Fast, generic and reliable control and simulation of soft robots using model order reduction

Olivier Goury, Christian Duriez

Fast, generic and reliable control and simulation of soft robots using model order reduction.

Olivier Goury[1,2] and Christian Duriez[1,2]

***Abstract*—Obtaining an accurate mechanical model of a soft deformable robot compatible with the computation time imposed by robotic applications is often considered as an unattainable goal. This paper should invert this idea. The proposed methodology offers the possibility to dramatically reduce the size and the online computation time of a Finite Element Model (FEM) of a soft robot. After a set of expensive offline simulations based on the whole model, we apply snapshot-proper orthogonal decomposition to sharply reduce the number of state variables of the soft robot model. To keep the computational efficiency, hyperreduction is used to perform the integration on a reduced domain. The method allows to tune the error during the two main steps of complexity reduction. The method handles external loads (contact, friction, gravity...) with precision as long as they are tested during the offline simulations. The method is validated on two very different examples of FE models of soft robots and on one real soft robot. It enables acceleration factors of more than 100, while saving accuracy, in particular compared to coarsely meshed FE models and provides a generic way to control soft robots.**

***Index Terms*—Soft Robotics, Robot simulation, Model Order Reduction, Proper Orthogonal Decomposition, Hyperreduction, Energy conserving sampling and weighting Method**

I. Introduction

Soft Robotics is an emerging field of robotics. Unlike traditional robots typically made of articulated rigid bodies, soft robots are made of soft materials, such as silicone, and their movements are created by deformation. The design of the soft robots is often inspired by nature: elephant's trunk [1], octopus' arm [2] or worms for example. It is thought that for some applications, soft robots may be more suited than their rigid counterparts. These applications may be manipulation of fragile objects, locomotion in difficult terrain, interaction with humans, or surgical applications to name a few. The potential of soft robots were identified in [3], [4]. The suitable mechanical impedance provided by the soft materials may also be achieved by Variable Stiffness Actuators (VSAs) or Series Elastic Actuators(SEAs)[5].

A. Simulation

The simulation of robots is an important field in robotics. Simulators provide assistance for robot prototyping, testing of control methods, online or offline planning of the motion within the environment, reward evaluation tests for learning methods or supervision. General platforms like V-Rep [6], Gazebo [7] or even more specialized simulation system, like openHRP [8] dedicated to humanoid robotics, provide mechanical simulation of articulated rigid robot dynamics. Interaction of robots with their environment could also be simulated thanks to collision detection and response available in physics engines like ODE [9], PhysX [10], Bullet [11]. See [12] for a comparison of these tools. The code of these engines is optimized to obtain very fast simulation, in particular to allow interactive and real-time simulation of the robot, which are very intuitive and useful for the users. However, such physics engines with real-time level of performance are not available for deformable robots in the general case.

In some particular cases, for instance when the robot is considered as a series of deformable rods (like concentric tube robots), very fast and accurate models could be used based on Kirchoff [13] or Cosserat rod theory [14], [15]. To model soft robots, tools such as Voxelyse [16] were developed, based on a representation of the robot structure in Voxels supported by a lattice modelled using beam theory. For more realistic mechanical simulations, finite element (FE) based method can be used to predict the robots deformation with great accuracy (see for example [17]). However that is an expensive method that does not allow a priori for interactivity. Real-time FE simulation and control of soft robots [18] were made possible using the SOFA framework[1], which is dedicated to the fast simulation of deformable bodies, and the SoftRobots plugin[2]. This method was used for various type of robots geometries and actuators [19], [20], [21]. However, the real-time constraints made the method possible only for relatively coarse meshes and simple material constitutive laws.

The objective of this study is to go much further on this limit of computation time and to allow a real-time simulation of flexible robots on meshes and models much more complex, while remaining generic. For that purpose, we propose to use a technique of model order reduction. This code is developed in a SOFA plugin[3]. In the future, the SOFA framework could be integrated in more traditional simulation frameworks like Gazebo, to provide a physics

[1]DEFROST team, Inria Lille-Nord Europe, 40 avenue du Halley, 59000 Lille, France olivier.goury@inria.fr

[2]University of Lille - CRIStAL UMR CNRS 9189, 59655 Villeneuve d'Ascq Cedex France

[1]www.sofa-framework.org

[2]https://project.inria.fr/softrobot

[3]https://project.inria.fr/modelorderreduction

engine to simulate soft robots.

B. Model order reduction

Model order reduction methods are well-used in the computational mechanics community to reduce the computational cost of FE simulations, with material design or virtual-prototyping applications in mind. Model order reduction (MOR) methods consist in projecting the large FE equations of motion onto attractive subspaces of smaller dimensions allowing much faster computations [22].

In practice it allows to dramatically reduce the number of degrees of freedom of the computed mechanical system, while keeping the precision. One of the main methods is the snapshot-POD (Proper orthogonal decomposition) [23], [24], [25], which proposes to represent the high-dimensional deformation of the structure within a small attractive subspace of much smaller dimension, defined by a basis of a few vectors. To find this small but representative subspace, the process involves an offline stage where all the potential movements of the structure studied are sampled and stored in what is called the snapshot. This offline stage is computationally intensive, since many fine simulations are performed, but it is performed only once to build the reduced order model. The snapshot space is condensed in a reduced basis using a singular value decomposition and a truncation. The resulting reduced-model of the structure has few degrees of freedom and the simulation can be performed at a much higher rate. To keep the method numerically efficient in the case of nonlinear material behaviour, some hyperreduction method [26], [25], [27], [28] are used to reduce that computation. Some notable contributions in the idea of using model order reduction technique for fast FE simulations can be found for a simple soft robot example using Proper Generalised Decomposition (PGD)[29], or in the context of computer animations [30], [31] or material design [32]. These latter methods generate the reduced basis by computing vibration modes and their derivatives to account for large deformations.

C. Contribution

The contribution of the paper is, to our best knowledge, the first published method to obtain a very fast mechanical model of a soft robot - compatible with real-time robotic constraints - based on a complex FE model reduced by a snapshot-POD method.

To demonstrate efficiency and genericity, the method will be tested on three very different cases: one deformable cable-driven parallel manipulator, a multigait pneumatic robot for locomotion, showing that the method is compatible with contact constraints with the floor, and a tendon-driven soft tentacle interacting with its environment. The method enables very impressive acceleration factors, while keeping a good accuracy and in all cases true real-time (computation time = simulation time) could be achieved. In the third example, the method is used to control a real tentacle soft robot.

Despite some limitations, especially on local deformations that are discussed in the paper, the approach is very comprehensive and opens completely new perspectives in terms of modeling and simulation for deformable robots.

II. Soft robots finite element model

In this section we define the finite element modeling framework used to model the deformation of soft robots in general. The robot may have any shape. Consider the deformation of a soft-robot subjected to external forces and contacts. Its dynamics is defined using Newton's second law:

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

where $\mathbf{q}$ is the vector of position, $\mathbf{v}$ is the vector of velocity, $\mathbf{M}(\mathbf{q})$ is the mass matrix, $\mathbf{P}$ gathers external forces, $\mathbb{F}$ accounts for internal forces (depending on the material the soft robot is made of) and $\mathbf{H}^T\lambda$ is the vector of constraint forces contributions, which are typically generated when the soft-robot boundary enters in contact with its environment, or when an actuator applies force on the soft-robot.

A. Space discretization

In this paper, we mainly focus on 3D geometries though the method would be applicable to shell models or beam models in principle. The geometry of the robots is meshed with tetrahedrons or hexahedrons. The finite element method is applied to express the deformation of the robot in a discrete way. In this paper, we will use linear tetrahedrons though in principle, the method would work for any order of elements.

B. Implicit time integration

The continuous time space is subdivided into discrete intervals: $[t_0, t_1, ..., t_{n_t}]$. Let's consider a small time interval $[t_n, t_{n+1}]$, and let $h = t_{n+1} - t_n$. Integrating equation (1) over this interval gives (assuming the mass matrix is constant):

$$\mathbf{M}(\mathbf{v}_{t_{n+1}} - \mathbf{v}_{t_n}) = \int_{t_n}^{t_{n+1}} (\mathbf{P}(t) - \mathbb{F}(\mathbf{q}, \mathbf{v}))\, dt + \; h\mathbf{H}^T\lambda \tag{2}$$

$$\mathbf{q}_{t_{n+1}} - \mathbf{q}_{t_n} = \int_{t_n}^{t_{n+1}} \mathbf{v}(t)dt \tag{3}$$

Using an implicit Euler integration scheme, we obtain:

$$\mathbf{M}(\mathbf{v}_{t_{n+1}} - \mathbf{v}_{t_n}) = h\left(\mathbf{p}_{t_{n+1}} - \mathbb{F}(\mathbf{q}_{t_{n+1}}, \mathbf{v}_{t_{n+1}})\right) + \; h\mathbf{H}^T\lambda \tag{4}$$

$$\mathbf{q}_{t_{n+1}} = \mathbf{q}_{t_n} + h\mathbf{v}_{t_{n+1}}, \tag{5}$$

where $\mathbf{p}_{t_{n+1}}$ is the value of the function $\mathbf{P}$ at time $t_{t_{n+1}}$ (known a priori). The internal forces $\mathbb{F}$ is a nonlinear function of the positions and the velocities, which means equation (4) can not be solved exactly in one step and one can use an iterative method such as Newton-Raphson.

A less expensive method, though less accurate when the timestep is large compared to the internal forces nonlinearity, is to solve the system by applying a Taylor series expansion to $\mathbb{F}$ leading to the following first order approximation:

$$\mathbb{F}(\mathbf{q}_{t_{n+1}}, \mathbf{v}_{t_{n+1}})) = \mathbb{F}\ (\mathbf{q}_{t_n} + d\mathbf{q}, \mathbf{v}_{t_n} + d\mathbf{v}) \tag{6}$$

$$= \mathbf{f}_{t_n} + \frac{\delta \mathbb{F}}{\delta \mathbf{q}} d\mathbf{q} + \frac{\delta \mathbb{F}}{\delta \mathbf{v}} d\mathbf{v} \tag{7}$$

Using $d\mathbf{q} = \mathbf{q}_{t_{n+1}} - \mathbf{q}_{t_n} = h\mathbf{v}_{t_{n+1}}$ and $d\mathbf{v} = \mathbf{v}_{t_{n+1}} - \mathbf{v}_{t_n}$, we obtain:

$$\underbrace{\left(\mathbf{M} + h\frac{\delta \mathbb{F}}{\delta \mathbf{v}} + h^2 \frac{\delta \mathbb{F}}{\delta \mathbf{q}}\right)}_{\mathbf{A}(\mathbf{q}_{t_n}, \mathbf{v}_{t_n})} d\mathbf{v} \tag{8}$$

$$= \underbrace{-h^2 \frac{\delta \mathbb{F}}{\delta \mathbf{q}} \mathbf{v}_{t_n} - h\left(\mathbf{f}_{t_n} + \mathbf{p}_{t_{n+1}}\right)}_{\mathbf{b}(\mathbf{q}_{t_n}, \mathbf{v}_{t_n})} + h\mathbf{H}^T \lambda. \tag{9}$$

This is exactly equivalent to perform the first iteration of the Newton-Raphson algorithm. In this paper, we choose this linearisation strategy for performance purposes, but the method we will describe in the following section would also work using the Newton-Raphson method. The computational cost of one Newton-Raphson iteration is identical to one linearisation step. When the mesh used to describe the soft robot is fine, equation (8) is of large dimension and prohibits an interactive simulation of the soft robot deformations.

C. Constitutive law of the soft robot material

The constitutive law of silicone is generally recognised as being hyperelastic, which establishes a nonlinear relationship between stresses and strains in the material. In this paper, to model the behaviour of the soft robot, we will consider two kinds of constitutive laws:

- Co-rotational linear elastic. This formulation assumes a linear stress-strain relationship within the frame attached to each element of the FE mesh. The co-rotational element stiffness contribution is computed using the rotation transformation between the deformed element compared to its rest position. This results in a non-linear constitutive law, since the displacements are needed to compute the stiffness matrix. This law can actually represent large deformations, as long as those deformations involve only large rotations but relatively small deformations of each individual elements.
- Mooney-Rivlin Hyperelastic. This is a well known model for hyperelasticity [33]. It assumes the following strain energy density function:

$$W = C_{01}(\bar{I}_1 - 3) + C_{10}(\bar{I}_2 - 3) + D(J-1)^2, \tag{10}$$

with $\bar{I}_1 = J^{-2/3} I_1$, $\bar{I}_2 = J^{-4/3} I_2$, $J = \det(\mathbf{F})$, $\mathbf{F}$ being the deformation gradient, I_1 and I_2 being the first and second strain invariants of the right Cauchy-Green deformation tensor.

These two constitutive laws allow to compute the internal forces vector $\mathbb{F}$ through the integration of partial differential equations coming from continuum mechanics. To have a good accuracy on the numerical model, it is often necessary to use a very fine discretisation which leads to systems of high dimension. Hence, there is a real need for reduction.

III. Reduction of the full order Finite element model

In this section, we present a general framework for the reduction of the full order dynamic model describing the robot's motion presented in the previous section. This reduction will be performed in two stages: a projection-based method to express the state variables in a low-dimensional subspace, and a system approximation method, to integrate the mechanical properties only on a small subset of elements of the large FE mesh.

A. Snapshot - Proper Orthogonal Decomposition

The proper orthogonal decomposition (POD) [23], [25], [24] is a method used widely in the computational mechanics community to generate relevant bases to be used in reduced-order models. The model to reduce is typically parametrized, and the reduced model is then valid for any values of the parameter. To apply the method in our context, we define $\boldsymbol{\Lambda}$, the parameter space of scenarios of constraints applied onto the robot (that may be from actuators or external contacts) parametrized by the constraints[4] $\boldsymbol{\lambda}$.

Then, the POD decomposes the position $\mathbf{q}(t)$, solution of (8) in a truncated expansion of orthonormal vectors (where we show clearly the dependency of $\mathbf{q}$ on the constraints $\boldsymbol{\lambda}$):

$$\mathbf{q}\,(t, \boldsymbol{\lambda}) \approx \mathbf{q}(0) + \sum_{i=1}^{N} \boldsymbol{\phi}_i \alpha_i(t, \boldsymbol{\lambda}) \tag{11}$$

$$= \mathbf{q}(0) + \boldsymbol{\Phi}\, \boldsymbol{\alpha}(t, \boldsymbol{\lambda}), \tag{12}$$

such that the orthonormal basis $(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_N)$ is minimizing the cost function representing the sum of all the projection errors of the exact solution onto the basis:

$$J(\boldsymbol{\Phi})^2 = \int_{\boldsymbol{\lambda}^* \in \boldsymbol{\Lambda}} \int_{t=t_0}^{t=t_{n_t}} \left\| \mathbf{q}(t, \boldsymbol{\lambda}^*) - \sum \left({\boldsymbol{\phi}_i}^T \mathbf{q}(t, \boldsymbol{\lambda}^*) \right) \boldsymbol{\phi}_i \right\|_2^2 dt. \tag{13}$$

In practice, this technique is not useful since it requires to know the exact solution at each point of time and for every scenario of loading. The snapshot-POD proposes to use a discrete approximation of this cost function, where

[4]The constraints are actually also defined by the direction parametrized by $\mathbf{H}$ but for the sake of simplicity of the notations we only show the constraint dependency with $\boldsymbol{\lambda}$ instead of $\mathbf{H}^T\boldsymbol{\lambda}$.

we define a discrete subset of $\boldsymbol{\Lambda}$: $\widehat{\boldsymbol{\Lambda}} = \{\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_{n_{\boldsymbol{\lambda}}}\}$. The cost function becomes:

$$\widehat{J}(\boldsymbol{\Phi})^2 = \sum_{\boldsymbol{\lambda}^* \in \widehat{\boldsymbol{\Lambda}}} \sum_{t=t_0}^{t=t_{n_t}} \left\| \mathbf{q}(t, \boldsymbol{\lambda}^*) - \sum \left({\boldsymbol{\phi}_i}^T \mathbf{q}(t, \boldsymbol{\lambda}^*) \right) \boldsymbol{\phi}_i \right\|_2^2. \quad (14)$$

The set $\left\{ \mathbf{q}(t, \boldsymbol{\lambda}) | \, t \in [t_0, t_{n_t}], \boldsymbol{\lambda} \in \widehat{\boldsymbol{\Lambda}} \right\}$ can be assembled in a snapshot matrix $\mathbf{S} = [\mathbf{q}(t_0, \boldsymbol{\lambda}_0), \ldots, \mathbf{q}(t_{n_t}, \boldsymbol{\lambda}_{n_{\boldsymbol{\lambda}}})]$. The solution to the problem of minimising the cost function defined in (14), under the constraint of orthonormality of the basis is given by performing a singular value decomposition (SVD): $\mathbf{S} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, with $\mathbf{U}$ and $\mathbf{V}$ unitary matrices and $\boldsymbol{\Sigma}$ a rectangular diagonal matrix containing the singular values of $\mathbf{S}$. The optimal basis of order p is then given by the first p left singular vectors (in $\mathbf{U}$) associated to the p largest singular values σ_i. The truncation error of a POD transform of order p is given by:

$$\nu^2 = \frac{\sum_{i=p+1}^{n_q} \sigma_i^2}{\sum_{i=1}^{n_q} \sigma_i^2} \quad (15)$$

The reduced basis $\boldsymbol{\Phi}$, minimising the cost function (14), will give a good approximation of the position $\mathbf{q}$ if the set $\widehat{\boldsymbol{\Lambda}}$ is exhaustive enough and represents well all the workspace of the robot (induced by all possible actuations) and the contacts the robot may encounter. Note that computing the cost function $\widehat{\mathbf{J}}$ requires the knowledge of the full order solution over the set $\widehat{\boldsymbol{\Lambda}}$, and it is a very expensive step. However, it is performed only once, and the reduced basis $\boldsymbol{\Phi}$ is computed once for all. The definition of the set $\widehat{\boldsymbol{\Lambda}}$, which is the set of scenarios of loading the robot will be subjected to in the offline phase, has a strong influence on what deformations the resulting reduced order model will then be able to reproduce. In the context of this paper, this set is defined by all possible combination of actuation. Assuming there are N_A actuators, and defining a range divided in a certain number of samples s_i for each actuator, the number of possible actuations is $\Pi_{i=1}^{N_A} s_i$. For example, a robot with 4 actuators with each actuator having a range divided in 10 steps leads to $10^4 = 10000$ configurations. In this paper, we will simplify the generation of the snapshot by only considering the 2 extreme values of the range of each actuator, leading to 2^{N_A} configurations and the robot will be simulated while moving directly from one actuation state to another, as described in Figure 1. Despite this simplification, the number of tests to perform to generate the snapshot grows exponentially with the number of actuators. In the case the number is not tractable with the computational power at hand, non-exhaustive sets can be selected through Latin-Hypercube sampling. In the general context of model order reduction, the problem of selecting an optimal snapshot to create accurate reduced models is an active research field. Beyond uniform sampling, other strategies exist to sample the workspace: random selection [34], error estimation to select the samples in an iterative way [35], gradient-optimisation [36] or statistical inference [37], [34]. These strategies may be used in future works.

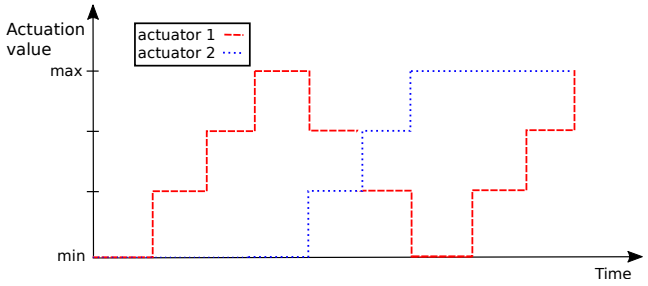

Fig. 1: Example of actuation sequence for a robot with 2 actuators and hence $2^2 = 4$ different extreme configurations. In 3 steps, the actuation moves from one configuration to another. This sampling of the space of actuation is not exhaustive but tends to give sufficient information to build an accurate reduced order model as it explores the extremities of the actuation space.

B. Galerkin projection of the equations of motion onto the reduced space

Once the basis is defined, the reduced expression of the state variables (position, velocity) can be introduced into the equation of motion (8). Note that by differentiating with respect to time, the reduced expansion (11) naturally leads to a reduced expression for the vector of velocities: $\mathbf{v}(t) = \dot{\mathbf{q}}(t) = \boldsymbol{\Phi}\, \dot{\boldsymbol{\alpha}}(t)$. Plugging these expressions into equation (8) and projecting onto the reduced subspace (thus a Galerkin projection) leads to:

$$\underbrace{\boldsymbol{\Phi}^T \mathbf{A}(\mathbf{q}_{t_n}, \mathbf{v}_{t_n}) \boldsymbol{\Phi}}_{\mathbf{A}_r} \dot{d\boldsymbol{\alpha}}(t) = \underbrace{\boldsymbol{\Phi}^T \mathbf{b}(\mathbf{q}_{t_n}, \mathbf{v}_{t_n})}_{\mathbf{b}_r} + \underbrace{(\mathbf{H}\,\boldsymbol{\Phi})^T}_{\mathbf{H}_r} \boldsymbol{\lambda}, \quad (16)$$

where $d\boldsymbol{\alpha} = \boldsymbol{\alpha}_{t_{n+1}} - \boldsymbol{\alpha}_{t_n}$ and $\dot{d\boldsymbol{\alpha}} = \dot{\boldsymbol{\alpha}}_{t_{n+1}} - \dot{\boldsymbol{\alpha}}_{t_n}$. Note that we performed a Galerkin projection (using the reduced basis itself as projector) for simplicity, but a Petrov-Galerkin projection could have been used (a study in the two different approaches was made in [25]). This is a reduced equation that is of the dimension N of the subspace generated by the basis $\boldsymbol{\Phi}$. The new unknown becomes $\boldsymbol{d\alpha}$ which is of much smaller dimension than the vector of positions $\mathbf{q}$ and of velocities $\mathbf{v}$.

C. System approximation of the Finite Element mesh for fast efficient construction of the nonlinear operators

The reduced equation (16), though of smaller dimension than the original full order model, is still expensive to build. Indeed, because the equations of motion are nonlinear, the operator $\mathbf{A}_r$ is dependent on the current state of the system and can not be pre-computed. Hence, it is still necessary to loop over all the elements of the high dimensional mesh to evaluate the full order operator $\mathbf{A}$ before projecting it onto the reduced basis $\boldsymbol{\Phi}$. The same is true for the right hand side $\mathbf{b}_r$. As a result, the gain in computational time provided by the reduced model will be negligible. This issue is well-known in the field of reduced-order modeling and can be tackled by applying a second approximation called hyperreduction or system

approximation. This method provides a way to solve the reduced equation while computing the mechanical properties on a small subset of the elements of the full order mesh only. The procedure has several variations in the literature. We can cite the hyperreduction method [26], the discrete empirical interpolation method [38] or more recently the energy-conserving sampling and weighting (ECSW) method [27] or empirical cubature method [28]. In this work, we focus on the ECSW method (very similar to the empirical cubature method) which builds stable and consistent reduced models. First, for the unique sake of simplicity of the notations, we rewrite equation (16):

$$\mathbf{\Phi}^T \mathbf{g}(t_n; \boldsymbol{\lambda}) = (\mathbf{H}\,\mathbf{\Phi})^T \boldsymbol{\lambda}, \tag{17}$$

with $\mathbf{g}(t_n; \boldsymbol{\lambda}) = \mathbf{A}(\mathbf{q}_{t_n}, \mathbf{v}_{t_n})\mathbf{\Phi}\dot{\boldsymbol{d\alpha}}(t) - \mathbf{b}(\mathbf{q}_{t_n}, \mathbf{v}_{t_n})$. Here, we show clearly the dependence of the operator $\mathbf{g}$ on the constraints $\boldsymbol{\lambda}$, which can be seen as a parameter of the simulation. The method proposes to go back to the element level. Indeed, the left-hand side can be decomposed as:

$$\mathbf{\Phi}^T \mathbf{g}(t_n; \boldsymbol{\lambda}) = \mathbf{\Phi}^T \sum_{e \in \mathcal{E}} \mathbf{Y}^{(e)} \mathbf{g}_e(t_n; \boldsymbol{\lambda}) \tag{18}$$

$$= \sum_{e \in \mathcal{E}} \mathbf{\Phi}_e^T \mathbf{g}_e(t_n; \boldsymbol{\lambda}), \tag{19}$$

where $\mathbf{Y}^{(e)} \in \mathbb{R}^{n_q \times n_e}$ is the assembly operator that maps degrees of freedom from the element level (e) to the global degrees of freedom of the entire system. $\mathbf{Y}^{(e)}$ is rectangular and only filled with zeros and ones. Hence, $\mathbf{\Phi}_e$ is the restriction of $\mathbf{\Phi}$ to the degrees of freedom corresponding to element (e). $\mathbf{g}_e(t_n; \boldsymbol{\lambda})$ is the contribution of element (e) to the global term $\mathbf{g}(t_n; \boldsymbol{\lambda})$.

Then, the ECSW method interprets the quantities $\mathbf{\Phi}_e^T \mathbf{g}_e(t_n; \boldsymbol{\lambda})$ as a set of energies and therefore proposes to approximate $\mathbf{g}$ by performing the sum over only a small subset of the elements $\widehat{\mathcal{E}}$ (the reduced integration domain or RID) weighted with some positive coefficients ξ_e:

$$\mathbf{\Phi}^T \mathbf{g}(t_n; \boldsymbol{\lambda}) \approx \sum_{e \in \widehat{\mathcal{E}}} \xi_e \mathbf{\Phi}_e^T \mathbf{g}_e(t_n; \boldsymbol{\lambda}). \tag{20}$$

This new expression is much less computationally intensive as the computation of all the operators will involve a loop over a small number of elements rather than the entire high-dimensional mesh.

Now, the set of elements of the reduced integration domain $\widehat{\mathcal{E}}$ and the corresponding weights $(\xi_e)_{e \in \widehat{\mathcal{E}}}$ can be found through an optimization problem, based on a database of unassembled elements contributions computed previously (This implies that another set of offline simulations needs to be performed to store these element contributions, and typically, these are done using the same actuation sequence that generated the snapshot in the previous stage). The problem can be expressed in the following way:

$$\begin{aligned} \boldsymbol{\xi} &= \arg\min \|\boldsymbol{\xi}^*\|_0 \\ \text{subject to:} \quad & \|\mathbf{G}\boldsymbol{\xi}^* - \mathbf{y}\|_2 \leq \tau \|\mathbf{y}\|_2 \\ & \boldsymbol{\xi}^* \geq 0\,, \end{aligned} \tag{21}$$

where τ is a tolerance defined by the user. The zero norm $\|\mathbf{x}\|_0$ is the number of non-zeros entries in $\mathbf{x}$. $\mathbf{G} = [\mathbf{G}_{ke}]_{k \in \widehat{\Lambda}, e \in \widehat{\mathcal{E}}}$ is a block matrix containing the unassembled element contributions for each configuration $\boldsymbol{\lambda} \in \widehat{\Lambda}$, with the column block $\mathbf{G}_{ke} = \mathbf{\Phi}_e^T \mathbf{g}_e(t^k, \boldsymbol{\lambda}^k)$ for the k^{th} time-parameter configuration and element number e. $\mathbf{y} = [\mathbf{y}_k]_{k \in \widehat{\Lambda}}$ is the assembled counterpart: $\mathbf{y}_k = \mathbf{\Phi}^T \mathbf{g}(t^k, \boldsymbol{\lambda}^k) = \sum_{e \in \widehat{\mathcal{E}}} \mathbf{G}_{ke}$.

This minimisation problem is a non-negative least square problem [39] and finding the optimal solution typically involves an intractable exhaustive combinatoric search. In practice, a suboptimal greedy algorithm, like described in [40], [27]), can be used to find an acceptable solution to problem (21).

D. Summary for building the reduced order model of a soft robot

In this section, we sum up the main steps of the reduction workflow in the form of an algorithm:

Algorithm 1 Workflow for building reduced order model of a soft robot

Ensure: The accuracy of the full order model is representative of the real soft robot.
- Evaluate number of actuators N_A and their range of actuation.
- Define number of samples s_i within each actuator range (minimum is 2)
if it is feasible to compute $\Pi_{i=1}^{N_A} s_i$ simulations to generate an exhaustive snapshot **then**
  Compute and store the positions (or state variables in general) in the snapshot
else
  Define an alternative non-exhaustive snapshot selection strategy (Latin Hypercube, Bayesian, etc...) to compute the snapshot
end if
- Perform a singular value decomposition on the snapshot matrix and truncate up to a tolerance ν defined in (15). This defines the reduced basis Φ.
- Compute the reduced model on the snapshot space again, storing the FE elements contributions along the way.
- Define a tolerance τ and solve the associated NNLS problem (21). This gives the reduced integration domain and the weights connected to it.

IV. Practical applications

The proposed method allows to reduce greatly the computational time of an FE model of a soft robot. First, two examples on two very different kinds of robots will be presented to show the method can be generic. The practical applications are hence primarily concentrated on the numerical simulation, assuming that a converged FE solution on a fine mesh can be used as "ground truth". In those sections, we show the application of the method on these two robots and analyse performance in terms of accuracy and computational time with respect to the full order model. Then, the applicability of the method will be shown to control a real tentacle robot.

A. A first example: fixed soft silicone robot actuated by four cables

Let us consider the parallel soft silicone robot described in [18]. This purely academic robot is displayed in Figure 2. It is entirely made of soft silicone and is actuated by four cables controlled by step motors located at its center. Pulling on the cables has the effect of lifting the effector located on top of the robot. The "game" with this robot is to control the position of the effector by pulling on the cables.

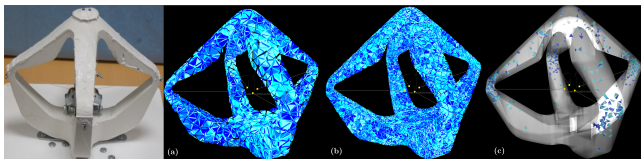

Fig. 2: From left to right: coarse mesh with 1400 nodes; fine mesh with 15553 nodes; reduced mesh used for hyper-reduction : 861 nodes and only 376 tetrahedra.

a) Finite element mesh refinement: In [18], the robot was controlled through real-time finite element simulation based on a mesh of 1628 nodes and 4147 tetrahedra. That size of mesh was manageable in real-time on a standard desktop computer. The simulation made using this underlying mesh was accurate enough to control the robot, only considering the displacement of the effector point, located on the top of the robot and with a limited range on the pulling of the cable actuators. However, even the full convergence of the FE model could require a larger number of elements (convergence of the displacement or of the stress field inside the domain which could be important for the equilibrium with external loads). In some configurations, in particular with large deformations, the actual position each of the four arms of the robot may not be accurately predicted for example. When considering an application where the robot arms may enter in contact with the environment, an accurate prediction of their position becomes relevant. In Figure 3, we show the difference in prediction between a coarse mesh and a much finer one. In this case, the error in displacement is concentrated in the region of the "elbows" of the robot. To get an idea of how fine the mesh should be to be accurate enough, we first perform a quick sensitivity analysis on the behaviour of the robot with increasingly finer meshes. We compare the results using four different meshes of increasing size. All the meshes are generated with the CGAL mesh generator[5]. Results are shown in Figure 3. From these experiments, we decide to pick the mesh with 15553 nodes, which presents a relative error of less than one percent with the finest mesh. Indeed, typically, the reduced order model will generate an error of that magnitude. It is hence unnecessary to go further in mesh refinement, unless a much greater accuracy is needed.

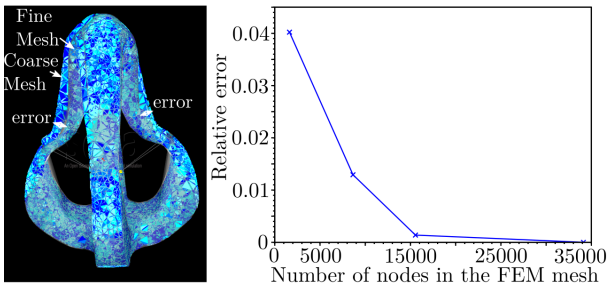

Fig. 3: (Left) Superposition of the robot deformation using two meshes with different refinement in one scenario of cables pulling. We can see that the differences between the fine model and the coarse model of the robot mostly concentrates along the arms. (Right) Relative error in the displacement of the arms of the robot with increasingly finer meshes. The error is computed relative to a finer FE mesh with 34118 nodes and 171603 elements. We consider that the 3^{rd} mesh with 15553 nodes is fine enough, given that the accuracy barely improves with a mesh more than twice as fine.

b) Finite Element model used: We will consider two full FE models using two different constitutive laws:

- A linear elastic law with a co-rotational formulation. The young Modulus is set to 450 MPa, which corresponds to sensible value for silicone.
- A hyperelastic law, namely a Mooney-Rivlin law. For consistency, the parameters values C_{01}, C_{01} and D (from (10)) are chosen to match those of the linear elastic law in the limit of small displacements.

c) Generation of the reduced basis: We proceed by "shaking" the robot following each cable actuator range. A range of $[0, 40]$mm is defined for each actuator, and the snapshot phase test all possible combinations of actuating cases at the ends of the range. With these 4 actuators with a 1-dimensional range, this results in a manageable set of simulations to perform. Indeed, we want to make the robot reach all $2^4 = 16$ different extremity states that are accessible within the range of the actuators. Dividing the loading from one state to another in 9 steps, this provides a set of $9 * 15 + 1 = 136$ positions of the robot, following the same principle displayed in Figure 1.

Performing a singular value decomposition to obtain the reduced basis, we define an error truncation criteria

[5]www.cgal.org

$\nu = 10^{-3}$. From equation (15) this criteria can be achieved with a selection of 34 basis vectors when using the elastic law, and 30 when using the Mooney-Rivlin hyperelastic formulation. This error criterion is only valid on the snapshot space. If that snapshot space is exhaustive enough, the actual error between the true model and the reduced model should be quite small. Some of those vectors are displayed in Figure 4.

The system approximation method (ECSW), described in the previous section, is then applied to reduce the computational costs with a tolerance τ varying from 0.03 to 0.1 in the minimisation (21). This leads to reduced integration domains of various sizes.

d) Numerical Results: In Tables I and II, we compare the accuracy of the simulation using the coarse mesh versus using the reduced order model with various sizes. A more graphical representation of the performances of the reduced order model is displayed in Figure 6. The error is measured with respect to the full order model evaluated on the fine mesh. Several remarks can be made.

- First of all, we can see that the reduced model can be evaluated at a much faster rate than its full order counterpart, no matter which constitutive law is used (elastic or hyperelastic). Indeed, for the ROM with 34 basis vectors and a reduced integration domain of 376 elements, the speedup is around 58.
- As expected, the accuracy is directly linked to the size of the ROM. Fewer elements in the ROM leads to a larger error.
- It becomes clear, especially on Figure 6, that, with an appropriate size of the ROM, a frame rate at least as large as the full order model on the coarse mesh model can be achieved while having a much better accuracy. In other words, the ROM based on the expensive but accurate full order model on the fine mesh keeps a good accuracy while being computationally as inexpensive as a full order model based on a much coarser mesh and hence quite inaccurate. This is particularly true for the simulation using the hyperelastic formulation, where the accuracy is almost an order of magnitude better than the full coarse model while having a runtime twice as low.

In the context of simulation of soft robots for real-time control applications, this opens up the possibility to simulate interactively the behaviour of soft robots with a higher accuracy. In particular, this allows to start from a real accurate digital description of the robot with the finite element method before applying the reduced order modelling workflow to make the simulation computationally tractable within the real-time control applications constraints.

Model	Goal Error	Arms Error	Tps
Coarse FOM	0.034	0.061	45
HROM (34 ,275)	0.015	0.014	50
HROM (34, 376)	0.008	0.010	41
ROM (34)	0.003	0.008	0.8
Fine FOM	0.000	0.000	0.7

TABLE I: Relative errors between the simulation with the full order models (FOM) on both coarse and fine meshes and the reduced order models (ROM(size of reduced basis)) and hyperreduced model (HROM(size of reduced basis, size of reduced integration domain)) using a co-rotational elastic constitutive law.

Model	Goal Error	Arms Error	Tps
Coarse FOM	0.036	0.048	22
HROM (30 , 227)	0.0060	0.0049	42
ROM (30)	0.0050	0.0044	1.0
Fine FOM	0.000	0.000	0.6

TABLE II: Relative errors between the simulation with the coarse mesh and the reduced order models and the finer mesh simulation using the Mooney-Rivlin Hyperelastic constitutive law. In that case, the hyperreduced model leads to a much smaller error than the full order model with the coarse mesh while still running about 2 times faster.

B. More complex robot geometry and predictable contact handling: Multigait Soft Robot

In this example, we consider the multigait soft robot, as described in [41]. The robot is displayed in Figure 7. This robot is made of two layers: one thick layer of soft silicone containing the cavities, and one stiffer and thiner layer of Polydimethylsiloxane (PDMS) that can bend easily but does not elongate. The robot is actuated by five air cavities that can be actuated independently. The effect of inflating each cavity is to create a motion of bending. Then, by actuating with various sequences each cavities, the robot can move along the floor. We propose here to make an interactive simulation of the robot crawling on the floor.

a) Finite Element model of the multigait soft robot: To model the multigait soft robot, we use two different internal force contributions, one volumetric one for the main body of the robot, and one surfacic one for the non-stretchy layer of PDMS. A membrane model is used for that layer: bending forces are ignored, and internal forces are only created by stretch. Using this 2D representation, we can avoid meshing that thin layer in 3D which would have required many elements. This decomposition is displayed in Figure 8. For the main body, internal forces are computed as a 3-dimensional force field, ans a corotational elastic constitutive law is used. Young's modulus constitutive parameter is taken from literature, to match the material used in [41]. For the material properties of the membrane representing the PDMS layer, a much higher stiffness is used to model the fact it is not extensible.

To model the actuation due to the pneumatic cavities, we do not use a dynamic model for the pressurised air and its interaction with the cavities, but rather assume a uniform pressure dispatched regularly over the cavities in a direction orthogonal to the surface of the cavity. This approach is explained in more details in [21]. In Figure 9, we can see the effect of the PDMS layer on the deformation

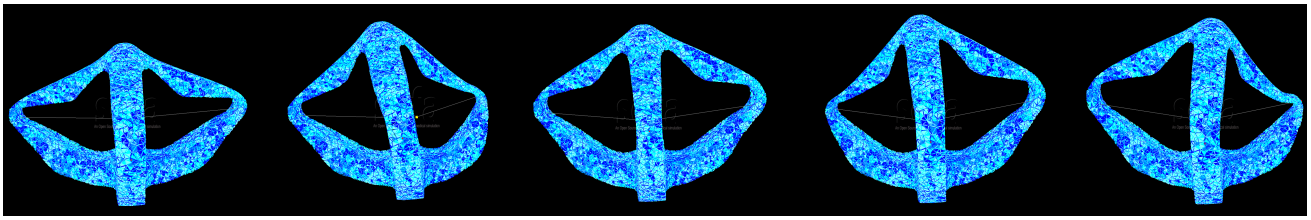

Fig. 4: First few basis vectors generated with the snapshot-POD method.

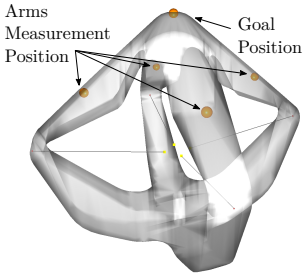

Fig. 5: Position where the error is measured for the arms on one hand, and the top of the robot (Goal) on the other hand.

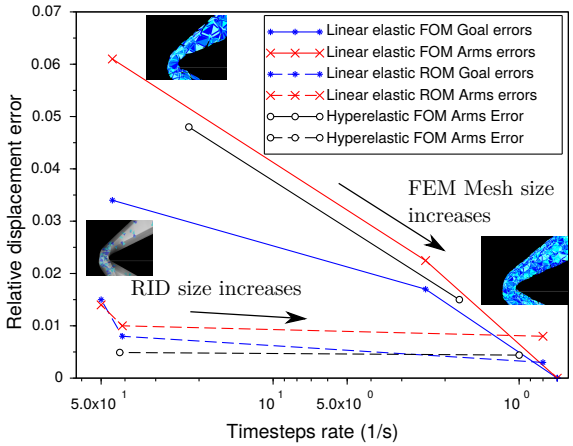

Fig. 6: Comparison between full order models (FOM) with several meshes and reduced order models (ROM) with increasing reduced integration domains, in terms of accuracy versus computational time. The error is measured on the goal point and on the arms of the robot (for the hyperelastic model, we display the error for the arms only for clarity of the graph). Values are extracted from Tables I and II. For the FOMs (continous lines), a more refined mesh implies a better accuracy, but at a high computational cost which results in a timestep rate decrease. To reach Timestep rates allowing interactivity (circa 25 fps for example), the FOM has to be based on a coarse mesh and hence has low accuracy. Comparatively, for the ROMs (discontinuous lines), the interactivity rate can be achieved with an error of an order of magnitude smaller than the FOMs. The timestep rate can increase dramatically while decreasing the size of the RID but the error remains low.

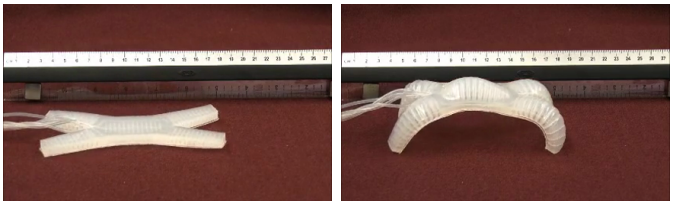

Fig. 7: Multigait soft robot. Picture taken from [41]

of the legs while inflating the corresponding cavity. The layer has the clear effect of bending the leg, which matches the behaviour of the real robot. In Figure 10, we can see a comparison between the real robot and its finite element simulation.

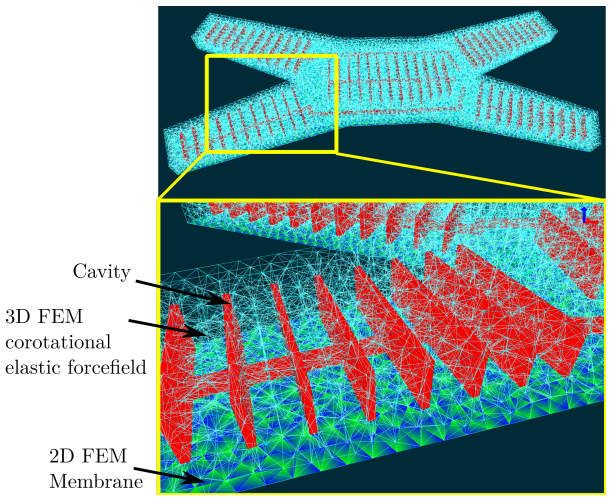

Fig. 8: Zoom in the mesh: a membrane constitutive law is applied onto the bottom layer of the soft robot. That contribution is added to the nodes located at the bottom of the robot.

b) Generation of the reduced basis and rigid body modes: Unlike the previous example, this robot is not fixed on the floor (so a dynamical model is necessary for the simulation) , and the contacts of the floor onto the robot influences its deformation. To generate a relevant snapshot, we perform the offline phase while simulating the robot inside its environment, i.e. onto the floor, to account for those specific deformations but without friction. So during the offline phase, the robot is not moving, it is only deforming while staying at its initial position. The robot may also crawl around on the floor since it is not

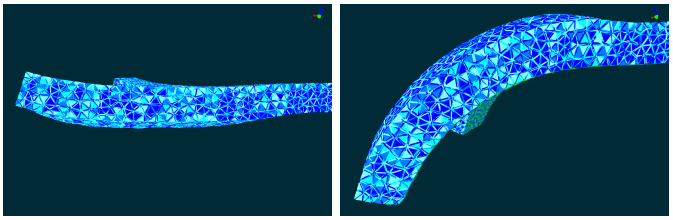

Fig. 9: Effect of the bottom layer on the deformation of the robot captured by the simulation. Left: without bottom layer, the material expands in no special direction. Right: with the bottom layer, the material does bend.

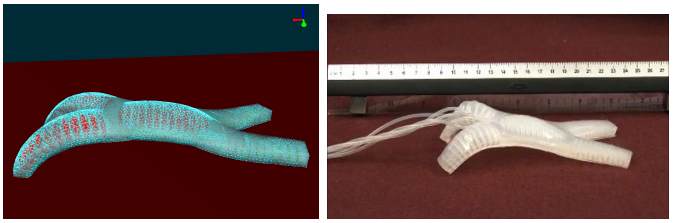

Fig. 10: Comparison between a fine FE model of the soft robot and the real robot (from [41]) with identical pressures in the cavities. The finer mesh is made of 15787 nodes and 59296 elements. In that case, we can see that it reproduces the behaviour of the real robot fairly well.

fixed. To allow the robot to move, the reduced basis should include the rigid body motions $\mathbf{R}$ (i.e. translations and rotations) of the robot to be able to move freely onto the floor. The reduced basis computed from the snapshot does not include those rigid modes $\mathbf{R}$, corresponding to pure translations ($\mathbf{R}_t$) and pure rotations ($\mathbf{R}_\theta$), which do not create internal forces on the robot. Hence, it is necessary to concatenate them to the reduced basis $\mathbf{\Phi}$, to allow for these movements on the reduced model, while erasing their projection onto it to keep the orthogonality of the basis and a good conditioning of the reduced system:

$$\mathbf{\Phi} = [\mathbf{R}\,, \quad \mathbf{\Phi}_{snap} - \sum_{i\in\{R_t,R_\theta\}} \left(\mathbf{R}_i^T \mathbf{\Phi}_{snap}\right) \mathbf{R}_i], \qquad (22)$$

where $\mathbf{\Phi}_{snap}$ is the reduced basis computed with the POD from the snapshot set.

The robot has 5 pneumatic actuators, which means the snapshot generation will involve at least $2^5 = 32$ extreme configurations, if considering only two cases for each actuator (no air pressure and maximum air pressure). For the sake of stability of the simulation, we do not apply the pressure all at once but iteratively between each configuration, over 15 steps. This leads to a snapshot with about 500 configurations. Performing the singular value decomposition with an error tolerance $\nu = 10^{-3}$ (respectively 10^{-2}) from equation (15) leads to a basis of 60 modes (respectively 30). Adding the rigid body modes for translation (we neglect the rotations) leads to a basis with 63 vectors. In Figure 11 are displayed only a few of them.

c) Application of the hyperreduction technique: In the same way to the previous example, to get numerical efficiency, we proceed to apply a method of hyperreduction. The application is slightly different in this case, since there are two distinct internal forces applying on the robot, a three-dimensional one on the body, and a two-dimensional one on the bottom surface. The ECSW procedure is applied independently for each internal force to output a reduced integration domain for each force. The resulting domains are displayed in Figure 12.

d) Speed-up and behaviour of the reduced-order model: Computational time gains are displayed in Table III. We can see that the reduced model is up to more than 100 times faster than the full order model. One interesting point is that unlike the previous example of the parallel cable robot, the reduced model without hyperreduction already has some noticeable gains in term of computational time over the full order model. This is due to the fact that contact handling can be computed in a much faster way with the reduced model since the sytems involved are of much smaller dimensions. In the video material associated with this paper, it can be seen that the reduced simulation of the robot is able to reproduce the undulating movement of the robot. In particular, it is able to replicate the forward movement of the robot following the same sequence of actuation.

Model	Timesteps per second
HROM 33	20
HROM 63	12
ROM 63	0.6
Fine FOM	0.2

TABLE III: Timestep rate for the full order simulation of the multigait soft robot, the reduced model (ROM), and the hyperreduced model (HROM) with 33 modes and 63 modes.

C. Application to the control of a real cable-driven soft robot using an inverse model with contact handling

To show the applicability of the method to inverse kinematics and control of a real robot, we apply the procedure to the tentacle robot described in [20]. This robot is divided in two sections, and actuated by 8 cables, 4 for each section. It also has one actuator allowing to move the whole robot forward in a translation. A drawing of its geometry is displayed in Figure 14. In this section, we will control the robot through inverse kinematics. A end-effector is defined at the tip section of the tentacle robot, and an inverse model based on the reduced order finite element simulation computes the actuation needed for the end-effector to reach a user-defined goal. The inverse problem formulation minimises the distance between the goal and the end-effector, while having a penalty for the mechanical energy of the robot deformation, which guarantees the uniqueness of the solution at each time step. The optimisation problem is solved using a Quadratic program with linear complementarity constraint (QPCC). More details on the definition of the inverse problem as well as its solving can be found in [20].

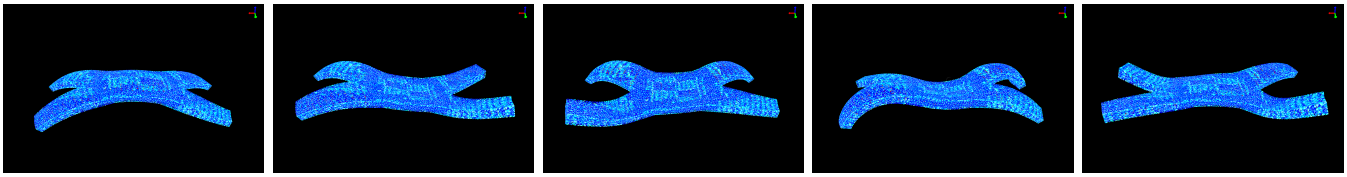

Fig. 11: First basis vectors of the quadruped reduced model. This does not include the translation vectors.

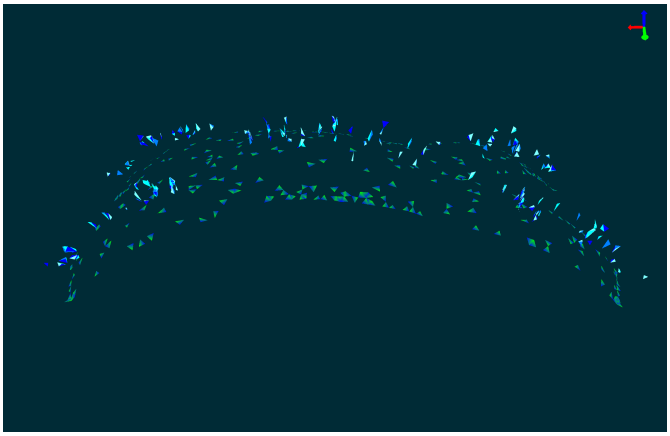

Fig. 12: Reduced integration domains for the multi-gait robot. An independent domain is selected for the three-dimensional force field on one hand, and the two-dimensional membrane force field on the other hand. The combination of both leads to the global reduced integration domain for the reduced model of the whole robot.

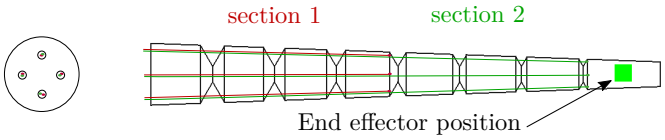

Fig. 14: Slice view and side view of the tentacle robot.

We define a fine discretisation with 28221 tetrahedra (compared to a coarser description with 1565 tetrahedra in [20]). Applying the the reduced order modelling strategies, the robot is tested in $2^8 = 256$ extremity cases of its workspace. With a tolerance set to 10^{-3}, this leads to a reduced model of dimension 33, and with a tolerance $\tau = 0.07$, to a reduced integration domain with 102 elements. The meshes are displayed in Figure 15. Though the coarse mesh was sufficient to control the tentacle robot in [20] with inverse kinematics, the reduced model allows for a greater accuracy, as can be seen in Figure 16. The reduced model simulation is then tested to control the real tentacle robot using an inverse model with contact handling, in the same way this was done in [20]. See Figure 17. The reduced model is compatible with the inverse method for control already used in [18], [20], and can also handle contacts in this case. Indeed, the deformation created by the contact of the tube onto the tentacle robot remains within the deformation space explored during the offline phase to build the reduced model.

V. Discussion

Applying model order reduction allows to have an accurate mechanical model of the robot, typically of the same order than an FE model of 50000 elements, and to achieve a real-time simulation constraint. It makes possible the use of the dynamical model of the robot for simulation as well as for real-time control (in the parallel robot example and on the tentacle robot, we are testing the inverse model, in the multigait robot example, the dynamic model is tested). The method can be used for contacts with the environment, as long as those contacts are taken into account in the snapshot computation phase (the phase where we shake the robot), or if those contacts do not create deformations outside of that space, like in the tentacle robot example. However, it is not able to cope with local contacts, especially if those where not simulated in the "shaking" phase and create deformations that are unreachable using the actuators only. Another point is that it requires a heavy offline stage which grows exponentially with the number of actuators. The use of our method in the design phase allows to test a virtual representation of the robot at fast rates (with an accurate mechanical model). It allows for instance to design the motion sequence of the actuators or to evaluate the obtained deformations, configurations, workspace. We could also use it to design the control loop. However, at each new change of the design (shape of the robot or type of material used), the offline stage needs to be recomputed. Even if this offline stage takes some hours, it takes much less time (and less efforts) than building the actual robot. In future works, we could avoid some re-computation. For example, beyond the range of actuation, it could be possible to vary the material properties to include them in an interactive design phase. However, this implies that the testing phase becomes even larger. Design involving change in the robot geometry can also be included, especially if the robot is to be made with fundamental building bricks. Reduced models of those buildings bricks can be constructed offline, and added interactively in the design phase. For more general arbitrary geometry changes, the method would require to discretise those geometry changes by creating several reduced models, one for each geometry, but at a tremendous computational cost and is left to future works to optimise.

VI. Conclusion and perspectives

The reduced order modelling technique for soft-robots presented in this paper is very useful to have a mechanical model in real-time and to be able to manage the

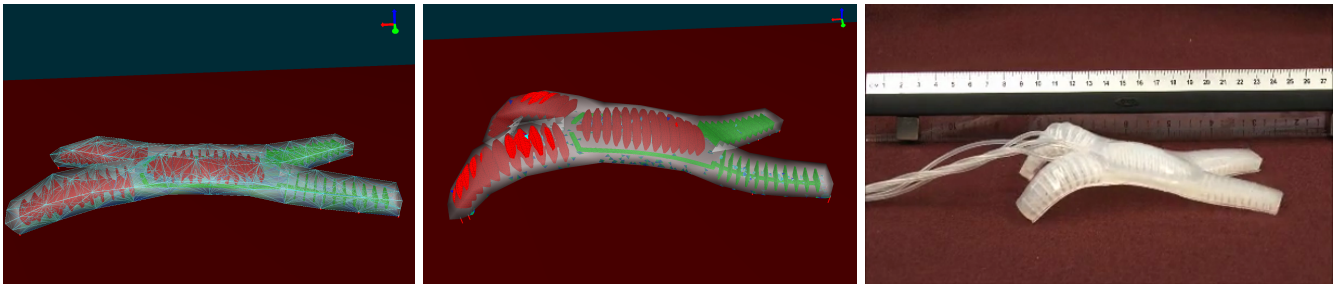

Fig. 13: Comparison between the hyperreduced model of the soft robot and a full order on a coarser mesh. The coarse mesh is made of 2509 nodes and 8222 elements. We can see that the deformation does not have a great amplitude. This is a known fact that FE methods applied on unrealistically coarse meshes tend to artificially stiffen the structure. On the other hand, the ROM has a good behaviour. In that case, we can see that it reproduces the behaviour of the real robot fairly well.

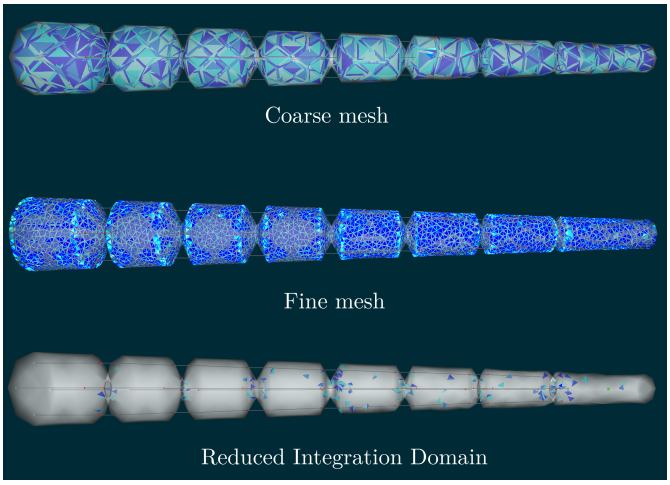

Fig. 15: Finite element meshes of the tentacle: the coarse mesh was used in [20] with 1565 tetrahedra, the fine mesh used to build the reduced model with 28221 tetrahedra, the reduced integration domain used for the real-time control with only 102 tetrahedra.

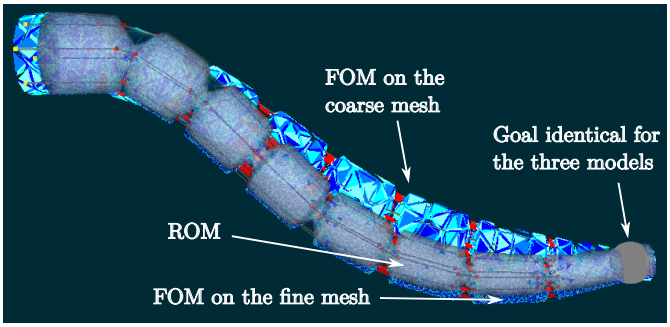

Fig. 16: Superposition of the simulation of the tentacle robot using the full order model (FOM) on the coarse mesh, the fine mesh and the reduced order model (ROM). For all three models, the actuation is managed so that end-effector (corresponding to the tip section of the tentacle) reaches a goal materialised by a grey sphere. We can see that the coarse mesh simulation, being artificially stiffer has some error compared to the fine one. The ROM approximates the fine model with a small error.

compromise between accuracy and computational time. This can be thought of as some sort of machine learning technique, in the sense that a lot of tests are made initially to accumulate data about the robot which is then used to build a reduced model. However, one key difference is that in the case of the reduced-order model, a mechanical model remains, which allows to keep coherency with respect to the physics. This is shown in the case of the multigait soft robot which moves forward while undulating in the reduced-order simulation even if it was tested without friction, and hence remained in the same position during the testing phase. It is however able to move forward in the online phase. Similarly, the snapshot space for the tentacle robot was generated without considering contacts, but the tentacle can deal with contacts in the online stage anyway.

Acknowledgment

The authors would like to thank the FEDER (European Regional Development Fund) that co-funded the COMOROS project from which this work is a part of.

References

[1] M. W. Hannan and I. D. Walker, "Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots," *Journal of Field Robotics*, vol. 20, no. 2, pp. 45–63, 2003.

[2] C. Laschi, M. Cianchetti, B. Mazzolai, L. Margheri, M. Follador, and P. Dario, "Soft robot arm inspired by the octopus," *Advanced Robotics*, vol. 26, no. 7, pp. 709–727, 2012.

[3] S. Kim, C. Laschi, and B. Trimmer, "Soft robotics: a bioinspired evolution in robotics," *Trends in Biotechnology*, vol. 31, no. 5, pp. 287 – 294, 2013.

[4] D. Trivedi, C. D. Rahn, W. M. Kier, and I. D. Walker, "Soft robotics: Biological inspiration, state of the art, and future research," *Appl. Bionics Biomechanics*, vol. 5, no. 3, pp. 99–117, July 2008.

[5] C. Della Santina, M. Bianchi, G. Grioli, F. Angelini, M. Catalano, M. Garabini, and A. Bicchi, "Controlling soft robots: balancing feedback and feedforward elements," *IEEE Robotics & Automation Magazine*, vol. 24, no. 3, pp. 75–83, 2017.

[6] M. Freese, S. Singh, F. Ozaki, and N. Matsuhira, "Virtual robot experimentation platform v-rep: A versatile 3d robot simulator," *Simulation, modeling, and programming for autonomous robots*, pp. 51–62, 2010.

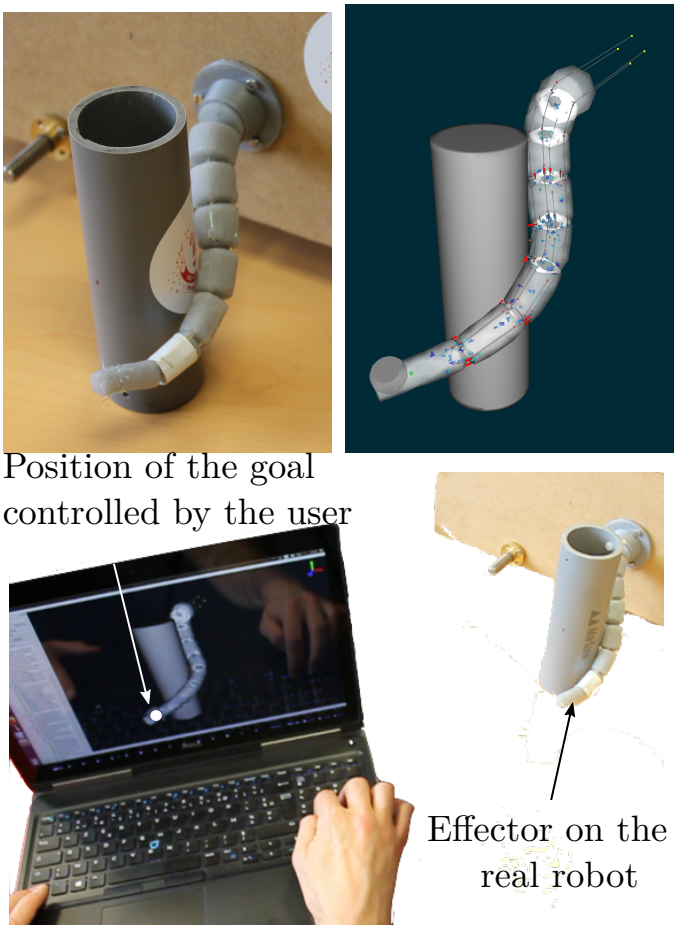

Fig. 17: Real-time control of the tentacle robot using the reduced order simulation and the inverse model with contact handling.

[7] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proceeding of the conference on Intelligent Robots and Systems. (IROS).* , vol. 3, Sept 2004, pp. 2149–2154 vol.3.

[8] F. Kanehiro, H. Hirukawa, and S. Kajita, "Openhrp: Open architecture humanoid robotics platform," *The International Journal of Robotics Research*, vol. 23, no. 2, pp. 155–165, 2004.

[9] R. Smith, "Open dynamics engine," 2005. [Online]. Available: http://www.ode.org

[10] J. Rieffel, F. Saunders, S. Nadimpalli, H. Zhou, S. Hassoun, J. Rife, and B. Trimmer, "Evolving soft robotic locomotion in physx," in *Proceedings of the 11th annual conference companion on genetic and evolutionary computation conference: Late breaking papers.* ACM, 2009, pp. 2499–2504.

[11] E. Coumans, "Bullet physics engine," *Open Source Software: http://bulletphysics.org*, vol. 1, 2010.

[12] T. Erez, Y. Tassa, and E. Todorov, "Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on.* IEEE, 2015, pp. 4397–4404.

[13] D. C. Rucker, B. A. Jones, and R. J. Webster III, "A geometrically exact model for externally loaded concentric-tube continuum robots," *IEEE Transactions on Robotics*, vol. 26, no. 5, pp. 769–780, 2010.

[14] J. Spillmann and M. Teschner, "Corde: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '07, 2007, pp. 63–72. [Online]. Available: http://dl.acm.org/citation.cfm?id=1272690.1272700

[15] D. Trivedi, A. Lotfi, and C. D. Rahn, "Geometrically exact models for soft robotic manipulators," *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 773–780, 2008.

[16] J. Hiller and H. Lipson, "Dynamic Simulation of Soft Multimaterial 3D-Printed Objects," *Soft Robotics*, vol. 1, no. 1, pp. 88–101, 2014.

[17] H. Zhang, Y. Wang, M. Y. Wang, J. Y. H. Fuh, and A. S. Kumar, "Design and analysis of soft grippers for hand rehabilitation," in *ASME 2017 12th International Manufacturing Science and Engineering Conference collocated with the JSME/ASME 2017 6th International Conference on Materials and Processing.* American Society of Mechanical Engineers, 2017, pp. V004T05A003–V004T05A003.

[18] C. Duriez, "Control of Elastic Soft Robots based on Real-Time Finite Element Method," in *IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, France, 2013, pp. 3982–3987. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00823766

[19] Z. Zhang, J. Dequidt, A. Kruszewski, F. Largilliere, and C. Duriez, "Kinematic modeling and observer based control of soft robot using real-time finite element method," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on.* IEEE, 2016, pp. 5509–5514.

[20] E. Coevoet, A. Escande, and C. Duriez, "Optimization-based inverse model of soft robots with contact handling," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1413–1419, 2017.

[21] C. Duriez, E. Coevoet, F. Largilliere, T. Morales-Bieze, Z. Zhang, M. Sanz-Lopez, B. Carrez, D. Marchal, O. Goury, and J. Dequidt, "Framework for online simulation of soft robots with optimization-based inverse model," in *Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR), IEEE International Conference on.* IEEE, 2016, pp. 111–118.

[22] E. Sifakis and J. Barbic, "Fem simulation of 3d deformable solids: A practitioner's guide to theory, discretization and model reduction," in *ACM SIGGRAPH 2012 Courses*, ser. SIGGRAPH '12. New York, NY, USA: ACM, 2012, pp. 20:1–20:50.

[23] L. Sirovich, "Turbulence and the dynamics of coherent structures. part I: coherent structures," *Quarterly of Applied Mathematics*, vol. 45, pp. 561–571, 1987.

[24] O. Goury, "Computational time savings in multiscale fracture mechanics using model order reduction," Ph.D. dissertation, Cardiff University, 2015.

[25] K. Carlberg, C. Bou-Mosleh, and C. Farhat, "Efficient non-linear model reduction via a least-squares petrov–galerkin projection and compressive tensor approximations," *International Journal for Numerical Methods in Engineering*, vol. 86, no. 2, pp. 155–181, 2011.

[26] D. Ryckelynck, "A priori hyperreduction method: an adaptive approach," *Journal of Computational Physics*, vol. 202(1), pp. 346–366, 2005.

[27] C. Farhat, P. Avery, T. Chapman, and J. Cortial, "Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency," *International Journal for Numerical Methods in Engineering*, vol. 98, no. 9, pp. 625–662, 2014.

[28] J. A. Hernandez, M. A. Caicedo, and A. Ferrer, "Dimensional hyper-reduction of nonlinear finite element models via empirical cubature," *Computer methods in applied mechanics and engineering*, vol. 313, pp. 687–722, 2017.

[29] J. Chenevier, D. González, J. V. Aguado, F. Chinesta, and E. Cueto, "Reduced-order modeling of soft robots," *PloS one*, vol. 13, no. 2, p. e0192052, 2018.

[30] F. S. Sin, D. Schroeder, and J. Barbič, "Vega: Non-linear fem deformable object simulator," in *Computer Graphics Forum*, vol. 32, no. 1. Wiley Online Library, 2013, pp. 36–48.

[31] J. Barbič and D. L. James, "Real-time subspace integration for st. venant-kirchhoff deformable models," in *ACM transactions on graphics (TOG)*, vol. 24, no. 3. ACM, 2005, pp. 982–990.

[32] H. Xu, Y. Li, Y. Chen, and J. Barbič, "Interactive material design using model reduction," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 2, p. 18, 2015.

[33] M. Mooney, "A theory of large elastic deformation," *Journal of applied physics*, vol. 11, no. 9, pp. 582–592, 1940.

[34] O. Goury, D. Amsallem, S. P. A. Bordas, W. K. Liu, and P. Kerfriden, "Automatised selection of load paths to construct

reduced-order models in computational damage micromechanics: from dissipation-driven random selection to bayesian optimization," *Computational Mechanics*, vol. 58, no. 2, pp. 213–234, 2016.

[35] A. Quarteroni, G. Rozza, and A. Manzoni, "Certified reduced basis approximation for parametrized partial differential equations and applications," *Journal of Mathematics in Industry*, vol. 1, no. 3, pp. 1–44, 2011.

[36] T. Bui-Thanh, K. Willcox, and O. Ghattas, "Model reduction for large-scale systems with high-dimensional parametric input space," *SIAM Journal on Scientific Computing*, vol. 30, no. 6, pp. 3270–3288, 2008.

[37] A. Paul-Dubois-Taine and D. Amsallem, "An adaptive and efficient greedy procedure for the optimal training of parametric reduced-order models," *International Journal for Numerical Methods in Engineering*, vol. 102, no. 5, pp. 1262–1292, 2015.

[38] S. Chaturantabut and D. Sorensen, "Discrete empirical interpolation for nonlinear model reduction," in *Proceedings of the 48th IEEE Conference on Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009.* Ieee, 2004, pp. 4316–4321.

[39] C. L. Lawson and R. J. Hanson, *Solving least squares problems*. SIAM, 1995.

[40] R. Peharz and F. Pernkopf, "Sparse nonnegative matrix factorization with ℓ^0-constraints," *Neurocomputing*, vol. 80, pp. 38–46, 2012.

[41] R. F. Shepherd, F. Ilievski, W. Choi, S. A. Morin, A. A. Stokes, A. D. Mazzeo, X. Chen, M. Wang, and G. M. Whitesides, "Multigait soft robot," *Proceedings of the National Academy of Sciences*, vol. 108, no. 51, pp. 20 400–20 403, 2011.