Condensed semi-implicit dynamics for trajectory optimization in soft robotics

Etienne Ménager, Alexandre Bilger, Wilson Jallet, Justin Carpentier, Christian Duriez

Condensed semi-implicit dynamics for trajectory optimization in soft robotics

Etienne Ménager[1*], Alexandre Bilger[1], Wilson Jallet[2,3], Justin Carpentier[2] and Christian Duriez[1]

***Abstract*— Over the past decades, trajectory optimization (TO) has become an effective solution for solving complex motion generation problems in robotics, ranging from autonomous driving to humanoids. Yet, TO methods remain limited to robots with tens of degrees of freedom (DoFs), limiting their usage in soft robotics, where kinematic models may require hundreds of DoFs in general. In this work, we introduce a generic method to perform trajectory optimization based on continuum mechanics to describe the behavior of soft robots. The core idea is to condense the dynamics of the soft robot in the constraint space in order to obtain a reduced dynamics formulation, which can then be plugged into numerical TO methods. In particular, we show that these condensed dynamics can be easily coupled with differential dynamic programming methods for solving TO problems involving soft robots. This method is evaluated on three different soft robots with different geometries and actuation.**

Keywords: Optimization and Optimal Control. Modeling, Control, and Learning for Soft Robots. Soft Robot Applications.

I. Introduction

In rigid robotics and humanoid robotics in particular, optimal control is an efficient way to formalize and solve multi-time step control problems [1], [2], [3]. The general idea of this method is to minimize a cost function over a time horizon while satisfying the system dynamics and various constraints, such as actuation constraints. Many methods exist to solve this kind of problem. Among them, Differential Dynamic Programming-based methods [4], [5] are often used because they efficiently exploit the temporal structure of the problem and satisfy the dynamics of the robot for every iterate of the method. Recent variations of this method [6], [7], [8], [3], [9] allow accounting for additional constraints related to the actuation or system state.

In soft robotics, Model Predictive Control (MPC) models have been studied in previous works and are used for multistep control. MPC is a model-based control method that predicts future trajectories and realizes online planning using this prediction horizon. However, in [10], [11], [12], the robots are controlled in the joint space, which limits their applications. In [13], the robot is controlled in the task space, but the problem formulation is based on the Piece-Wise Constant Curvature (PWCC) assumption, which limits the type of robot or actuation that can be used. Finite Element Methods (FEM) is a good candidate for accurate simulation of a wide range of geometries and actuations. However, FEM simulation is based on the discretization of continuous mechanics and represents the soft robot as a mesh. Such a representation is unsuitable for large optimization problems such as multistep predictive control, because it involves manipulating high-dimensional mathematical quantities. Some learned-based methods can be used to speed up simulation and write MPC problems for soft robots. In [14], authors attempt to capture the dynamics of the system using a spectral submanifold. However, the method uses an approximation of the dynamics, and the applications are limited to robots with equilibrium points. Other works such as [15] use Reduced Order Finite Element Models to implement optimal control methods. The model allows achieving computational efficiency but is based on the construction of a reduced order piecewise affine model to approximate the FEM. To the best of the authors' knowledge, there is no efficient optimal control method based on the FEM modeling of a soft robot.

Contributions: Inspired by the current one-step control in soft robotics and the trajectory optimization tools developed in rigid robotics, we propose a formulation of the trajectory optimization problem in soft robotics based on a condensed semi-implicit dynamics. The general idea of the method is to use the high-dimensional FEM model of the robot to simulate its behavior and a low-dimensional linearized condensed dynamics to optimize the actuation. This method is tested in simulation on three robots with different geometries and actuation.

II. Background

A. FEM simulation of soft robot

The behavior of soft robots can be described via the mechanical equations of continuous media. FEM simulation software is used to obtain an approximate solution of these equations [16], [17]. They can be written as:

$$M\ddot{q}_t = f_{\text{int}}(q_t, \dot{q}_t) + f_{\text{ext}} + H_a^T u \tag{1}$$

where M is the mass matrix, f_{int} the internal forces, f_{ext} the external forces, $\ddot{q}_t$, $\dot{q}_t$, q_t are respectively the acceleration, the velocity and position of the nodes of the FEM mesh, H_a can be seen as the Jacobian matrix of the actuation and depends on the type of the actuator, and u is the Lagrange multiplier which represents the actuation constraints.

The dynamics equation is discretized in time using an integration scheme. In soft robotics, an implicit Euler integration scheme is generally used because of its unconditional stability and its capacity to adapt to the "time-stepping" used in non-regular mechanics to deal with speed jumps on

[1]Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France. [2]Inria, Département d'informatique de l'ENS, Ecole normale supérieur, CNRS, PSL Research University, Paris, France. [3]LAAS-CNRS, 7 av. du Colonel Roche, 31400 Toulouse.

*Corresponding author: etienne.menager@inria.fr

contact. In addition, internal forces are linearized using a first-order approximation. Using h the time step, $q_k = q_{kh}$ and $\dot{q}_k = \dot{q}_{kh}$ the discretized position and velocity, and $d\dot{q} = h\ddot{q} = \dot{q}_{k+1} - \dot{q}_k$, the discretized dynamics is written:

$$\begin{aligned} A &= M - hD - h^2K \\ b(q_k, \dot{q}_k) &= hf_{\text{int}}(q_k, \dot{q}_k) + h^2K\dot{q}_k + hf_{\text{ext}} \\ Ad\dot{q} &= b(q_k, \dot{q}_k) + hH_a^T u \end{aligned} \tag{2}$$

where K and D are respectively the compliance and the damping matrices.

The system $Ad\dot{q} = b + hH_a^T u$ is solved to find the next position and velocity of the FEM mesh points. First, the free configuration of the robot $Ad\dot{q}^{free} = b$ obtained with no actuation $u = 0$ is solved using a linear solver. Then, this value is corrected with the value of the actuation according to $d\dot{q} = d\dot{q}^{free} + hA^{-1}H_a^T u$. Finally, the position and velocity of all the mesh points are computed using the integration scheme.

B. One-step control in soft-robotics

Until now, the control of soft robots using FEM simulation is done on a time step using inverse modeling of the soft robot and a condensed description of its behavior [18]. The movement of a soft robot is controlled using specific points called effectors. The concept of effector and actuator in soft robotics is illustrated in Figure 1, using the Diamond soft robot as an example (see Section IV)).

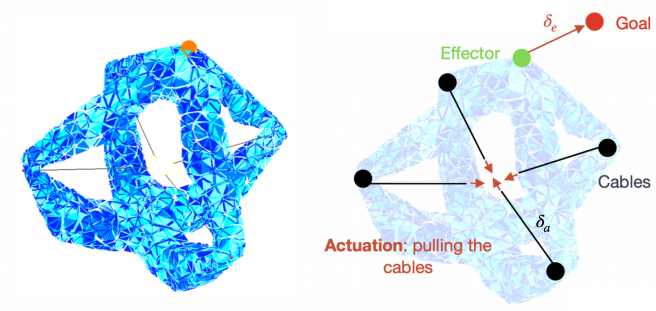

Fig. 1. Example of the Diamond soft robot's actuation and end-effector, that will be developed in Section IV. The effector is a particular point on the robot that we want to control. Its position is modified by applying forces on the actuators.

To achieve position control of the soft robot, the idea is to minimize the distance between the effectors and goals:

$$\delta_e = p - p_{\text{goal}} \tag{3}$$

where p is the position of the effectors.

From this equation, the Jacobian matrix of the effector H_e can be defined. It corresponds to the identity for the corresponding point in the mesh and 0 otherwise. As with the forward modeling described by equation (2), inverse modeling is solved in two steps. In the first step, the free configuration of the robot is computed. Then the violation of the constraints in the free configuration of the robot δ_i^{free} for $i \in \{a, e\}$ is evaluated. Finally, this value is corrected using the actuation by making a linear approximation of the displacement of the nodes with a FEM interpolation. Using the condensed FEM modeling [19], this approximation is written:

$$\begin{aligned} \delta_e &= W_{ea}u + \delta_e^{\text{free}} \\ \delta_a &= W_{aa}u + \delta_a^{\text{free}} \end{aligned} \tag{4}$$

with $W_{ij} = H_i A^{-1} H_j^T$ for $i, j \in \{a, e\}$ the projections of A in the constraint space. The matrices W_{ij} represent the mechanical coupling between the effectors e and the actuators a. These quantities can be used in an optimization scheme to find the actuation that minimizes the distance δ_e and respects some constraints on the actuator's displacement δ_a.

C. Multi-step control in robotics

We consider the following numerical optimal control problem [20]:

$$\begin{aligned} \min_{x,u} J(x,u) &= \sum_{k=0}^{N-1} l_k(x_k, u_k) + l_N(x_N) \\ \text{s.t. } x_{k+1} &= f_k(x_k, u_k) \\ x(0) &= \bar{x}_0 \end{aligned} \tag{5}$$

with J the objective function, N the number of time steps, $\boldsymbol{x} = [x_0, ..., x_N]$ a sequence of states $x_k = [q_k, \dot{q}_k]$, $\boldsymbol{u} = [u_0, ..., u_{N-1}]$ a sequence of control inputs, f_k the discrete dynamics, l_k the cost function at time k depending on the state and the actuation, and l_N the terminal cost function depending on the final state.

There exist several methods [1] for solving this problem. In this article, we consider Differential Dynamic Programming (DDP) [21]. DDP is based on the Bellman principle of optimality, which gives the following recursive formula:

$$V_k(x) = \min_u l_k(x,u) + V_{k+1}(f(x,u)) \tag{6}$$

where V_k is the value function at time k, with the terminal condition $V_N(x) = l_N(x)$.

To solve this equation, DDP computes a Newton step δu in the control input $u_k + \delta u$ for a given variation $x_k + \delta x$ of the previous state x_k during the backward pass. To do so, DDP relies on a quadratic approximation of the so-called Hamiltonian function, defined by:

$$Q_k(x,u) = l_k(x,u) + V_{k+1}(f(x,u))$$

This quadratic approximation around the nominal values (x, y) is given by:

$$\begin{aligned} &Q_k(x + \delta x, u + \delta u) - Q_k(x,u) \approx \\ &\frac{1}{2}\begin{bmatrix} 1 \\ \delta z \\ \delta u \end{bmatrix}^T \begin{bmatrix} 0 & Q_x^T & Q_u^T \\ Q_x & Q_{xx} & Q_{xu} \\ Q_u & Q_{ux} & Q_{uu} \end{bmatrix} \begin{bmatrix} 1 \\ \delta x \\ \delta u \end{bmatrix} \end{aligned} \tag{7}$$

where the different quantities Q_{ij} with $i, j \in \{x, u\}$ are written as a function of the data of the problem, i.e., l, f and V_{k+1} and their first and second derivatives.

As Q_k is quadratic, V_k is too, allowing the backward propagation of its value along the temporal structure of the

problem starting from the end. We can add constraints to the equation [7] and solve it thanks to augmented Lagrangian-based approaches [9]. Once the backward step has been performed, i.e., once the δu sequence has been computed, it is possible to find the step size of the optimization process thanks to backtracking line-search procedure [5] and then perform the forward step. The forward step allows the calculation and evaluation of a new nominal trajectory from a given control sequence.

III. Method

A. Condensed dynamics in soft robotics

To solve multi-step control problem with DDP, we have to write the dynamics of the system using Equation (5) formalism. Equation (2) gives the general dynamics equation of a soft robot using an implicit Euler integration scheme with only one linearization of the internal forces per time step. By isolating $d\dot{q}$ and writing the time dependence of actuation u_k, we can write a semi-implicit scheme:

$$d\dot{q} = A^{-1}b(q_k, \dot{q}_k) + hA^{-1}H_a^T u_k \tag{8}$$

Equation (3) gives the relation between the effectors and the points of the mesh $p = H_e q$. Equation (8) projected in the effector space can be written:

$$d\dot{p} = H_e A^{-1}b(q_k, \dot{q}_k) + hH_e A^{-1}H_a^T u_k \tag{9}$$

where we recognize the condensed FEM matrices in the right member of the equation. Using the implicit integration scheme, the condensed dynamics of the system can be written as:

$$\begin{aligned} \dot{p}_{k+1} &= \dot{p}_k + H_e A^{-1}b(q_k, \dot{q}_k) + hW_{ea}u_k \\ p_{k+1} &= p_k + h\dot{p}_{k+1} \end{aligned} \tag{10}$$

The condensed dynamics of the soft robot can be written in matrix notation:

$$\begin{aligned} X_{k+1} &= \begin{bmatrix} h^2 W_{ea,k} \\ hW_{ea,k} \end{bmatrix} u_k + \begin{bmatrix} I & hI \\ 0 & I \end{bmatrix} X_k \\ &\quad + \begin{bmatrix} hI \\ I \end{bmatrix} H_e A^{-1} b(q_k, \dot{q}_k) \\ \Leftrightarrow X_{k+1} &= f_k(X_k, u_k) \end{aligned} \tag{11}$$

where $X_k = [p_k, \dot{p}_k]^T$.

B. Derivative of the condensed dynamic

As explained in section II-C, DDP uses the derivative of the dynamics to compute the Hamiltonian value. By differentiating Equation (11) according to u_k and X_k we have for the control part:

$$\frac{\partial f_k}{\partial u} = \begin{bmatrix} h^2 W_{ea,k} \\ hW_{ea,k} \end{bmatrix} \tag{12}$$

and for the state using the chain rules and the fact that H_e is its own pseudo-inverse:

$$\begin{aligned} \frac{\partial f_k}{\partial X} &= \underbrace{\begin{bmatrix} I & hI \\ 0 & I \end{bmatrix}}_{\frac{\partial f_k}{\partial X}\text{ linear}} \\ &+ \underbrace{\begin{bmatrix} H_e A^{-1} & 0 \\ 0 & H_e A^{-1} \end{bmatrix} \begin{bmatrix} h^2 K & h^3 K + h^2 D \\ hK & h^2 K + hD \end{bmatrix} \begin{bmatrix} H_e^T & 0 \\ 0 & H_e^T \end{bmatrix}}_{\frac{\partial f_k}{\partial X}\text{ non-linear}} \end{aligned} \tag{13}$$

This quantity is composed of two terms: one corresponding to the derivative of the linear part of the dynamics (with respect to X_k) and the other one to the derivative of the non-linear part.

C. Constraint on the actuation

We can define additional mathematical constraints $h_k(X, u) \leq 0$ in the optimization problem. In this paper, these constraints correspond to actuator limits.

They can be expressed as:

$$h_k(X, u) = h_k(u) = \begin{bmatrix} -I \\ I \end{bmatrix} u + \begin{bmatrix} u_{\min} \\ -u_{\max} \end{bmatrix} \leq 0 \tag{14}$$

D. General method to solve the trajectory optimization problem

The function f_k corresponds to the projection of the robot dynamics into the effectors/actuators space. As explained in section II-B, this projection has already been used for one-time step control. The idea is to use the condensed FEM model to find the actuation that minimizes the cost function around each step of the nominal trajectory.

The derivatives used for the optimization problem depend on the mechanical quantities W_{ea}, A^{-1}, K, and D. Technically, these quantities depend on the position and velocity of all the mesh points of the robot. We assume that we are looking for an actuation around a nominal trajectory and that the mathematical quantities vary slightly around this trajectory. This means that these quantities are constant for one time step of the backward phase. Moreover, by improving the nominal trajectory with a succession of forward and backward steps, the error of the linear approximation is reduced (small displacement around the nominal trajectory) and there is little change in the values of W.

It is, therefore, necessary to compute a faithful nominal trajectory to obtain useful admittance, stiffness, and damping values for the optimization problem. To do so, the forward phase uses the complete dynamics of the robot, and the simulation is used to compute the next values of W_{ea}, A^{-1}, K, and D for the backward phase. When only the derivative of the linear part of the dynamics is used in the backward phase, only W_{ea} is recovered from the simulation.

The global method is summarized in Figure 2. Trajectory optimization is achieved through several forward and backward phases. At each iteration of the optimization process, the actuation sequence is tested with the complete FEM

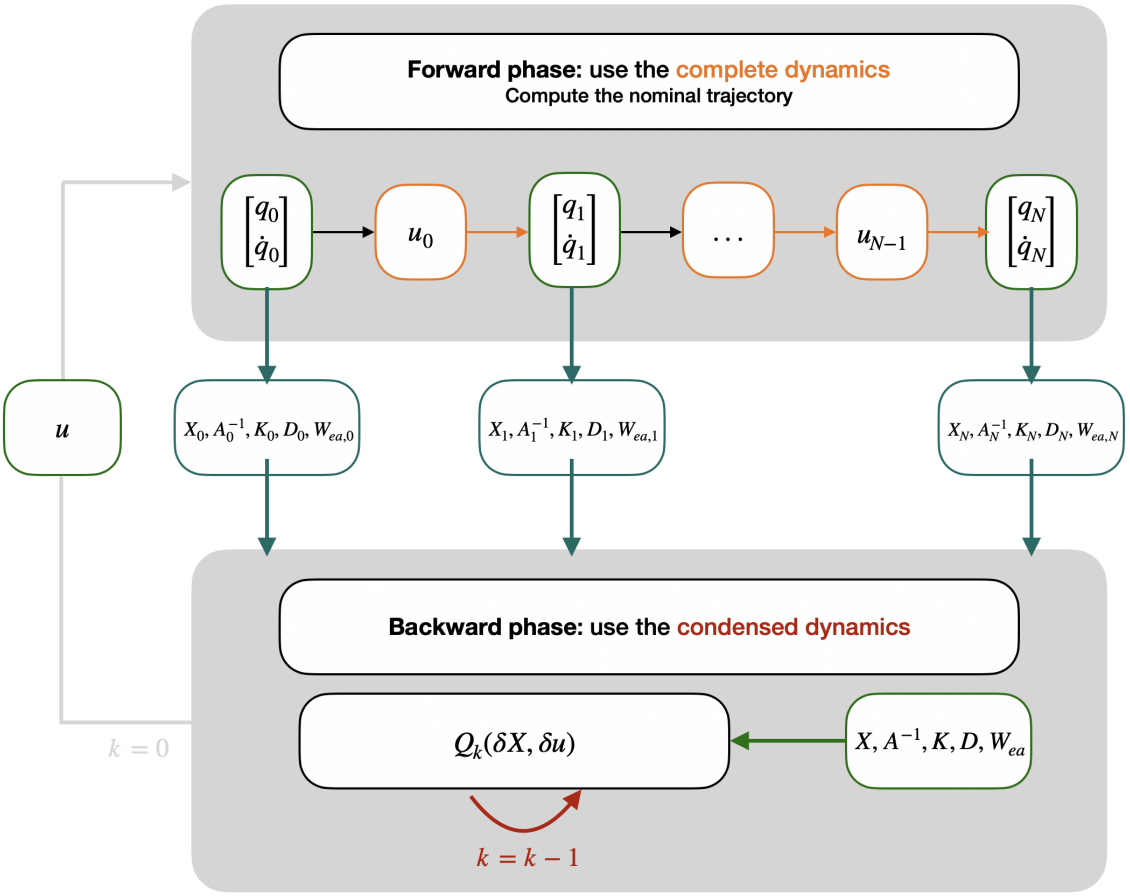

Fig. 2. Illustration of the trajectory optimization pipeline. Starting from an initial position, the forward phase computes the different positions and velocities of the complete mesh using the full dynamics (in orange). From the simulation data, a nominal trajectory is extracted, and the mechanical quantities necessary for the DDP are saved (in blue). The nominal trajectory (green) is used with the condensed trajectory (red) to compute the new action sequence that will be given to the forward pass. Several forward and backward steps may be necessary to converge to a solution.

model during the forward phase and a new nominal trajectory is computed. Mechanical quantities are extracted from this nominal trajectory to create the condensed FEM model. This model is used in the backward phase to calculate a new actuation sequence that minimizes the cost function.

IV. Materials and Results

A. Soft robots

We propose to illustrate the method using three different robots. These robots have different geometry and/or actuation and are simulated using the FEM model based on SOFA framework [17].

- The Diamond robot is a soft parallel silicone robot actuated by 4 cables. The FEM model of this robot has been validated in [16].
- The Stiff-Flop robot is a slender soft robot actuated by pneumatic cavities. Each section of the two sections of the robot is composed of three cavities, allowing the bending in two directions. The FEM model of this robot has been validated in [22].
- The Liandford robot is a soft parallel robot with 6 deformable legs developed for haptic applications. Each end of the legs is attached to a DC motor, whose rotation causes the legs to deform. The effector is attached to a central bar, whose movement is imposed by the deformation of the legs.

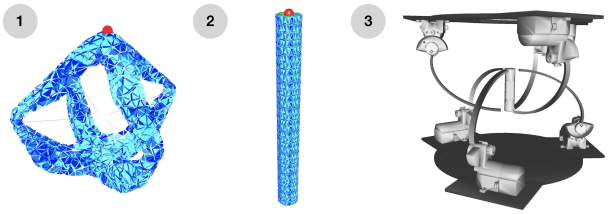

Fig. 3. Representation of the robots used in this work, simulated using SOFA. 1) The Diamond robot. 2) The Stiff-Flop robot. 3) The Liandford robot.

Figure 3 shows these different robots in simulation.

B. Trajectory optimization

The trajectory optimization is performed using the Prox-DDP solver [3], [9] implemented within the ALIGATOR library [23]. This solver uses a primal-dual augmented Lagrangian approach to solve the optimization problem. The

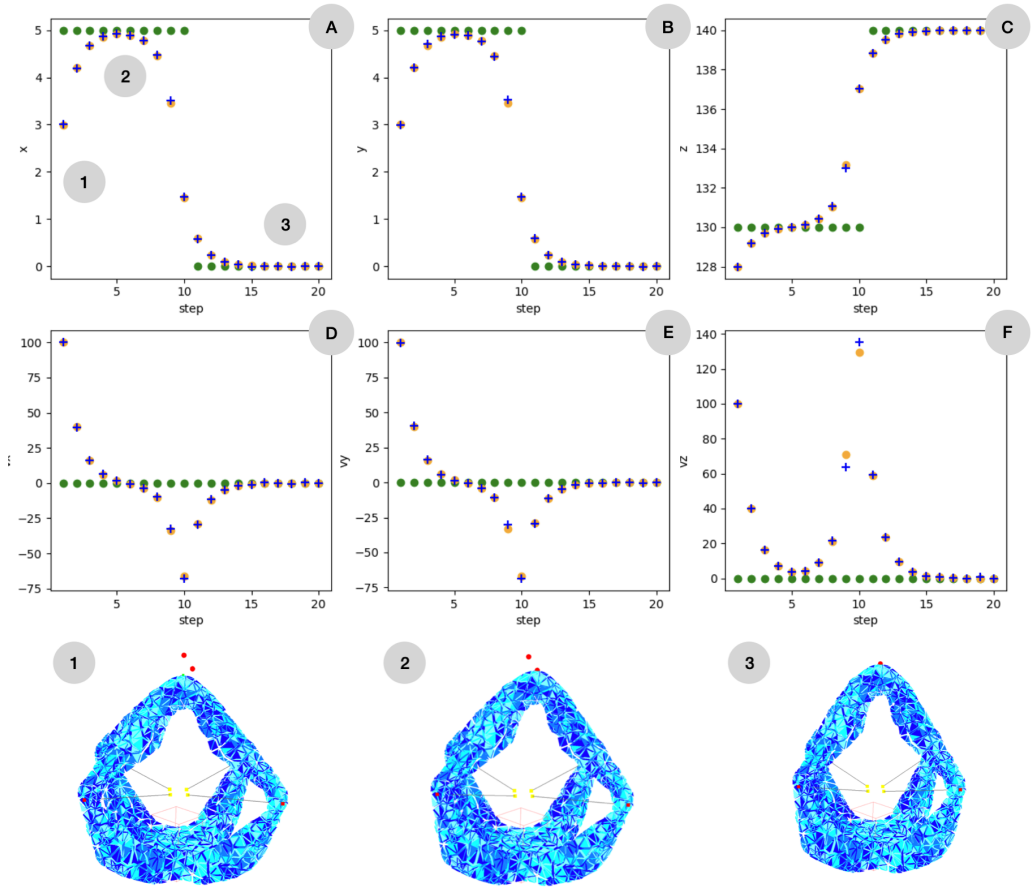

Fig. 4. Results obtained for Diamond robot control. End effector position along x (A), y (B), and z (C) axes. End effector speed along x (D), y (E), and z (F) axes. Green dot: the objective. Blue cross: results obtained by considering the derivative of the non-linear part of the condensed dynamics. Orange dot: results obtained by considering only the linear part. Three robot positions along the trajectory (1, 2, 3) are shown.

user interface allows for defining the underlying optimal control problem using a node-by-node basis, to write the dynamics and constraints, and to specify the differences between forward and backward stages. The forward step has been adapted in order to link with SOFA. Each time step in the simulation corresponds to 0.01 seconds. The different goals are selected according to the points that can be reached with one-time step-control approach. This allows to obtain scenarios with goals in the workspace of the robot. The code is available in Open Source on the GitHub page of the project. The results are presented for a fixed maximum number of iterations and tolerance.

1) Scenario 1: Use both derivatives of the linear and nonlinear part of the dynamic in the backward step: The first scenario compares the positioning performance obtained when the derivative of the non-linear part of the dynamics is used in the backward phase and when this is not the case. This comparison is illustrated with the Diamond robot and shown in Figure 4. The goal is to position the effector of the Diamond at two successive positions.

From these results, we can see that using condensed dynamics allows the control of the position and velocity of the robot, both with and without the derivative of the non-linear part of the condensed dynamics. The results in terms of positioning error are similar with and without the use of $\frac{\partial f_i}{\partial X}_{\text{non-linear}}$. However, the number of iterations to reach the same performances is more critical with the nonlinear part (reach the maximum of 1000 iterations set for this experiment) rather than without (208 iterations). In addition, using the derivative of the non-linear part leads to additional calculations, as the resulting matrix is not calculated directly in the forward stage. More precisely, the matrix W is obtained with $O(m^2)$ operations, where m is the number of constraints. The non-linear part of the derivative is obtained with $O(m \times n)$ operations, where n is the number of elements in the FEM mesh. n is a large number that depends on the accuracy of the FEM model. Therefore, considering only the linear part of the derivative of the dynamics according to X in the backward stage improves the calculation time.

This scenario shows that in the case of the Diamond robot, using the derivative of the linear part alone in the backward is sufficient to control the soft robot. One possible explanation of this result is that the full non-linear FEM model is used to find the nominal trajectory, and linearization around this nominal trajectory in the backward is sufficient to converge to a solution. According to this result and the computation

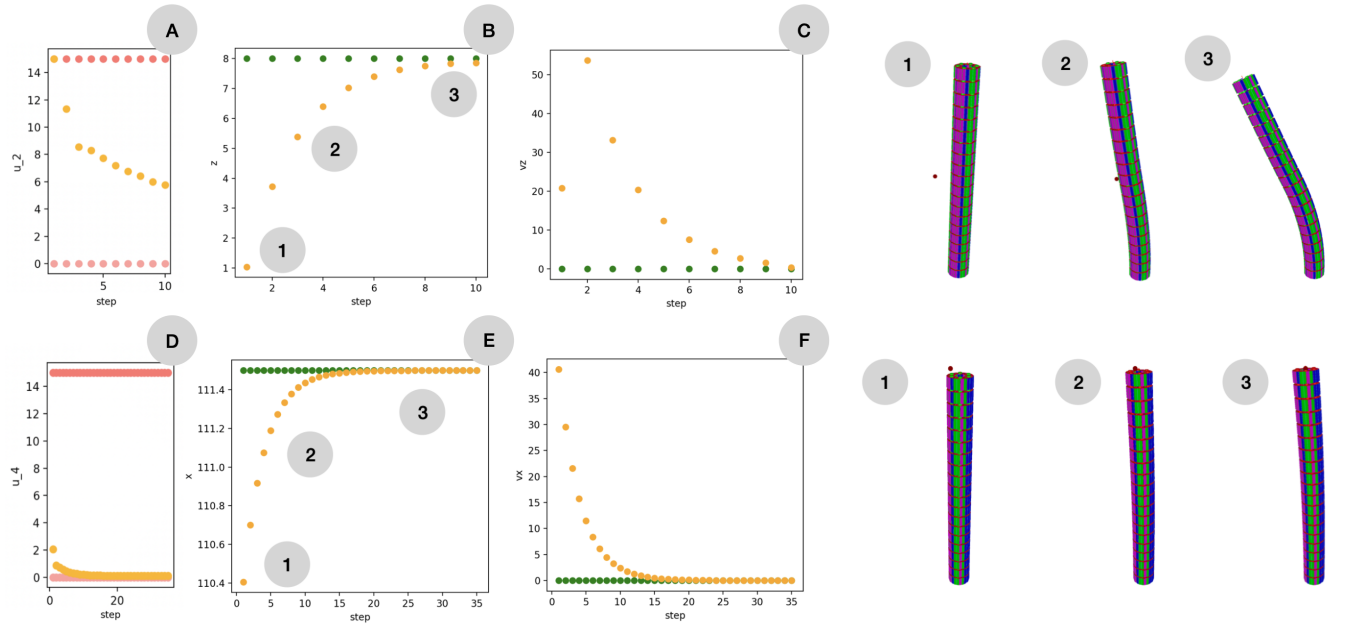

Fig. 5. Results obtained for the Stiff-Flop robot. First row: Effector at the middle of the robot. Second row: Effector at the end of the robot. Example of actuation (A-D); position of the effectors along one axis (B-E); velocity of the effectors along the same axis (C-F); three configurations of the robots (1, 2, 3). Green: the objective. Orange: position and velocity of the effector. Pink: actuation bounds.

time reduction, only the derivative of the linear part of the dynamics is used in the remainder of this work. The practical impact of this approximation for more general cases, such as highly non-linear materials, is left for future work.

2) Scenario 2: Constraint actuation with the Stiff-Flop robot: The second scenario shows that we can constrain the actuation during the trajectory. This scenario is illustrated using the Stiff-Flop robot. The results are shown in Figure 5. This robot illustrates the method with a different type of actuation. The objective is to position the end-effector attached to the center or the end-effector of the robot.

The results are shown for one axis and one actuator, but similar results can be plotted for the other axis and actuators. The framework enables to take into account constraints on the actuation and can integrate various kinds of geometry and actuation. Different deformations can be obtained depending on the position of the end-effector and the target. In practice, all the positions in the robot's workspace can be reached if enough time steps are available to solve the task.

C. Scenario 3: Multiple goals with the Liandford robot

The first two scenarios focus on positioning one effector with a maximum of two targets. In addition, the two previous robots are simulated with volumic FEM models, while the Liandford robot is simulated with a lattice of beam FEM elements. The third scenario involves optimizing a trajectory with multiple goals. We consider three different cases. The first is a spiral trajectory, where each time step corresponds to a new goal. The second is a pick-and-place task where the effector at the center of the robot has to move back and forth between two positions. The last is an orientation task using two effectors at the same time. The results are shown in Figure 6 and Figure 7.

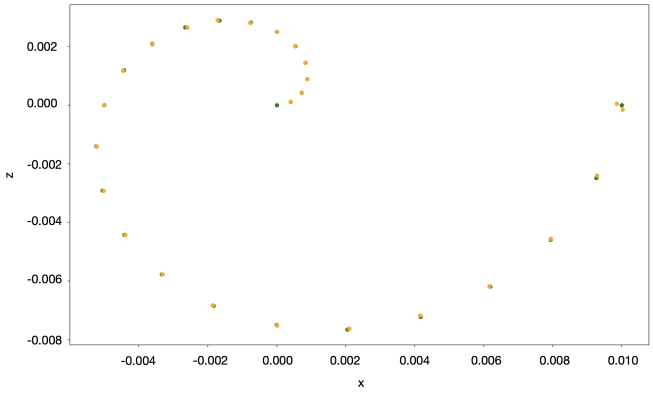

Fig. 6. Results obtained for the Liandford robot control for the first case: position on the x-z plane for the spiral trajectory. Green: the objective. Orange: position of the effector on the considering plane.

The first task shows that the robot can be controlled over several time steps, with a goal that varies at each time step. The obtained trajectory corresponds to the one that minimizes the cost function and follows the constraints at the tolerance level set. There are a few differences between the target trajectory and the computed trajectory: the target trajectory only takes into account the positioning of the effector, whereas the computed trajectory minimizes an objective that includes not only the position but also the velocity of the effector and the actuation value. Changing the weight of these elements in the loss leads to different results. Here, the emphasis is on positioning. The second task shows that it is possible to control the robot's dynamics on more complex trajectories, such as tasks where it has to navigate between

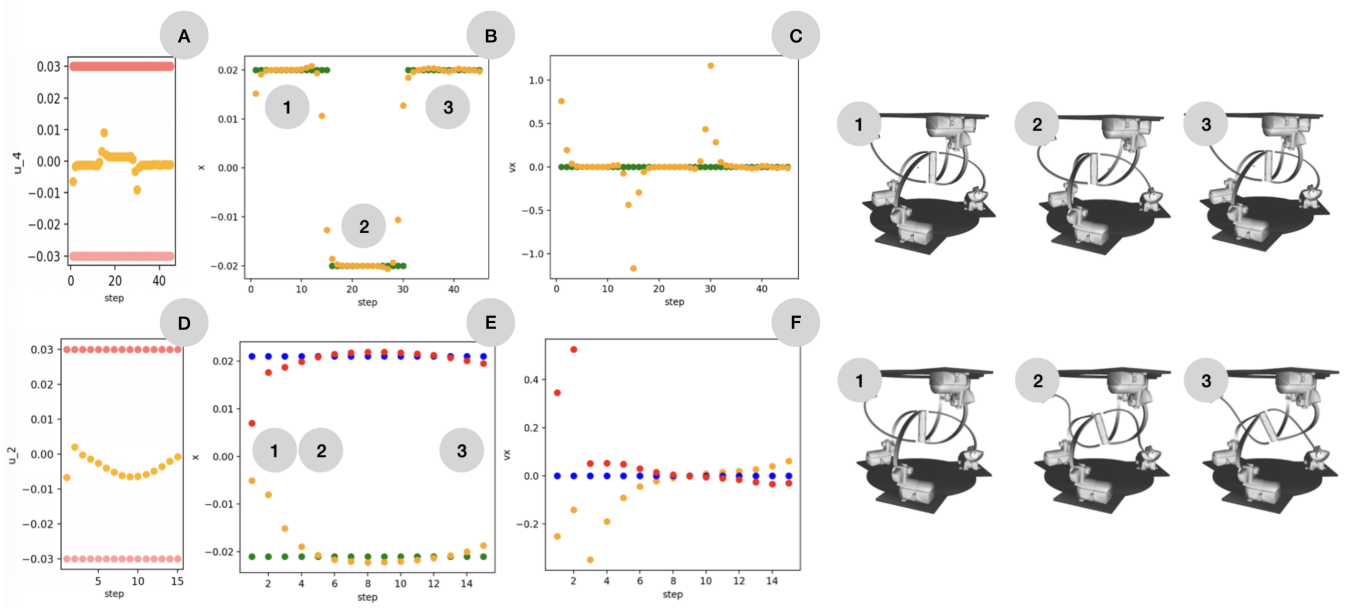

Fig. 7. Results obtained for the Liandford robot control for the second and third cases. First row: actuation (A), position (B), velocity (C), and three configurations of the robot for the pick and place task. Green: the objective. Orange: position and velocity of the effector. Pink: actuation bounds. Second row: actuation (D), position (E), velocity (F), and three configurations of the robot for the orientation task. Green and Blue: the objective. Orange and Red: position and velocity of the effector. Pink: actuation bounds.

two different positions. Finally, the third task shows that it is possible to control several effectors simultaneously.

V. Discussion and Conclusion

We have proposed a method for optimizing the trajectory of soft robots using dynamic FEM models. This method is based on the joint use of a full FEM model to obtain an accurate nominal trajectory in the forward phase and the use of a condensed FEM model to find the actuation in the backward phase. This method can handle several types of robot geometry (parallel or slender), several types of FEM elements (tetra, hexa, beam), several types of actuation (cable, pneumatic, joint), and several types of task (one goal, several goals, one trajectory). We have used three different examples to illustrate all these features.

For the moment, we realize trajectory optimization and not an interactive control. The trajectory is optimized during an offline phase and then applied to the robot in an open loop. Future work will be to mix this method with a correction loop to obtain a robust controller of the soft robots. In addition, the method is based on the simulation of all the FEM models during the forward phases, which is a bottleneck for real-time simulation. The simulation time depends on the length of the trajectory and the size of the FEM mesh. One way to overcome this limitation is to use model reduction or a learned model in the forward phase. The advantage of this method over previous ones is that the optimization does not depend on the choice of model used for the simulation, the condensation being valid for FEM models, reduced models, or beam models. The only criterion that the forward model must satisfy is to provide the mechanical quantities needed to build the condensed model.

The fact that only the derivative of the linear part of the dynamics is used in the backward step means that no additional computations are required to be compared with the forward step and speed up the convergence of the method. The Diamond example showed that the two approaches lead to similar results. Using the derivative of the non-linear part of the dynamics with the other robots didn't yield the same results as those obtained with just the linear part, with the same number of iterations. To achieve the same performance, some additional work on algorithmic performance and computation is required. However, the linear part alone allows the algorithm to converge on an optimal solution to our problem. In addition, the dynamics used in the forward stage and the linearized condensed dynamics used in the backward stage are not the same. This leads to different gradients between those calculated from the position of the end-effectors and those calculated from the condensed model. This can lead to the framework's dual error not converging, although in practice, this does not affect the control of the robots.

It is also possible to extend the capabilities of the method, in particular by adding different constraints. For example, it would be possible to imagine constraints on the maximum displacement increment of actuators or constraints on the position of the end effector by defining zones where it is not allowed to go. Another extension would be to consider perturbations in the trajectory. This would enable the development of tasks where a force is applied to the robot at a given point in the trajectory. Finally, we are using a line search approach to update the variables, but other methods, such as trust region approaches, could be considered.

ACKNOWLEDGMENTS

This work was supported in part by the French government under the management of Agence Nationale de la Recherche through the project NIMBLE (ANR-22-CE33-0008) and as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute), and the Louis Vuitton ENS Chair on Artificial Intelligence. It was also supported by a grant overseen by the French National Research Agency (ANR) and France 2030 as part of the Organic Robotics Program (PEPR O2R). Finally, this work was supported by the TIRREX project, grant ANR-21-ESRE-0015.

REFERENCES

[1] M. Diehl, H. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," *Lecture Notes in Control and Information Sciences*, vol. 340, 07 2007.

[2] G. Schultz and K. D. Mombaur, "Modeling and optimal control of human-like running," *IEEE/ASME Transactions on Mechatronics*, vol. 15, pp. 783–792, 2010. [Online]. Available: https://api.semanticscholar.org/CorpusID:42855787

[3] W. Jallet, N. Mansard, and J. Carpentier, "Implicit Differential Dynamic Programming," in *International Conference on Robotics and Automation (ICRA 2022)*. Philadelphia, United States: IEEE Robotics and Automation Society, May 2022. [Online]. Available: https://hal.science/hal-03351641

[4] D. H. Jacobson and D. Q. Mayne, "Differential dynamic programming. modern analytic and computational methods in science and mathematics," *American Elsevier*, 1970.

[5] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4906–4913, 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:6652724

[6] G. Lantoine and R. P. Russell, "A hybrid differential dynamic programming algorithm for constrained optimal control problems. part 2: Application," *Journal of Optimization Theory and Applications*, vol. 154, pp. 418–442, 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:8512224

[7] Z. Xie, C. K. Liu, and K. K. Hauser, "Differential dynamic programming with nonlinear constraints," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 695–702, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:7648776

[8] A. Pavlov, I. Shames, and C. Manzie, "Interior point differential dynamic programming," *IEEE Transactions on Control Systems Technology*, vol. 29, pp. 2720–2727, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:216553894

[9] W. Jallet, A. Bambade, N. Mansard, and J. Carpentier, "Constrained differential dynamic programming: A primal-dual augmented lagrangian approach," *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 13 371–13 378, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:250478342

[10] B. T. Lopez, J.-J. E. Slotine, and J. P. How, "Dynamic tube mpc for nonlinear systems," *2019 American Control Conference (ACC)*, pp. 1655–1662, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:196622543

[11] L. Hewing, K. P. Wabersich, and M. N. Zeilinger, "Recursively feasible stochastic model predictive control using indirect feedback," 2019.

[12] C. M. Best, M. T. Gillespie, P. Hyatt, L. Rupert, V. Sherrod, and M. D. Killpack, "Model predictive control for pneumatically actuated soft robots," 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:35211429

[13] F. A. Spinelli and R. K. Katzschmann, "A unified and modular model predictive control framework for soft continuum manipulators under internal and external constraints," *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9393–9400, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:248476270

[14] J. I. Alora, M. Cenedese, E. Schmerling, G. Haller, and M. Pavone, "Data-driven spectral submanifold reduction for nonlinear optimal control of high-dimensional robots," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2627–2633, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:252367304

[15] S. Tonkens, J. Lorenzetti, and M. Pavone, "Soft robot optimal control via reduced order finite element models," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12 010–12 016, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:226246100

[16] C. Duriez, "Control of elastic soft robots based on real-time finite element method," *2013 IEEE International Conference on Robotics and Automation*, pp. 3982–3987, 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:1617784

[17] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, I. Peterlík, and S. Cotin, "Sofa: A multi-model framework for interactive physical simulation," 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:62022467

[18] E. Coevoet, T. Morales-Bieze, F. Largilliere, Z. Zhang, M. Thieffry, M. Sanz-Lopez, B. Carrez, D. Marchal, O. Goury, J. Dequidt, and C. Duriez, "Software toolkit for modeling, simulation, and control of soft robots," *Advanced Robotics*, vol. 31, pp. 1–17, 11 2017.

[19] E. Coevoet, "Optimization-based inverse model of soft robots, with contact handling," Theses, Université de Lille 1, Sciences et Technologies, Jan. 2019. [Online]. Available: https://hal.science/tel-02446416

[20] S. Gros and M. Diehl, "Numerical optimal control," 2019.

[21] D. Q. Mayne, "Differential dynamic programming–a unified approach to the optimization of dynamic systems," *Control and dynamic systems*, vol. 10, pp. 179–254, 1973. [Online]. Available: https://api.semanticscholar.org/CorpusID:124945755

[22] P. Chaillou, J. Shi, A. Kruszewski, I. Fournier, H. Wurdemann, and C. Duriez, "Reduced finite element modelling and closed-loop control of pneumatic-driven soft continuum robots," in *2023 IEEE International Conference on Soft Robotics (RoboSoft)*. Singapore, France: IEEE, Apr. 2023, pp. 1–8. [Online]. Available: https://hal.science/hal-04101252

[23] W. Jallet, A. Bambade, E. Arlaud, S. El-Kazdadi, N. Mansard, and J. Carpentier, "PROXDDP: Proximal Constrained Trajectory Optimization," Dec. 2023, working paper or preprint. [Online]. Available: https://inria.hal.science/hal-04332348