Control of Elastic Soft Robots based on Real-Time Finite Element Method

Christian Duriez

Control of Elastic Soft Robots based on Real-Time Finite Element Method

Christian Duriez
INRIA Shacra Team
University of Lille
Email: christian.duriez@inria.fr

Abstract—In this paper, we present a new method for the control of soft robots with elastic behavior, piloted by several actuators. The central contribution of this work is the use of the Finite Element Method (FEM), computed in real-time, in the control algorithm. The FEM based simulation computes the nonlinear deformations of the robots at interactive rates. The model is completed by Lagrange multipliers at the actuation zones and at the end-effector position. A reduced compliance matrix is built in order to deal with the necessary inversion of the model. Then, an iterative algorithm uses this compliance matrix to find the contribution of the actuators (force and/or position) that will deform the structure so that the terminal end of the robot follows a given position. Additional constraints, like rigid or deformable obstacles, or the internal characteristics of the actuators are integrated in the control algorithm. We illustrate our method using simulated examples of both serial and parallel structures and we validate it on a real 3D soft robot made of silicone.

Index Terms—Soft robots, Elastic robots, Finite Element Method (FEM), Real-time, constraints-based mechanics, Mechanical Compliance, Control algorithms.

I. INTRODUCTION

Traditional robots are based on articulated rigid structures and the design usually aims at finding the best trade off between a high stiffness and a large workspace of these rigid robots. High stiffness of the structure is often a goal so that vibration and/or deformations will not deviate the robot from its prescribed motion. Moreover, this high stiffness leads to a natural use of rigid bodies in the kinematic and mechanical models of the robot.

At the opposite, since about ten years, some work is centered on the use of soft robots that may a low stiffness on purpose. These approaches are often inspired from examples found in nature. Indeed, soft robots are particularly adapted to exploration and manipulation in a fragile environment. Their natural compliance allows to tolerate collision while reducing the risks for the robot and the environment. This is particularly suitable, for instance, for medical applications. Soft robots can be made of a unique piece and a variety of actuations could be used: pneumatic, hydraulic, cables, electroactive polymers, piezzo...

Contrary to rigid robots, the number of degrees of freedom (dof) of soft robots is infinite. This makes their control tricky: On the one hand, it seems natural to use a large number of actuators, to be able to impose a deformation everywhere. On the other hand, the actuators are coupled together by the deformation of the robot and controlling this redundant and coupled actuation is far from trivial. One important step towards the control soft robots is to be able to estimate, in any configuration, the mechanical coupling between the actuators and between the actuators and the end effector.

In this paper, we propose to use a real-time implementation of the Finite Element Method (FEM) in order to accurately model the deformations that are the cause of this mechanical coupling. Over other approaches for modeling deformations, FEM has the advantage to start from the constitutive law of the material which is measurable experimentally. From the FEM model, we extract the reduced mechanical compliance in the space of the actuators and of the end-effector thanks to the use of Lagrange multipliers. Then, a constraint-based approach, solved by an iterative Gauss-Seidel (GS) type algorithm, allows to find a contribution of the actuators which moves (while deforming) the end effector to the targeted position. Additional constraints (like the specification of the actuators and the contact response with the environment) is also included in the constraint solver. This work has been implemented and tested in SOFA [1], an open-source framework that contains fast implementations of FEM as well as constraint-based models[1].

To our best knowledge, this is the first paper that proposes a control method for deformable robots directly based on a real-time computation of the FEM. The first results of this work suggest that this type of approach may provide a part of the answer to the problem of control for soft robots.

The following of the paper begins by a quick overview of the related work. The FEM model of volume deformation is presented in section 3 (in the concrete case of robots made of a soft silicon) with the computation of the reduced compliance and the constraint-based models. Section 4 provides the details about the iterative algorithm which provides the control of the robot. Numerical results as well as concrete testing on a naive soft-robot are shown in section 5. A study of the computation time performance of the control method is also presented before the conclusion and the perspectives of this work.

II. RELATED WORK

An excellent state of the art [2] summaries the motivations and the novel capabilities of the soft robots. Several challenges

---

[1] www.sofa-framework.org

like the design and the realization are presented but the paper insists on the difficulties for modeling and controlling the robot. Indeed, the deformations of soft robots are difficult to anticipate and, in addition, the actuators are often integrated into the structure leading to coupled actuation. The actuator itself is sometimes complex, like when using pneumatic artificial muscle (PAM) [3]. Unlike traditional rigid robots, that have often only few degrees of freedom and can be controlled by solving a small system of kinematic equations, deformable robots have an infinite number of degrees of freedom. This makes their controllability particularly challenging.

Several advances have been made for controlling the deformations of *continuum* robots. These robots have the structure of a deformable rod that is continuously curving (see for instance [4]). Due to their relatively simple geometrical shape (deformable curve), fast models based on Cosserat rod theory [5] can be used. Moreover, the computation of the kinematic models and the compliance matrices can be obtained by propagating the deformations along the curvilinear structure of the robot [6]. However, this type of modeling is not adequate for volume deformations or actuation on deformable parallel architectures (i.e. similar to parallel robots). The control method is only adapted to curve-like robots (it limits the geometrical design), actuated in a serial manner. In addition, it supposes that the robot's structure is stiffer than its surrounding environment (i.e. the kinematic equations are not impacted by contact response).

Biomimetics is often a motivation for the work on soft robots. Many existing approaches, that uses the Finite Element Method, aims at an accurate simulation of biological muscular tissue dynamic motion. For instance, [7] studies the behavior of muscle during active and passive conditions and [8] reproduces the hyperelastic behavior of an octopus arm through simulation. This type of work tries to reproduce or predict the mechanical behavior of the living tissues. However, it does not seek to leverage the simulation for an interactive control of these tissue deformations, which would necessitate an inverse approach of their model.

For an interactive control of a soft robot, real-time computation of FEM is needed. Given the complexity of the models, obtaining this performance is challenging. Nonetheless, recent advances in interactive medical simulation yet obtain fast simulation of complex deformations like visco-hyperelastic models [9].

In any case, solving the FEM model only provides the motion created on a deformable object under known external loads or displacements (like a direct kinematic model). When computing the robot control, the forces or the displacements applied by the actuators are the unknown output of the command. This problem is similar when computing multi-point contact response on deformable objects: the contact force / displacement needed to fulfill the unilateral constraint is not known before computing the deformations [10]. The solving process makes use of Lagrange multipliers and finds a solution using a solver for complementarity problem. Recently, this approach was generalized to other type of mechanical

interaction (like insertion of flexible needles in soft-tissue) and was extended to haptic feedback rendering in [11]. Finally, the computation time can be greatly reduced for large problems thanks to algorithms adapted to GPU [12]. The new approach for piloting soft robots presented in this paper is strongly inspired by these works on real-time constraint-based modeling on deformable objects.

III. MODELS

In this section, FEM modeling is quickly presented, by using an example of soft silicone material. After measuring the constitutive law, a volume-based approach is used, based on tetrahedral elements. Then, we introduce the mathematical formulation of the constraints. This formulation brings up a compliance matrix of a reduced size, on which we will rely (in the following section) to solve inverse kinematics. Finally, we present a set of constraint models for end-effector, actuators and contact response.

A. FEM model of the soft-robot

The non-linear deformation of the volumetric soft robot is computed using FEM. The method integrates over the whole of the structure, a constitutive law of the deformable material. In this work, a soft silicone was used in order to get pure elastic deformation for quite large strain. Strain/stress measures were performed on 3 samples (see figure 1). A good repeatability
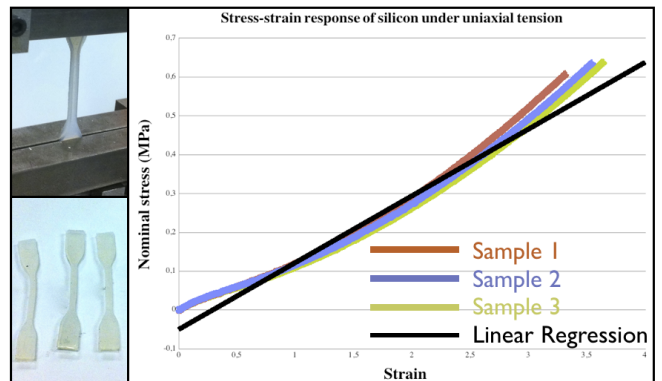

Fig. 1. Mechanical experiments to obtain the constitutive law of the Silicone. (Left) Photos of the uniaxial tension experiment (Right) stress/strain response of the silicon.

was observed and the measures show that the strain/stress ratio is not far from linear over quite large strains. Consequently, Hooke's law was used to characterize the behavior of this silicone with a Young Modulus of $150kPa$.

The corotational implementation of volume FEM, presented in [13], is particularly suitable for linear elasticity under the hypothesis of large transformations. The approach relies on a decomposition of each element's motion in a rigid rotation and a *pure* deformation. The strain/stress relation is evaluated in the frame of the element and the linear elasticity hypothesis allows some local pre-computations which accelerate the on-line execution. The computed rotations handle the geometrical non-linearities inherent to a volume deformation and after

assembling each element, the global stiffness model is no more linear. The approach is particularly fast to compute, numerically stable and an implementation in C++ is available freely in the open-source framework SOFA.

During each step i of the control, based on the simulation, the following linearization of the internal forces is computed:

$$\mathbf{f}(\mathbf{x_i}) \approx \mathbf{f}(\mathbf{x_{i-1}}) + \mathbf{K}(\mathbf{x_{i-1}})d\mathbf{x} \tag{1}$$

where $\mathbf{f}$ provides the volumetric internal stiffness forces at a given position $\mathbf{x}$ of the nodes, $\mathbf{K}(\mathbf{x})$ is the tangent stiffness matrix that depends on the actual position of the nodes and $d\mathbf{x}$ is the difference between positions $d\mathbf{x} = \mathbf{x_i} - \mathbf{x_{i-1}}$. The lines and columns that correspond to fix nodes are removed from the system to improve the condition number of the matrix $\mathbf{K}$. A dynamic model could also be used in our approach, using implicit integration but it would add the problem of real-time temporal integration (the time step used in the simulation must be strictly equal to the computation time). In a first approach, a quasi-static approach is chosen as the control of the robot is performed at low velocities.

One seeks to establish static equilibrium (sum of external and internal force equal to zero) at each step:

$$-\mathbf{K}(\mathbf{x_{i-1}})d\mathbf{x} = \mathbf{p} + \mathbf{f}(\mathbf{x_{i-1}}) + \mathbf{J}^T\boldsymbol{\lambda}. \tag{2}$$

where $\mathbf{p}$ represents the external forces (e.g. gravity) and $\mathbf{J}^T\boldsymbol{\lambda}$ gathers the contributions of the actuators and the contact forces (if applicable).

It should be noticed that the matrix $\mathbf{K}$ is highly sparse. A conjugate gradient solver is used and preconditioned by a sparse $\mathbf{LDL}^T$ decomposition. For a mesh composed of about 1000 nodes and about 3000 tetrahedral element, a refresh rate of $60Hz$ is obtained with the implementation available in SOFA. For larger meshes or faster updates, the $\mathbf{LDL}^T$ decomposition is unsynchronized and the resulting matrices be *warped* as it is explained in [14].

B. Reduced compliance on the constraint space

Complementary to the FEM formulation, a constraint-based approach accounts for the actuation and the contact with the environment. Each constraint adds a new direction that is gathered in the matrix $\mathbf{J}^T$ in a sparse manner[2]. $\boldsymbol{\lambda}$ is not known at the beginning of the process. This is a specificity of the constraint-based solver that solves the constraints in a separate process, while using specific laws for each of them.

Thus three steps are followed: (I), a free configuration $\mathbf{x}_{\text{free}}$ of the robot is found by solving the structure with $\boldsymbol{\lambda} = 0$. For each constraint, a violation is estimated $\boldsymbol{\delta}^{\text{free}}$ given the free configuration. (II) The constraint-based solver computes the value of $\boldsymbol{\lambda}$ given the laws of the constraint (between $\boldsymbol{\delta}$ and $\boldsymbol{\lambda}$) and the value of $\mathbf{W}$:

$$\boldsymbol{\delta} = \underbrace{\left[\mathbf{J}\mathbf{K}^{-1}\mathbf{J}^T\right]}_{\mathbf{W}}\boldsymbol{\lambda} + \boldsymbol{\delta}^{\text{free}} \tag{3}$$

---

[2]The direction is mapped to the FE nodes using barycentric coordinates if the contact point does not coincide with any FE node.

(III) The final configuration of the soft robot, at the end of the time step, is corrected by using the value of the constraint response:

$$\mathbf{x}_t = \mathbf{x}_{\text{free}} + \mathbf{K}^{-1}\mathbf{J}^T\boldsymbol{\lambda}.$$

It should be emphasized that one of the main difficulty is to compute $\mathbf{W}$ in a fast manner. No pre-computation is possible as the value of $\mathbf{W}$ changes at each iteration. The implementation of these 3 steps in SOFA is explained in [1] .

C. Terminal effector, actuator and contact models

As presented above, the method relies on the computation of the compliance along the constraint directions. If both actuator, effector and contact models rely on setting constraints, we will get a measure of the mechanical compliance between them based on the FEM model. However, a direction in matrix $\mathbf{J}$ and a law between $\boldsymbol{\delta}$ and $\boldsymbol{\lambda}$ must be defined for each constraint.

The constraint for the terminal effector is very simple: it consists on setting three constraint directions (along x, y and z) on a given point, mapped on the mesh. As there is no actuation on the terminal effector, the constraint law is very simple: $\boldsymbol{\lambda} = 0$.

Actuator model takes into account its physical characteristics. For instance, if the actuator is a cable, the direction $\mathbf{d}$ of the constraint is equal to the direction in which the cable pulls. In such case, the actuation is unilateral ($\lambda \geq 0$) and the actuator stroke can also be included by imposing ($\delta \in [\delta_{min}\ \delta_{max}]$). If the actuation is between two points, the direction and its opposite are respectively mapped on the mesh using matrix $\mathbf{J}$. It would be also possible to emulate a pneumatic actuation by selecting the triangles of the deformable cavity that should inflate with pressure. If the pressure is supposed to be uniform in the cavity, each triangle contributes to a single line of $\mathbf{J}$ by weighting its direction by its area and by adding the results on the column of the concerned nodes.

Finally, the contact response follows the Signorini's law $0 \leq \boldsymbol{\delta} \perp \boldsymbol{\lambda} \geq 0$, where $\boldsymbol{\delta}$ is the vector that gathers the distances at each pair of contacting points (see [10] for integrating Signorini's law with FEM). Directions of contact are given by a proximity query algorithm between meshes.

IV. Control algorithm based on reduced FEM compliance

In this section, we develop the successive steps of the control algorithm based on the models described below. First, it should be noted that a kind of direct geometric model can be obtained by solving the FEM model while imposing the actuator positions p_a in the constraint law $\delta = p_a$. However, the control algorithm aims at computing the inverse. (i.e. moving the end-effector at the right place by assigning a command on the actuators). The beginning of this section shows that this goal is somehow reached with the compliance matrix condensed in the constraint space $\mathbf{W}$ presented above. Then, by building an iterative resolution algorithm upon this matrix, a solution of the inverse kinematic relation between the terminal effector position to the actuators can be found. This leads to the position control algorithm of the robot and

then, the integration of contact constraints. Some numerical aspect as well as the implementation are finally discussed.

A. Coupled Kinematic Equations

Using the compliance operator $\mathbf{W}$, we can get a measure of the mechanical coupling between effector and actuator, and also between actuators.

For instance, we can get the displacement $\Delta\delta_i$ created on the end-effector (on a direction stored on the line i of matrix $\mathbf{J}$) by a unitary force λ_j applied by the actuator which is stored at the line j of matrix $\mathbf{J}$. We get: $\delta_i = w_{ij}\lambda_j$. We can also link λ_j to the position of the actuator using the local compliance $\delta_j = w_{jj}\lambda_j$, in order to get a simple kinematic link between $\Delta\delta_i$ and δ_j: $\Delta\delta_i = \frac{w_{ij}}{w_{jj}}\delta_j$

However, an other actuator k may also influence the displacement of the actuator j. (and vice-versa). This mechanical coupling can also be measured using the reduced compliance operator $\Delta\delta_j = w_{jk}\lambda_k = \frac{w_{jk}}{w_{kk}}\delta_k$ (see figure 2).
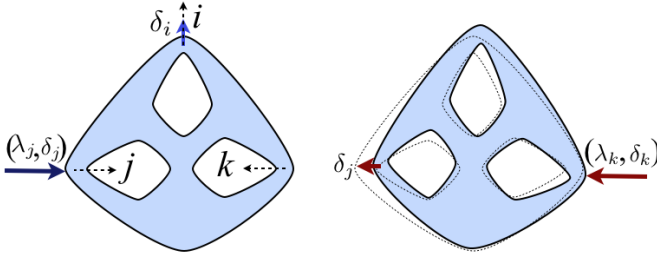

$$\delta_i \uparrow i$$

$$(\lambda_j, \delta_j) \cdots \leftrightarrow j \quad k \leftrightarrow \cdots \quad \delta_j \leftarrow \qquad (\lambda_k, \delta_k)$$

Fig. 2. Actuation (λ_j, δ_j) along direction j can be controlled to impose a displacement δ_i on the end effector. However, an other actuator (λ_k, δ_k) along direction k may create an unwanted displacement δ_j (in red) of the actuator j. This illustrates the coupling between kinematic equations.

This coupling between actuators prevents from obtaining kinematic equations through an *usual* jacobian matrix. Moreover, soft robots are often highly redundant, which would be problematic, anyway, for jacobian inversion.

B. Position control

Instead of using (inverted) kinematic equations, we build the position control of the soft robot on the compliance $\mathbf{W}$ operator. Let's consider that $\Delta\delta_i$ provides the shift between the position of the end effector and a desired position along direction i: $\Delta\delta_i = \delta_i - \delta_i^{\text{desired}}$. The position control algorithm founds a position on the actuators that deforms the structure so that $\Delta\delta_i = 0$.

In this work, a Gauss-Seidel (GS) iterative solver is used to find a solution among the possible solution (if a solution exists). During each iteration it, the algorithm updates one by one the contribution λ_j of each actuator j, while *freezing* the contribution λ_k of the other actuators.

$$\delta_j - w_{jj}\lambda_j^{(it)} = \sum_{k=0}^{j-1} w_{jk}\lambda_k^{(it)} + \sum_{k=j+1}^{N} w_{jk}\lambda_k^{(it-1)} + \delta_j^{free} \quad (4)$$

As we use a GS algorithm, when doing a local update on actuator j, the *frozen* contribution of the actuators $[0 \to j-1]$ comes from the current iteration it whereas the contribution

of actuators $[j+1 \to N]$ (where N is the total number of actuators), comes from the previous iteration $it-1$. The local update of actuator j provides a new contribution $\lambda_j^{(it)}$ and the effector position δ_i, along direction i, is easily updated:

$$\delta_i \mathrel{+}= w_{ij}(\lambda_j^{(it)} - \lambda_j^{(it-1)}) = w_{ij}\Delta\lambda_j \quad (5)$$

Thus, during the update of j, we can measure how the actuator can (or not) reduce the value of $\Delta\delta_i$. When performing this update in 3 dimensions, direction i is alternatively replaced by x, y and z. Imposing a variation of λ_j will create a 3D motion of the end effector:

$$\underbrace{\begin{bmatrix} \delta_x \\ \delta_y \\ \delta_z \end{bmatrix}}_{\delta_*} \mathrel{+}= \underbrace{\begin{bmatrix} w_{xj} \\ w_{yj} \\ w_{zj} \end{bmatrix}}_{\mathbf{w}_{*j}} \Delta\lambda_j \quad (6)$$

The control should try to reduce the shift between actual and desired positions, measured by $\Delta\delta_*$. But the actuator j can only move the effector along the direction given by $\frac{\mathbf{w}_{*j}}{\|\mathbf{w}_{*j}\|}$. Consequently, we search for a value of $\Delta\lambda_j$ so that:

$$(\mathbf{w}_{*j}\Delta\lambda_j) \cdot (\frac{\mathbf{w}_{*j}}{\|\mathbf{w}_{*j}\|}) = -\Delta\delta_* \cdot (\frac{\mathbf{w}_{*j}}{\|\mathbf{w}_{*j}\|}) \quad (7)$$

This value can be obtained by using:

$$\Delta\lambda_j = -\left(w_{xj}\Delta\delta_x + w_{yj}\Delta\delta_y + w_{zj}\Delta\delta_z\right)/\|\mathbf{w}_{*j}\|^2 \quad (8)$$

Given the new value of $\Delta\lambda_j$, one can update $\lambda_j^{(it)}$ and the position δ_j of the actuator j using the GS equation (4).

C. Control in a constrained deformable environment

The internal constraints of the soft robot (capacity and stroke of the actuators, self-collision), and the external contact constraints imposed by the nearby objects are added in the GS algorithm presented above.

If the actuation is unilateral, $\lambda_j^{(it)} \geq 0$. If $\lambda_j^{(it)} < 0$ after solving equation (8) during the local update of actuator j, we simply impose $\lambda_j^{(it)} = 0$ before leaving the local update and processing the following actuator $j+1$. In the same way, we can impose a maximal actuator force $\lambda_j^{(it)} \leq f_{max}$.

For imposing that the computed displacement of the actuators stays inside the stroke, a displacement correction δ_j^c can be applied locally. In such case, we modify the value of λ_j, so that $\lambda_j \mathrel{+}= \delta_j^c/w_{jj}$.

Auto-collision and contact constraint with the environment can be quite easily integrated in this control method. Indeed, the GS algorithm is inspired by contact constraint resolution between deformable objects [10]. For each contact (internal or external), an additional unilateral constraint is added in the system. Both actuator constraints and unilateral contact constraints are solved in the same process. At each iteration, the GS algorithm first updates the contribution of the actuators (as described above) and then, it updates the contact constraints. The local update is performed by imposing the Signorini's law: On a contact j, we first compute $\delta_j^{(it)}$ using equation (4). Then we compute, $\Delta\lambda_j = -\delta_j^{(it)}/w_{jj}$ and iff $\lambda_j^{(it)} < 0$, we impose a unilateral contribution: $\lambda_j^{(it)} = 0$.

D. Discussion

There is no assumption about the way the actuators are placed on the deformable structure. Consequently, the method presented above is valid for both serial and parallel actuations.

Numerical experiments seem to show that if a solution exists, the GS algorithm is converging to this solution. A formal proof of convergence is not yet established in this work as it can be observed that when redundant actuation is used, the solution is not unique. In such case, the presented method tends to find a solution which involves more importantly the actuators that are numbered first in the GS algorithm. Sometimes, the algorithm oscillates between two valid solutions. To remove this problem, we use a successive over-relaxation technique: from one iteration to an other, the value $\Delta\lambda_j$ is weighted by ω ($0 < \omega < 1$) in order to impose a limited variation of δ_j and to facilitate the convergence.

The contact constraints are set using a collision detection based on proximity queries. Contact response can be obtained with both rigid and deformable nearby objects. If deformable, the object must be modeled and simulated with FEM and a new compliance is combined with the compliance of the robot in equation (3).

V. RESULTS

The algorithms were implemented in C++ in the SOFA framework. We have re-used the implementation of the FEM model and the Lagrange multipliers system. Our control algorithm was implemented as a new way of solving the constraints (*ConstraintSolver* in SOFA) on a deformable model.

In the following, two examples are presented to illustrate the capability of the control method and especially the compatibility with both parallel and serial structures. The first example is a numerical experiment on a deformable beam with redundant serial actuation and contact constraints. The second example is the interactive control of a 3D soft robot made of silicone with an analysis of the precision. We also provide the computation performance of the method which is the key of our approach.

A. 3D "staircase" Beam in a constrained environment

For this first example, the design of the structure is a bit *naive* but is chosen to, numerically, evaluate the method with serial redundant actuation and with contact constraints. A beam of square cross-section is deformed by 8 actuators distributed alternately on each side of the deformable robot (see figure 3(a)). A solution is found at each step so that the end effector follows the trajectory (orange sphere in figure 3(b)). Then a deformable rod is placed on the trajectory of the robot and the contact response is included in the resolution process. The computation of the actuation is adapted to the new situation, while still finding a solution that follows the provided trajectory.(see figure 3(c)). Other experiments were performed with a collision on a stiffer obstacle. It was verified that if the contact prevents the robot to reach the target, the algorithm remains stable.
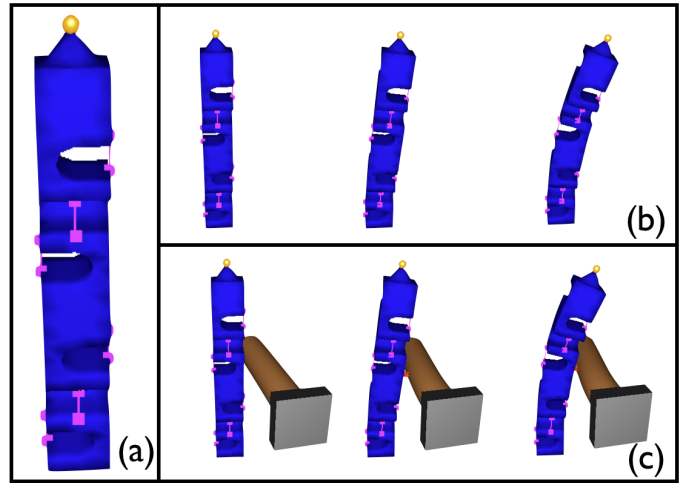

Fig. 3. (a) A beam is deformed by 8 bilateral actuators (like pistons) placed on recessed parts of the structure (pink lines). The end effector is placed on the top of the mesh and follows the trajectory (orange sphere). (b) In the free environment, the algorithm finds a first solution. (c) The robot is contacting a deformable rod. The control is adapted to find an other solution that is coherent with contact response and the stiffness of the obstacle.

B. Validation on a 3D silicone made robot, actuated with cables

A second example has been chosen to validate the approach presented in the paper. A real 3D deformable robot, made of soft silicone, which design is inspired by parallel robots (see figure 4), is controlled by simulation approach presented in the paper. The robot naturally deforms and sinks under the action of gravity, but 4 unilateral actuators (servo-motors that are connected to the structure of the robot with cables) are placed on each leg to prevent and pilot the deformation.

In the simulation, the effector position is placed on the upper part of the robot and its trajectory is imposed at each step using a new desired position. The trajectory can be recorded and replay but the user can also interactively impose it as the high refresh rate of the simulation makes possible the real-time control of the servo-motors.

As illustrated in figure 4 we measure the position of the end effector, in the real world, using a motion capture system[3]. We measure the discrepancy between the desired positions and the obtained positions on static positions distributed in a workspace of $40mm \times 40mm \times 20mm$ around the rest-position of the robot. On a sample of 36 positions, we obtain a mean error of 1.4 mm with a standard deviation is 0.6 mm and the maximum error is 2.9 mm. We emphasize that these results are obtained using an open-loop: the control of the robot only relies on the model based on FEM that is computed in the interactive simulation. This experiment shows that the FEM approach presented in this paper can be quite accurate. Even better results would certainly be obtained using the information of the motion capture for correcting the robot

[3]We use the *OptiTrack*® system, commercialized by NatrualPoint company. The system has a precision of $0.1mm$ for measuring the 3D displacements of a target point.

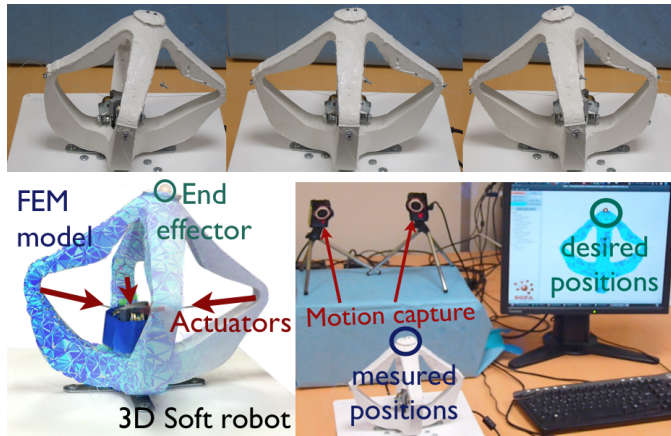position in real-time, in the simulation (in a closed-loop).

Fig. 4. Top: Pictures of the 3D soft robot. Bottom: (left) Registered images of the FEM model and the 3D Soft robot and (right) experimental validation of the approach. The real position of the end effector is measured using motion capture and compared to the desired position.

C. Performance

One critical aspect of the method is the computation time of the FE models in the simulation and the computation of the matrix $\mathbf{W}$ that requires an inversion of a quite large but sparse matrix system. To obtain good performances, we rely on the LDL^T factorization of the matrix computed using CSPARSE Library[4] and on the method presented in [14]. The table 5 provides an indication of the computation time on the different simulations presented in this paper (a more detailed analysis of the performance is provided in [14]).

Model / Perf	Size of the mesh	Size of matrix W	Comp. time
Silicone-made soft robot	1628 Nodes 4147 Tetra	7	30 ms
3D Beam (free)	935 Nodes 2853 Tetra	11	23 ms
3D Beam + Obstacle	935 + 174 Nodes 2853 + 400 Tetra	23	33ms

Fig. 5. Computation time observed for the simulations presented in the paper

VI. Conclusion and Future work

This paper proposes a new approach for the control of a soft robot. Thanks to a real-time FEM simulation of the robot and the use of lagrange multipliers, we compute a reduced compliance matrix between end-effector and actuators. A dedicated constraint-based solver algorithm uses this matrix to find a solution to the command. The method is applied to several experiments (numerical and real).

It is a first step in this direction, many future work possibilities are open. First, we could analyze the stability of the

control and try to close the loop using the motion capture. We also need to assess the quality of the solution found by the Gauss-Seidel algorithm when several solutions are possible. Then, the approach can be extended to force control and other type of actuation. Finally some questions remains open like the controllability of the soft objects in contact with the robot.

References

[1] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, I. Peterlik, and S. Cotin, "Sofa: A multi-model framework for interactive physical simulation," in *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*, ser. Studies in Mechanobiology, Tissue Engineering and Biomaterials, Y. Payan, Ed. Springer Berlin Heidelberg, 2012, vol. 11, pp. 283–321.

[2] D. Trivedi, C. D. Rahn, W. M. Kier, and I. D. Walker, "Soft robotics: Biological inspiration, state of the art, and future research," *Applied Bionics and Biomechanics*, vol. 5, no. 3, pp. 99–117, 2008. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/11762320802557865

[3] F. Daerden and D. Lefeber, "Pneumatic artificial muscles: actuators for robotics and automation," *European journal of Mechanical and Environmental Engineering*, vol. 47, pp. 10–21, 2000.

[4] L. G. Torres and R. Alterovitz, "Motion planning for concentric tube robots using mechanics-based models," in *International Conference on Intelligent Robots and Systems (IROS),*, 2011.

[5] D. Pai, "Strands: Interactive simulation of thin solids using the cosserat models," in *Eurographics*, G. Drettakis and H. P. Seidel, Eds., vol. 21, no. 3. Blackwell Publishers, 2002.

[6] D. Ducker and R. Webster, "Computing jacobians and compliance matrices for externally loaded continuum robots," in *International Conference on Robotics and Automation (ICRA)*, 2011.

[7] C. Tang, G. Zhang, and C. Tsui, "A 3d skeletal muscle model coupled with active contraction of muscle fibres and hyperelastic behaviour," *Journal of biomechanics*, vol. 42, pp. 865–872, 2009.

[8] V. Vavourakis, A. Kazakidi, D. Tsakiris, and J. Ekaterinaris, "A finite element method for non-linear hyperelasticity applied for the simulation of octopus arm motions," in *IV Int. Conf. Comp. Meth. Coupl. Probl. Sci. Eng*, 2011.

[9] S. Marchesseau, T. Heimann, S. Chatelin, R. Willinger, and H. Delingette, "Multiplicative jacobian energy decomposition method for fast porous visco-hyperelastic soft tissue model," *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2010*, pp. 235–242, 2010.

[10] C. Duriez, C. Andriot, and A. Kheddar, "Signorini's contact model for deformable objects in haptic simulations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004, pp. 3232–3237.

[11] I. Peterlik, M. Nouicer, C. Duriez, S. Cotin, and A. Kheddar, "Constraint-based haptic rendering of multirate compliant mechanisms," *EEE Trans. Haptics*, vol. 4, no. 3, pp. 175–187, Jul. 2011. [Online]. Available: http://dx.doi.org/10.1109/TOH.2011.41

[12] H. Courtecuisse, H. Jung, J. Allard, C. Duriez, D. Y. Lee, and S. Cotin, "Gpu-based real-time soft tissue deformation with cutting and haptic feedback," *Progress in Biophysics and Molecular Biology*, 2011, special Issue on Soft Tissue Modelling. [Online]. Available: http://codrt.fr/allardj/pub/2011/CJADLC11

[13] C. Felippa, "A systematic approach to the element-independent corotational dynamics of finite elements," in *Technical Report, Center for Aerospace Structures*, 2000.

[14] H. Courtecuisse, J. Allard, C. Duriez, and S. Cotin, "Preconditioner-based contact response and application to cataract surgery," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, sept 2011. [Online]. Available: http://www.lifl.fr/~courtecu/

[4] http://www.cise.ufl.edu/research/sparse/CSparse/