

# Interactive Robotic Manipulation of Elastic Objects

Simon Duenser<sup>1</sup>, James M. Bern<sup>1</sup>, Roi Poranne<sup>1,2</sup>, and Stelian Coros<sup>1</sup>

**Abstract**—In this paper, we address the challenge of robotic manipulation of elastically deforming objects. To this end, we model elastic objects using the Finite Element Method. Through a quasi-static assumption, we leverage sensitivity analysis to mathematically model how changes in the robot’s configuration affect the deformed shape of the object being manipulated. This enables an interactive, simulation-based control methodology, wherein user-specified deformations for the elastic objects are automatically mapped to joint angle commands. The optimization formulation we introduce is general, operates directly within a robot’s workspace and can readily incorporate joint limits as well as collision avoidance between the links. We validate our control methodology on a YuMi<sup>®</sup> IRB 14000, which we use to manipulate a variety of elastic objects.

## I. INTRODUCTION

Deformable objects such as cables, textiles, elastic metal rods, plush toys, cushions and paper are ubiquitous on construction sites, in factories, hospitals and in our own homes. Robotic manipulation of such physical objects therefore holds tremendous promise for many applications. However, deformable objects have infinite-dimensional configuration spaces, which makes them unsuitable for established motion planning and control techniques developed under the assumption that objects are rigid. As a consequence, solutions that allow robots to *effectively* manipulate and work with such flexible objects remain largely out-of-reach. In this paper we propose an efficient method for robotic manipulation of elastically-deforming objects. Our goal is to provide a system that is computationally efficient such that user-provided high-level task specifications can be translated into appropriate robot commands almost instantaneously.

The task of robotic manipulation of deformable objects has a long history. Shape control of deformable objects was explored early on by Kosuge et al., who leverage a finite element model to relate the bend angle of a piece of sheet metal to the placement of a robot’s grippers [1]. The overall task of soft body manipulation was also explored early on from the standpoint of path planning, with Lamiraux et al. finding paths for abstract robot end effectors to guide an elastic object around obstacles, while respecting that object’s physical model [2].

A large amount of work has been done on the specific problem of manipulating deformable linear objects (DLOs), for example in the context of knot tying [3]. The work on DLO manipulation includes approaches based on numerical simulation [4], as well as analytic methods [5]. In this work

we will use numerical simulation to tackle the more general problem of manipulating arbitrarily-shaped, elastically-deforming objects based on high-level user-provided specifications.

The use of physical simulation to guide the manipulation of non-rigid objects is starting to be explored for a variety of problems. We refer the reader to [6] for a survey. Recent works demonstrate the predictive power of numerical simulations, accomplishing tasks such as mechanical assembly with a deformable ring [7], folding of clothes [8], and liquid pouring [9]. However, the computational cost of most existing approaches typically requires at least parts of the optimization process to be run offline [7], [8], [9], or necessitates the use of heuristics [9]. To avoid the computational burden, other works on elastic object manipulation avoid high-fidelity models altogether. For instance [10] uses computer vision to estimate simple linear models on the fly. Learning and demonstration-based approaches show promise in the context of this challenging problem domain as well [11], [12], but they can often still benefit from accurate simulation models [13].

In this paper, we develop a new, computationally-efficient formulation that leverages finite element simulation and sensitivity analysis to endow robots with the knowledge required to manipulate complex, elastically-deforming objects. The only assumption made by our model is that the motion of the elastically-deforming objects being manipulated is quasi-static. Our mathematical formulation differs from recent related work (e.g. [7], [8], [9]) in two main respects. First by applying sensitivity analysis in a similar fashion to [14], we are able to dramatically speed up the generation of control signals without compromising the predictive power of the underlying simulation models. This yields an approach that can be run in real-time as an interactive control method for practical problem sizes. Second, the variables of our optimization are joint angles as opposed to end effector positions. This fact has two benefits: First it makes it simple to incorporate objectives and constraints related to the robot’s configuration, e.g. joint limits or collision avoidance. Second, it operates directly within the configuration space of the robots. This eliminates the need for workspace approximations and subsequent projection steps, which could cause deterioration of results or even failure.

Succinctly, our contributions are:

- A general, computationally-efficient method for interactively controlling the shape of elastic objects with a general-purpose robot.
- Multiple demonstrations of elastic objects being dexterously manipulated by a YuMi<sup>®</sup> IRB 14000.

<sup>1</sup> Department of Computer Science, ETH Zurich, Switzerland. s.duenser@gmx.at, jamesmbern@gmail.com, roi.poranne@inf.ethz.ch, scoros@gmail.com

<sup>2</sup> Department of Computer Science, University of Haifa, Israel

We present our physical simulation model in Section II, our control framework in Section III, describe the interaction modalities available in our graphical user interface in Section IV, and present our results in Section V.

## II. SIMULATION MODEL

In this section we derive a computationally efficient control model for the manipulation of elastic objects using typical robots. To this end, we couple a simple link and joint representation of the robot to a finite element model (FEM) of an elastic object.

The configuration of a fully-actuated robot with  $m$  independently controlled joints is defined by joint angles  $\theta \in \mathbb{R}^m$ . The shape of a finite element mesh with  $n$  nodes can be written as a vector  $\mathbf{x} = [\mathbf{p}_1^\top, \dots, \mathbf{p}_n^\top]^\top$ , where  $\mathbf{p}_i \in \mathbb{R}^3$  is the position of the  $i$ -th node. We often refer to  $\mathbf{x}$  as the *pose* of the mesh.

We assume the overall system is quasi-static. Given robot joint angles  $\theta$ , our model computes a corresponding statically stable pose  $\hat{\mathbf{x}}(\theta)$  of the elastic object. This is done by using Newton's method to minimize the total energy of the system  $E = E(\mathbf{x}; \theta)$  over the space of nodal positions.

$$\hat{\mathbf{x}}(\theta) = \arg \min_{\mathbf{x}} E(\mathbf{x}; \theta) \quad (1)$$

The first order condition for a minimum has that  $\frac{\partial E}{\partial \mathbf{x}}(\hat{\mathbf{x}}(\theta); \theta) = \mathbf{0}$ . Recall that the nodal forces  $\mathbf{F} = [\mathbf{f}_1^\top, \dots, \mathbf{f}_n^\top]^\top$  are given by  $\mathbf{F} = -\frac{\partial E}{\partial \mathbf{x}}$ , where  $\mathbf{f}_i \in \mathbb{R}^3$  is the force acting on the  $i$ -th node. Therefore we know that a solution of Equation (1) also satisfies the equation  $\mathbf{F}(\hat{\mathbf{x}}(\theta); \theta) = \mathbf{0}$ . So  $\hat{\mathbf{x}}(\theta)$  adheres the typical definition of static equilibrium, i.e. that all forces are zero.

The total energy of the system is defined as the sum of the *FEM deformation energy*  $E_{\text{FEM}}(\mathbf{x})$  and a term  $E_{\text{grip}}(\mathbf{x}; \theta)$  that we call the *grip energy*. The FEM deformation energy models the energy that is stored when the elastic material is deformed in order to achieve a certain objective. The grip energy models the physical connection between the grippers of the robot and the contact region on the elastic object.

### A. The FEM energy

Our approach to computing the FEM energy is the same as in [14], and we summarize it here. We model the elastic object as a tetrahedral finite element mesh, composed of linear finite elements with a compressible neo-Hookean material model, to strike a balance between simulation accuracy and computational cost. We note that our modeling approach is agnostic to the specific material model used, so long as the material model has well defined second derivatives.

We denote the *rest pose* of the finite element mesh as  $\mathbf{X}$ , while  $\mathbf{x}$  is the *current pose*. The deformation energy density of each element is defined using a compressible Neo-Hookean material model,

$$\Psi(\mathbf{x}, \mathbf{X}) = \frac{\mu}{2} \text{tr}(\mathcal{F}^\top \mathcal{F} - I) - \mu \ln J + \frac{\kappa}{2} (\ln J)^2, \quad (2)$$

where  $\mathcal{F}$  is the deformation gradient,  $\mu$  and  $\kappa$  are material parameters,  $I$  is the identity matrix and  $J = \det(\mathcal{F})$ .

Our assumption of linear finite elements means that the deformation gradient is constant across each element. This in turn implies that the energy density is constant across each element as well. Therefore the energy stored in a given element is simply its energy density multiplied by its volume. The total deformation energy  $E_d(\mathbf{x})$  stored in the mesh is computed by summing up the energy stored in all elements.

In addition to the deformation energy, we also account for the gravitational potential energy, which is given by

$$E_g(\mathbf{x}) = \mathbf{m}^\top g \mathbf{h}, \quad (3)$$

where  $\mathbf{m}$  is a vector containing the mass associated with each node,  $g$  is the gravitational constant, and  $\mathbf{h} = \mathbf{h}(\mathbf{x})$  is a vector containing the height of each node. The total FEM energy is then

$$E_{\text{FEM}}(\mathbf{x}) = E_d(\mathbf{x}) + E_g(\mathbf{x}). \quad (4)$$

### B. The grip energy

In order to express the notion of a typical robot gripping an elastic object, we formulate an energy that relates the position and orientation of the end effectors to the positions of gripped nodes in the elastic object simulation mesh. The specific robot we will use in this work is a YuMi<sup>®</sup> IRB 14000, which consists of a pair of independently controlled articulated robotic arms, but we note that our method is general to any robot with defined joint angles.

With  $\theta$  being the vector of the robot's joint angles, the position and orientation of each *end effector*  $e$  is given by the forward-kinematics function  $T_e(\theta) \in SE(3)$ . Initially, the overall system (i.e. the robot and the elastic object it is gripping) is in a known *reference configuration*, with  $\theta^0$  and  $\mathbf{x}^0$ , or  $\mathbf{p}_i^0$  respectively. Each end effector  $e$  is assumed to be gripping a predetermined region on the surface of the object, which corresponds to a subset of *gripped nodes*  $G^e$  in the finite element mesh. We wish to force the positions of these gripped nodes to match the positions of the end effectors predicted by the forward kinematics. This can be expressed by the following *grip energy*

$$E_{\text{grip}}(\mathbf{x}; \theta) = \frac{1}{2} K \sum_{e} \sum_{\mathbf{p}_i \in G^e} \left\| \mathbf{p}_i - T_e(\theta) [\mathbf{p}_i^0] \right\|^2, \quad (5)$$

which corresponds to a spring energy with stiffness  $K$ , where we choose  $K$  sufficiently large to approximate a rigid coupling of the nodes.

## III. CONTROL FORMULATION

We leverage the model derived in the previous section to enable interactive shape control of an elastic object. We propose a modular optimization framework for shape control, which easily incorporates additional goals for collision avoidance and joint limits.

During a control session with our system, the user interactively specifies a *target shape*  $\mathbf{x}' = [\mathbf{p}'_1{}^\top, \dots, \mathbf{p}'_n{}^\top]^\top$ , where  $\mathbf{p}'_i \in \mathbb{R}^3$  is the target position of the  $i$ -th node. This target shape is incorporated into the *control objective*  $\mathcal{O} = \mathcal{O}(\mathbf{x}, \theta; \mathbf{x}')$ . This objective encodes the goal for the

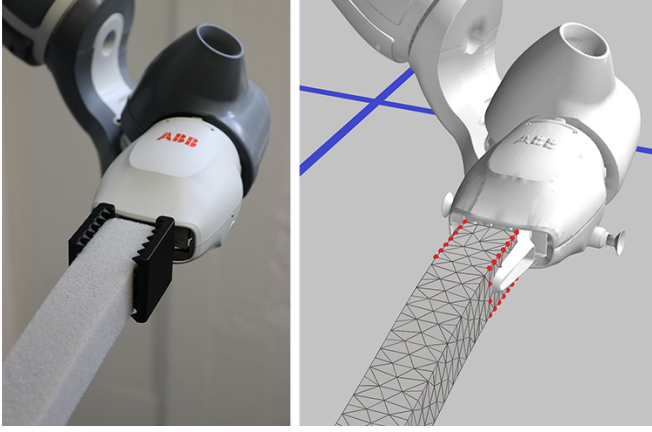


Fig. 1. The real-world YuMi<sup>®</sup> IRB 14000 gripping an elastic object with custom 3D-printed grippers (left). The corresponding simulated robot (rendered with standard grippers) and finite element mesh, with gripped nodes highlighted in red (right).

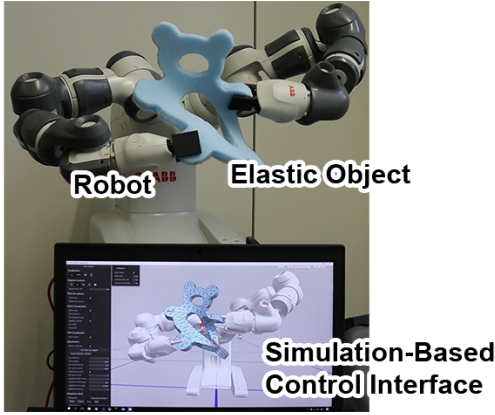


Fig. 2. An overview of our system. We simulate the real-world robot and the elastic object it is manipulating. The user poses the simulated elastic object in the space of robot-achievable shapes. Optimal joint angles are sent to the real-world robot in real time.

shape of the elastic object  $x$  to approximate the target shape  $x'$ , as well as any additional goals written in terms of  $x$ ,  $\theta$  and  $x'$ .

Recall that our system is assumed to be quasi-static. A choice of joint angles  $\theta$  implies a corresponding statically stable pose  $\hat{x}(\theta)$ . To score the overall statically stable system implied by  $\theta$ , we define a *reduced form* of the control objective  $\hat{\mathcal{O}}(\theta; x') = \mathcal{O}(\hat{x}(\theta), \theta; x')$ , which evaluates the original objective at  $\hat{x}(\theta)$ . We use a quasi-Newton method to minimize this reduced control objective over the space of joint angles.

$$\theta^*(x') = \arg \min_{\theta} \hat{\mathcal{O}}(\theta; x') \quad (6)$$

This yields optimal robot joint angles  $\theta^*(x')$ , which deform the elastic object such that it approximates the user-specified target shape.

The specific control objective we use in this work is defined as the sum of the *shape objective*  $\mathcal{O}_{\text{shape}}(x; x')$ , the *collision avoidance objective*  $\mathcal{O}_{\text{collisions}}(\theta)$ , and the *joint limits objective*  $\mathcal{O}_{\text{joints}}(\theta)$ . The shape objective measures

how close the mesh is to the target shape. The collision avoidance objective prevents the robot from colliding with itself. Finally, the joint limits objective enforces bounds on robot joint angles.

#### A. The shape objective

We encode the goal that the mesh pose  $x$  match the target shape  $x'$  into the *shape objective*

$$\mathcal{O}_{\text{shape}}(x; x') = \frac{1}{2} \sum_{p_i \in S} \|p_i - p'_i\|^2, \quad (7)$$

where  $S$  is the subset of mesh nodes for which the user chooses to specify a target position.

*Sensitivity Analysis:* Performing the optimization specified in Equation (6) requires the gradient  $\frac{d\hat{\mathcal{O}}_{\text{shape}}}{d\theta}$ , where  $\hat{\mathcal{O}}_{\text{shape}}(\theta; x') = \mathcal{O}_{\text{shape}}(\hat{x}(\theta); x')$  is the reduced form of the shape objective. This gradient is prohibitively expensive to estimate numerically. This is because  $\hat{\mathcal{O}}_{\text{shape}}$  is defined in terms of the statically stable pose  $\hat{x}(\theta)$ , which requires performing the nonlinear optimization in Equation (1) to evaluate. Motivated by this fact, we derive an analytic expression for the gradient.

We apply the chain rule to yield

$$\frac{d\hat{\mathcal{O}}_{\text{shape}}}{d\theta} = \frac{\partial \mathcal{O}_{\text{shape}}}{\partial x} \frac{d\hat{x}}{d\theta}, \quad (8)$$

and employ sensitivity analysis to solve for the Jacobian  $\frac{d\hat{x}}{d\theta}$ , which describes how changes in the robot's joint angles affect the corresponding statically shape of the elastic object. Recall from Section II that  $\hat{F}(\theta) = 0$ , where  $\hat{F}(\theta) = F(\hat{x}(\theta); \theta)$ . These are the forces on the elastic object *after* being deformed into statically stable shape  $\hat{x}(\theta)$  by a robot with joint angles  $\theta$ . They are always zero. Since  $\hat{F}(\theta)$  is always zero, the total derivative of  $\hat{F}$  with respect to  $\theta$  will be zero as well, giving us the expression

$$\frac{d\hat{F}}{d\theta} = \frac{\partial F}{\partial \theta} + \frac{\partial F}{\partial x} \frac{d\hat{x}}{d\theta} = 0. \quad (9)$$

This linear system can be solved for the Jacobian of interest  $\frac{d\hat{x}}{d\theta}$ , provided we have a means of computing the two other Jacobians in the equation. The first Jacobian  $\frac{\partial F}{\partial x} = -\frac{\partial^2 E}{\partial x^2}$  is the negative Hessian of the energy  $E$ , which we already compute to solve statics. To compute the remaining Jacobian  $\frac{\partial F}{\partial \theta}$ , we first recognize that  $E_{\text{grip}}$  is the only energy term that depends explicitly on the joint angles  $\theta$ , and so  $\frac{\partial F}{\partial \theta} = -\frac{\partial}{\partial \theta} \frac{\partial E_{\text{grip}}}{\partial x}$ . We substitute into the definition of the grip energy in Equation (5) and take the required partial derivatives. This yields the expression

$$\frac{\partial F}{\partial \theta} = \begin{bmatrix} \frac{\partial f_1}{\partial \theta} \\ \vdots \\ \frac{\partial f_n}{\partial \theta} \end{bmatrix}, \text{ with } \frac{\partial f_i}{\partial \theta} = \sum_e \begin{cases} K \frac{\partial T_e}{\partial \theta}(\theta) [p_i^0] & \text{if } p_i \in G^e \\ 0 & \text{otherwise,} \end{cases} \quad (10)$$

where  $\frac{\partial T_e}{\partial \theta}(\theta) [p_i]$  is a typical Jacobian used in manipulation describing how changes in the joint angles  $\theta$  affect the position of a point on end effector  $e$ .

## B. Collision Avoidance

It is imperative that the optimal joint angles returned by our approach imply a collision-free configuration of the robot. Collisions would either be between the two arms of the robot, or between two links of the same arm. We begin with a 3D model of the YuMi<sup>®</sup> IRB 14000, with individual surface meshes for each link. We then approximate each of these links with collision spheres (see Figure 3). Then, for

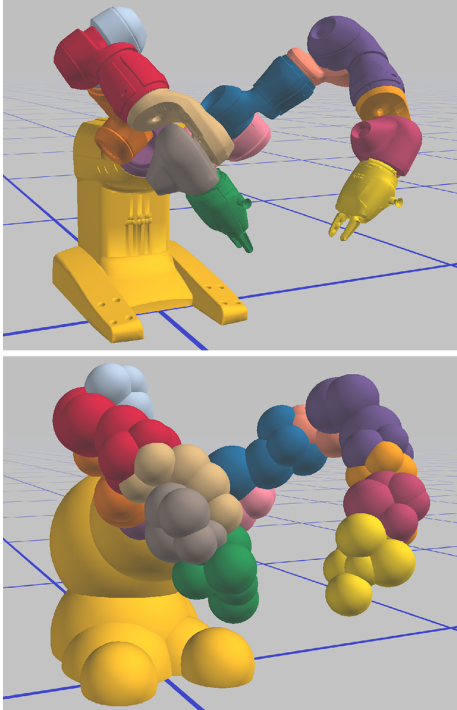


Fig. 3. The simulated YuMi<sup>®</sup> IRB 14000 with each rigid link in a different color (top). The corresponding collision spheres used in our optimization framework (bottom).

each pair of spheres in non-consecutive links, we add the following constraint to our optimization

$$|s_i - s_j| \geq r_i + r_j, \quad (11)$$

where  $s_i, s_j$  are the center positions of the spheres, and  $r_i, r_j$  are the radii.

We treat these constraints as soft so that they can be incorporated into our overall unconstrained optimization for  $\theta^*$ . For each collision constraint we add a *barrier function* to  $\mathcal{O}_{\text{collisions}}$ . This is a function that vanishes when the constraint holds, and rapidly increases as the constraint is violated. Our particular choice of barrier function is the  $\mathcal{C}^2$  one-sided quadratic explained in [14].

## C. Joint Limits

Since we are optimizing over joint angles, as opposed to e.g. end effector positions, it is simple to incorporate joint limits into our approach. Joint limits are constraints of the form

$$\underline{\theta} \leq \theta \leq \bar{\theta}, \quad (12)$$

where  $\underline{\theta}$  is the lower limit on joint angle  $\theta$ , and  $\bar{\theta}$  is the upper limit. We replace each such pair of joint limits with two barrier functions, one penalizing  $\theta < \underline{\theta}$  and the other penalizing  $\theta > \bar{\theta}$ .

## IV. GRAPHICAL USER INTERFACE

The user can prescribe target shapes for the elastic object in real-time through a simple graphical user interface. The primary interaction modality is to specify target positions for nodes in the mesh using the mouse, as shown in Figure 4.

For the special case of elastic deformable linear objects (DLOs) our system has an additional input modality, in which the user sketches a target center-line for the entire DLO (Figure 5).

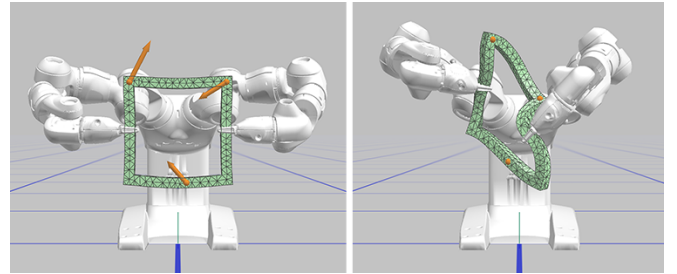


Fig. 4. The user specifies target positions for several nodes in the simulation mesh of the frame, rendered as the orange arrows (left). The result of our optimization with those target positions (right).

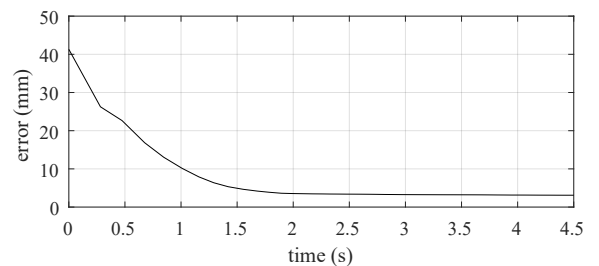
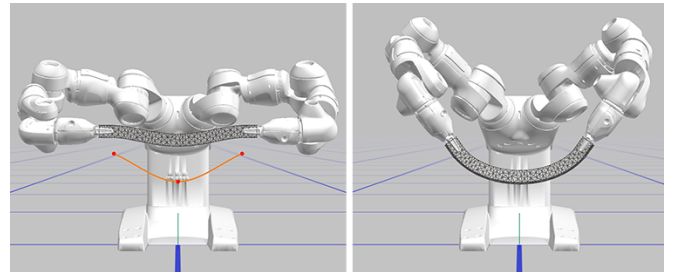


Fig. 5. The user specifies a target center-line for the simulation mesh of an initially-straight 320mm foam bar, rendered as the orange curve (top-left). The result of running our optimization with that target center-line (top-right). A plot showing the convergence of this example over time (bottom). The error is the simulated average euclidean distance of nodes subject to a shape objective from their target position.

## V. RESULTS

We outfit a YuMi<sup>®</sup> IRB 14000 with custom 3D-printed fingers for gripping soft objects. A posing session begins with the robot gripping the elastic object in a reference

configuration (e.g. top-left sub-figure of Figure 5). The simulation model described in Section II is initialized in the same configuration, and the gripped nodes are set to match the region of the surface gripped by the real-world robot.

The user is then free to interactively specify targets using the simple drag and drop GUI described in Section IV. The optimization from Section III is run continuously, and the current best joint angles are sent to the real-world YuMi<sup>®</sup> IRB 14000 in real-time (Figure 2). In Figure 6 we illustrate that the optimal joint angles found by our control methodology carry over well from simulation to the real-world. More results can be seen in Figure 9. We refer the reader to our video for footage of actual posing sessions, which further show how the real-world robot and elastic object track their simulated counterparts.

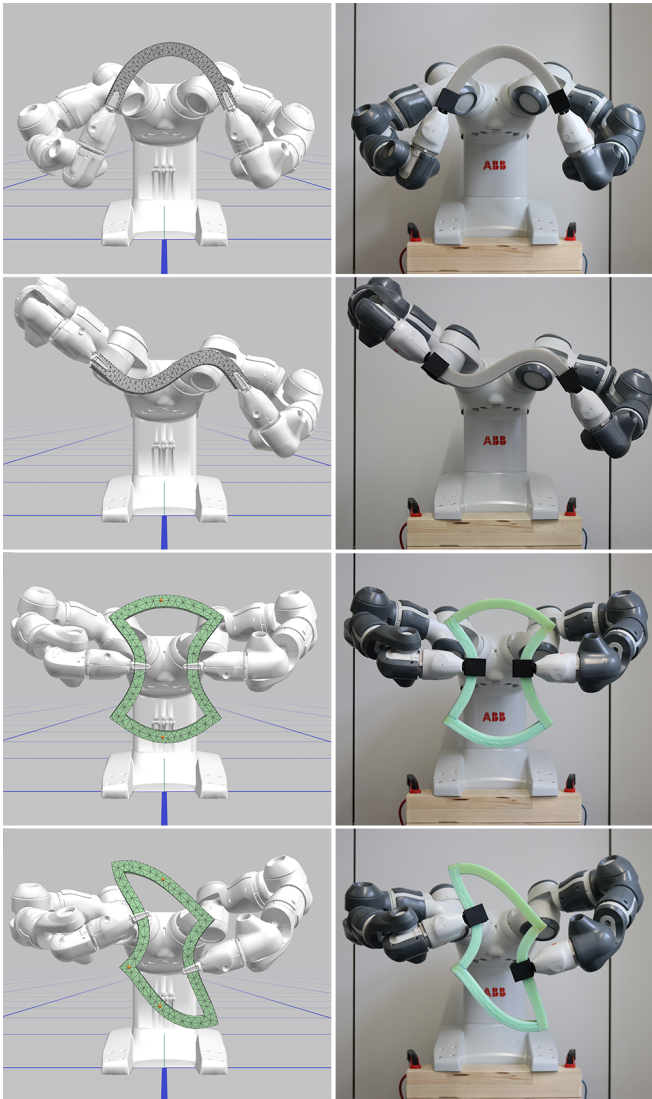


Fig. 6. Optimized joint angles running on the simulated robot and finite element mesh (left). The same joint angles running on the real-world robot.

### A. Performance

The performance boost gained by employing the sensitivity analysis to compute the Jacobian  $\frac{\partial x}{\partial \theta}$  is considerable. For

the bar mesh from Figure 5, which contains 1472 elements, we observe a 8.2x speedup compared to estimating the Jacobian using a finite-difference approach.

We provide quantitative data on the convergence properties of our method in Figures 5 and 7. In Figure 5 we show the typical convergence behavior for our method, going from the reference configuration to the optimized configuration. Our method quickly makes progress, and reaches an optimal set of joint angles in under two seconds. This enables real-time interaction with the system. The computational times reported were obtained on an Intel<sup>®</sup> Xeon<sup>®</sup> CPU E3-1505M v6 running at 3.6GHz.

In Figure 7 we show a similar plot for a full interaction session. Over the course of 20 seconds the user is able to explore multiple different target poses for the elastic object, and corresponding optimized robot configurations.

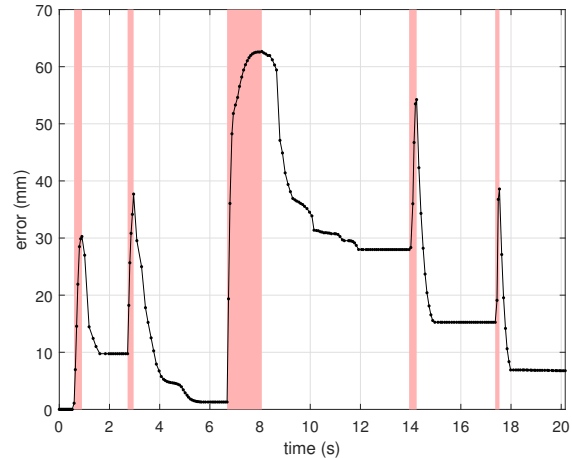


Fig. 7. A plot showing the error over the course of a typical 20 second control session, wherein a user is controlling the shape of an initially-straight 320mm foam bar. The optimization is run continuously. During the times indicated by red shading, the user is actively specifying target nodal positions. Following the specification of a target position the optimization converges to an optimal set of joint angles in approximately one second. The error is the simulated average euclidean distance of nodes subject to a shape objective from their target position.

## VI. DISCUSSION

### A. Abstractions for Shape Control

In this work the user is responsible for directly specifying target shapes for the finite element mesh. This is done either by specifying target positions for individual nodes, or for the special case of deformable linear objects by specifying a target center-line. One avenue of future work would be to extend the notion of specifying a target center-line to the case of nonlinear elastic objects, perhaps by leveraging the rich body of work on curve-skeletons [15]. Another avenue of future work would be to investigate higher-level manipulation goals, e.g. to command the robot to fold up a sheet without having to explicitly specify a sequence of target shapes.

## B. Limitations

Our approach is naturally limited by the capabilities of its finite element model. We do not currently consider self-collision of the deformable object, and also cannot guarantee that buckling will behave exactly as predicted. Indeed, for some configurations of the end effectors, multiple energy-minimal poses of the objects can exist (see Figure 8), and the one that manifests in the real world depends on the history of the deformation—along the dynamics of the system—neither of which we account for. Additionally, we do not consider regrasps, and assume that the gripper positions are specified from the beginning by the user. In the future, we could add visual feedback into our system, which would enable the ability to make corrections to the simulation in real-time.

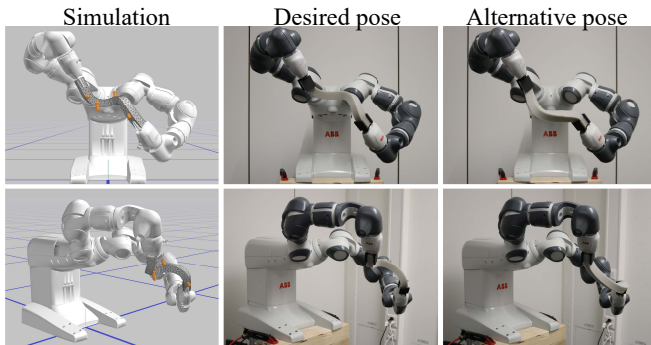


Fig. 8. In certain situations there could be multiple possible poses for the same end effector configuration. Our model is not guaranteed to predict the correct one. In this figure our model (left column) predicted the outcome on the middle column. However, The alternative pose on the right column also exist.

## VII. CONCLUSION

We presented a method for shape control of an elastic object with a standard robot. We began by deriving a physical simulation model, which coupled a general finite element model of the elastic object to the grippers of the robot. Our control methodology leveraged sensitivity analysis to speed up computation of the Jacobian relating changes in robot joint angles to changes in the shape of the elastic object. We demonstrated the use of our method by controlling the shape of various elastic objects with a YuMi<sup>®</sup> IRB 14000.

## ACKNOWLEDGMENT

We would like to thank Simon Zimmerman for his help.

## REFERENCES

- [1] K. Kosuge, H. Yoshida, T. Fukuda, M. Sakai, and K. Kanitani, “Manipulation of a flexible object by dual manipulators,” in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 1. IEEE, 1995, pp. 318–323.
- [2] F. Lamiraux and L. E. Kavraki, “Planning paths for elastic objects under manipulation constraints,” *The International Journal of Robotics Research*, vol. 20, no. 3, pp. 188–208, 2001.
- [3] M. Saha and P. Ito, “Manipulation planning for deformable linear objects,” *IEEE Transactions on Robotics*, vol. 23, no. 6, pp. 1141–1150, 2007.

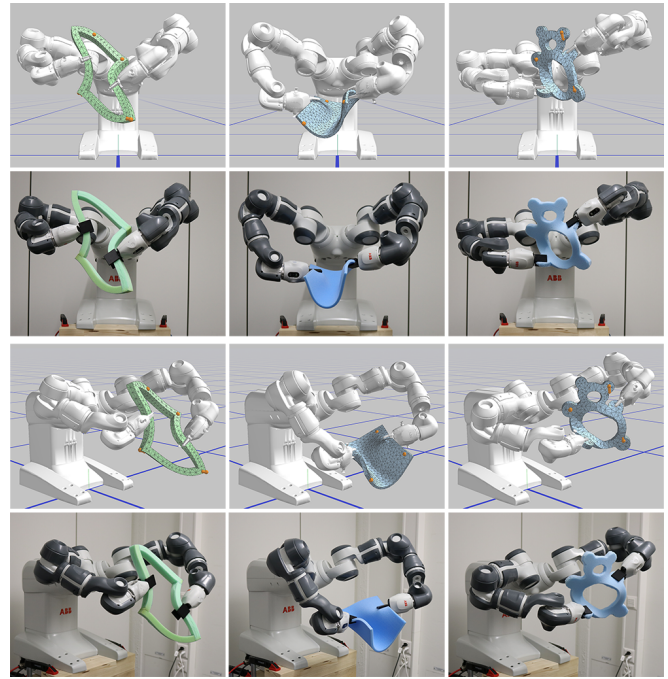


Fig. 9. More results interactively obtained using our system. Please refer to the video for footage of the actual posing sessions.

- [4] M. Moll and L. E. Kavraki, “Path planning for deformable linear objects,” *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 625–636, 2006.
- [5] T. Bretl and Z. McCarthy, “Quasi-static manipulation of a kirchhoff elastic rod based on a geometric analysis of equilibrium configurations,” *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 48–68, 2014.
- [6] P. Jiménez, “Survey on model-based manipulation planning of deformable objects,” *Robotics and computer-integrated manufacturing*, vol. 28, no. 2, pp. 154–163, 2012.
- [7] E. Yoshida, K. Ayusawa, I. G. Ramirez-Alpizar, K. Harada, C. Duriez, and A. Kheddar, “Simulation-based optimal motion planning for deformable object,” in *2015 IEEE International Workshop on Advanced Robotics and its Social Impacts, ARSO 2015, Lyon, France, June 30 - July 2, 2015*, 2015, pp. 1–6.
- [8] Y. Li, Y. Yue, D. Xu, E. Grinspun, and P. K. Allen, “Folding deformable objects using predictive simulation and trajectory optimization,” in *IEEE/RSJ IROS 2015*. IEEE/RSJ, 2015.
- [9] Z. Pan, C. Park, and D. Manocha, “Robot motion planning for pouring liquids.” [Online]. Available: <http://www.aaii.org/ocs/index.php/ICAPS/ICAPS16/paper/view/13009>
- [10] 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.
- [11] A. M. Howard and G. A. Bekey, “Intelligent learning for deformable object manipulation,” *Auton. Robots*, vol. 9, no. 1, pp. 51–58, Aug. 2000. [Online]. Available: <https://doi.org/10.1023/A:1008924218273>
- [12] M. Rambow, T. Schau, M. Buss, and S. Hirche, “Autonomous manipulation of deformable objects based on teleoperated demonstrations,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 2809–2814.
- [13] 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. PP, no. 99, pp. 1–18, 2018.
- [14] J. M. Bern, G. Kumagai, and S. Coros, “Fabrication, modeling, and control of push robots,” in *Proceedings of the International Conference on Intelligent Robots and Systems*, 2017.
- [15] N. D. Cornea, D. Silver, and P. Min, “Curve-skeleton properties, applications, and algorithms,” *IEEE Transactions on visualization and computer graphics*, vol. 13, no. 3, pp. 0530–548, 2007.