Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

LPV framework for Non-Linear Dynamic Control of Soft Robots using Finite Element Model

Maxime Thieffry * **Alexandre Kruszewski** ***
Thierry-Marie Guerra ** **Christian Duriez** ***

* *Sorbonne University, CNRS, ISIR UMR 7222, Paris, France*
** *Polytechnic University Hauts-de-France, CNRS, UMR 8201 - LAMIH F-59313 Valenciennes, France*
*** *Defrost team, Inria, University of Lille, Centrale Lille, CRIStAL - Centre de Recherche en Informatique Signal et Automatique de Lille - UMR 9189, France*

Abstract: This work presents a methodology to control soft robots using a reduced order nonlinear finite element model. The Linear Parameter-Varying (LPV) framework is used both to model the robot along a prescribed trajectory and to design its control law. Model reduction algorithms along with radial basis functions network are used to identify the nonlinear behavior of the robot. Finally, the method is validated through simulation experiments.

Keywords: Soft robotics, Control, Linear parameter-varying systems, Lyapunov methods, Model Reduction

1. INTRODUCTION

The desire to design robots to interact with humans and/or in confined space in contact with the environment pushed the robotics community to invent new paradigms, among which is soft robotics. Biology provides a major inspiration in the design of soft robots: the elephant trunk, octopus arm or snake body lead to the design of the first soft robots (Majidi, 2014; Kim et al., 2013). By soft robots, we mean robots made of deformable materials whose motion are obtained by deformation and where no joints are presents in the structure. Soft robots have an theoretical infinite number of degrees of freedom. Developing a mathematical model suitable to describe the dynamics of such systems is a challenging task as it must be at the same time computationally affordable and sufficiently accurate.

Different methods have been proposed to solve this challenge, like constant curvature (Webster and Jones, 2010). This method eases the modeling of soft structure due to geometrical assumptions that restrict the design of the robots to beam-like structures. It has been successfully applied to many continuum robots with different actuation systems such as cables or pneumatic chambers (Marchese et al., 2014). Initially developed for kinematics studies, this method has been extended to dynamics modeling and used in closed-loop control experiments in (Katzschmann et al., 2019b).

In addition, Cosserat theory provides a geometrically exact method to model soft robots, taking into account large deformations and displacements and handling material nonlinearities (Trivedi et al., 2008). As for the constant curvature method, it is intended for beam-like robots.

Numerical methods have been proposed to model soft robots without limitations on the geometry. The Finite Element Method (FEM) consists in discretizing the structure into a finite number of small elements, the underlying equations coming from continuum mechanics are then solved for each of these elements. A modeling and simulation software dedicated to soft robots is implemented upon the open-source framework SOFA along with a plugin for the specific needs of soft robots (Coevoet et al., 2017). This modeling method can handle most of the geometries, provided that a CAD file for the structure exists.

To design feedback controllers for soft robot, depending if one is interested in controlling the kinematics or dynamics of the structure, it can lead to different controller designs. Many different methods exist to tune a closed-loop controller, depending on the modeling strategy, the targeted application or an optimization criterion. If some results have shown significant results for kinematics control, few methods exist for dynamic control (see (Thuruthel et al., 2018)).

Kinematics controllers may not be sufficient to perform high speed tasks (such as jumping) or high speed obstacle avoidance. Recently, different authors have proposed new methods to control the dynamic behavior of soft robots. A dynamic controller based on constant curvature model is presented in (Falkenhahn et al., 2015) where the model is used to generate a feed-forward action coupled with a PID controller. Also based on piece-wise constant curvature model, authors of (Della Santina et al., 2018) present a dynamic controller that enables dynamic trajec-

⋆ This research was conducted with support of ANR (Project ANR-17-ERC2-0029), the European Union through the European Regional Development Fund (ERDF), the French Ministry of Higher Education and Research, the National Center for Scientific Research (CNRS), and the Hauts-de-France Region.

tory tracking for a continuous soft robot while handling interactions with the environment. Combining Cosserat and Lagrange dynamic models with Ritz-Galerkin methods, (Sadati et al., 2018) presents a real-time model of continuum manipulators that enables nonlinear impedance and configuration control. In addition, model predictive control has been used recently to perform trajectory following tasks (Bruder et al., 2019). Authors use Koopman operator and system identification methods to construct a discrete-time linear model based on which a linear model predictive controller is designed.

2. CONTRIBUTIONS

This paper aims at extending our previous work by considering the non-linear model of the robot for the control design. In (Thieffry et al., 2019), we have presented a control strategy based on a finite element model of soft robot dynamics. and it has been validated on experimental platform in (Katzschmann et al., 2019a). However, the control law was designed using a linear time invariant (LTI) model, this is a known limitation of this work as it constrains the guaranteed domain of the controller. This papers presents a methodology to build reduced order linear parameter varying (LPV) models of soft robots based on a finite element model. A linear controller is then designed for this LPV model.

3. FINITE ELEMENT MODEL

3.1 Linear large-scale model

To overcome the difficulty of providing realistic analytical models of soft robots, we rely on numerical methods to solve the equations coming from continuum mechanics. The finite element method is commonly used in computational mechanics and is implemented within the SOFA framework (Coevoet et al., 2017) to obtain real-time performances. This modeling method consists of discretizing the deformable solid into small elements. The equations are then solved for each of these elements and assembled to obtain the behavior of the hole structure of the solid.

Let us start with the formulation given by the second law of Newton that models the dynamic behavior of a body as:

$$\mathbf{M}(q)\ddot{q} = \mathbf{P}(q) - \mathbf{F}(q,\dot{q}) + \mathbf{H}^T(q)\lambda \tag{1}$$

where $q \in \mathbb{R}^n$ is the vector of generalized degrees of freedom, $\mathbf{M}(q) : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is the inertia matrix, $\mathbf{F}(q,\dot{q}) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ represents the internal forces applied to the structure and $\mathbf{P}(q) : \mathbb{R}^n \to \mathbb{R}^n$ gathers known external forces. Finally, matrix $\mathbf{H}(q) : \mathbb{R}^m \to \mathbb{R}^{n\times m}$ is the matrix containing the actuation directions while $\lambda \in \mathbb{R}^m$ is the vector of actuators forces.

Let $q_0 \in \mathbb{R}^n$ be a stable equilibrium point. It is induced by the gravity field $\mathbf{P}(q_0)$ and the actuation input λ_0. As q_0 is an equilibrium point, it holds:

$$0 = \mathbf{P} - \mathbf{F}(q_0,0) + \mathbf{H}^T(q_o)\lambda_0 \tag{2}$$

The internal force vector $\mathbf{F}$ is a nonlinear function of the positions and the velocities. We apply a Taylor series expansion to $\mathbf{F}$ and make the following first order approximation around this configuration q_0 and it holds:

$$\mathbf{F}(q,v) = \mathbf{F}(q_0,0) - \left.\frac{\partial \mathbf{F}(q,v)}{\partial q}\right|_{\substack{q=q_0\\v=0}} \partial q - \left.\frac{\partial \mathbf{F}(q,v)}{\partial v}\right|_{\substack{q=q_0\\v=0}} \partial v \tag{3}$$

We define the stiffness matrix $\mathbf{K}$ and the damping matrix $\mathbf{D}$ as:

$$\begin{aligned} \mathbf{K} &: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{n\times n} \quad \mathbf{K}(q,v) = \frac{\partial \mathbf{F}(q,v)}{\partial q} \\ \mathbf{D} &: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{n\times n} \quad \mathbf{D}(q,v) = \frac{\partial \mathbf{F}(q,v)}{\partial v} \end{aligned} \tag{4}$$

Let us define the vector of velocity $v \in \mathbb{R}^n$, $v = \dot{q}$, the state $x \in \mathbb{R}^{2n}$ of the robot is then defined as:

$$x = \begin{pmatrix} v \\ q \end{pmatrix} \tag{5}$$

In the following, we assume that the external forces $\mathbf{P}(q)$, the mass matrix $\mathbf{M}(q)$, the stiffness $\mathbf{K}(q,v)$, the damping $\mathbf{D}(q,v)$ and the actuation direction $\mathbf{H}(q)$ are constant around q_0:

$$\begin{aligned} \mathbf{P} &= \mathbf{P}(q) = \mathbf{P}(q_0) \\ \mathbf{H} &= \mathbf{H}(q) = \mathbf{H}(q_0) \\ \mathbf{M} &= \mathbf{M}(q) = \mathbf{M}(q_0) \\ \mathbf{K} &= \mathbf{K}(q,v) = \mathbf{K}(q_0,0) \\ \mathbf{D} &= \mathbf{D}(q,v) = \mathbf{D}(q_0,0) \end{aligned} \tag{6}$$

Finally, we define the displacement vector d as

$$d = q - q_0 \tag{7}$$

From (1) and (3), the equation of motion around this equilibrium point q_0 writes:

$$\mathbf{M}\dot{v} = \mathbf{P} - \mathbf{F}(q_0,0) - \mathbf{K}d - \mathbf{D}v + \mathbf{H}^T\lambda \tag{8}$$

Computing (8)-(2) yields to the following equation, modeling the motion around an equilibrium point q_0:

$$\mathbf{M}\dot{v} = -\mathbf{K}d - \mathbf{D}v + \mathbf{H}^T(\lambda - \lambda_0) \tag{9}$$

Under this assumption and defining a new input $u = \lambda - \lambda_0$, we obtain a linear model of the behaviour of the robot described by the following equation:

$$\mathbf{M}\dot{v} = -\mathbf{K}d - \mathbf{D}v + \mathbf{H}^T u \tag{10}$$

Considering the state vector $x = (v\ d)^T$ and provided that the matrix $\mathbf{M}$ is regular, equation (10) can be written as a Linear Time Invariant (LTI) model:

$$\begin{cases} \dot{x} = \begin{pmatrix} -\mathbf{M}^{-1}\mathbf{D} & -\mathbf{M}^{-1}\mathbf{K} \\ I & 0 \end{pmatrix} x + \begin{pmatrix} -\mathbf{M}^{-1}\mathbf{H}^T \\ 0 \end{pmatrix} u \\ y = Cx \end{cases} \tag{11}$$

where I is the identity matrix of dimension n and the matrix C is a sparse matrix defining the end-effector coordinates.

3.2 Reduced Order Model

In equation (1), the vector q is the displacement of each nodes of the mesh in the three dimensions of space x, y and z. The more precise the model is, the more variables there are in the model. For soft robotics applications, the

mesh of the robot is often made of thousands of variables, that makes system (11) a model of large dimensions. To overcome the dimensionality issue arising when studying these systems, model order reduction is used in this work. Consider a nonlinear model:

$$\dot{x}(t) = f(x(t), u(t)) \ , \ x \in \mathbb{R}^n, \ n \gg 1 \tag{12}$$

projection-based model order reduction consists of decomposing the full-order state x into two parts, a low-order state $x_r \in \mathbb{R}^r$ and a neglected state $x_{\bar{r}} \in \mathbb{R}^{n-r}$ such that:

$$x = V_r x_r + V_{\bar{r}} x_{\bar{r}} \quad \text{with} \quad \begin{cases} x_r = W_r^T x \\ x_{\bar{r}} = W_{\bar{r}}^T x \end{cases} \tag{13}$$

The problem is thus to find two projectors $V_r \in \mathbb{R}^{n \times r}$ and $W_r \in \mathbb{R}^{n \times r}$ to compute a x_r with $r \ll n$ such that:

$$x_r = W_r^T x \ \ ; \ \ x \approx V_r x_r \tag{14}$$

In other words, the approximation method consists of finding the matrix V_r and W_r such that:

$$\dot{x}_r(t) - W_r^T f(V_r x_r(t), u(t)) \approx 0 \tag{15}$$

This provides an approximation of the large-scale system:

$$\dot{x}_r(t) \approx W_r^T f(V_r x_r(t), u(t)) \ , \ x_r \in \mathbb{R}^r, \ r \ll n \tag{16}$$

Singular Value Decomposition (SVD) based methods and Krylov (moment-matching) based methods are commonly used in the literature (Benner et al., 2017). Proper Orthogonal Decomposition (POD) is a SVD-based method that is directly usable for non-linear large-scale systems, that makes is suitable for our application.

The underlying idea behind POD algorithm is to collect samples of the state of the studied system and find a reduced basis that approximates these samples. This method is well suited for applications where simulation is available, as it is easy to get a collection of samples of states. The SOFA framework used to model soft robots includes algorithms to perform model reduction using POD (Goury and Duriez, 2018).

Let $\Sigma : \dot{x} = f(x(t), u(t))$ be a nonlinear system and let S be a collection of s snapshots.

$$S = (x(t_1) \ x(t_2) \ \dots \ x(t_N)) \in \mathbb{R}^{n \times s}$$

Computing a SVD of this matrix, we get V, the left singular matrix of S :

$$V \Sigma \Omega^T = S \tag{17}$$

Let us define V_r as the r^{th} first columns of V, we have the following approximation:

$$S = V \Sigma \Omega^T = V_r \Sigma_r \Omega_r^T + \Delta \tag{18}$$

where Σ_r contains the r-first singular values of S and Δ represents the model order reduction errors.

Then, the projectors V_r and W_r are simply obtained as:

$$V_r = W_r = V_r \tag{19}$$

so that it holds:

$$x_r = W_r^T x \ ; \ x \approx V_r x_r \tag{20}$$

In addition, projectors are orthogonal and it holds:

$$W_r^T V_r = I_r \text{ and } W_r^T V_{\bar{r}} = 0 \tag{21}$$

Applying POD to the linear model (11), the reduced order model writes:

$$\begin{cases} \dot{x}_r = W_r^T A V_r x_r + W_r^T B u = A_r x_r + B_r u \\ y_r = W_r^T C x_r = C_r x_r \end{cases} \tag{22}$$

4. ILLUSTRATIONS

4.1 Trunk-like robot

This experimental platform is a trunk-like robot presented in figure 1 with a schematic view in figure 2. It is made entirely of silicone, it is 18 centimeters long and the thickness at its base and its tip are respectively 2.5 and 1 centimeters. The structure is driven by 4 cables - actuated by 4 servomotors whose entry are the cable length - that permit to work in the 3 dimensions of space. The output of the system is the position of the tip (red point in figure 1) in the three dimensions of space x, y and z.

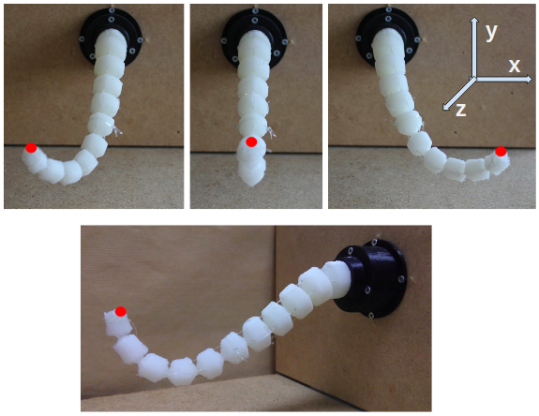

Fig. 1. Soft robot used for experimental validation, fully made of silicone. Top: front view of the robot. Bottom: side view. Red = end-effector, location of the sensor.

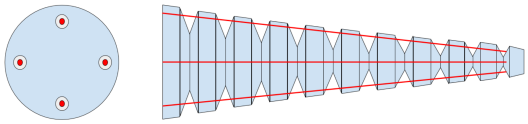

Fig. 2. Design of the robot: slice view (left) and side view (right).
The robot is actuated with 4 cables in red.

A comparison of different FEM mesh with different accuracy is given in figure 3. A mesh with 210 nodes is not accurate enough to represent the geometry faithfully. Conversely, a mesh with 6012 nodes does not give more accurate results compared to a medium size mesh of 1557 elements. This medium size mesh is a good compromise and will be used in this work.

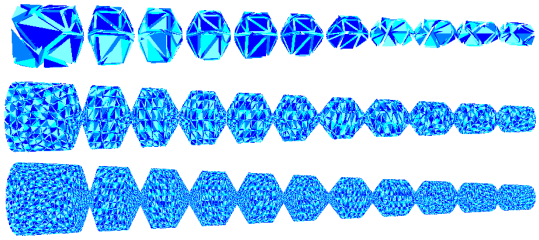

Fig. 3. Comparison of different FEM mesh of the Trunk-like robot with different accuracy.
Top: FEM mesh with 210 nodes, Middle: FEM mesh with 1557 nodes, Bottom: mesh with 6012 nodes.

With 1557 nodes in the mesh, the dimensions of the state vector in system (11) is also $1557 \times 3 \times 2 = 9342$ state variables (3 directions of space for displacement and velocity). The output is the position of the end-effector (red-point of figure 1) in the three directions of space.

A linear controller has been designed in Thieffry et al. (2018) where results show how to control the dynamics of the robot in simulation experiments. Simulation and real-world experiment shows the effectiveness of the approach. The algorithm proposed in previous work is able to control the dynamic behavior of the soft robots. However, as the model was linear, the workspace where the algorithm has been validated was limited to a region around the equilibrium point. In addition, when the target position is outside of the region of validity of the linearization assumption, the results are unpredictable and can lead to undesired behavior, as shown in figure 4.

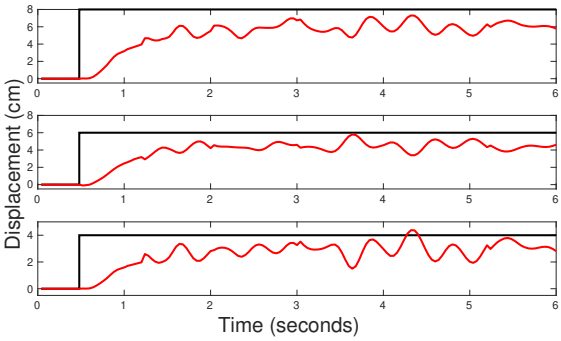

Fig. 4. Unsuccessful attempt to reach a more target distant from the linearization point. From top to bottom: displacement of end-effector along x, y and z axis in red with the desired value in black.

This limitation may have several explanations but it is probably due to the fact that the target position is outside the validity range of the linearization. Next section presents tracks to model this kind of soft robots using LPV models.

5. LPV MODEL OF SOFT ROBOTS

5.1 Basic Concepts of LPV models

Linear Parameter varying (LPV) systems have gained popularity during the 1990s, benefiting from the extension of $\mathcal{H}_\infty$ optimal control. Many nonlinear systems can be written as quasi-LPV systems, which has two advantages: the first one is to avoid writing a nonlinear model that requires precise knowledge of the process studied. The second one is to take advantage of all the techniques developed for LTI systems. Polytopic systems are a common way of modeling LPV systems (Apkarian and Tuan, 2000) and are of the form:

$$\begin{bmatrix} A(\rho(t)) & B(\rho(t)) \\ C(\rho(t)) & D(\rho(t)) \end{bmatrix} = \sum_{i=1}^{N} \rho_i(t) \begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix} \tag{23}$$

The birth of LPV systems comes from gain scheduling techniques, where the idea is to linearize nonlinear systems around different operating points yielding to a collection of local LTI models. Then, interpolation functions link each

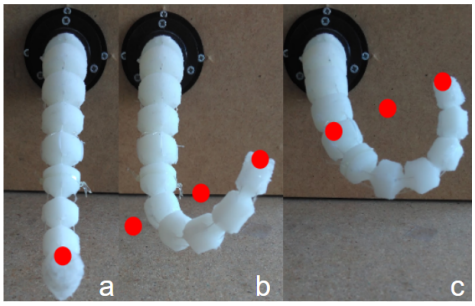

Fig. 5. Trajectory along which the model is linearized.

local subsystems. These interpolation functions are called scheduling function and describe the change of operating point. The scheduling signal is referred to as ρ. Therefore, the resulting controller are dependent on the varying signal ρ, thus resulting in *parameter varying* systems.

5.2 Multiple linearizations along a trajectory

To extend the guaranteed domain of the control algorithms, a solution is to design a controller valid for the nonlinear system (1), i.e. without assumption about area of validity. With $x = (v, q)^T$, this model writes:

$$\dot{x} = A(x)x + B(x)u \tag{24}$$

This defines a non-linear large-scale state-space system. While it is already challenging to design a controller for nonlinear systems, the complexity of the problem increases with the dimensions of the system.

Let us consider a fixed number $\mathcal{K}$ of equilibrium point x_{e_k}, $k \in \{1, \ldots, \mathcal{K}\}$ along the trajectory defined in figure 5. These equilibrium points are induced by the gravity field and a collection of inputs u_{e_k}, such that:

$$0 = A(x_{e_k})x_{e_k} + B(x_{e_k})u_{e_k} \tag{25}$$

Around each of the equilibrium point x_{e_k}, let us define $z_k = x - x_{e_k}$ and $u_k = u - u_{e_k}$ to write the collection of linear systems:

$$\dot{z}_k = A_k z_k + B_k u_k, \; k \in \{1, \ldots, \mathcal{K}\} \tag{26}$$

The POD reduction algorithm is well suited for nonlinear systems. The snapshots are captured so that the entire workspace of the robot is modeled, in other words all the subsystems (A_k, B_k) are included in the snapshot space. The reduction provides the projection matrices V_r and W_r that are valid for all of the subsystems. For the entire workspace of the robot, the projection writes:

$$x_r = W_r x \; ; \; x \approx V_r x_r \tag{27}$$

The key-point of using the POD algorithm, is that all subsystems (A_k, B_k) share the same projectors, yielding to a collection of low order linear systems:

$$\dot{z}_{r_k} = A_{r_k} z_{r_k} + B_{r_k} u_i \tag{28}$$

where $z_{r_k} = x_r - x_{r_{e_k}}$ and with $k \in \{1, \ldots, \mathcal{N}\}$.

5.3 Approximation of the collection of linear systems

Let us study the evolution of the coefficient of the reduced order system matrix A_r of the cable-driven soft robot (see figure 1). The reduction is done using POD algorithm and

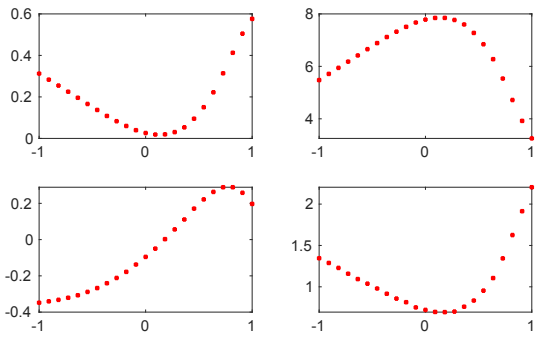

Fig. 6. From left to right, top to bottom : evolution of coefficient a_{11}, a_{14}, a_{21} and a_{25} of matrix A_r along the trajectory, from equation (29).

the reduced order system is of dimension 6. The matrix A_r also writes:

$$A_{r_k} = \begin{pmatrix} a_{11_k} & a_{12_k} & \dots & a_{16_k} \\ a_{21_k} & \ddots & \ddots & a_{26_k} \\ a_{61_k} & \dots & a_{65_k} & a_{66_k} \end{pmatrix} \tag{29}$$

The coefficient of the matrix A_r are saved at 23 different locations (i.e. $\mathcal{K} = 23$) along the trajectory depicted in figure 5. The evolution of some of the matrix coefficients is shown in figure 6. One interesting track is to use a sparse approximation technique to reconstruct all of the coefficients from matrices A_r and B_r with a minimum of variables. According to the nice properties of radial basis functions, such as sparse universal approximation, these first trials are focused on these functions.

A radial basis function is a real-valued function f whose value depends only on the distance from the origin, so that $f(x) = f(\|x\|)$; or alternatively on the distance from a given point c, so that $f(x, c) = f(\|x - c\|)$. This point c is then called a center. Gaussian function of the form $f(x) = e^{-x^2}$ are commonly used.

A radial basis network is created to approximate a function defined by a set of data points, in our case the coefficients (entries) of each matrices coming from the linearized systems along the trajectory. A radial basis network is a network with two layers, a hidden layer of radial basis neurons and an output layer of linear neurons. The MATLAB functin `newrb` is used to create the radial basis functions network.

The inputs of the network are the coefficient of the systems matrices (A_{r_k}, B_{r_k}) for each subsystems, the networks output is the function that approximates those coefficient that writes:

$$\lambda_i(\phi) = e^{-(b_i\|\phi - c_i\|)^2}, i \in \{1, \dots, \mathcal{N}\} \tag{30}$$

where $\mathcal{N}$ is the number of functions λ used to approximate the matrix coefficients. In addition, b_i are the bias values, ϕ are the input weight values and c_i are the centers values.

5.4 Construction of LPV model

From this approximated function, we get the following system:

$$\dot{x}_r = \sum_{i=1}^{\mathcal{N}} \lambda_i (A_{r_i} x_r + B_{r_i}) u \tag{31}$$

However, due to the construction of the radial basis function network, only $\lambda > 0$ holds and there is no reason for the functions to share the convex sum property, i.e. $\sum_{i=1}^{\mathcal{N}} \lambda_i \neq 1$.

Nevertheless, the sector nonlinearity approach is a systematic way to derive a polytopic model (Tanaka and Wang, 2004). Values of λ_i are bounded and it holds $\lambda_i \in [\underline{m}, \overline{m}] \in [0, 1]$, $i \in \{1, \dots, \mathcal{N}\}$, one can also write $\lambda_i = \omega_0^i \underline{m} + \omega_1^i \overline{m}$, where:

$$\omega_0^i = \frac{\overline{m} - \lambda_i}{\overline{m} - \underline{m}} \quad ; \quad \omega_1^i = \frac{\lambda_i - \underline{m}}{\overline{m} - \underline{m}} \tag{32}$$

and the function ω_i^i satisfy the convex sum property: $\omega_0^i + \omega_1^i = 1$ and $\omega_i^i \geq 0$. Let us define functions h_i (Bernal and Guerra, 2010):

$$h_{1+i_1+i_2\times 2+\dots+i_{\mathcal{N}}\times 2^{\mathcal{N}-1}} = \prod_{j=1}^{\mathcal{N}} \omega_{i_j}^j \tag{33}$$

We finally end with a polytopic model (34), based on which the design of a controller is detailed in next section.

$$\dot{x}_r = \sum_{i=1}^{2^{\mathcal{N}}} h_i (\tilde{\mathbf{A}}_{r_i} x_r + \tilde{\mathbf{B}}_{r_i} u) \tag{34}$$

where

$$\begin{aligned} \tilde{\mathbf{A}}_{1+i_1+i_2\times 2+\dots+i_{\mathcal{N}}\times 2^{\mathcal{N}-1}} &= \lambda_1(\phi_{i_1}) A_{r_1} + \dots + \lambda_{\mathcal{N}}(\phi_{i_{\mathcal{N}}}) A_{r_{\mathcal{N}}} \\ &= \sum_{j=1\dots\mathcal{N}} (\lambda_j(\phi_{i_j}) A_{r_j}) \end{aligned} \tag{35}$$

with $\phi_0 = \min(\phi)$ and $\phi_1 = \max(\phi)$.

6. CONTROL DESIGN

6.1 Theory

The simplest control design is a linear state feedback controller

$$u = -Lx_r \tag{36}$$

corresponding to the closed-loop system:

$$\dot{x}_r = \sum_{i=1}^{2^{\mathcal{N}}} h_i (\tilde{\mathbf{A}}_{r_i} - \tilde{\mathbf{B}}_{r_i} L) x_r \tag{37}$$

The closed-loop LPV polytopic system (37) is quadratically stable if and only if there exists a matrix $P = P^T > 0$ such that:

$$(\tilde{\mathbf{A}}_{r_i} - \tilde{\mathbf{B}}_{r_i} L)^T P + P(\tilde{\mathbf{A}}_{r_i} - \tilde{\mathbf{B}}_{r_i}) < 0 \tag{38}$$

hold for all $i = 1, \dots, 2^{\mathcal{N}}$, see (Boyd et al., 1994).

Condition (38) corresponds to a set of N LMI constraints that, depending on the number of vertices of the polytope, can sometimes lead to a huge number of constraints to solve. As the number of parameters N growths, so does the complexity of the condition to satisfy. It can also be time and memory consuming as the number of LMIs to solve is $2^{\mathcal{N}}$.

7. APPLICATION TO STUDIED SOFT ROBOT

7.1 Approximation using RBF

The evolution of the coefficients along the trajectory studied in this work is done using the RBF network that

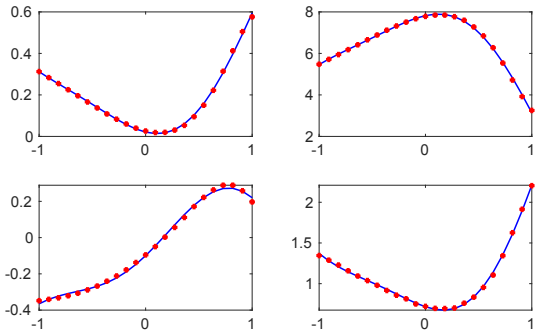

Fig. 7. Comparison of matrix coefficients a_{11}, a_{14}, a_{21} and a_{25} along the trajectory (red dots) and the outputs of the radial basis network for these coefficients (blues lines).

provides 4 functions λ, i.e. $\mathcal{N} = 4$. The mean square error (MSE) between the network input (matrix coefficients along the trajectory) and the network output for the studied soft robot is $MSE = 0.39$ given that the maximum values for a coefficient along the trajectory is 7.85. This represents a maximum error of 4.97%. A comparison of the matrix coefficients a_{11}, a_{14}, a_{21} and a_{25} and the output of the radial basis network for these coefficients is shown in figure 7.

7.2 Validation of control laws in simulation

Simulation experiments are conducted on the trunk like robot to validate the linear control law designed in equation (36). Two different experiments are done, the first one aims at driving the position around its rest position, where a linear model is valid. The robots starts at its rest shape (figure 5 a) and converges to a slightly deformed position (figure 5 b). Results are shown in figure 8 shows the comparison between open and closed-loop behavior.

Then, a second experiment aims at driving the robot from its rest shape to a highly deformed position, where a linear model is not able to capture the dynamic behavior. The robots starts at its rest shape (figure 5 a) and converges to a highly deformed position (figure 5 c). Results are shown on figure 9. Compared to results of previous works shown in figure 4, figure 9 shows that designing a controller based on LPV models makes it possible to control the robot in a wider workspace.

8. DISCUSSION AND FUTURE WORK

Simulation experiments shows the validity of the approach using a linear controller for the LPV system. Better performances could be reached with more elaborate control laws, such as the parallel distributed compensation (PDC) (Tanaka and Wang, 2004). This is the topic of ongoing research. It is composed of linear feedbacks assembled together with the same nonlinear function h_i as in the model (34):

$$u = -\sum_{i=1}^{2^{\mathcal{N}}} h_i L_i x \tag{39}$$

yielding to corresponding closed-loop model:

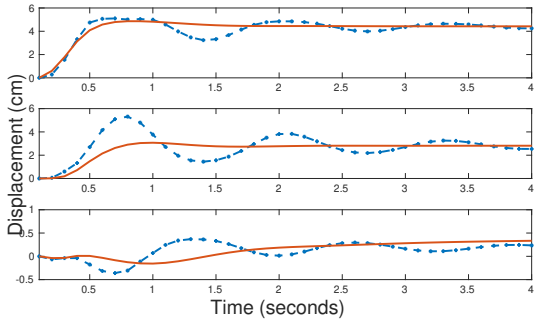

Fig. 8. Displacement of end-effector along x (top) y (center) and z axis (bottom) in open-loop (dashed line) and closed-loop (plain line). The robot starts from its rest shape (Fig. 5 a) and converges to a slightly deformed position (Fig. 5 b).

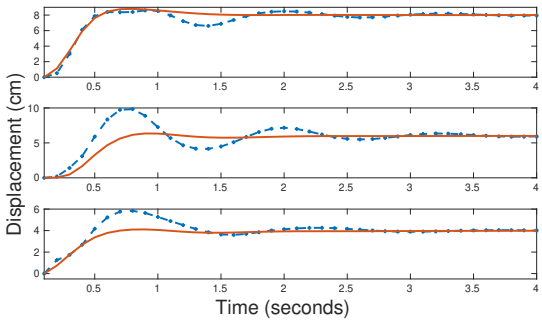

Fig. 9. Displacement of end-effector along x (top) y (center) and z axis (bottom) in open-loop (dashed line) and closed-loop (plain line). The robot starts from its rest shape (Fig. 5 a) and converges to a highly deformed position (Fig. 5 c).

$$\dot{x}_r = \sum_{i=1}^{2^{\mathcal{N}}} \sum_{j=1}^{2^{\mathcal{N}}} h_i h_j (A_i - B_i L_j) x_r \tag{40}$$

In addition, to study the robustness of the control law, one should study the modeling errors coming from the reduction algorithm. Indeed, the POD algorithm decompose the full order state x into two parts, a reduced part x_r that represents faithfully x and a neglected part $x_{\bar{r}}$ that is neglected such that:

$$x = V_r x_r + V_{\bar{r}} x_{\bar{r}} \approx V_r x_r \tag{41}$$

Results of Thieffry et al. (2019) shows how to design a controller that takes into account this reduction error $x_{\bar{r}}$ based on linear models. Its extension to LPV models has to consider a global error coming from the convex aggregation of the polytopic vertices. This extension is left for future research.

9. CONCLUSION

This work presents new results about dynamic control of soft robots using a nonlinear reduced-order numerical model. The finite element method provides us with a large-scale nonlinear model that is linearized along a trajectory to build a linear parameter varying (LPV) model. This work is an important step towards nonlinear control of soft robots modeled using finite element method. Further

research will focus on the design on a nonlinear observer to enable experimental validation of the approach.

REFERENCES

Apkarian, P. and Tuan, H.D. (2000). Parameterized lmis in control theory. *SIAM journal on control and optimization*, 38(4), 1241–1264.

Benner, P., Cohen, A., Ohlberger, M., and Willcox, K. (2017). *Model Reduction and Approximation: Theory and Algorithms*, volume 15. SIAM.

Bernal, M. and Guerra, T.M. (2010). Generalized nonquadratic stability of continuous-time takagi–sugeno models. *IEEE Transactions on Fuzzy Systems*, 18(4), 815–822.

Boyd, S., El Ghaoui, L., Feron, E., and Balakrishnan, V. (1994). *Linear matrix inequalities in system and control theory*, volume 15. Siam.

Bruder, D., Gillespie, B., Remy, C.D., and Vasudevan, R. (2019). Modeling and control of soft robots using the koopman operator and model predictive control. *arXiv preprint arXiv:1902.02827*.

Coevoet, E. et al. (2017). Software toolkit for modeling, simulation, and control of soft robots. *Advanced Robotics*, 31(22), 1208–1224.

Della Santina, C., Katzschmann, R.K., Biechi, A., and Rus, D. (2018). Dynamic control of soft robots interacting with the environment. In *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, 46–53. IEEE.

Falkenhahn, V., Hildebrandt, A., Neumann, R., and Sawodny, O. (2015). Model-based feedforward position control of constant curvature continuum robots using feedback linearization. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 762–767. IEEE.

Goury, O. and Duriez, C. (2018). Fast, generic, and reliable control and simulation of soft robots using model order reduction. *IEEE Transactions on Robotics*, 1–12.

Katzschmann, R., Thieffry, M., et al. (2019a). Dynamically closed-loop controlled soft robotic arm using a reduced order finite element model with state observer. In *IEEE 2019 International Conference on Soft Robotics*.

Katzschmann, R.K., Della Santina, C., Toshimitsu, Y., Bicchi, A., and Rus, D. (2019b). Dynamic motion control of multi-segment soft robots using piecewise constant curvature matched with an augmented rigid body model. In *2019 IEEE International Conference on Soft Robotics (RoboSoft)*.

Kim, S., Laschi, C., and Trimmer, B. (2013). Soft robotics: a bioinspired evolution in robotics. *Trends in biotechnology*, 31(5), 287–294.

Majidi, C. (2014). Soft robotics: a perspective — current trends and prospects for the future. *Soft Robotics*, 1(1), 5–11.

Marchese, A.D., Komorowski, K., Onal, C.D., and Rus, D. (2014). Design and control of a soft and continuously deformable 2d robotic manipulation system. In *2014 IEEE international conference on robotics and automation (ICRA)*, 2189–2196. IEEE.

Sadati, S.M.H., Naghibi, S.E., Walker, I.D., Althoefer, K., and Nanayakkara, T. (2018). Control Space Reduction and Real-Time Accurate Modeling of Continuum Manipulators Using Ritz and Ritz–Galerkin Methods. *IEEE Robotics and Automation Letters*.

Tanaka, K. and Wang, H.O. (2004). *Fuzzy control systems design and analysis: a linear matrix inequality approach*. John Wiley & Sons.

Thieffry, M., Kruszewski, A., Duriez, C., and Guerra, T.M. (2019). Control design for soft robots based on reduced-order model. *IEEE Robotics and Automation Letters*, 4(1), 25–32.

Thieffry, M., Kruszewski, A., Guerra, T.M., and Duriez, C. (2018). Reduced Order Control of Soft Robots with Guaranteed Stability. In *European Control Conference ECC18*. Limassol, Cyprus.

Thuruthel, T.G. et al. (2018). Control strategies for soft robotic manipulators: A survey. *Soft robotics*.

Trivedi, D., Lotfi, A., and Rahn, C.D. (2008). Geometrically exact models for soft robotic manipulators. *IEEE Transactions on Robotics*, 24(4), 773–780.

Webster, R.J. and Jones, B.A. (2010). Design and kinematic modeling of constant curvature continuum robots: A review. *The International Journal of Robotics Research*, 29(13), 1661–1683.