An efficient approach to closed-loop shape control of deformable objects using finite element models

A. Koessler, N. Roca Filella, B.C. Bouzgarrou, L. Lequievre, J.-A. Corrales Ramon

An efficient approach to closed-loop shape control of deformable objects using finite element models

A. Koessler[1], N. Roca Filella[1], B.C. Bouzgarrou[1], L. Lequièvre[1] and J.-A. Corrales Ramon[2]

***Abstract*— Robots are nowadays faced with the challenge of handling deformable objects in industrial operations. In particular, the problem of shape control, which aims at giving a specific deformation state to an object, has gained interest recently in the research community. Among the proposed solutions, approaches based on finite elements proved accurate and reliable but also complex and computationally-intensive.**

In order to mitigate these drawbacks, we propose a scheme for shape control that does not require to run a real-time simulation or to solve an implicit optimization problem for computing the control outputs. It is based on a partition of the nodal coordinates that allows deriving a control law directly from tangent stiffness matrices. This formulation is also coupled with the introduction of reduced finite element models. Simulation and experimental results in the context of linear deformable object manipulation demonstrate the interest of the proposed approach.

I. Introduction

Robotic manipulation of deformable objects is a blooming field in robotics research. This can be explained by its numerous applications in the fields of manufacturing, service industry or surgery for instance, where robots are facing objects that are inherently deformable such as elastic bodies, ropes, clothes or living tissues. Those objects come in different topologies (linear, planar, three-dimensional) and with different deformation behaviours (isometric, elastic, elastoplastic) [1]. As described in [2], methods developed for rigid object manipulation cannot fulfill industrial expectations, since the state of deformation of the manipulated object needs to be taken into account at every step of the robotic task: system modeling, motion planning, robot control and state perception. This justifies the many contributions aimed at deformable objects that have been proposed on those topics over the last decade, as shown by recent surveys such as [1], [3].

Shape control: One particular problem we wish to tackle in this paper is the problem of *shape control* (or *shape servoing*) of deformable objects. It consists in driving the actuators of a robotic system so that the manipulated deformable object reaches a given target shape. Solving this problem is necessary in contexts where the success of the task relies on the final deformation state of the manipulated object. Industrial applications are draping processes, multi-layer product assembly or cable routing for instance.

[1] Université Clermont Auvergne, Clermont Auvergne INP, CNRS, Institut Pascal, F-63000 Clermont-Ferrand, France {firstname.lastname}@sigma-clermont.fr

[2] Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS), Universidade de Santiago de Compostela, 15782, Santiago de Compostela, Spain {firstname.lastname}@usc.es

Contributions on this subject all seek to answer a core question: how to quantify the interaction between the displacement or force exerted by the actuators and the change in shape of the object? The mathematical definition of this problem is often referred to as *deformation Jacobian* estimation. The methods to answer this question can be separated in two main categories, depending on their reliance on a deformation model.

Model-free approaches: The first category regroups methods where the deformation Jacobian is numerically estimated from sensor measurements. We can think of the works [4], [5], [6] for instance. In model-free approaches, the control problem is generally formulated in sensor measurement space. These methods take full advantage of machine learning to quantify the interaction [7] and reuse learned trajectories [8], but adaptive control [9] or predictive control [10] can also be used to improve model accuracy. These methods avoid the step of mechanical identification of the manipulated object, but quality of initial estimates for the interaction remains crucial, and loss of generality or over-training in specific configurations remain hazards for learned models.

Model-based approaches: Even though popular older works are known, as evidenced by the review [11], model-based shape servoing gets less attention than its model-free counterpart nowadays. Non-mechanical models such as templates [12] are computationally inexpensive and well suited to visual servoing, but their results guarantee the visual convergence of some points over the surface of the object; consequently, the mechanical constraints related with its material properties are not taken into account. In some cases, low-complexity mechanical models can be used, such as catenaries for isometric linear objects [13]. In contrast, *finite element models* (FEM), which are based on discrete or continuum mechanical equations, can handle many materials and shapes of objects. These advantages justify the research interest on FEM-based methods. Shaping of objects using a mass-spring model and torque control is described in [14], with an emphasis on motion planning to reach a given shape. We also can mention the contributions of [15] and [16] which have in common their open-loop structure and their formulation of the control law as the solution to an optimization problem. In both approaches, resolution of the finite element problem at each time step is required, which is the major drawback of the method, since it is resource-consuming and involves mechanical expertise. However, very precise results can be obtained from such mechanical models.

More FEM-based approaches can be found in the neigh-

boring research field of soft robotics [17], where the estimation of the deformation Jacobian is also of crucial relevance for controlling end-effector pose accurately. Unlike in deformable objects manipulation, model-based closed-loop control of soft robots has recently been addressed in [18], [19]. The real-time tracktability of FEM simulation in these works is allowed by proper orthogonal decomposition techniques, in order to reduce the dimension of state-space equations for control. Nonetheless, these works still rely on the computationally-intensive real-time FEM simulation, and accounting for reduction errors makes the control problem more complex.

Contribution: In the proposed work, we wish to introduce a method for shape control based on a finite element formulation, which is an accurate and versatile method. Through a novel expression of the control problem, we show how to get rid of the need for real-time simulation of the system in the control algorithm. The control problem is then solved using straightforward linear control techniques that enable control loop closure. The efficiency of the proposed control law is validated experimentally, in the context of manipulation of a linear elastic product. In this case study, we will prove that using hand-made FE models with very low node count is sufficient to obtain good results.

II. Finite element-based control law

In this section, we present the mechanical finite element model that is used in order to derive a closed-loop control law. This presented formulation is valid under the hypothesis that inertial effects are negligible (quasi-static motion).

A. Finite element modeling and notations

Finite element method is a widely used mathematical method in engineering, especially in the field of structural analysis [20]. It consists in discretizing a body in small elements, forming a mesh consisting of nodes and vertices. In the following, we consider that the shape control problem is resolved when the nodes of the mesh are able to reach a given position in space.

The basic formula for finite element analysis is the static equilibrium equation, which is often written down as

$$\mathbf{K}_g\mathbf{u} = \mathbf{f} \tag{1}$$

where $\mathbf{K}_g$ is the global stiffness matrix of the structure, $\mathbf{u}$ is the vector of nodal displacements and $\mathbf{f}$ is the vector of external nodal forces. This formula encapsulates the important idea that FEM provides a way to link nodal displacements to nodal forces through the concept of stiffness matrix assembly (see for instance section 3.3. of [20]). However, it is not valid in the general case, where large displacements and large deformations can arise. Since we seek a more generic approach, we will write the static equilibrium equation in the local formulation

$$\mathbf{K}(\mathbf{u})\boldsymbol{\Delta}\mathbf{u} = \boldsymbol{\Delta}\mathbf{f} \tag{2}$$

where $\mathbf{K}$ is the locally valid tangent stiffness matrix (which depends on node coordinates) linking a small nodal displacement $\boldsymbol{\Delta}\mathbf{u}$ with a small variation in external nodal forces $\boldsymbol{\Delta}\mathbf{f}$.

B. Partition between nodes for control

To serve the purpose of shape control, the robot must be given a goal shape and the capability to exert mechanical actions on the body. In the finite element model, this translates as different properties that distinguish nodes between each other in terms of control and perception. For control purposes, there will be three types of nodes:

- Driven nodes (subscript d) are the nodes which are bound to the end-effector or the gripper of the robot. They can be imposed a given force or displacement and undergo actions that modify the state of the structure;
- Target nodes (subscript t) are the nodes whose goal position $\mathbf{u}_t^*$ is specified. They define the target shape of the object;
- Free nodes (subscript f) are nodes which are neither driven nor target nodes. They do not play a role for control but add precision to the model.

This partition turns (2) into the following system of equations:

$$\mathbf{K}_{dd}\boldsymbol{\Delta}\mathbf{u}_d + \mathbf{K}_{dt}\boldsymbol{\Delta}\mathbf{u}_t + \mathbf{K}_{df}\boldsymbol{\Delta}\mathbf{u}_f = \boldsymbol{\Delta}\mathbf{f}_d \tag{3}$$

$$\mathbf{K}_{td}\boldsymbol{\Delta}\mathbf{u}_d + \mathbf{K}_{tt}\boldsymbol{\Delta}\mathbf{u}_t + \mathbf{K}_{tf}\boldsymbol{\Delta}\mathbf{u}_f = \boldsymbol{\Delta}\mathbf{f}_t \tag{4}$$

$$\mathbf{K}_{fd}\boldsymbol{\Delta}\mathbf{u}_d + \mathbf{K}_{ft}\boldsymbol{\Delta}\mathbf{u}_t + \mathbf{K}_{ff}\boldsymbol{\Delta}\mathbf{u}_f = \boldsymbol{\Delta}\mathbf{f}_f. \tag{5}$$

The main idea behind this partition is that it allows straightforward definition of the control problem. While it has not been already reported this way to our knowledge, it can be related to the partition presented in [21] for soft robot control or [22] for soft robot compliance analysis. From (3)-(5), one can see which quantities are computed by the control law, which ones need to be estimated or perceived. From this, shape control can be achieved through force or position servoing. We will focus on the latter: the driven nodes are imposed a position, $\boldsymbol{\Delta}\mathbf{u}_d$ is the control output and a control law h under the form $\boldsymbol{\Delta}\mathbf{u}_d = h(\boldsymbol{\Delta}\mathbf{u}_t, \boldsymbol{\Delta}\mathbf{u}_f)$ should be found so that $\mathbf{u}_t$ converges towards $\mathbf{u}_t^*$. We will now synthesize this control law h.

C. Velocity control law synthesis

From now on, we will drop small variations $\boldsymbol{\Delta}$ for dotted time derivatives, which explains why this is actually a velocity-based control law. Under the assumption that target and free nodes are submitted to zero or constant external forces (eg. weight), we have $\dot{\mathbf{f}}_t = \dot{\mathbf{f}}_f = \mathbf{0}$ and (3)-(5) become

$$\mathbf{K}_{dd}\dot{\mathbf{u}}_d + \mathbf{K}_{dt}\dot{\mathbf{u}}_t + \mathbf{K}_{df}\dot{\mathbf{u}}_f = \dot{\mathbf{f}}_d \tag{6}$$

$$\mathbf{K}_{td}\dot{\mathbf{u}}_d + \mathbf{K}_{tt}\dot{\mathbf{u}}_t + \mathbf{K}_{tf}\dot{\mathbf{u}}_f = \mathbf{0} \tag{7}$$

$$\mathbf{K}_{fd}\dot{\mathbf{u}}_d + \mathbf{K}_{ft}\dot{\mathbf{u}}_t + \mathbf{K}_{ff}\dot{\mathbf{u}}_f = \mathbf{0}. \tag{8}$$

From this system of equations, we want to link control law output $\dot{\mathbf{u}}_d$ to controlled quantity $\dot{\mathbf{u}}_t$. Only two of the three equations are needed to do this. Since measuring $\dot{\mathbf{f}}_d$ would need a costly dedicated sensor, we will only combine (7) and (8) to obtain the following relation:

$$\mathbf{K}_d\dot{\mathbf{u}}_d + \mathbf{K}_t\dot{\mathbf{u}}_t = \mathbf{0} \tag{9}$$

where $\mathbf{K}_d = \mathbf{K}_{td} - \mathbf{K}_{tf}\mathbf{K}_{ff}^{-1}\mathbf{K}_{fd}$ and $\mathbf{K}_t = \mathbf{K}_{tt} - \mathbf{K}_{tf}\mathbf{K}_{ff}^{-1}\mathbf{K}_{ft}$. Since our goal is to drive the error $\mathbf{e}_t = \mathbf{u}_t^* - \mathbf{u}_t$ towards $\mathbf{0}$, we seek to impose the following error equation with first-order dynamics:

$$\dot{\mathbf{e}}_t + \mathbf{G}_p \mathbf{e}_t = \mathbf{0}. \tag{10}$$

This relation ensures exponentially stable convergence towards zero if $\mathbf{G}_p$ is a positive definite matrix.

Remarking that $\dot{\mathbf{e}}_t = -\dot{\mathbf{u}}_t$ and substituting (9) into (10), we have

$$\mathbf{K}_t^{-1}\mathbf{K}_d\dot{\mathbf{u}}_d + \mathbf{G}_p\mathbf{e}_t = \mathbf{0} \tag{11}$$

and isolating control law output we obtain the control law

$$\dot{\mathbf{u}}_d = -\mathbf{K}_d^{+}\mathbf{K}_t\mathbf{G}_p\mathbf{e}_t \tag{12}$$

where the superscript $+$ denotes the pseudo-inverse of a matrix. This control law computes velocities of driven nodes such that the coordinates of target nodes converge towards their goal values. We find here a control law that follows the principle of pseudo-inverse allocation coined in [21].

D. Perception and estimation of nodal coordinates

For real implementation of this control law, all nodal coordinates must be known, as they are used in the computation of the error $\mathbf{e}_t$, but also of the matrices $\mathbf{K}_d$ and $\mathbf{K}_t$. In the general case, it is impossible to resolve this problem through perception only, since nodes may be hidden within the object, vision may be partial or occluded, etc. Literature overcomes this by solving implicitely the whole FE problem. However, the whole shape of an object can be deduced from few known nodal coordinates, as they define boundary conditions that allow solving the local equilibrium problem.[1] Consequently, we can introduce a novel partition of the nodes:

- Perceived nodes (subscript p) are nodes whose coordinates can be measured;
- Estimated nodes (subscript e) are nodes whose coordinates cannot be measured, and should hence be computed.

Given this partition, static local equilibrium equation can be written as

$$\mathbf{K}_{pp}\dot{\mathbf{u}}_p + \mathbf{K}_{pe}\dot{\mathbf{u}}_e = \dot{\mathbf{f}}_p \tag{13}$$

$$\mathbf{K}_{ep}\dot{\mathbf{u}}_p + \mathbf{K}_{ee}\dot{\mathbf{u}}_e = \dot{\mathbf{f}}_e. \tag{14}$$

The derivative of effort is nonzero for driven nodes, which are generally perceived (through proprioception in the actuation chain). Consequently, $\dot{\mathbf{f}}_p$ is usually non null while $\dot{\mathbf{f}}_e$ is. Taking only the second line, the following relation can then be obtained:

$$\dot{\mathbf{u}}_e = \mathbf{K}_{ee}^{-1}\mathbf{K}_{ep}\dot{\mathbf{u}}_p. \tag{15}$$

Provided an initial estimate can be given (through measurement or offline simulation), estimated node velocity can then be integrated numerically at each time step k as

$$\mathbf{u}_e[k+1] = \mathbf{u}_e[k] + \dot{\mathbf{u}}_e[k]\Delta t \tag{16}$$

[1]This is not true when instabilities such as buckling occur, but we deem them unlikely to happen in usual object manipulation.

where $\dot{\mathbf{u}}_e[k]$ is computed thanks to (15). The use of real updated values $\dot{\mathbf{u}}_p$ in the computation is expected to provide precise enough estimated values $\mathbf{u}_e$, so that the estimated FE model remains truthful to real object shape.

E. Remarks on actuation and perception dimensions

In order to use efficiently the proposed controller, the choice for type of nodes has to be adapted to the use case. Apart from the obvious consideration that enough actuation degrees must be available to reach certain shapes, there is a need to specify the target nodes in a coherent manner. If the task requires to position multiple nodes, then user-defined values $\mathbf{u}_t^*$ must define a physically feasible solution. Through pseudo-inverse computation, the control law (12) can only minimize a global error on all target nodes, but not the local error on each of the target nodes.

Regarding perception, a fair hypothesis is that the position of every driven node is known through proprioception in the actuation chain. This means that the shape of the object could be known in theory from those sole measurements, by solving equilibrium equations (1) or (2), but it relies on good knowledge of manipulated object deformation model. In contrary, perceiving additional nodes, through vision for instance, allows to robustify the control process with regards to modeling errors. This is especially interesting when using reduced FE models with low element count and simple kinematics, which is the direction we will explore in our case study.

III. Case Study

The application in which we will showcase our contribution consists in manipulating a thin sheet of vulcanized rubber with a robot, as shown on Fig. 1. To this end, we propose a reduced finite element model of the elastic object and a ROS implementation for our controller to be run on a UR5 robot.

A. Object FE model

The object we manipulate shows a typical elastic behavior with large displacements due to bending under its own weight. To make modeling easier, we will not consider shearing and torsion, so that the shape can be fully described in the longitudinal plane of symmetry of the object. Hence, we see the rubber sheet as a linear object (in the sense of Sanchez's classification [1]) deforming in a 2D plane.

Given that the object exhibits bending stiffness, the logical choice would be to use beam elements in the FE modeling. However, they introduce an angular coordinate which is tricky to perceive, so we settled on using even simpler 1D rod elements. For a perfectly elastic behavior, the elemental stiffness matrix of the 1D rod element i in a local frame (O_i, X_i, Y_i) is

$$\mathbf{K}_i = \frac{E_i A_i}{\ell_{0i}} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{17}$$

where E_i denotes the Young's modulus of the material, A_i the cross-section area of the object and ℓ_{0i} the element's

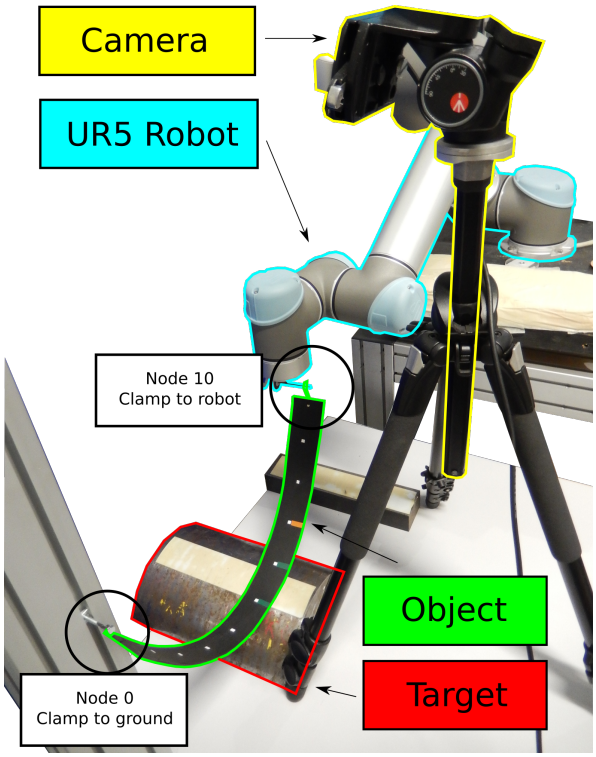

Fig. 1: Picture of the experimental setup.

length at rest. Elemental matrices are translated to the global (O,X,Y) frame and assembled to obtain the locally-valid tangent stiffness matrix of the structure :

$$\mathbf{K} = \text{assemble}_i \left(\mathbf{R}(\theta_i) \mathbf{K}_i \mathbf{R}(\theta_i)^T \right) \tag{18}$$

where $\mathbf{R}$ is the usual 2×2 rotation matrix and θ_i the angle between the x-axes of local element frame and global frame. These rotations introduce reliance on nodal coordinates, since for an element i linking nodes j and $j+1$ we use the relation $\theta_i = \text{atan2}(u^y_{j+1} - u^y_j, u^x_{j+1} - u^x_j)$ to compute the θ_i angle.

B. Hardware and software implementation

To actuate the system, a UR5 robot from Universal Robots company is used. Its motions are controlled through a dedicated ROS driver that provides a joint velocity interface. The control loop was programmed based on ROS Control and its main loop structure is highlighted on Fig. 2. Programs are available on GitHub[2].

One can see two interactions with other programs: user-made goal specification and sensor data acquisition. For the latter, we use the proprioception of the UR5 robot and a vision system. Vision perception is performed with an Intel RealSense D435 RGB-D camera which allows to measure node coordinates through point tracking. The algorithm that performs tracking is programmed using Python and OpenCV. Due to image acquisition which is time-consuming, the loop rate is capped to 30 Hz in our implementation.

[2]Repository link: github.com/adkoessler/mrod_ur_ip_controllers

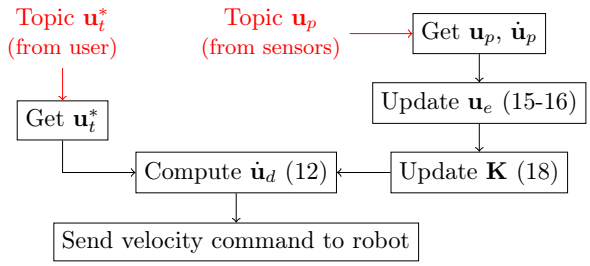

Fig. 2: Schematic representation of the controller.

IV. Experimental validation

Following controller description, we perform shape control experiments on the real system. The proposed experiments evaluate the impact of modeling errors, compute the level of precision that can be expected experimentally and show to which extent the estimation of non-perceived coordinates can be trusted.

Setup: The object is a 60 cm-long sheet of rubber. We settle on dividing it in 11 nodes and 10 elements, making the model low-dimensional but still capable to represent deformations faithfully. From characterization, we have the real values $\bar{E} = 5.82$ MPa, $\bar{\ell}_0 = 60$ mm and $\bar{A} = 172$ mm^2 for object parameters. Node 0 (leftmost) is clamped to the ground, node 10 (rightmost) is driven by the UR5 robot and free nodes 1 to 9 may be perceived or estimated.

A. Sensitivity to modeling errors

Since the presented control law is model-based, a very desirable property would be robustness with regard to modeling errors. In the presented case study, these errors could come from misidentified parameters (namely Young's modulus) or ignored phenomena such as bending.

To evaluate this aspect, we skew voluntarily Young's modulus value. In the controller, we setup $E = 1.50$ MPa, which is a typical value for this family of materials[3]. The task we want to accomplish consists in positioning nodes 5 and 6 relatively with a piece of tape: relevant goal coordinates in the global frame for these nodes are resp. $[-.880; -.163]$ and $[-.826; -.147]$ (m). Node 1 to 6 are perceived with the RGB-D camera.

This task is illustrated on Fig. 3 but also in the video attachment to this paper. Tuning the controller by hand, we found that $\mathbf{G}_p = \text{diag}(0.2)$ gives satisfactory results in terms of stability and velocity. The specified goal position is reached after five seconds, when the final norm of error vector $\mathbf{e}_t$ remains steadily under 6 mm. We conclude that the mismatch in modeling parameters introduced voluntarily does not have an impact on the precision of the result.

Analysis: As shown by expressions (17) and (18), stiffness matrices have a linear dependency on E. From control law (12), the terms in E in $\mathbf{K}_d$, $\mathbf{K}_t$ cancel out and the error signal still follows the first-order convergent behavior defined by (10). Practically, misidentification of the Young's

[3]Found on matweb.com under the "Natural Rubber, Vulcanized" category.

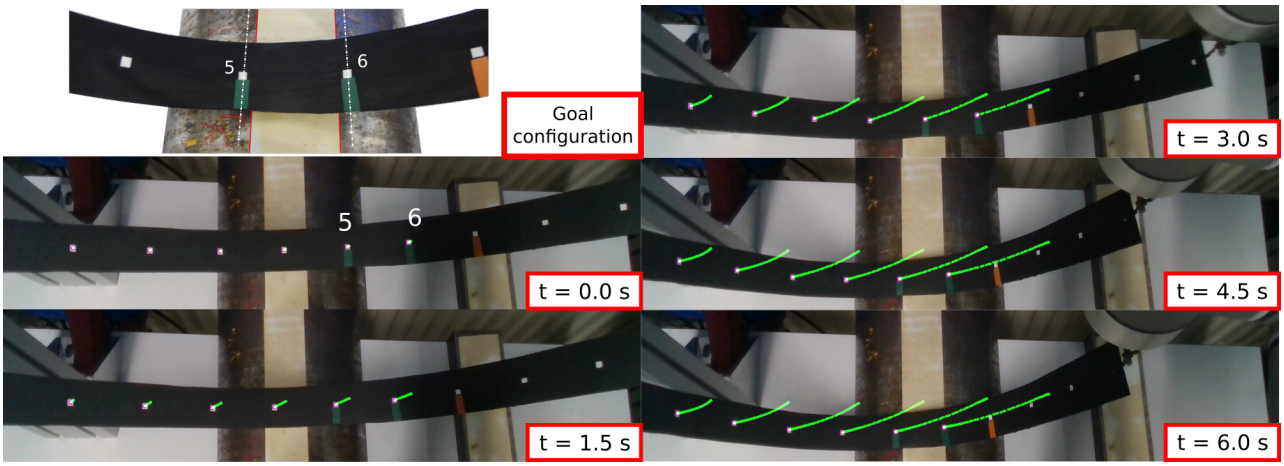

Fig. 3: Tracking of nodes during trajectory execution.

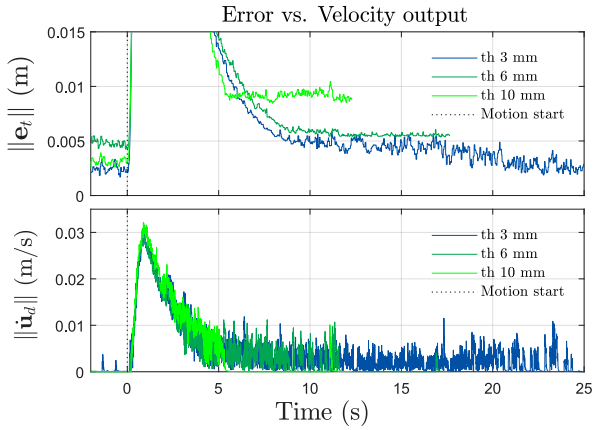

Fig. 4: Target error and computed velocity output during motion.

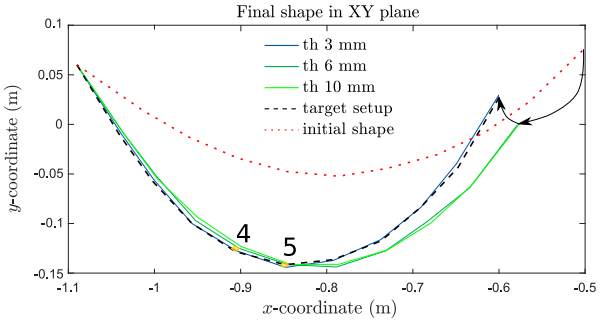

Fig. 5: Final shapes obtained for 3, 6 and 10 mm threshold.

modulus and neglect of other effects such as bending can be compensated by tuning the proportional gain matrix $\mathbf{G}_p$ accordingly.

B. Positioning precision evaluation

On the real system, it is impossible to reach a perfect precision for target nodes positioning, either because measurements are subject to noise (especially depth values of the RGB-D camera) or because the specified goal is not physically feasible. To answer this problem, we propose to terminate the control loop when the norm of the error vector $\mathbf{e}_t$ is smaller than a certain threshold.

In this experiment, we set the goal coordinates for target nodes 4 and 5 to resp. $[-.905; -.126]$ and $[-.849; -.142]$ (m). First, we try to reach this configuration when setting the error threshold to 10 mm. From Fig. 4, we can see that the controller succeeds in reducing the error under this value and stops. Afterwards, we repeat this procedure, but lowering threshold value every time. An interesting case arises when the threshold value is down to 3 mm. From Fig. 4, one can see that the error cannot durably settle under this value due to noises. Consequently, termination condition is never met and velocities are computed indefinitely. It can be said that positioning precision cannot be better than 3 mm with the current implementation.

Analysis: This value for maximal precision makes sense, since depth measurement spread for our camera is typically 1% of camera to object distance, whose value is 300 to 500 mm in our setup. Aside from this, comparing global object shapes on Fig. 5, it can be seen that lowering the threshold has made the global shape much closer to the expected result. By raising the y-coordinate of the actuator compared to 6 and 10 mm cases, the controller has found a way towards a better fitting configuration.

C. Reliance on node coordinate estimation

As said in section II, estimation of node position is needed to handle cases where the vision system cannot measure the coordinates of all nodes. To illustrate this, we perform tasks from sections IV-A and IV-B but this time we act as if the right side of the object cannot be perceived. Different cases will be explored:

1) second experiment, target nodes (4,5) are perceived, nodes 6 to 9 are estimated
2) second experiment, target nodes (4,5) are estimated as well as nodes 6 to 9

3) first experiment, target nodes (5,6) are perceived, nodes 7 to 9 are estimated
4) first experiment, target nodes (5,6) are estimated as well as nodes 7 to 9

For this experiment, we track all nodes with the vision system to obtain a ground truth, but choose to only use values for perceived nodes in the control loop. We also keep the modeling error introduced for the first experiment. In each case, we measure the final value of three different error signals:

- the norm of expected control error $\mathbf{e}_t$, which is used with a 10 mm precision threshold,
- the norm of real control error $\mathbf{u}_t^* - \bar{\mathbf{u}}_t$, where the bar denotes the ground truth value,
- the norm of the estimation error $\mathbf{u}_e - \bar{\mathbf{u}}_e$, where the bar denotes the ground truth value.

Target nodes are perceived in cases 1 and 3; consequently, no error is introduced on target node coordinate and $\mathbf{e}_t$ is equal to $\mathbf{u}_t^* - \bar{\mathbf{u}}_t$. Results are compiled in the table I.

Analysis: The main difference between cases 1-2 and 3-4 is that the goal configuration involves a greater change in object shape in the latter case. Coincidentally, estimation error values are greater in cases 3 and 4. This means that estimation is subject to drift: the bigger the displacement, the larger the estimation error. A fair hypothesis is that the sharp filtering on RGB-D camera signals needed to eliminate the noise induces imprecision on $\dot{\mathbf{u}}_p$ which is propagated to $\mathbf{u}_e$ by (15) and (16). For case 4, the improper estimation is illustrated on Fig. 6. From this graph, nodes 6 to 8 exhibit the largest estimation errors. Estimation errors possibly stack up: the nodes located furthest to a perceived node will be even more subject to estimation drift.

Cases 2 and 4 hence have a very different outcome. While in case 2 the real control error remains just shy of the threshold value, in case 4 the object ends up ill-positioned by several centimeters. Moreover, the expected control error remains largely above the threshold in case 4, meaning that termination is never met and that the controller is stuck in a configuration from which it cannot improve. However, cases 1 and 3 are not prevented from reaching the goal in spite of estimation errors.

Overall, we can conclude that perception of target nodes is the best way to ensure success. When it is not possible, the estimation should be handled with care as it is subject to a slow drift. In future works, this problem could be mitigated by using the simulation-based approaches of the literature to carry out this estimation. However, satisfactory results in case 2 show that simulation solving could be executed at a drastically diminished rate, and not at each time step of the control loop as in the state of the art.

V. Conclusion

Contribution: In this article, we have proposed a method for closed-loop shape control of deformable objects based on a finite element formulation. The expression of the static equilibrium conditions have allowed us to link nodal displacements with each other. From this, we have synthesized a control law to reach a given shape and a procedure to estimate the current shape. Both together, they allowed us to write a controller whose capabilities have been illustrated and experimentally.

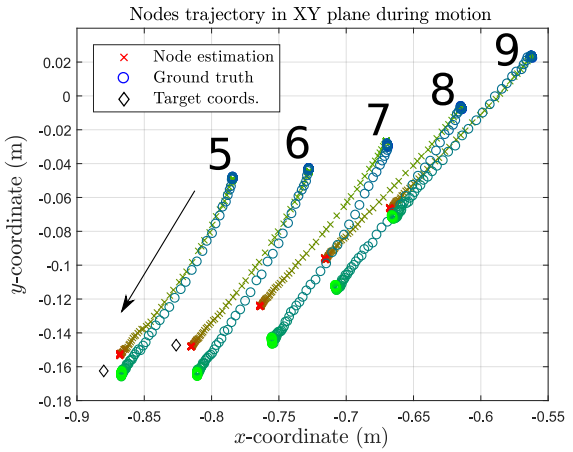

Fig. 6: Evolution of coordinates of rightmost nodes during motion in case 4.

TABLE I: Experimental error results

case	estimated e nodes	perceived p nodes	$\mathbf{e}_t$ mm	$\mathbf{u}_t^* - \bar{\mathbf{u}}_t$ mm	$\mathbf{u}_e - \bar{\mathbf{u}}_e$ mm
1	6 to 9	0 to 5, 10	9.58	9.54	15.77
2	4 to 9	0 to 3, 10	9.59	10.30	15.79
3	7 to 9	0 to 6, 10	9.73	9.73	41.78
4	5 to 9	0 to 4, 10	19.01	27.08	37.33

Advantages: Using an inexpensive RGB-D camera to close the control loop, we have shown our method capable of imposing a goal shape to an object without resorting to resource-intensive simulations and numercial optimizations. The robustness of the method with regards to modeling errors is another important merit, especially if object parameters are changeable or difficult to identify. Partial perception and estimation of hidden nodes was also validated experimentally.

Limitations and perspectives: An important perspective is to extend the presented work to more generic cases. Those cases include tasks with richer deformation patterns and 3D objects, but also textureless objects which cannot be handled by the current perception solution. We have also considered handling drift in node estimation with a computationally-inexpensive simulation method. Dynamic switching between perception and estimation for nodes in order to handle occlusions could also prove a worthy addition.

Acknowledgements

This research was funded by the French government IDEX-ISITE initiative 16-IDEX-001 (CAP 20-25) and by the European Union through the project COMMANDIA SOE2/P1/F0638 (Interreg Sudoe Programme and European Regional Development Fund). Authors wish to thank C. Weigel and A. Gravier from SIGMA Clermont for their help in setting up the experiments.

REFERENCES

[1] J. Sanchez, J. A. Corrales, B. C. Bouzgarrou, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey," *International Journal of Robotics Research*, vol. 37, no. 7, pp. 688–716, 2018.

[2] F. F. Khalil and P. Payeur, "Dexterous Robotic Manipulation of Deformable Objects with Multi-Sensory Feedback - a Review," *Robot Manipulators Trends and Development*, 2010.

[3] F. Nadon, A. J. Valencia, and P. Payeur, "Multi-modal sensing and robotic manipulation of non-rigid objects: A survey," *Robotics*, vol. 7, no. 4, 2018.

[4] D. Navarro-Alarcón, Y. H. Liu, J. G. Romero, and P. Li, "Model-free visually servoed deformation control of elastic objects by robot manipulators," *IEEE Transactions on Robotics*, vol. 29, no. 6, pp. 1457–1468, 2013.

[5] D. Navarro-Alarcon, Y. H. Liu, J. G. Romero, and P. Li, "On the visual deformation servoing of compliant objects: Uncalibrated control methods and experiments," *International Journal of Robotics Research*, vol. 33, no. 11, pp. 1462–1480, 2014.

[6] J. Zhu, B. Navarro, P. Fraisse, A. Crosnier, and A. Cherubini, "Dual-arm robotic manipulation of flexible cables," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.

[7] Y. Li, Y. Wang, Y. Yue, D. Xu, M. Case, S. F. Chang, E. Grinspun, and P. K. Allen, "Model-Driven Feedforward Prediction for Manipulation of Deformable Objects," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 4, pp. 1621–1638, 2018.

[8] T. Tang, C. Wang, and M. Tomizuka, "A framework for manipulating deformable linear objects by coherent point drift," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3426–3433, 2018.

[9] D. Navarro-Alarcon, H. M. Yip, Z. Wang, Y. H. Liu, F. Zhong, T. Zhang, and P. Li, "Automatic 3-D Manipulation of Soft Objects by Robotic Arms with an Adaptive Deformation Model," *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 429–441, 2016.

[10] C. Shin, P. W. Ferguson, S. A. Pedram, J. Ma, E. P. Dutson, and J. Rosen, "Autonomous tissue manipulation via surgical robot using learning based model predictive control," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2019-May, pp. 3875–3881, 2019.

[11] P. Jiménez, "Survey on model-based manipulation planning of deformable objects," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 154–163, 2012.

[12] M. Aranda, J. A. Corrales, Y. Mezouar, A. Bartoli, and E. Ozgur, "Monocular Visual Shape Tracking and Servoing for Isometrically Deforming Objects," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

[13] S. Flixeder, T. Glück, and A. Kugi, "Force-based cooperative handling and lay-up of deformable materials: Mechatronic design, modeling, and control of a demonstrator," *Mechatronics*, vol. 47, pp. 246–261, 2017.

[14] J. Das and N. Sarkar, "Autonomous shape control of a deformable object by multiple manipulators," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 62, no. 1, pp. 3–27, 2011.

[15] S. Duenser, J. M. Bern, R. Poranne, and S. Coros, "Interactive Robotic Manipulation of Elastic Objects," *IEEE International Conference on Intelligent Robots and Systems*, pp. 3476–3481, 2018.

[16] F. Ficuciello, A. Migliozzi, E. Coevoet, A. Petit, and C. Duriez, "FEM-Based Deformation Control for Dexterous Manipulation of 3D Soft Objects," *IEEE International Conference on Intelligent Robots and Systems*, pp. 4007–4013, 2018.

[17] C. Duriez, "Control of elastic soft robots based on real-time finite element method," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3982–3987, 2013.

[18] R. K. Katzschmann, M. Thieffry, O. Goury, A. Kruszewski, T. M. Guerra, C. Duriez, and D. Rus, "Dynamically closed-loop controlled soft robotic arm using a reduced order finite element model with state observer," *RoboSoft 2019 - 2019 IEEE International Conference on Soft Robotics*, pp. 717–724, 2019.

[19] M. Thieffry, *Model-Based Dynamic Control of Soft Robots*. PhD thesis, Université Polytechnique des Hauts-de-France, 2019.

[20] M. Bonnet, A. Frangi, and C. Rey, *The finite element method in solid mechanics*. McGraw Hill Education, 2014.

[21] Z. Zhang, T. M. Bieze, J. Dequidt, A. Kruszewski, and C. Duriez, "Visual servoing control of soft robots based on finite element model," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2017-Septe, pp. 2895–2901, 2017.

[22] M. Koehler, A. M. Okamura, and C. Duriez, "Stiffness Control of Deformable Robots Using Finite Element Modeling," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 469–476, 2019.