Soft robots locomotion and manipulation control using FEM simulation and quadratic programming

Eulalie Coevoet, Adrien Escande, Christian Duriez

Soft robots locomotion and manipulation control using FEM simulation and quadratic programming.

Eulalie Coevoet[1], Adrien Escande[2] and Christian Duriez[1]

***Abstract*— In this paper, we propose a method to control the motion of soft robots able to manipulate objects or roll from one place to another. We use the Finite Element Method (FEM) to simulate the deformations of the soft robot, its actuators, and surroundings when deformable. To find the inverse model of the robot interacting with obstacles, and with constraints on its actuators, we write the problem as a quadratic program with complementarity constraints. The novelty of this work is that friction contacts (sticking contact only) are taken into account in the optimization process, allowing the control of these specific tasks that are locomotion and manipulation. We propose a formulation that simplifies the optimization problem, together with a dedicated solver. The algorithm has real-time performance and handles evolving environments as long as we know them. To show the effectiveness of the method, we present several numerical examples, and a demonstration on a real robot.**

I. INTRODUCTION

Soft robots are mainly used for their compliance in interaction and because they are safer for interaction with humans and fragile objects. Indeed, their soft structure allows them to bend and squeeze to fit obstacles, and reduce the stress induced by contact over both surroundings and the robot's surface. They are particularly appropriate for locomotion in uneven and/or sensitive environment, and manipulation tasks of delicate items. As mentioned in [1]:

> "The promise of soft robots is perhaps best realized in environments and applications that require interaction with soft materials and organisms..."

Soft robots have been intensely investigated and developed for grasping applications. Indeed, with traditional hard robots, these tasks have always been complex to perform. It requires for example to have force sensor mounted on the robot to avoid damaging the grasped object. Also, for other applications than grasping, contacts are generally avoided with hard robots. With soft robots, contacts are not so much a concern anymore, and we can even seek to use contacts with obstacles to better reach a target.

The particularity of soft robots is that they change their initial shape to create the motion. They have a theoretical infinite number of degrees of freedom (DoFs) while having a finite number of actuators. Consequently, these robots are inherently under actuated. Also, the material properties have a direct influence on the motion of the robots, so that interactions play a bigger role than for rigid robots. Their modeling and control are thus more complex than for their traditional rigid counterparts, as mentioned in [2], and [3]. Handling interaction with obstacles is even more challenging.

The method we propose opens the control of some soft robots locomotion and manipulation, with the assumption that each contact is either sticking or inactive. In particular, we propose a solution with real-time performance, which is often required by robotic applications.

II. RELATED WORK

Piecewise-Constant Curvature (PCC) methods have been used to model continuum robots with rod-like shape [4]. The method is a geometrical approach that simplifies the problem, but the hypothesis requires a specific design of the robot. In [5] Wang *et al.* propose a visual servo controller based on PCC kinematics to control an eye-in-hand system in constrained environments. Other controllers are based on neural network learning approaches [6]. While these approaches are interesting to get real-time controller, they do not account for interaction with the environment. A model-less approach is proposed by Yip and Camarillo [7] to control continuum robots interacting into an unknown environment. The method relies on empirical estimates of the robot's Jacobian computed in real-time using measurements of actuators and the position of the end-effector. While the method offers a closed-loop task-space control, it is for now designed for static environment only.

In [8] and [9], Petit *et al.* and Zhang *et al.* propose to use FEM and camera for contact force sensing acting on soft robots. While Petit *et al.* assume the contact point between the objects are known, Zhang *et al.* are also able to locate these points. In [10], we propose to use FEM to control a large variety of soft robots in real-time. We recently proposed an extended method to handle contact, again in real-time [11]. This method assume that the environment is known: 1) the environment has to be modeled in the simulation, and 2) evolving environment must be tracked and the simulation must be consequently updated. This method is limited by the fact it does not handle friction contact and then does not allow the control of soft robot locomotion and manipulation. Inverse kinematics of soft structures with contact handling using FEM simulation has also been studied in the field of computer graphics [12], [13], [14]. The idea is to enhance animations by making them physically more realistic. In [12] and [13] the authors propose to control the motion of virtual skeleton-driven deformable characters in real-time. In [14], Tan *et al.* use FEM and muscle fibers actuation to simulate soft bodies character's locomotion.

[1]INRIA and University of Lille, France

[2]CNRS-AIST Joint Robotic Laboratory (JRL) UMI3218/RL, Tsukuba, Japan

*Adrien Escande is supported by the CNRS-AIST-Airbus Joint Research Program.

They also use a complementarity formulation to model contacts, as it is often used in rigid robotics for locomotion and manipulation, where robots are frequently making and breaking contact in an unpredictable manner [15]. In [15], Posa *et al.* formulate the problem as a mathematical program with complementarity constraints, and use a sequential QP to solve the problem, leaving real-time performance as future work.

In the present work, we propose to handle friction in the problem, but with the assumption of sticking contacts only. While making the problem more simple, this assumption allows us to control some soft robots locomotion and manipulation tasks in real time.

The remainder of the paper is organized as follows. In section III we describe the main steps of the method, including the formulation of the inverse problem and the algorithm to solve it in real-time. In section IV we present and discuss several numerical examples and a real robot on which we applied our controller. We finally discuss the method and conclude in section V.

III. METHOD

We want to find how to actuate a soft robot to obtain a desired motion. In particular, we are interested in the control of soft robots that use friction contact to perform a task (motion on the ground or object manipulation for instance).

A. Dynamics Equations

To simulate the deformation of the volume structure, we use the FEM implementation provided by the simulation framework SOFA [16]. The configuration of the robot at a given time is obtained by solving the equations of dynamics given by Newton's second law:

$$M\dot{v} = f(x,v) + f_{ext} + H_a^T\lambda_a + H_c^T\lambda_c \tag{1}$$

where x and v are respectively the vector of the FEM nodes position and velocity, M is the mass matrix, $f(x,v)$ represents internal forces of the deformable structure, f_{ext} gather the known external forces (such as gravity), and $H_a^T\lambda_a$ and $H_c^T\lambda_c$ respectively are the contributions of actuators and contacts. The matrices H^T are filled with the direction of the effort applied by the constraints (actuators and contacts) on the FEM nodes and λ are the Lagrange multipliers corresponding to the intensity of this effort.

The internal forces $f(x,v)$ are a nonlinear function. At each time step of the simulation, we compute a *linearization* of $f(x,v)$ by applying a Taylor series expansion, leading to the following first order approximation:

$$f(x+dx, v+dv) = f(x,v) + \frac{\partial f}{\partial x}dx + \frac{\partial f}{\partial v}dv \tag{2}$$

We use an implicit scheme (backward Euler) to integrate the equation of the dynamics over time. Given a time step h and a current state (x_t, v_t) at a time t, equation (1) becomes the following linear system:

$$\begin{aligned} \underbrace{\left(M + h\frac{\partial f}{\partial x}dx + h^2\frac{\partial f}{\partial v}dv\right)}_{A} dv &= -hf(x_t,v_t) - h^2\tfrac{\partial f}{\partial x}v_t \\ &+ hf_{ext} \\ &+ hH_a^T\lambda_a + hH_c^T\lambda_c \end{aligned} \tag{3}$$

The solution dv of this equation is then used to update the Euler scheme: $v_{t+h} = v_t + dv$ and $x_{t+h} = x_t + hv_{t+h}$. It depends only on the values of λ_a and λ_c, which will be the variables of our control problem.

The computation of the internal forces depends on the deformable model we use. In this paper we use a co-rotational model, which allows for large displacements and geometrical non-linearities. This model offers good compromise between computational efficiency and accuracy. However, SOFA also provides implementation of different hyper-elastic models based on non-linear models, such as St-Venant-Kirchhoff or NeoHookean. The difficulty with these models is that they require more parameters that are difficult to acquire (can be measured only with specific equipment). Note that the optimization method we propose to control our robots does not depend on the model we choose.

B. Friction Contacts

Coulomb's law is usually used to model the effects of sticking and sliding contacts [12], [13], [14]. The contact force is thus given by the composition of a normal force $H_n^T\lambda_n$ and a tangential (friction) force $H_t^T\lambda_t$, given $H_c^T\lambda_c = H_n^T\lambda_n + H_t^T\lambda_t$. The Coulomb's model imposes the contact force to lie in a circular cone, called friction cone. If the force is (strictly) inside the cone, the contact is sticking. If the force is on the cone boundary, the contact is sliding.

The circular cone leads to a non-linear complementarity problem which is difficult to solve numerically. Therefore the usual approach in treating this problem has been to formulate a Linear Complementarity Problem (LCP) with a polygonal approximation of the friction cone [17]. However, the global problem of solving both the actuation and friction contact is more complex, and solving this problem in real time is very challenging. To simplify the resolution, we make the assumption of sticking contact only, which can be sufficient for the control of some soft robot locomotion or for manipulation tasks. More detail is given in the following sections.

C. Inverse Problem Formulation

The system of equations of motion (3) can be of high dimension, as its size is directly related to the number of nodes of the FEM mesh. To allow real time performances, we use the Schur complement to solve the optimization in the space of the constraints variables (actuators and contacts) instead of the motion space, leading to (see [10] for more details):

$$\begin{bmatrix}\delta_e\\ \delta_a\\ \delta_c\end{bmatrix} = \begin{bmatrix}W_{ea} & W_{ec}\\ W_{aa} & W_{ac}\\ W_{ca} & W_{cc}\end{bmatrix}\begin{bmatrix}\lambda_a\\ \lambda_c\end{bmatrix} + \begin{bmatrix}\delta_e^{\text{free}}\\ \delta_a^{\text{free}}\\ \delta_c^{\text{free}}\end{bmatrix} \tag{4}$$

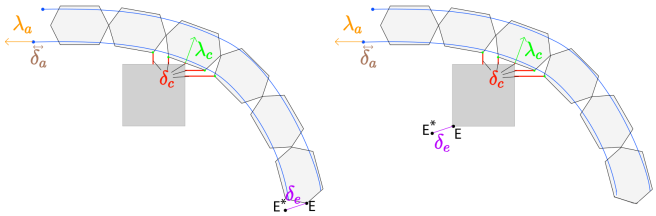

Fig. 1: Illustration of the quantities of equation (4) for a soft robot with cable actuation (in blue), colliding with an obstacle (grey square). E^* is the desired position for the controlled point E. Left: Usual case where we control the position of the end-effector. Right: We can also control the position of the object with which the robot interacts.

where matrices $W_{ij} = H_i A^{-1} H_j^T$ $(i, j = e, a, c)$ gather the mechanical coupling between effector points e, actuators a and contacts c, λ_a and λ_c are respectively the actuator and contact efforts, δ_e gives the shift between the controlled points and their desired positions, δ_a is the displacement of actuator (cable displacement for instance), and δ_c is the gap between two colliding points. See Fig. 1 for an illustration of these quantities. The values δ_e^{free}, δ_a^{free} and δ_c^{free} respectively correspond to the shift δ_e, the displacement δ_a, and the gap δ_c computed during a free motion, that is when solving the structure without any constraints, i.e $\lambda_a = 0$ and $\lambda_c = 0$ (see [10]). Note that the matrices H_i^T correspond to the Jacobian $\frac{\delta \delta_i}{\delta x}$ $(i, j = e, a, c)$. For an effector, $\frac{\delta \delta_e}{\delta x}$ is the identity matrix $I \in \mathbf{R}_{3,3}$. Then the matrix H_e^T is filled with identity matrices at the indices of the controlled points. Let us note n_a and n_c the number of actuator and contact forces, with the contact points being provided by one of SOFA's contact detection methods. We formulate the constraints on contact as follow. Given a contact c:

$$\begin{cases} \text{either:} & \lambda_n = \lambda_t = 0 \text{ and } \delta_n \geq 0 \quad \text{(inactive)} \\ \text{or:} & \lambda_n \geq 0 \text{ and } \delta_n = \delta_t = 0 \quad \text{(sticking)} \end{cases} \tag{5}$$

where δ_n and δ_t are the normal and tangential distances. To control the robot, we want to find the actuation λ_a so that effectors reach their desired positions. This corresponds to minimizing the norm $\|\delta_e\|$, while respecting the constraints (5), leading to the following Quadratic Program with (linear) Complementarity Constraints (QPCC):

$$\begin{aligned} &\min_{\lambda_c, \lambda_a} \ \left\| W_{ec}\lambda_c + W_{ea}\lambda_a + \delta_e^{\text{free}} \right\|^2 \\ &\text{s.t. } (5) \end{aligned} \tag{6}$$

Note that we can add linear constraints on actuators such as limits on the cables displacements $\delta_{min} \leq \delta_a \leq \delta_{max}$, or directly limits on forces $\lambda_{min} \leq \lambda_a \leq \lambda_{max}$. For instance, in the case of cable actuator we set $\lambda_{min} = 0$, allowing the cable to pull only. When no potential collision has been detected ($n_c = 0$), the problem is a simple Quadratic Program (QP). Note that λ_c is part of the optimization variables, which allows the controller to make use of contact forces to achieve the desired motion.

D. Solver

In [11], we propose a specific solver to handle the complementarity constraints introduced by Signorini's law for frictionless contact. The algorithm is based on decomposition method. The same approach can be used here to solve the new QPCC (6). The main idea of the method is to use the disjunctive structure of the complementarity constraints to formulate series of QP. At each iteration of the QPCC solver, we specified two subsets I_1 and I_2 with $I_1 \cup I_2 = \{1, \ldots, n_c\}$, respectively distinguishing the inactive and active (sticking) contacts, yielding:

$$\begin{aligned} &\min_{\lambda_a, \lambda_c} \ \left\| W_{ea}\lambda_a + W_{ec}\lambda_c + \delta_e^{free} \right\|^2 \\ &\text{s.t. } \delta_a^{min} \leq \delta_a = W_{aa}\lambda_a + W_{ac}\lambda_c + \delta_a^{free} \leq \delta_a^{max} \\ &\qquad (\lambda_n)_i = (\lambda_t)_i = 0 \text{ and } (\delta_n)_i \geq 0, \text{for } i \in I_1 \\ &\qquad (\lambda_n)_i \geq 0 \text{ and } (\delta_n)_i = (\delta_t)_i = 0, \text{for } i \in I_2 \end{aligned}$$

Each QP is a piece of the global problem and we solve it using the qpOASES library [18]. The goal is to find the two subsets I_1 and I_2 that give the best solution. To this end, we look at the constraints that have reach their boundary at the end of the optimization, as pivoting these constraints could lead to a better solution. We take each of these constraints as candidate for pivot. In our algorithm only one constraint is pivoted at a time. To choose the best candidate for pivot we look at their dual variable, and select the candidate with the greater one.

For the decomposition method to converge, the iteration has to start from initial feasible subsets I_1 and I_2. To find these feasible subsets, we consider the actuation fixed. As in our previous work, for this stage, either the actuation computed during the previous time step of the simulation is available (warm start) or we set the actuation to be equal to zero. With the actuation fixed, we can express the contact problem with the usual LCP formulation. We use a Gauss Seidel algorithm to solve it. In this resolution, we allow contacts to lie outside the friction cone to obtain strictly sticking contacts. Note that it can happen that the problem is not convex, in such case the solver does not aim at finding the global optimum. When there is no more candidate for pivot the iterations stop, and the solution, which can be a local minimum, is the one given by the last QP resolution. A scheme of the solver is given in Fig. 2.

E. Well posed problem

As already presented in [11], the matrix of the QPCC is symmetric, and positive definite if the number of controlled points (in terms of DoFs) is greater or equal to the number of actuation. In that case, each QP with its subsets I_1 and I_2 has a unique solution. For some robots, the number of actuators may be greater than the number of controlled points. In such case, the matrix of the QP is no longer positive definite, but positive semi-definite. The problem has multiple solutions, and the solver may swing from one solution to another. To avoid that, we add to the objective an expression of the actuators mechanical work E. The objective becomes

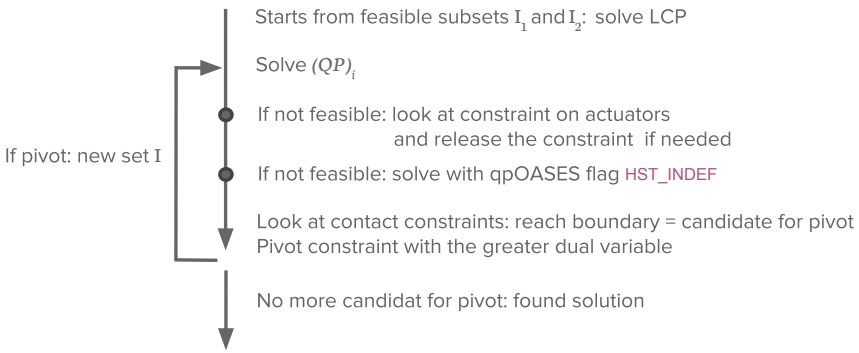

Fig. 2: Strategy scheme to solve the inverse problem with friction contact handling.

$(||\delta_e||^2+\epsilon E)$, and the QP matrix is regularized (with ϵ chosen sufficiently small to keep a good accuracy on the effectors).

For manipulation tasks, we want to control the motion of the grasped object, instead of the gripper itself (see section IV-B for examples). Yet, if the squeezing force (the real gripper applies on the object) is not sufficient, the assumption of sticking contacts we make in the simulation, may not be satisfied in the real world; sliding contacts may appear between the real gripper and the object. Also, in some configurations, dropping the object may lead to a better solution at a given time step; usually when the target is far from the grasped object we want to control. Thus, to prevent sliding contacts on the real robot, and also to prevent the gripper from releasing the object when possible, we need to constrain the gripper tightening. To this end, we insert into the optimization problem an additional constraint on the contacts force. The simplest formulation is to constrain the sum of the contacts force (along the normal direction) to be greater than a given value, yielding:

$$\mathbb{I}^T \lambda_c \geq \lambda_{c,min}$$

with $\mathbb{I}$ a column vector of zeros and ones, with $\mathbb{I}_i = 1$ if i in the set of the contacts normal indices. To estimate a good value for $\lambda_{c,min}$, we simply run a simulation where the object is being hold against the gravity and store a corresponding range for the product $\mathbb{I}^T \lambda_c$. The configuration is easily obtained by having the effectors target located in the robot's working space.

We noticed, from experiments with locomotion, that some constraints on actuators may become infeasible, due to a motion induced by contacts and the dynamic of the robot. For example, it is assumed, in the simulation, that cables are inextensible and straight. Yet, if the robot enters a contact with certain velocity (see Figure 3), it may happen that the motor mounted on the real robot does not pull the cable fast enough (to ensure that the cable remains straight), or that the course of the cable is already at its maximum. The corresponding constraints in the optimization problem (i.e. limits on cable maximum/minimum displacements and/or cable displacements variation) may no longer be satisfied. In the implementation, when such cases are detected, we notify the user and we temporarily remove the corresponding constraints from the optimization problem.

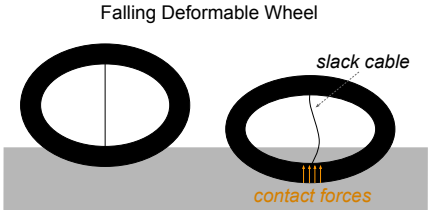

Fig. 3: Example of a cable that cannot remain straight. In such cases, the cable constraint (i.e. limit on displacement) is removed from the QP, for a certain amount of time that corresponds to the time the motors take to pull on the cable.

It appears sometimes that the matrix of the QPCC is no longer positive definite, nor positive semi-definite (e.g. when the size of the effectors space is too low), but indefinite. The cause of this problem has not been yet identified, while the problem appears only in few experiments. However, when it is detected, we use the implementation provided by qpOASES for indefinite Hessian, when solving a $\mathrm{QP_I}$, piece of the global QPCC problem. Unfortunately in that case, the convergence of the algorithm is no longer guaranteed. In practice, the solver finds a solution that is not far from the previous one (i.e. the solution computed at the previous time step), but can either way produce small oscillations.

IV. RESULTS

We applied the method on several numerical examples, showing some locomotion, and manipulation controls. We also demonstrate the effectiveness of the method on a real soft robot performing a manipulation task. We invite the reader to watch the attached video in which each experiment is presented.

A. Locomotion

Circular soft robot. Numerous research projects aim at designing soft robots able to walk [19], crawl [20], or roll [21]. In [20], Sugiyama *et al.* propose a soft circular robot actuated with eight shape-memory alloy coils capable of crawling and jumping. To control the robot they define by hand a periodic voltage pattern that produces the motions. Based on their

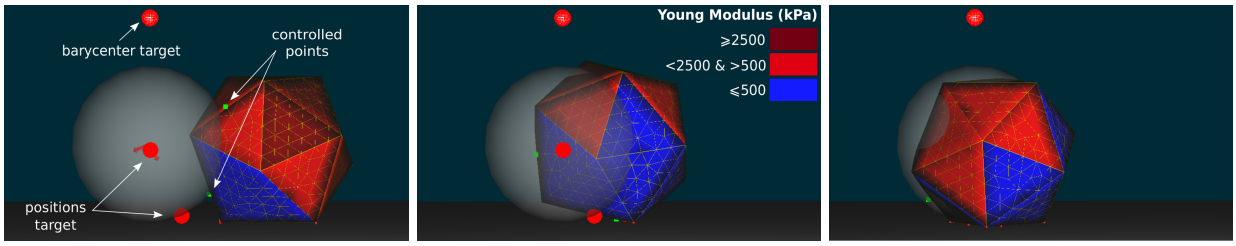

Fig. 4: Spherical robot simulation. The inverse problem outputs a different Young modulus for each cell, and a pressure for the inner cavity, to make it roll to a target position. Here we control points on the soft body surface and the position of its geometric center.

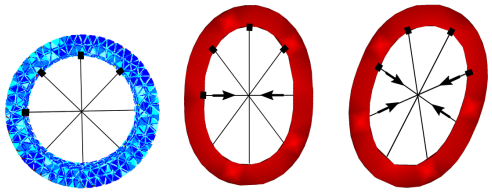

Fig. 5: Circular cable-driven soft robot. Left: rest shape and FEM mesh. Middle, right: deformed shape with corresponding actuation.

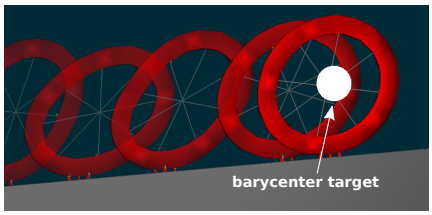

Fig. 6: Simulation of the circular robot locomotion. The algorithm finds how to actuate the four cables to control the position of the deformable structure geometric center. Moving the geometric center target (white sphere) forward or backward makes the structure roll.

design, we model a circular soft robot actuated with four cables (see Fig.5). We built a simulation in which we are able to drive the circular robot on a slope by optimizing the position of its geometric center (see Fig. 6). Simply moving the geometric center target makes the deformable structure start to roll and reach the desired position. The rolling pattern is easy to figure, but when dealing with inclined terrain, the temporal rate of the pattern changes depending on the degree of the steepness. That is why using our optimization algorithm to control the motion of the robot may be better suited. Furthermore, as our controller has real-time performance, we can track the gradient of the terrain and control the circular robot's progression on evolving platforms.

Spherical soft robot. In [21], Steltz *et al.* propose a spherical robot composed of cells around its outer perimeter (each cell being filled with jamming material), and a centeral actuated cavity. By unjamming a subset of outer cells, and inflating the central actuator, the robot is capable of rolling.

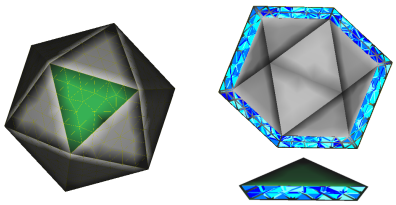

Fig. 7: Spherical soft robot. Left: global shape with one of the 20 cells shown in green. Top right: cross-section of the robot showing the central cavity and the FEM mesh. Bottom right: one isolated cell. Note that the FEM mesh follows the cells boundary.

We modeled a similar spherical robot, an icosahedron with a central cavity (see Fig. 7). We proposed in [22], among others, to optimize the material parameter of deformable structures (for instance the Young Modulus). Using the same approach we were able to optimize both the stiffness (Young modulus) of each cell and the pressure to apply in the central cavity to drive the structure (see Fig. 4). To control the robot's displacement, we optimize the position of its geometric center, and two additional points located on its surface. To make the robot roll, we move the desired position of the geometric center and compute the kinematic of a corresponding rigid sphere to obtain the target of the two other controlled points. In comparison with the circular robot's example, we have to control more than just the geometric center; the two additional points allow us to describe a global movement.

B. Manipulation

Trunk. In this example we propose to control the position/orientation of a grasped object instead of directly controlling the position of the soft robot. We modified a bit the shape of the soft trunk proposed in [11] to make it able to grab objects. The new shape is a bit longer and have higher low-cut joints. In the simulation both the cup and the trunk are deformable. In Fig. 8 we show results of the simulation with different orientation target for the cup. In Fig. 9 we control the orientation of a real plastic cup online using our simulation framework. In Fig. 10, we give the trajectories of the plastic cup center of mass. A magnetic sensor was placed at the bottom of the cup. The sensor gives both the position and the orientation of the real cup. Measured error for the

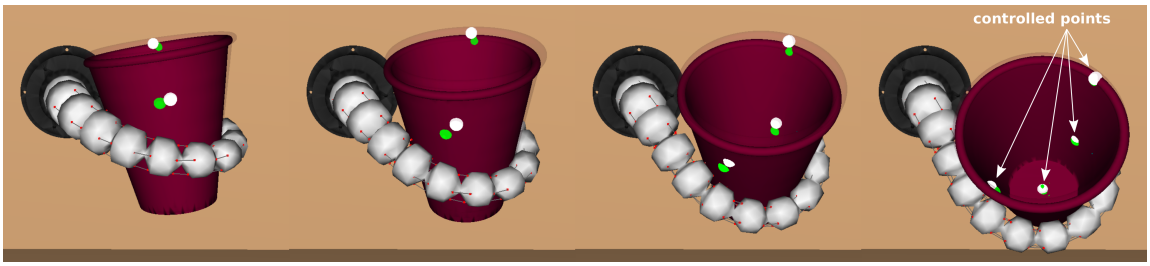

Fig. 8: Simulation of a soft gripper holding a deformable cup subject to gravity. Here we optimize the cables displacements to control the position/orientation of the cup. A phantom of the cup target is shown in transparency. The cup have four controlled points represented by the green spheres. The corresponding targets are represented by the white spheres.

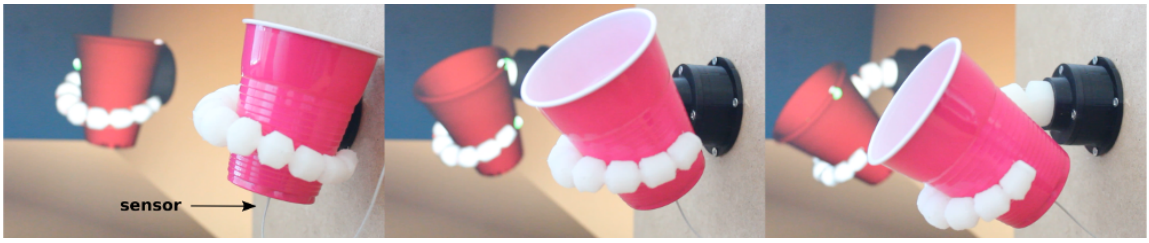

Fig. 9: Real soft robot actuated online using the output of the simulation. In this scenario, using our control framework, we are able to control the orientation of the real plastic cup (see the attached video).

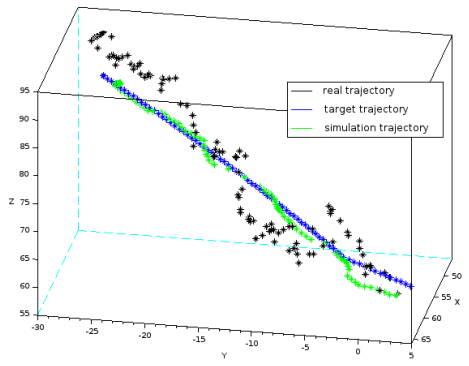

Fig. 10: Trajectories of the plastic cup center of mass: (black) real trajectory from the magnetic sensor, (blue) target trajectory, and (green) simulation trajectory.

	Target vs. Simulation		Simulation vs. Reality	
	mean	stdev	mean	stdev
c. of mass	1.34*mm*	0.37*mm*	8.69*mm*	3.79*mm*
orientation	2.21*deg*	1.60*deg*	5.79*deg*	3.58*deg*

TABLE I: Mean error and standard deviation of the plastic cup center of mass and orientation (for the trajectories given in Fig. 10). The error of the orientation is calculated on the plan of the greater displacement (see Fig. 9).

entire animation are given in Table I (please see the video for a better understanding of the results). We see that with this open-loop system we can obtain good match between the simulation and the robot. Yet, the robot should be equipped with sensors to correct the model on the tightening force. Indeed, if this force is not sufficient, the cup may slide. Note that if the target is out of the robot's working space, the algorithm will try to optimize the displacement to fit at best the target, but within the space of possible configurations.

Rigid fingers. We propose again to control the grasped object instead of directly controlling the position of the robot. In this example we use our algorithm to control not only the position, but the shape of a structure deformed by three rigid *fingers* (see Fig. 11). For each *finger*, we consider only the tip, which has three DoFs (the three translations). Thanks to the transmission of forces allowed by contacts, we are able to control the *fingers* tip position so that the deformable object reach a desired shape. This scenario has been proposed in [23]. The difference here is that we solve the actual problem by considering contacts in the inverse resolution. The deformable structure has no fixed part and is, like in the other simulations, subject to gravity. We control three positions on the deformable object surface (see Fig. 11).

C. Performance

In this section, we give the computation time of each simulation shown in this paper. The two main computation steps of the simulation are the computation of the matrices W_{ij} $(i, j = e, a, c)$ and the resolution of the sequence of QP problems. In Table. II, we show the average computation time for these two main steps. For more complex geometries, a large number of nodes may be required. In that case, the size of the FEM matrix will be large as well and the computation of $\mathbf{W}$ will take time. For example in the trunk and the cup simulation, there is a high number DoFs. In such case we can use a GPU based algorithm to compute the matrix W. Using this approach we were able to run the simulation at near real-time.

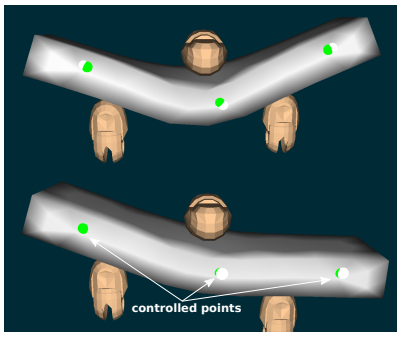

Fig. 11: Simulation of a soft beam subject to gravity, and held and deformed by three rigid *fingers*. The optimization finds the *fingers* position that leads to a desired deformation of the beam, and also prevents it from falling. The controlled points are the green spheres and their target are the white spheres.

Examples	Cont.	DoFs	W(ms)	QPs(ms)
Circular	9	2238	7.91	0.21
Spherical	39	3003	79.56	10.67
Trunk(GPU)	81	2127, 3765	76.53	35.47
Fingers	12	1335, 9	2.87	0.98

TABLE II: The different examples simulated, with the average number of contact variables, the number of DoFs (3*number of nodes) and the computation time in ms of the matrices W_{ij} construction and sequence of QPs resolution.

V. CONCLUSION & DISCUSSION

In this paper, we proposed a generic algorithm to control the motion of soft robots able to manipulate objects or roll from one place to another, with the assumption that each contact is either sticking or inactive. The method has real-time performance, which allow us to interactively control our robots and deal with evolving environments, as long as they are known. The method is mainly limited by the fact it does not handle dynamic friction (sliding contacts). Thus, we should only use this method on scenarios where sliding effects are negligible. As mentioned in the section IV, in the soft trunk example, the robot does not grab the cup well enough to always prevent it to slide. We think that this problem should vanish by either improving the robot's design or using a closed-loop strategy with sensor mounted on the robot to correct the model on the tightening force. As a future work, we would also like to test our method on a real gripper, and work on a pipeline to be able to switch from a control of the end-effector to a control of the grasped object. The real time performance is limited by both the complexity of the robot's geometry and the number of contacts. If the number of DoFs and/or contacts is high, the computation of the matrix W may take time. To reduce this computation time we can use methods based on model order reduction [24]. Yet, for the examples given in this paper, the use of GPU was satisfactory enough. Other methods can also be used to significantly reduce the number of contacts, regardless of the geometric complexity [25].

REFERENCES

[1] C. Majidi, "Soft robotics: A perspective, current trends and prospects for the future," *Soft Robotics*, 2014.

[2] "Soft robotics: a bioinspired evolution in robotics," *Trends in Biotechnology*, vol. 31, no. 5.

[3] D. Rus and M. Tolley, "Design, fabrication and control of soft robots," *Nature*, 2015.

[4] I. Robert J. Webster and B. A. Jones, "Design and kinematic modeling of constant curvature continuum robots: A review," *The International Journal of Robotics Research*, vol. 29, no. 13.

[5] H. Wang, B. Yang, Y. Liu, W. Chen, X. Liang, and R. Pfeifer, "Visual servoing of soft robot manipulator in constrained environments with an adaptive controller," *Transactions on Mechatronics*, vol. 22, 2017.

[6] M. Giorelli, F. Renda, G. Ferri, and C. Laschi, "A feed-forward neural network learning the inverse kinetics of a soft cable-driven manipulator moving in three-dimensional space," in *International Conference on Intelligent Robots and Systems*, 2013.

[7] M. C. Yip and D. B. Camarillo, "Model-less feedback control of continuum manipulators in constrained environments," *Transactions on Robotics*, vol. 30, no. 4, 2014.

[8] A. Petit, F. Ficuciello, G. A. Fontanelli, L. Villani, and B. Siciliano, "Using Physical Modeling and RGB-D Registration for Contact Force Sensing on Deformable Objects," in *International Conference on Informatics in Control, Automation and Robotics*, 2017.

[9] Z. Zhang, J. Dequidt, and C. Duriez, "Vision-based sensing of external forces acting on soft robots using finite element method," *Robotics and Automation Letters*, vol. 3, no. 3, 2018.

[10] E. Coevoet and T. M.-B. et. al., "Software toolkit for modeling, simulation, and control of soft robots," *Advanced Robotics*, vol. 31.

[11] E. Coevoet, A. Escande, and C. Duriez, "Optimization-based inverse model of soft robots with contact handling," *Robotics and Automation Letters (Proc. ICRA)*, vol. 2, no. 3, 2017.

[12] J. Kim and N. Pollard, "Direct control of simulated non-human characters," vol. 31, no. 4, 2011.

[13] L. Liu, K. Yin, B. Wang, and B. Guo, "Simulation and control of skeleton-driven soft body characters," *ACM Transactions on Graphics*, vol. 32, no. 6, 2013.

[14] J. Tan, G. Turk, and C. K. Liu, "Soft body locomotion," *Transactions on Graphics*.

[15] M. Posa, C. Cantu, and R. Tedrake, "Direct trajectory optimization of rigid body dynamical systems through contact," in *Algorithmic foundations of robotics X*, 2013.

[16] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, and et. al., "Sofa: A multi-model framework for interactive physical simulation," in *Studies in Mechanobiology Tissue Engineering and Biomaterials*, 2012.

[17] M. Anitescu and F. A. Potra, "Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems," *Nonlinear Dynamics*, vol. 14, no. 3, Nov 1997.

[18] H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, 2014.

[19] M. T. Tolley, R. F. Shepherd, B. Mosadegh, K. C. Galloway, M. Wehner, M. Karpelson, R. J. Wood, and G. M. Whitesides, "A resilient, untethered soft robot," *Soft Robotics*.

[20] Y. Sugiyama and S. Hirai, "Crawling and jumping by a deformable robot," *The International Journal of Robotics Research*, vol. 25, no. 5-6.

[21] E. Steltz, A. Mozeika, N. Rodenberg, E. Brown, and H. M. Jaeger, "Jsel: Jamming skin enabled locomotion," in *International Conference on Intelligent Robots and Systems*, 2009.

[22] E. Coevoet, N. Reynaert, E. Lartigau, L. Schiappacasse, J. Dequidt, and C. Duriez, "Registration by interactive inverse simulation: application for adaptive radiotherapy," *International Journal of Computer Assisted Radiology and Surgery*, vol. 10, 2015.

[23] F. Ficuciello, A. Migliozzi, E. Coevoet, A. Petit, and C. Duriez, "Fem-based deformation control for dexterous manipulation of 3d soft objects," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[24] O. Goury and C. Duriez, "Fast, generic and reliable control and simultation of soft robots using model order reduction," *Transactions on Robotics*, 2018.

[25] J. Allard, F. Faure, H. Courtecuisse, F. Falipou, C. Duriez, and P. G. Kry, "Volume contact constraints at arbitrary resolution," *Transactions on Graphics*, vol. 29, 2010.