along $\vartheta-$axis. (c) Tracking performance along $\varphi-$axis. (d) Force control input along $\varphi-$axis.

large, the ILC controller is not effective anymore to track the desired reference. In particular, the end-effector is not able to reach the robot configuration with 90° bending deformation, which is not the case of the proposed LPV controller. The validation video for both scenarios of this test can be found at: https://bit.ly/3xLb3CH.

For a quantitative performance analysis, we define the following normalized square tracking error (nSTE) index to avoid the impact of different reference amplitudes:

$$\text{nSTE} = \frac{1}{\|\boldsymbol{x}_r\|^2_{\ell_\infty}} \sum_{k=1}^{\Delta t} \|\boldsymbol{e}_k\|^2$$

where Δt is the tracking time. Figure 18 summarizes the tracking performance in terms of nSTE index obtained with the proposed LPV control and the compared ILC controller for both small- and large-range trajectory references. Remark that for the small-range tracking, a clear performance improvement can be observed loop after loop for ILC control until a high tracking accuracy can be achieved. However, for the large-range tracking, despite a loop-after-loop improvement, the nSTE values obtained with ILC controller are large compared to that obtained with the proposed LPV controller, which gives similar nSTE values for both test scenarios. These confirm the tracking control results in Figures 16 and 17.

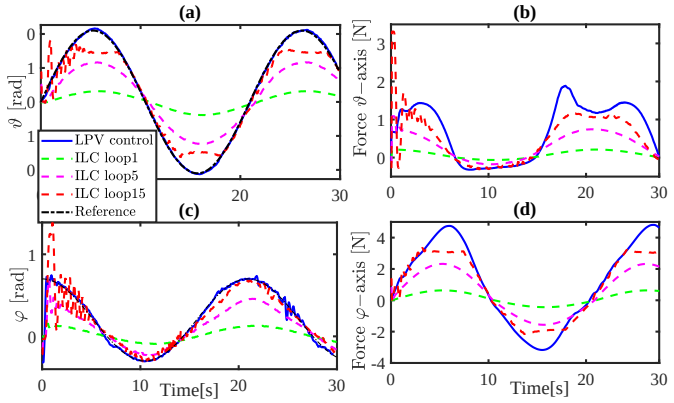


Figure 17: Tracking performance comparison between LPV control and different loops of ILC control for the large-deformation reference (87). (a) Tracking performance along $\vartheta-$axis. (b) Force control input along $\vartheta-$axis. (c) Tracking performance along $\varphi-$axis. (d) Force control input along $\varphi-$axis.

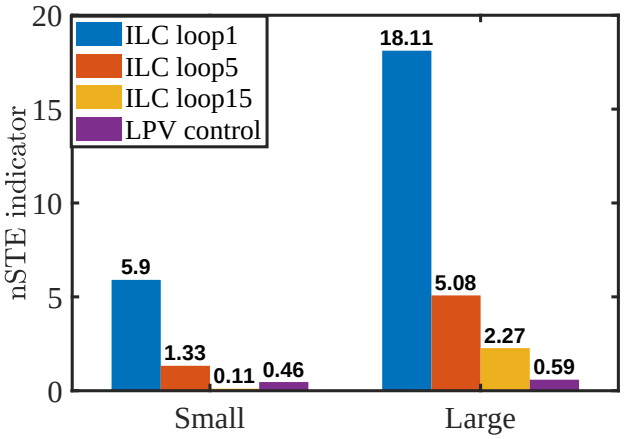


Figure 18: Comparisons between LPV controller and ILC controller in terms of nSTE performance for both small and large deformation test scenarios.

5.3.4. *Test 4: Real-Time Marker Tracking*

To further explore the potential of the proposed LPV control method, a more realistic experiment is conducted. The task is to imitate the elephant trunk to reach a given target, *i.e.*, to minimize the distance between the end-effector and the target, which is materialized by a marker as illustrated in Figure 19. Note that the marker can be manually and arbitrarily moved within the workspace of the soft Trunk robot.

The 3D trajectories of the end-effector and the target are shown in Figure 20. For this test, we pay a special attention

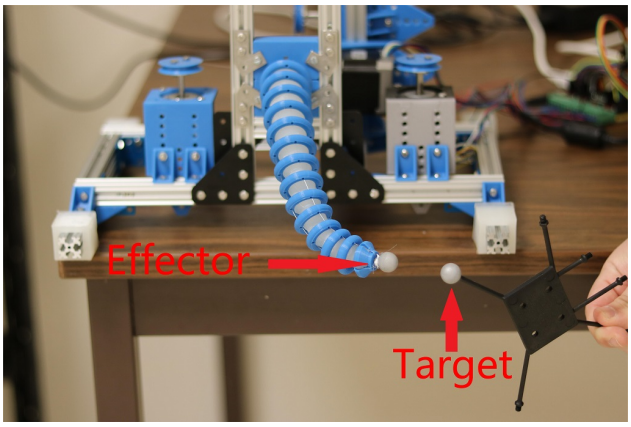


Figure 19: Illustration of a real-time target tracking task with the Trunk robot.

on the highly deformed configuration of the Trunk robot and the random and fast time-varying features of the target trajectory, which makes the tracking task much more challenging compared to the previous tests. Figure 21 shows that the real-time marker tracking is successfully achieved with the proposed LPV control method. The validation video for this test can be found at: https://bit.ly/3KMDIOr.

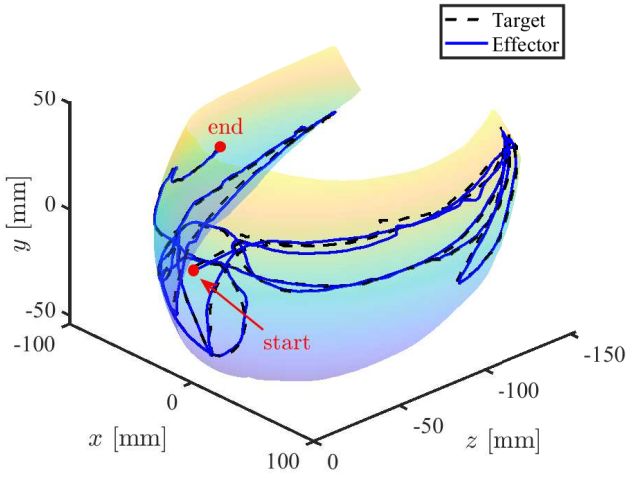


Figure 20: A 3D view of the real-time marker tracking task (Test 4).

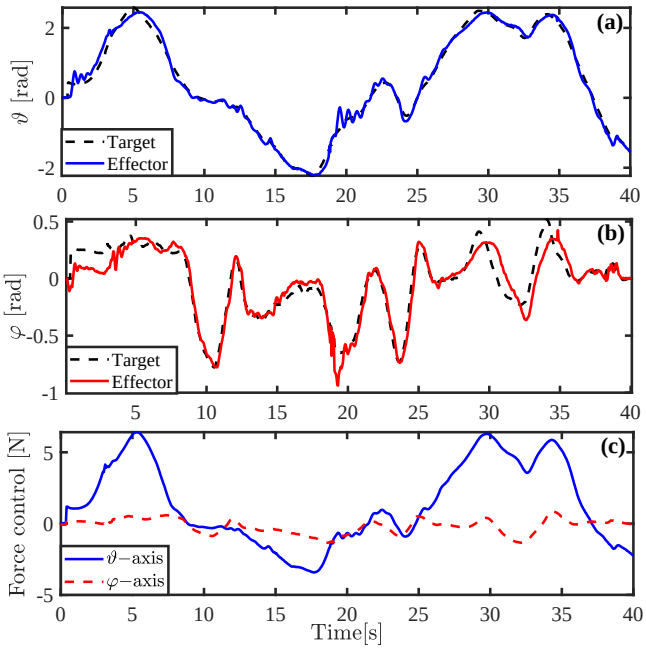


Figure 21: Real-time marker tracking task in the parametric space (Test 4). (a) Tracking performance along $\vartheta-$axis. (b) Tracking performance along $\varphi-$axis. (c) Force control inputs.

6. Concluding Remarks

A dynamic FEM model-based framework has been proposed for LPV tracking control of elastic soft robots. Based on a POD model reduction method, we first generate a set of reduced-order linear models with the same mechanical structure for different operating points, covering the whole robot workspace. Using RBF networks, these local linearized models are interpolated to build a reduced-order LPV model, which can capture the nonlinear dynamics of soft robots with large deformations. Then, an EID-based control scheme is developed for LPV dynamic tracking control, which is composed of three key components, *i.e.*, feedforward control, feedback control, error-compensation control. The LPV feedforward control is constructed from an interpolation of local control input values, obtained at the operating points considered for LPV modeling. The feedback control and the error-compensation control are designed with a generalized proportional integral observer structure, incorporating the low-frequency information of external disturbances and/or modeling uncertainties. Using Lyapunov stability theory, sufficient LMI conditions are derived to design both the LPV feedback controller and the LPV extended observer such that the closed-loop soft robot system is globally uniformly $\ell_\infty-$stable. Various experimental tests have been carried out with a soft Trunk robot under configurations with both small and large deformations to validate the proposed LPV modeling and to show the effectiveness of the proposed LPV control method over existing linear tracking control results. Thanks to its generic feature, the proposed LPV control framework can be extended to deal with soft robots with more complex structures and/or with more constraints on the robot segments. However, these complex situations require a special attention on the selection of robot sensors and of the scheduling variables to characterize adequately the variation of the robots characteristics. In particular, we have to obtain appropriate feasible references of the robot states for the LPV feedback controller, considered as a "low level" robot controller as in rigid robotics, to achieve desired tracking tasks in these cases. This will require further research investigations on kinematics and inverse kinematics of soft robots. Extensions of the proposed LPV control results for contact handling of soft robots in interaction with an unstructured environment is another promising future direction.

Appendix A. Norm-Optimal ILC Controller Design

To design two decentralized ILC controllers, the reduced-order robot model (16) is decoupled into two SISO models corresponding to the control along the $\vartheta-$axis and the $\varphi-$axis, respectively. The norm-optimal ILC design for these both controllers follows the same procedure, which is summarized hereafter with the same formulation. More related technical details can be found in [86].

Under zero initial condition, the following impulse response matrix is derived from the linear robot model (16):

$$\mathbf{P} = \begin{bmatrix} C_p B_{rp} & 0 & \cdots & 0 \\ C_p A_r B_{rp} & C_p B_{rp} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ C_p A_r^{T-1} B_{rp} & C_p A_r^{T-2} B_{rp} & \cdots & C_p B_{rp} \end{bmatrix}$$

where T is the iteration length, *i.e.*, $k \in [0, T]$. The pth input-output matrix pair (B_{rp}, C_p) corresponds to the SISO control loop along the $\vartheta-$axis or the $\varphi-$axis. The system output of the ith iteration can be represented by

$$\boldsymbol{y}_{k(i)} = \mathbf{P}\boldsymbol{u}_{k(i)} \tag{A.1}$$

where $\boldsymbol{u}_{k(i)}$ is the control input of the ith iteration. The norm-optimal ILC control design can be reformulated as the following optimization problem [86]:

$$\min_{\boldsymbol{u}_{k(i+1)}} \boldsymbol{e}_{k(i+1)}^{\top} W_e \boldsymbol{e}_{k(i+1)} + u_{k(i+1)}^{\top} W_u \boldsymbol{u}_{k(i+1)} \tag{A.2}$$

$$\text{subject to } \left[\boldsymbol{u}_{k(i+1)} - \boldsymbol{u}_{k(i)}\right]^{\top} \left[\boldsymbol{u}_{k(i+1)} - \boldsymbol{u}_{k(i)}\right] \leq \delta \tag{A.3}$$

where $\boldsymbol{e}_{k(i)} = \boldsymbol{r}_k - \boldsymbol{y}_{k(i)}$, and $\boldsymbol{r}_k$ is the output reference. The weighting matrices W_e and W_u determine the tradeoff between performance and input energy. The constraint (A.3) can be taken into account in the optimization problem (A.2) via a Lagrange multiplier λ as

$$\min_{\boldsymbol{u}_{k(i+1)}} \mathscr{J}_{k(i+1)} \tag{A.4}$$

with

$$\begin{aligned} \mathscr{J}_{k(i+1)} = {} & \boldsymbol{e}_{k(i+1)}^{\top} W_e \boldsymbol{e}_{k(i+1)} + \boldsymbol{u}_{k(i+1)}^{\top} W_u \boldsymbol{u}_{k(i+1)} \\ & + \lambda \left[\left(\boldsymbol{u}_{k(i+1)} - \boldsymbol{u}_{k(i)}\right)^{\top} \left(\boldsymbol{u}_{k(i+1)} - \boldsymbol{u}_{k(i)}\right) - \delta \right]. \end{aligned}$$

Then, the optimal solution of the optimization problem (A.4) can be determined from the equation $\frac{\partial \mathscr{J}_{k(i+1)}}{\partial \boldsymbol{u}_{k(i+1)}} = 0$, leading to

$$\boldsymbol{u}_{k(i+1)} = \mathbf{W}_{\text{opt}}^{-1} \left(\lambda \boldsymbol{u}_{k(i)} + \mathbf{P}^{\top} W_e (\mathbf{I} - \mathbf{P}) \boldsymbol{r}_k \right) \tag{A.5}$$

with $\mathbf{W}_{\text{opt}} = \lambda \mathbf{I} + \mathbf{P}^{\top} W_e \mathbf{P} + W_u$. Substituting (A.1) into (A.5), the update law of the control input at the ith iteration can be obtained as

$$\boldsymbol{u}_{k(i+1)} = \mathbf{Q} \left[\boldsymbol{u}_{k(i)} + \mathbf{L} e_{k(i)} \right] \tag{A.6}$$

where the filter matrices $\mathbf{Q}$ and $\mathbf{L}$ are defined as

$$\begin{aligned} \mathbf{Q} &= \mathbf{W}_{\text{opt}}^{-1} \left(\lambda \mathbf{I} + \mathbf{P}^{\top} W_e \mathbf{P} \right) \\ \mathbf{L} &= \left(\lambda \mathbf{I} + \mathbf{P}^{\top} W_e \mathbf{P} \right)^{-1} \mathbf{P}^{\top} W_e. \end{aligned} \tag{A.7}$$

Remark 12. For ILC tuning, we select $W_e = I$ and $W_u = \rho I$, with $\rho \in [0, 1]$, to limit the computational burden of the control update law (A.6)–(A.7). The tuning of the two design parameters λ and ρ can be done as follows. Since the soft robot dynamics cannot be accurately described with a linear model, the value of λ should be large enough to maintain the system stability by avoiding an aggressive update in (A.6). The value of ρ should be small to minimize the tracking error $\boldsymbol{e}_{k(i)}$. For the experimental results in Section 5, we select $\lambda = 1.2$ and $\rho = 0.1$ for both SISO norm-optimal ILC controllers.

Acknowledgment

The present work (under the RITMEA research program) has been supported by the European Community, the Délégation Régionale à la Recherche et à la Technologie, the Ministère de l'Education Nationale, de la Recherche et de la Technologie, the Hauts-de-France region and the Centre National de la Recherche Scientifique. It is also supported by the University of Lille, Centrale Lille. The authors gratefully acknowledge the support of these institutions.

References

[1] D. Rus, M. Tolley, Design, fabrication and control of soft robots, Nature 521 (7553) (2015) 467–475.

[2] D. Trivedi, C. Rahn, W. Kier, I. Walker, Soft robotics: Biological inspiration, state of the art, and future research, Appl. Bionics. Biomech. 5 (3) (2008) 99–117.

[3] M. Wehner, R. Truby, D. Fitzgerald, B. Mosadegh, G. Whitesides, J. Lewis, R. Wood, An integrated design and fabrication strategy for entirely soft, autonomous robots, Nature 536 (7617) (2016) 451–455.

[4] C. Laschi, B. Mazzolai, M. Cianchetti, Soft robotics: Technologies and systems pushing the boundaries of robot abilities, Sci. Robot. 1 (1) (2016) eaah3690.

[5] T. Umedachi, V. Vikas, B. Trimmer, Softworms: the design and control of non-pneumatic, 3D-printed, deformable robots, Bioinspir. Biomim. 11 (2) (2016) 025001.

[6] J. Cao, W. Liang, Y. Wang, H. Lee, J. Zhu, Q. Ren, Control of a soft inchworm robot with environment adaptation, IEEE Trans. Indus. Electron. 67 (5) (2020) 3809–3818.

[7] B. Gamus, L. Salem, A. Gat, Y. Or, Understanding inchworm crawling for soft-robotics, IEEE Robot. Autom. Lett. 5 (2) (2020) 1397–1404.

[8] A. Marchese, R. Katzschmann, D. Rus, A recipe for soft fluidic elastomer robots, Soft Robot. 2 (1) (2015) 7–25.

[9] Q. Li, J. Qian, Z. Zhu, X. Bao, M. Helwa, A. Schoellig, Deep neural networks for improved, impromptu trajectory tracking of quadrotors, in: IEEE Int. Conf. Robot. Autom. (ICRA), 2017, pp. 5183–5189.

[10] J. Shintake, V. Cacucciolo, H. Shea, D. Floreano, Soft biomimetic fish robot made of dielectric elastomer actuators, Soft Robot. 5 (4) (2018) 466–474.

[11] A. Jusufi, D. Vogt, R. Wood, G. Lauder, Undulatory swimming performance and body stiffness modulation in a soft robotic fish-inspired physical model, Soft Robot. 4 (3) (2017) 202–210.

[12] M. Rolf, J. Steil, Efficient exploratory learning of inverse kinematics on a bionic elephant trunk, IEEE Trans. Neural Netw. Learn. Syst. 25 (6) (2013) 1147–1160.

[13] Q. Guan, J. Sun, Y. Liu, N. Wereley, J. Leng, Novel bending and helical extensile/contractile pneumatic artificial muscles inspired by elephant trunk, Soft Robot. 7 (5) (2020) 597–614.

[14] J. Zhang, X. Chen, P. Stegagno, C. Yuan, Nonlinear dynamics modeling and fault detection for a soft Trunk robot: An adaptive NN-based approach, IEEE Robot. Autom. Lett. 7 (3) (2022) 7534–7541.

[15] S. Li, A. Kruszewski, T.-M. Guerra, A.-T. Nguyen, Equivalent-input-disturbance-based dynamic tracking control for soft robots via reduced-order finite-element models, IEEE/ASME Trans. Mechatron. 27 (5) (2022) 4078–4089.
[16] C. Laschi, M. Cianchetti, B. Mazzolai, L. Margheri, M. Follador, P. Dario, Soft robot arm inspired by the octopus, Adv. Robot. 6 (7) (2012) 709–727.
[17] Z. Xie, A. Domel, N. An, C. Green, Z. Gong, T. Wang, E. Knubben, J. Weaver, K. Bertoldi, L. Wen, Octopus arm-inspired tapered soft actuators with suckers for improved grasping, Soft Robot. 7 (5) (2020) 39–48.
[18] T. Bieze, A. Kruszewski, B. Carrez, C. Duriez, Design, implementation, and control of a deformable manipulator robot based on a compliant spine, Int. J. Rob. Res. 39 (14) (2020) 604–619.
[19] X. Yang, T.-H. Huang, H. Hu, S. Yu, S. Zhang, X. Zhou, A. Carriero, G. Yue, H. Su, Spine-inspired continuum soft exoskeleton for stoop lifting assistance, IEEE Robot. Autom. Lett. 4 (4) (2019) 4547–4554.
[20] G. Qin, A. Ji, Y. Cheng, W. Zhao, H. Pan, S. Shi, Y. Song, A snake-inspired layer-driven continuum robot, Soft Robot. 9 (4) (2022) 788–797.
[21] B. Jones, I. Walker, Kinematics for multisection continuum robots, IEEE Trans. Robot. 22 (1) (2006) 43–55.
[22] Z. Patterson, A. Sabelhaus, C. Majidi, Robust control of a multi-axis shape memory alloy-driven soft manipulator, IEEE Robot. Autom. Lett. 7 (2) (2022) 2210–2217.
[23] D. Bruder, X. Fu, B. Gillespie, D. Remy, R. Vasudevan, Data-driven control of soft robots using Koopman operator theory, IEEE Trans. Robot. 37 (3) (2020) 948–961.
[24] G. Fang, X. Wang, K. Wang, K.-H. Lee, J. Ho, H.-C. Fu, D. Fu, K.-W. Kwok, Vision-based online learning kinematic control for soft robots using local Gaussian process regression, IEEE Robot. Autom. Lett. 4 (2) (2019) 1194–1201.
[25] T. Thuruthel, Y. Ansari, E. Falotico, C. Laschi, Control strategies for soft robotic manipulators: A survey, Soft Robot. 5 (2) (2018) 149–163.
[26] J. Bern, Y. Schnider, P. Banzet, N. Kumar, S. Coros, Soft robot control with a learned differentiable model, in: 3rd IEEE Int. Conf. Soft Robot. (RoboSoft), 2020, pp. 417–423.
[27] C. Della Santina, C. Duriez, D. Rus, Model based control of soft robots: A survey of the state of the art and open challenges, arXiv preprint arXiv:2110.01358 (2021).
[28] A. Berlinet, C. Thomas-Agnan, Reproducing Kernel Hilbert Spaces in Probability and Statistics, Springer US, 2004.
[29] M. Thieffry, A. Kruszewski, C. Duriez, T.-M. Guerra, Control design for soft robots based on reduced-order model, IEEE Robot. Autom. Lett. 4 (1) (2018) 25–32.
[30] E. Coevoet, A. Escande, C. Duriez, Optimization-based inverse model of soft robots with contact handling, IEEE Robot. Autom. Lett. 2 (3) (2017) 1413–1419.
[31] C. Della Santina, R. Katzschmann, A. Bicchi, D. Rus, Model-based dynamic feedback control of a planar soft robot: Trajectory tracking and interaction with the environment, Int. J. Robot. Res. 39 (4) (2020) 490–513.
[32] H. Wang, B. Yang, Y. Liu, W. Chen, X. Liang, R. Pfeifer, Visual servoing of soft robot manipulator in constrained environments with an adaptive controller, IEEE/ASME Trans. Mechatron. 22 (1) (2016) 41–50.
[33] R. Webster, B. Jones, Design and kinematic modeling of constant curvature continuum robots: A review, Int. J. Rob. Res. 29 (3) (2010) 661–683.
[34] Z. Zhang, J. Dequidt, A. Kruszewski, F. Largilliere, C. Duriez, Kinematic modeling and observer based control of soft robot using real-time finite element method, in: 2016 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), 2016, pp. 5509–5514.
[35] S. Satheeshbabu, N. K. Uppalapati, G. Chowdhary, G. Krishnan, Open loop position control of soft continuum arm using deep reinforcement learning, in: Int. Conf. Robot. Autom. (ICRA), 2019, pp. 5133–5139.
[36] A. Centurelli, L. Arleo, A. Rizzo, S. Tolu, C. Laschi, E. Falotico, Closed-loop dynamic control of a soft manipulator using deep reinforcement learning, IEEE Robot. Autom. Lett. 7 (2) (2022) 4741–4748.
[37] M. Ishige, T. Umedachi, T. Taniguchi, Y. Kawahara, Exploring behaviors of caterpillar-like soft robots with a central pattern generator-based controller and reinforcement learning, Soft Robot. 6 (5) (2019) 579–594.
[38] G. Mengaldo, F. Renda, S. L. Brunton, M. Bächer, M. Calisti, C. Duriez, G. S. Chirikjian, C. Laschi, A concise guide to modelling the physics of embodied intelligence in soft robotics, Nat. Rev. Phys. (2022) 1–16.
[39] D. Cao, R. W. Tucker, Nonlinear dynamics of elastic rods using the Cosserat theory: Modelling and simulation, Int. J. Solids Struct. 45 (2) (2008) 460–477.
[40] D. Trivedi, A. Lotfi, C. Rahn, Geometrically exact models for soft robotic manipulators, IEEE Trans. Robot. 24 (4) (2008) 773–780.
[41] C. Rucker, R. Webster, Statics and dynamics of continuum robots with general tendon routing and external loading, IEEE Trans. Robot. 27 (6) (2011) 1033–1044.
[42] H. Wang, C. Wang, W. Chen, X. Liang, Y. Liu, Three-dimensional dynamics for cable-driven soft manipulator, IEEE/ASME Trans. Mechatron. 22 (1) (2016) 18–28.
[43] F. Renda, F. Boyer, J. Dias, L. Seneviratne, Discrete Cosserat approach for multisection soft manipulator dynamics, IEEE Trans. Robot. 34 (6) (2018) 1518–1533.
[44] C. Duriez, Control of elastic soft robots based on real-time finite element method, in: IEEE Int. Conf. Robot. Autom., 2013, pp. 982–987.
[45] E. Coevoet, et al., Software toolkit for modeling, simulation, and control of soft robots, Adv. Robot. 31 (22) (2017) 1208–1224.
[46] S. Navarro, S. Nagels, H. Alagi, L.-M. Faller, O. Goury, T. Morales, H. Zangl, B. Hein, R. Ramakers, W. Deferme, G. Zheng, C. Duriez, A model-based sensor fusion approach for force and shape estimation in soft robotics, IEEE Robot. Autom. Lett. 5 (4) (2020) 5621–5628.
[47] W. Zimmerman, Multiphysics Modeling with Finite Element Methods, Vol. 18, World Scientific Publishing Company, 2006.
[48] M. Thieffry, A. Kruszewski, T.-M. Guerra, C. Duriez, Trajectory tracking control design for large-scale linear dynamical systems with applications to soft robotics, IEEE Trans. Control Syst. Technol. 29 (2) (2021) 56–66.
[49] M. Thieffry, A. Kruszewski, T.-M. Guerra, C. Duriez, LPV framework for non-linear dynamic control of soft robots using finite element model, IFAC-PapersOnLine 53 (2) (2020) 7312–7318.
[50] J. De Caigny, R. Pintelon, J. Camino, J. Swevers, Interpolated modeling of LPV systems, IEEE Trans. Control Syst. Technol. 22 (6) (2014) 32–46.
[51] Q. Zhang, L. Ljung, R. Pintelon, On local LTI model coherence for LPV interpolation, IEEE Trans. Autom. Control 65 (8) (2020) 3671–3676.
[52] S. Vijayakumar, S. Schaal, Locally weighted projection regression: An o(n) algorithm for incremental real time learning in high dimensional space, in: 7th Int. Conf. Machine Learning, Vol. 1, 2000, pp. 288–293.
[53] J.-H. She, X. Xin, Y. Pan, Equivalent-input-disturbance approach–Analysis and application to disturbance rejection in dual-stage feed drive control system, IEEE/ASME Trans. Mechatron. 16 (2) (2010) 330–340.
[54] S. Boyd, L. El Ghaoui, E. Feron, V. Balakrishnan, Linear Matrix Inequalities in System and Control Theory, Vol. 15, SIAM, Philadelphia, 1994.
[55] J.-C. Butcher, Numerical Methods for Ordinary Differential Equations, John Wiley & Sons, 2016.
[56] P. Benner, M. Ohlberger, A. Cohen, K. Willcox, Model Reduction and Approximation: Theory and Algorithms, SIAM, Philadelphia, 2017.
[57] O. Goury, B. Carrez, C. Duriez, Real-time simulation for control of soft robots with self-collisions using model order reduction for contact forces, IEEE Robot. Autom. Lett. 6 (2) (2021) 3752–3759.
[58] F. Schwenker, H. A. Kestler, G. Palm, Three learning phases for radial-basis-function networks, Neural Netw. 14 (4-5) (2001) 439–458.
[59] J. Park, I. Sandberg, Universal approximation using radial-basis-function networks, Neural Comput. 3 (2) (1991) 246–257.
[60] M. Buhmann, Radial Basis Functions: Theory and Implementations, Vol. 12, Cambridge University Press, 2003.
[61] S. Chen, C. Cowan, P. Grant, Orthogonal least squares learning algorithm for radial basis function networks, IEEE Trans. Neural Netw. 2 (2) (1991) 302–309.
[62] R. Toth, Modeling and Identification of Linear Parameter-Varying Systems, Springer-Verlag Berlin Heidelberg, 2010.
[63] M. Johansson, A. Rantzer, K.-E. Arzen, Piecewise quadratic stability of fuzzy systems, IEEE Trans. Fuzzy Syst. 7 (6) (1999) 713–722.
[64] H. Schulte, H. Hahn, Fuzzy state feedback gain scheduling control of servo-pneumatic actuators, Control Eng. Pract. 12 (5) (2004) 639–650.
[65] L. Rodrigues, S. Boyd, Piecewise-affine state feedback for piecewise-affine slab systems using convex optimization, Syst. Control Lett. 54 (9) (2005) 835–853.
[66] W.-H. Chen, J. Yang, L. Guo, S. Li, Disturbance-observer-based control and related methods–An overview, IEEE Trans. Indus. Electron. 63 (2) (2015) 1083–1095.
[67] A. Mohammadi, M. Tavakoli, H. Marquez, F. Hashemzadeh, Nonlinear

disturbance observer design for robotic manipulators, Control Eng. Pract. 21 (3) (2013) 253–267.
[68] J. Huang, S. Ri, L. Liu, Y. Wang, J. Kim, G. Pak, Nonlinear disturbance observer-based dynamic surface control of mobile wheeled inverted pendulum, IEEE Trans. Control Syst. Technol. 23 (6) (2015) 2400–2407.
[69] J. Guerrero-Castellanos, H. Rifaï, V. Arnez-Paniagua, J. Linares-Flores, L. Saynes-Torres, S. Mohammed, Robust active disturbance rejection control via control Lyapunov functions: Application to actuated-ankle–foot-orthosis, Control Eng. Pract. 80 (2018) 49–60.
[70] A. San-Miguel, V. Puig, G. Alenyà, Disturbance observer-based LPV feedback control of a N-DoF robotic manipulator including compliance through gain shifting, Control Eng. Pract. 115 (2021) 104887.
[71] S. Zak, Systems and Control, Vol. 198, New York: Oxford University Press, 2003.
[72] T.-M. Guerra, L. Vermeiren, LMI-based relaxed nonquadratic stabilization conditions for nonlinear systems in the Takagi–Sugeno's form, Automatica 40 (5) (2004) 823–829.
[73] J. Yoneyama, M. Nishikawa, H. Katayama, A. Ichikawa, Output stabilization of Takagi–Sugeno fuzzy systems, Fuzzy Sets Syst. 111 (2) (2000) 253–266.
[74] A.-T. Nguyen, P. Chevrel, F. Claveau, Gain-scheduled static output feedback control for saturated LPV systems with bounded parameter variations, Automatica 89 (2018) 420–424.
[75] P. Li, A.-T. Nguyen, H. Du, Y. Wang, H. Zhang, Polytopic LPV approaches for intelligent automotive systems: State of the art and future challenges, Mech. Syst. Signal Process. 161 (2021) 107931.
[76] K. Tanaka, T. Ikeda, H. Wang, Fuzzy regulators and fuzzy observers: Relaxed stability conditions and LMI-based designs, IEEE Trans. Fuzzy Syst. 6 (2) (1998) 250–265.
[77] J. Löfberg, YALMIP: A toolbox for modeling and optimization in Matlab, in: IEEE Int. Symp. Comput. Aided Control Syst. Des., Taipei, 2004, pp. 284–289.
[78] A. Marchese, D. Rus, Design, kinematics, and control of a soft spatial fluidic elastomer manipulator, Int. J. Robot. Res. 35 (7) (2016) 840–869.
[79] A. Bajo, R. Goldman, N. Simaan, Configuration and joint feedback for enhanced performance of multi-segment continuum robots, in: IEEE Int. Conf. Robot. Autom., 2011, pp. 2905–2912.
[80] K. Moore, Iterative Learning Control for Deterministic Systems, Springer Science & Business Media, 2012.
[81] L. Hladowski, K. Galkowski, W. Nowicka, E. Rogers, Repetitive process based design and experimental verification of a dynamic iterative learning control law, Control Eng. Pract. 46 (2016) 157–165.
[82] L. Blanken, T. Oomen, Multivariable iterative learning control design procedures: From decentralized to centralized, illustrated on an industrial printer, IEEE Trans. Control Syst. Technol. 28 (4) (2019) 1534–1541.
[83] Y. Chen, B. Chu, C. T. Freeman, Y. Liu, Generalized iterative learning control with mixed system constraints: A gantry robot based verification, Control Eng. Pract. 95 (2020) 104260.
[84] J. Ratcliffe, P. Lewin, E. Rogers, J. Hatonen, D. Owens, Norm-optimal iterative learning control applied to gantry robots for automation applications, IEEE Trans. Robot. 22 (6) (2006) 1303–1307.
[85] S.-K. Oh, J. M. Lee, Iterative learning model predictive control for constrained multivariable control of batch processes, Comput. Chem. Eng. 93 (2016) 284–292.
[86] M. Norrlöf, Iterative Learning Control: Analysis, Design, and Experiments, Ph.D. thesis, Linköping University (2000).