Sim-to-Real for Soft Robots Using Differentiable FEM: Recipes for Meshing, Damping, and Actuation

Mathieu Dubied, Mike Yan Michelis, Andrew Spielberg, and Robert Kevin Katzschmann, *Member, IEEE*

***Abstract*—An accurate, physically-based, and differentiable model of soft robots can unlock downstream applications in optimal control. The Finite Element Method (FEM) is an expressive approach for modeling highly deformable structures such as dynamic, elastomeric soft robots. In this paper, we compare virtual robot models simulated using differentiable FEM with measurements from their physical counterparts. In particular, we examine several soft structures with different morphologies: a clamped soft beam under external force, a pneumatically actuated soft robotic arm, and a soft robotic fish tail. We benchmark and analyze different meshing resolutions and elements (tetrahedra and hexahedra), numerical damping, and the efficacy of differentiability for parameter calibration using a simulator based on the fast Differentiable Projective Dynamics (DiffPD). We also advance FEM modeling in application to soft robotics by proposing a predictive model for pneumatic soft robotic actuation. Through our recipes and case studies, we provide strategies and algorithms for matching real-world physics in simulation, making FEM useful for soft robots.**

***Index Terms*—Modeling, control, and learning for soft robots, dynamics, optimization and optimal control, simulation and animation.**

I. Introduction

SOFT robotics promises the rise of a new generation of robots with built-in compliance, damping, elastic energy storage, continuous actuation, and other morphologically-encoded behavioral features.

However, one of the key challenges in realizing the potential of this field is to develop a simulation model that is useful for analyzing these effects and developing real-world controllers.

The Finite Element Method (FEM), when combined with highly stable implicit Euler integration, shows promise for real-world soft robotic models due to its speed and accuracy. Furthermore, the differentiability of some FEM models [1] enables the fast and accurate computation of gradients for efficient optimization of a robot's design and control.

Manuscript received September 9, 2021; accepted January 27, 2022. Date of publication February 24, 2022; date of current version March 8, 2022. This letter was recommended for publication by Associate Editor C. Duriez and Editor C. Laschi upon evaluation of the reviewers' comments. This work was supported by a generous gift from Credit Suisse to the ETH Foundation.

Mathieu Dubied, Mike Yan Michelis, and Robert Kevin Katzschmann are with the Soft Robotics Lab, Department of Mechanical and Process Engineering, 8092 ETH Zurich, Switzerland (e-mail: mdubied@ethz.ch; mike.michelis@srl.ethz.ch; rkk@ethz.ch).

Andrew Spielberg is with the Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139 USA (e-mail: aespielberg@csail.mit.edu).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2022.3154050, provided by the authors.

Digital Object Identifier 10.1109/LRA.2022.3154050

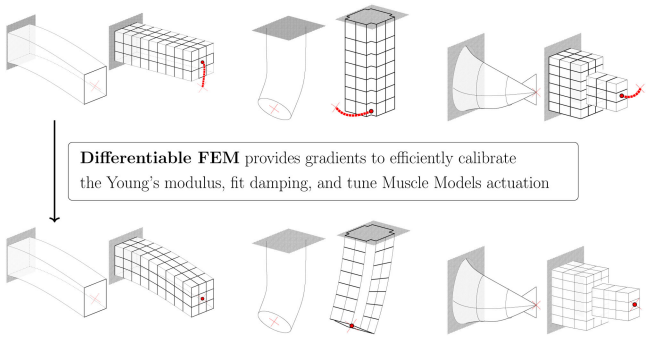


Fig. 1. We show that differentiable FEM can be used to improve models and decrease modeling error, bringing simulation closer to reality.

This approach, however, introduces several challenges when applied to soft robotic modeling. Although the implicit Euler integration is unconditionally stable, it also causes significant numerical damping — a side effect that causes simulated structures to exhibit mechanical damping, even when mechanical damping is not an explicit feature of the material model. Moreover, despite FEM being based on continuum mechanics, it attempts to approximate continuous phenomena *via* discretization. The choice of this discretization can have serious ramifications on the model's accuracy. These challenges must be addressed for FEM to be useful to the robotics community.

In this paper, we evaluate the ability of dynamic FEM to model deformable soft robots with fluidic actuation. In particular, we employ Projective Dynamics (PD), an FEM formulation that can be made efficiently differentiable. We rely on the differentiable nature of the Differentiable Projective Dynamics (DiffPD) framework [1] to overcome the sim-to-real gap. Our three contributions toward this goal are:

1) *Benchmarking FEM (in particular, PD) against reality for different meshes*. We perform characterizing experiments on three different soft robotic structures, and compare the measurement data to the PD simulations. We consider different mesh resolutions and two different mesh element types (tetrahedra and hexahedra). We also benchmark the obtained results against a commercial FEM solution.
2) *Characterizing and adapting numerical damping to match material damping in reality*. We characterize the numerical damping, which originates from the implicit Euler scheme. We benchmark this numerical damping against an analytical solution we derive. We then propose a new method to adjust the numerical damping so that it matches the material damping.
3) *Creating FEM models with soft robotic actuation matching reality*. We compare PD simulations to reality for geometrically complex soft robotic actuators. Since a precise

model of the pressure chambers leads to unsatisfactory results in terms of speed and accuracy, we present "Muscle Models" as an alternative means of modeling fluidic chambers.

These contributions address the most challenging aspects when simulating robots using FEM and transferring them to reality. Thus, we provide a guide for practitioners to quickly simulate and optimize real-world soft robots in the future.

II. Related Work

In this section, we begin by describing existing methods for simulating soft robots. We then summarize related progress toward overcoming their sim-to-real gap.

A. Soft Robotic Simulation

Many approaches for accurately simulating soft robots have been proposed over the years. One approach is to use a model based on simplifying assumptions, such as the Augmented Rigid Body model [2] that is based on the Piecewise Constant Curvature model [3]. A second approach is to develop data-driven models that are based on measurements that have been performed on the robot [4]–[6]. A third approach is to employ models that discretize the robot's continuum mechanics and solve the equations of motions for every element of the discretization [7].

In this work, we focus on discretization approaches, specifically FEM. In the field of soft robotics, the Simulation Open Framework Architecture (SOFA) is widely used. SOFA accurately models highly deformable structures, and it can be used as a basis for fast simulations that use both traditional and reduced order simulation methods [7]–[9]. Other promising simulation frameworks have recently emerged, such as the DiffPD framework [1], which is based on PD [10]. PD employs a projection-based implicit Euler method to solve the equations of motion and to achieve more stable and efficient simulations than explicit schemes (although this comes at the expense of accuracy). It has been optimized for performance over recent years [11]–[13]. DiffPD, and other simulators [14], [15], can even be designed to handle contact both accurately and efficiently.

Notably, DiffPD extends PD to the family of simulators that are *differentiable*. Differentiable simulators allow for model-based computation of gradients of any variable (state, model, or control) in a system with respect to any other variable. Differentiable simulators have been used to efficiently solve control and design optimization tasks in soft-body regimes [15]–[18].

B. Sim-to-Real for Soft Robotics

As the field of soft robotics has grown, so too has interest in translating simulated results to the physical world. In the case of linear manipulators such as arms, simplified models have been effective. For example, soft robots modeled as rigid links with torsional spring joints have been used to motion plan soft robotic arms through real-world maze-like environments [19], [20]. More complex PCC models have also been used to create dynamic soft robotic arms with optimal control [2], which translate faithfully to reality.

Finite element models have long garnered special attention, however, due to their ability to scale to more complex geometry. COMSOL, Abaqus, and other commercial and open-source FEM packages have been used for the modeling of soft structures and robots. These programs have been particularly useful for designing physical soft robotic structures [21], [22]. SOFA [23] have successfully been deployed in a number of applications of kinematic control of real-world soft robots, including real-time control [24]–[26]. Similar work has demonstrated dynamic analysis and control of soft robots by relying on reduced order models for real-time tractability [9], [27]. Other relevant work includes demonstrations of trajectory optimization and control on physical soft walking robots and grippers [28], [29]. While the above listed work takes steps to faithfully model soft robots and compare physical and virtual performance, they do not investigate systematic approaches and analyses needed in order to put the virtual and physical models into correspondence; this step is especially important but also difficult when dealing with dynamic robots, and it is often not treated in-depth.

The research works most similar to ours (namely, the works that take systematic approaches to overcoming the sim-to-real gap) include [30] and [31]. A data-driven approach to building complex voxel-based soft robots [30] tunes parameters for simpler designs and scales up complexity using a spring-based simulator [32]. Repeatedly applying system identification of model *parameters* in the DiffPD framework improves the performance of robotic swimmers, but does not address issues inherent to the model itself [31]. Our work analyzes the important nuances of soft robotic modeling, specifically in a PD framework, examining the effects of design choices such as meshing, material and numerical damping, and actuation models.

III. Theoretical Background

A. Soft Robotic Finite Element Model

PD, like other classical FEM solvers, solves the PDEs specified by continuum mechanics through spatial and time discretization.

1) Spatial Discretization: Starting from a continuous body with an infinite number of DOFs, the spatial discretization in finite elements (FE) reduces the system's dimensionality to a tractable, finite number. The shape of the body is approximated by a mesh of finite elements, each spanned by n nodes. For volumetric elements in 3D space, each node has 3 translational DOFs and is described by its position and velocity vectors through Newton's second law:

$$\boldsymbol{M}\dot{\boldsymbol{v}} = \boldsymbol{f}_{\text{ext}} + \boldsymbol{f}_{\text{int}}(\boldsymbol{q}) \tag{1}$$

where $\boldsymbol{q} = \boldsymbol{q}(t) \in \mathbb{R}^{3n}$ describes the position of every node, $\boldsymbol{v} = \boldsymbol{v}(t) \in \mathbb{R}^{3n}$ describes the nodes' velocities, $\boldsymbol{M} \in \mathbb{R}^{3n\times 3n}$ is the mass matrix, $\boldsymbol{f}_{\text{ext}} = \boldsymbol{f}_{\text{ext}}(t) \in \mathbb{R}^{3n}$ accounts for external forces, and $\boldsymbol{f}_{\text{int}}(\boldsymbol{q}) = \boldsymbol{f}_{\text{int}}(\boldsymbol{q}, t) \in \mathbb{R}^{3n}$ accounts for internal forces. We note that the formulation for PD's internal forces does not include system velocities; thus, velocity-dependent forces such as damping forces are modeled as *external* forces (see Section IV-B).

2) Time Discretization: PD and classic FEM solvers consider (1) for discrete time steps by, for example, using the implicit Euler method. The implicit Euler method approximates the derivative of a function $u_i = u(x_i, t_i)$ as

$$\frac{\partial u}{\partial t}(x_i, t_i) = \dot{u}(x_i, t_i) \approx \frac{u_{i+1} - u_i}{h} \tag{2}$$

where h is the time step defined as $h = t_{i+1} - t_i$ and $i = 0, 1, 2, \ldots$. Applying the implicit Euler method to (1), we obtain

the following discretized equations of motion:

$$\begin{aligned} \boldsymbol{q}_{i+1} &= \boldsymbol{q}_i + h\boldsymbol{v}_{i+1} \\ \boldsymbol{v}_{i+1} &= \boldsymbol{v}_i + h\boldsymbol{M}^{-1}(\boldsymbol{q})[\boldsymbol{f}_{\text{int}}(\boldsymbol{q}_{i+1}) + \boldsymbol{f}_{\text{ext}}] \end{aligned} \tag{3}$$

B. Projective Dynamics

1) Optimization Formulation: PD formulates (3) as an optimization problem to solve this equation efficiently:

$$\min_{\boldsymbol{q}_{i+1}} \left\{ \frac{1}{2h^2} ||\boldsymbol{M}^{\frac{1}{2}}(\boldsymbol{q}_{i+1} - \boldsymbol{y}_i)||_2^2 + E_{\text{int}}(\boldsymbol{q}_{i+1}) \right\} \tag{4}$$

with $\boldsymbol{f}_{\text{int}}(\boldsymbol{q}_{i+1}) = -\nabla E_{\text{int}}(\boldsymbol{q}_{i+1})$ and $\boldsymbol{y}_i = \boldsymbol{q}_i + h\boldsymbol{v}_i + h^2\boldsymbol{M}^{-1}\boldsymbol{f}_{\text{ext}}$. The first term is the momentum potential energy and the second term E_{int} is the elastic potential energy.

2) Elastic Energy Formulation: In nonlinear continuum mechanics, the elastic energy $E_{\text{int}}(\boldsymbol{q}_{i+1})$ is a nonlinear function that serves as a restorative spring-like potential energy that attracts each state to its undeformed state. Such potential energies can yield linear or nonlinear spring-like forces. Some common potential energies for modeling are described in [33]. PD introduces an elastic energy in a form that is convenient for efficient computation. The elastic energy's form is introduced in [34] and is also used by DiffPD [1].

C. Material Damping

Damping is a phenomenon that reduces the internal velocity of a system by dissipating energy. It is commonly associated with amplitude oscillatory responses.

With *material* damping, we refer to the damping scenario in which the source of damping is physical, and not numerical. Numerical damping is discussed in Section III-D.

1) Single DOF System and Damping Ratio: The prototypical equation of motion of a single DOF system undergoing free vibration can be expressed as

$$\ddot{x} + 2\zeta\omega_0\dot{x} + \omega_0^2 x = 0 \tag{5}$$

with damping ratio ζ and undamped frequency ω_0.

We can classify the solutions $x = x(t)$ of (5) depending on the value of the damping ratio ζ. Important for this paper are underdamped vibrations, where $0 \leq \zeta < 1$.

2) Underdamped Vibrations: In the case of underdamped vibrations, the solution of (5) can be written as

$$x(t) = e^{-\zeta\omega_0 t}\left[C_1\cos(\omega_d t) + C_2\sin(\omega_d t)\right] \tag{6}$$

where $\omega_d = \omega_0\sqrt{1-\zeta^2} < \omega_0$ is the damped natural frequency.

As an inverse problem, it is also possible to determine ζ based on data. We first need to compute the logarithmic decrement δ, which is defined as follows:

$$\delta = \frac{1}{m}\cdot\ln\left(\frac{X_k}{X_{k+m}}\right) \tag{7}$$

where X_k is the k-th peak amplitude, and X_{k+m} is the amplitude of the peak m periods afterward. In this paper, we consider $m \in \{1,2,3,4\}$ depending on the time step h of our simulations. From the logarithmic decrement δ, one can determine the damping ratio ζ:

$$\zeta = \frac{\delta}{(4\pi^2 + \delta^2)^{1/2}} \tag{8}$$

D. Numerical Damping

Unlike material damping which occurs due to physical phenomena (such as viscous drag, heat exchange, etc.), numerical damping occurs because of the time discretization method we use. The implicit Euler method, used in PD, is known to be A-stable but it also induces numerical damping [35], [36].

We illustrate this fact through a simple example.

We first define the following linear system:

$$\begin{cases} \frac{dx}{dt} = i\omega x(t), \quad \omega \in \mathbb{R},\ t \in [0, +\infty) \\ x(0) = x_0 = 1 \end{cases} \tag{9}$$

The exact solution of this equation is an oscillation whose amplitude does not change over time:

$$x(t) = e^{i\omega t} = \cos(\omega t) + i\sin(\omega t) \tag{10}$$

The solution obtained using the implicit Euler method is

$$\frac{x_{j+1} - x_j}{h} = i\omega x_{j+1} \ \Rightarrow\ x_{j+1} = \frac{1}{1 - hi\omega}x_j, \text{ for } j \in \mathbb{N} \tag{11}$$

Using $x_0 = 1$, we can rewrite this relation as

$$x_j = \left(\frac{1}{1-hi\omega}\right)^j \cdot x_0 = \left(\frac{1}{1-hi\omega}\right)^j \tag{12}$$

We can observe that for a finite value of h and $\omega \in \mathbb{R}^+$, the amplitude of the oscillation is decreasing over time:

$$|x_j| = \left|\left(\frac{1}{1-hi\omega}\right)^j\right| \to 0 \quad \text{for } j \to \infty \tag{13}$$

This phenomenon is called *numerical* damping, as the source of the damping is purely an artifact of the numerical method that is used for time discretization.

IV. Modeling: Meshing, Damping, and Actuation

A. Meshing

In this work, we investigate two FE types for meshing: hexahedra and tetrahedra. Hexahedral meshes are traditionally locked to axis-aligned voxel grids in their rest configuration. This leads to aliasing when one attempts to fit them to curved or angled surfaces. However, while more geometrically accurate, meshes based on linear tetrahedra suffer from *locking* when used to model (nearly) incompressible materials ($\nu \to 0.5$) [37] or when applied to systems with highly nonlinear dynamical behavior [38]. These meshes yield a much stiffer behavior of the simulated structure than the real equivalent.

B. Damping

Numerical damping is inherently present when using the implicit Euler method (as in PD) and can be quite large (cf. Section VI-A). This can lead to large error in the simulated dynamic motion. In order to compensate for this error and fit the dynamic behavior of real structures, we model velocity-dependent state forces that correct for numerical damping:

$$f_i^l = \Lambda \cdot v_{i-1}^l \tag{14}$$

where $\Lambda \in \mathbb{R}$ is a tuning parameter and l the vertex to which the force is applied. We use the simulated velocity v_{i-1}^l of the previous time step to compute the current state force f_i^l. We

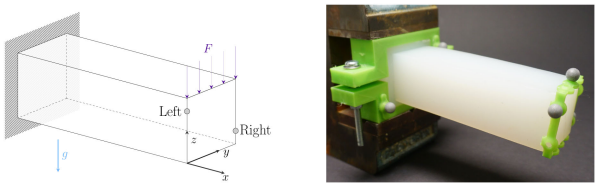
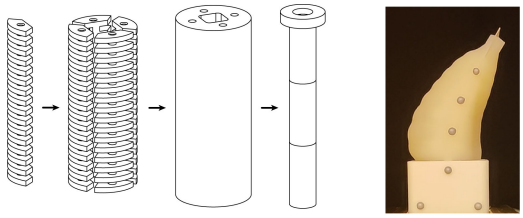


Fig. 2. Schematic with edge forces (left) and a photo (right) of the setup, including motion markers, for the clamped beam scenario.



(a) Soft Arm (b) Soft Fish

Fig. 3. (a) Arm segment with ribbed pressure chambers [27]. (b) Soft fish tail.

analyse in Section VI-A the effect of this force on the stability of the system. If the value of Λ is correctly chosen, the simulation results remain stable.

C. Actuation

We design "Muscle Models" in order to predict the behavior of elastic pneumatics. As will be shown, these are a helpful alternative to the exact modeling of pressure chambers. We employ a muscle energy [1], [39] which adds an additional energy term to E_{int} in (4); this term can be understood as a spring-like energy that is attributed to the mesh's elements:

$$E(\boldsymbol{q}) = \frac{w}{2}||(1-a)\boldsymbol{F}\boldsymbol{m}||^2 \tag{15}$$

where w can be interpreted as the stiffness of the spring, $a \in \mathbb{R}$ is the actuation signal, $\boldsymbol{F}$ is the deformation gradient, and $\boldsymbol{m}$ is the direction of the actuation. If $a > 1$, extension occurs; if $a < 1$, contraction occurs. While springs-like models are not pneumatic models as derived from first principles, they are much simpler to model in a PD simulator, and, as we demonstrate in section Section VI-B can be tuned to provide dynamic behavior that closely matches reality.

V. Experimental and Simulation Setups

A. Experimental Setup

In this section, we discuss the three deformable structures that will serve as a testbed for our analysis.

1) Clamped Beam Under External Force: To benchmark the accuracy of PD, we begin by considering a simple, yet important structure: a clamped beam made of a highly deformable silicone elastomer (Fig. 2).

We consider two scenarios: *1)* we keep the beam at its horizontal starting position and then release it so it bends downwards under gravity; and *2)* in addition to gravity, we apply an external force F along the edge of the tip. To track the two reference points ("Left" and "Right"), we use a Motion Capture System from Qualisys. See Fig. 2 for details.

2) Soft Robotic Arm: The second structure is a pneumatically actuated soft robotic arm made of a silicone elastomer.

Our arm is comprised of a single segment of the soft robotic arm described in [27]. It has 4 pressure chambers, which have a complex geometry, as shown in Fig. 3. We place motion markers on the robotic arm to track the tip's center position.

3) Soft Fish Tail: The last structure we consider is a pneumatic soft fish tail (Fig. 3). We focus on the position of the last motion marker placed on the tail's spine.

TABLE I
Summary of the Loading Cases for the Clamped Beam

Case	Load Scenario	DOFs Hex	DOFs Tet	DOFs COMSOL
A-1	$F = 0$ N	4608	3834	7110
A-2	$F = 0$ N	4608	183516	7110
B	$F = 0.510$ N	4608	3834	7110
C	$F = 0.510$ N	9900	71688	7110
D	$F = 0.991$ N	4608	3834	7110
E	$F = 0.991$ N	9900	71688	7110

B. Simulation Setup

In this section we explain our simulation setup. For the remainder of this paper, unless specified otherwise, our simulations use a time step h of 0.01 s.

1) Material Parameters: The silicone elastomer used for the beam is Smooth-On Dragon Skin 10 with Shore Hardness 10 A. For silicones, we assume incompressibility, *i.e.*, $\nu = 0.5$ ($\nu = 0.499$ to avoid numerical singularities). The material density specified by Smooth-On is $\rho = 1070\ \text{kg m}^{-3}$.

Precise knowledge of the material properties is needed to achieve an accurate simulation. This is difficult with elastomers, which generally show large variability due to their fabrication process. We use COMSOL as a ground truth, but we will show in Section VI-A that DiffPD can also be used to calibrate material parameters using its differentiability.

We simulate Case E (see Table I) and tune the Young's modulus so that the static solution from COMSOL matches the real data. We choose Case E because the large load F is dominant in the final deformation state, and the effects of setup inaccuracies are negligible. This method leads to a Young's modulus of $E = 263824$ Pa. This value is similar to the values found in related literature [40], [41].

2) Material Models: We simulated our structures, experimenting with corotational linear elastic and Neo-Hookean material models [33]. In our experiments, both models yield similar results; thus we only present results obtained with the corotational linear elastic model, which is numerically simpler to employ.

3) Optimization Algorithm: When optimizing any variable α in DiffPD, the following steps are taken. First, we choose an initial guess for α randomly. Then, we choose a loss function $\mathcal{L}$ that mathematically specifies our objective. We then optimize α to match the target; specifically, we iteratively 1) simulate the trajectory of the soft structure given the current α, 2) use DiffPD to compute model-based gradients $\frac{\partial \mathcal{L}}{\partial \alpha}$, and 3) perform an optimization step to update the guess for α using L-BFGS. We repeat this process until convergence.

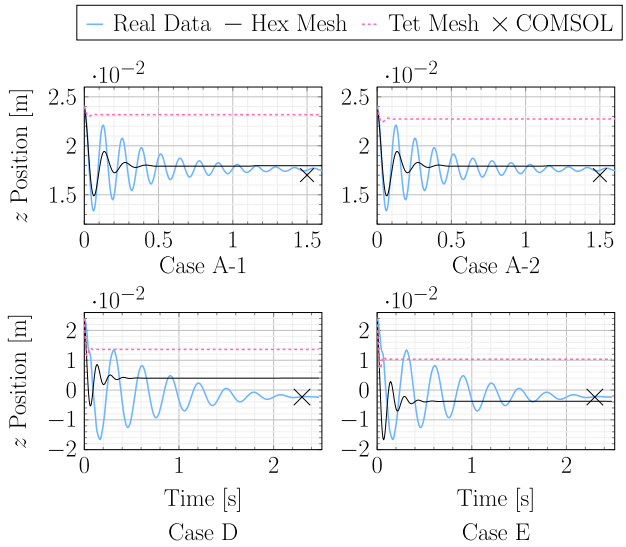


Fig. 4. Comparison of real data and simulated results using the Hex Mesh and the Tet Mesh, along with the COMSOL static solution.

VI. Results

A. Clamped Soft Beam Under External Force

Studying a clamped beam structure allows us to solve for certain variables and thus reduce the number of factors that influence more complex simulations. For this mechanical structure, we focus on the vertical deformation of the beam's tip (z axis). Here, we present the results for the point "Left" as defined in Fig. 2. The results for point "Right" are similar and therefore not shown here.

1) Meshing, Position Tracking, and Steady State Comparison: We experiment with *three* loading scenarios in this section (Table I). These scenarios were obtained by varying the edge force F . Additionally, we distinguish cases by their number of Degrees of Freedom (DOFs) and by the type of mesh elements that were used; namely, hexahedra ("Hex Mesh") or tethrahedra ("Tet Mesh").

For Case A-1, we find close correspondence between the real data, the COMSOL solution, and the Hex Mesh solution: all three steady state solutions are within 1 mm of each other. On the other hand, the Tet Mesh response is too stiff; it has 6 mm absolute error (Fig. 4 Case A-1).

In Case A-2, we verify if the Tet Mesh behavior is an artifact stemming from a poor choice of the mesh resolution. We perform the same simulation again while increasing the number of DOFs for the Tet Mesh from 3834 to 183516. As shown in Fig. 4 Case A-2, only a very small difference (0.438 mm difference) can be observed.

Keeping the same number of DOFs for the two mesh types as in Case A-1, we add an edge force at the tip of the beam for Case B and D. Under this configuration, the simulation yields incorrect results for both mesh types. However, increasing the number of DOFs of the Hex Mesh resolves this issue and leads to more accurate results. For the Tet Mesh, this refining step is not enough to elicit a sufficient response to match real data. We only present the results for Case D and E in Fig. 4, as Case B and C exhibit similar results. Since the DiffPD simulation for case E with the Hex Mesh is accurate, we can use the differentiability of the framework to calibrate the Young's Modulus. Following the steps described in Section V-B3 with $\mathcal{L} = \|\boldsymbol{q}_{\text{sim,final}} - \boldsymbol{q}_{\text{real,final}}\|^2$ and for different initial values, the average optimized Young's Modulus is $E = 283271$ Pa, which is close to the solution we found using COMSOL ($E = 263824$ Pa, Section V-B).

2) Numerical Damping by the Implicit Euler Method

a) Analytical Solution: As introduced in Section III-C, we can describe the damping of an oscillation using the damping ratio ζ. Here, we show an analytical relationship between the implicit Euler time step h and the induced numerical damping, characterized by the damping ratio ζ.

We consider the system (9), whose exact solution is the free undamped vibration (10). We can reformulate the solution obtained using the implicit Euler method (12) as:

$$x_n = \left(\frac{1}{1 - hi\omega}\right)^n = \frac{(1+\omega hi)^n}{(1+\omega^2 h^2)^n} = \frac{e^{n\cdot\angle(1+\omega hi)}}{(1+\omega^2 h^2)^{n/2}} \quad (16)$$

The values of the positive peaks of this oscillation can be found at step n^* as follows:

$$n^* \cdot \angle(1+\omega hi) = k \cdot 2\pi \;\Rightarrow\; n^* = \frac{k \cdot 2\pi}{\angle(1+\omega hi)},\; k \in \mathbb{N} \quad (17)$$

The amplitude of the oscillation at n^* is

$$X_k = \frac{1}{(1+\omega^2 h^2)^{n^*/2}} = \frac{1}{(1+\omega^2 h^2)^{\frac{k\pi}{\angle(1+\omega hi)}}} \quad (18)$$

To express the damping ratio ζ, we first need to find an expression for the logarithmic decrement δ (7):

$$\begin{aligned}\delta(h) &= \frac{1}{m} \cdot \ln\left(\frac{X_k}{X_{k+m}}\right) \\ &= \frac{1}{m} \cdot \ln\left\{\frac{(1+\omega^2 h^2)^{\frac{(k+m)\pi}{\angle(1+\omega hi)}}}{(1+\omega^2 h^2)^{\frac{k\pi}{\angle(1+\omega hi)}}}\right\} \\ &= \frac{\pi}{\angle(1+\omega hi)} \cdot \ln\left(1+\omega^2 h^2\right)\end{aligned} \quad (19)$$

Knowing the logarithmic decrement δ, we can write the damping ratio ζ as a function of h using (8):

$$\zeta(h) = \frac{\delta(h)}{\left(4\pi^2 + \delta(h)^2\right)^{1/2}} \quad (20)$$

b) Numerical Damping in FEM: To experimentally describe the relation between ζ and h, we simulated the Hex Mesh beam for different h (Fig. 5).

With the values we gathered from the simulations, we can determine ζ using (7) and (8). The vertical displacement of the tip points of the beam can be approximated as a single DOF vibration, and we can observe that our analytical solution (20) closely matches the value computed from the measurement data (Fig. 6). We note that this result generalizes to other FEM simulators using implicit Euler solvers, as shown by simulation results using COMSOL with implicit Euler method.

3) Material Damping

a) Matching the Real-World Dynamic Response: As introduced in Section IV-B, we let the numerical damping be the

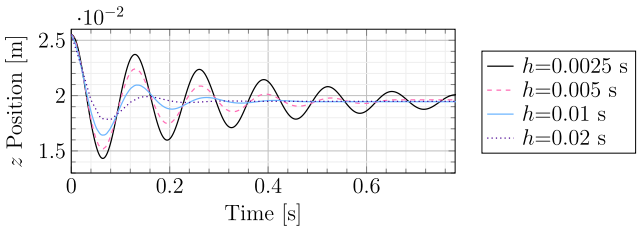


Fig. 5. Simulation results of Case A-1 for a selection of different time steps h.

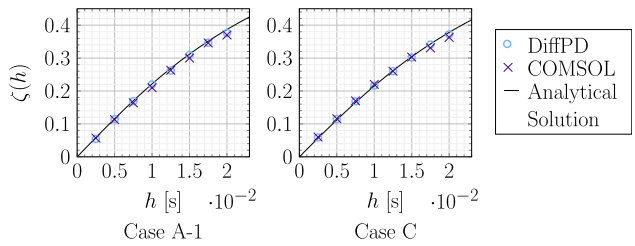


Fig. 6. Comparison between the computed values of $\zeta(h)$ from the DiffPD and COMSOL (using implicit Euler) simulations and our analytical solution.

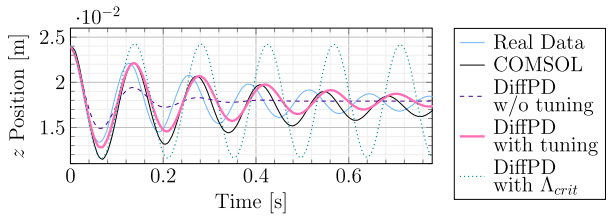


Fig. 7. Material damping modeling results using a velocity dependent state force.

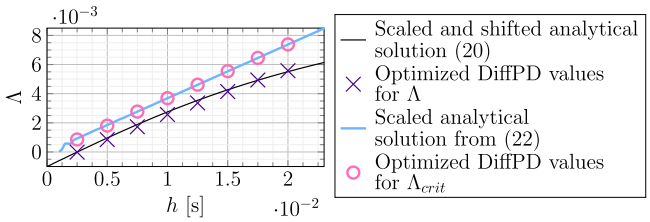


Fig. 8. Λ and Λ_{crit} as a function of h.

only source of damping in the FEM simulation and use the differentiability of DiffPD to find Λ so that the simulated solution closely approximates the real-world physical response. We define the optimization objective to be that the simulated dynamic oscillation should match the envelope of the real oscillation (Fig. 7). Plotting the tuning parameter Λ as a function of time step h, a similar relation to the one between ζ and h (20), can be observed (Fig. 8).

b) Stability Analysis: The notion of A-stability describing the implicit Euler method is not sufficient to assess the system's stability when active forces are considered. The velocity-dependent force (14) can lead the simulations to become unstable if Λ is larger than Λ_{crit}. To illustrate this, we derive an analytical criterion based on the linear system (9) that includes

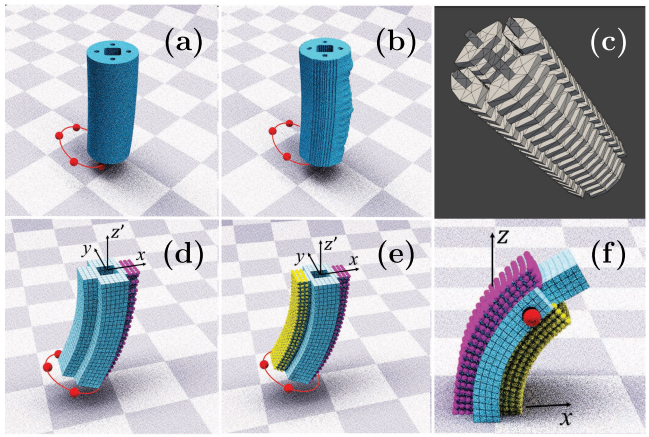


Fig. 9. Simulation results for the soft robotic arm and soft fish tail. The red dots are the motion markers positions of the real robot. The pink elements correspond to muscle fibers in extension, and the yellow elements to muscle fibers in contraction.

a velocity-dependent force:

$$\frac{d}{dt}\begin{bmatrix} x \\ \dot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2 & \Lambda \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{21}$$

Using the implicit Euler method on (21) leads to a discrete system of the form $\underline{x}_{j+1} = A\underline{x}_j$:

$$\begin{bmatrix} x_{j+1} \\ v_{j+1} \end{bmatrix} = \frac{1}{1 - h\Lambda + h^2\omega^2}\begin{bmatrix} 1 - h\Lambda & h \\ -h\omega^2 & 1 \end{bmatrix}\begin{bmatrix} x_j \\ v_j \end{bmatrix} \tag{22}$$

(22) is stable iff the eigenvalues of A are in the unit circle. This is the case if $\Lambda < \Lambda_{crit}$. Λ_{crit} can be represented as a function of h (Fig. 8). The corresponding values from the DiffPD simulations can be obtained using their differentiable property by defining an optimization objective that requires that the oscillation has constant amplitude. The obtained values closely follow our analytical solution up to a constant factor, which is likely primarily due to the force being applied at multiple nodes, and not at only one point particle as in (22).

B. Soft Robotic Arm

In this section, we simulate a more complex structure that includes pneumatic actuation, namely, a soft robotic arm.

The complex inner geometry of the pressure chambers is difficult to model accurately. Their modeling requires a very fine mesh (Fig. 9(c)) and is therefore computationally costly. A Tet Mesh suffers from the same stiff behavior as the beam simulations in Section VI-A (Fig. 9(a)). Meanwhile, a Hex Mesh cannot precisely capture the fine ribbing geometry (Fig. 9(b)) due to the regular shape of its elements.

As an alternative solution, we propose lower-order "Muscle Models" to simulate the arm. These models are based on the "Muscle Energy" model, introduced in Section IV. We specify "muscle fibers" along a simplified hexahedral mesh (Fig. 9), along which every hexahedron in the fiber is given the same actuation signal, as a means of emulating pneumatic behavior. We compare the performance of two possible muscle configurations in Fig. 11: (d) AC1 has a single muscle, which extends, while (e) AC2 has an extending muscle and an antagonistic contracting

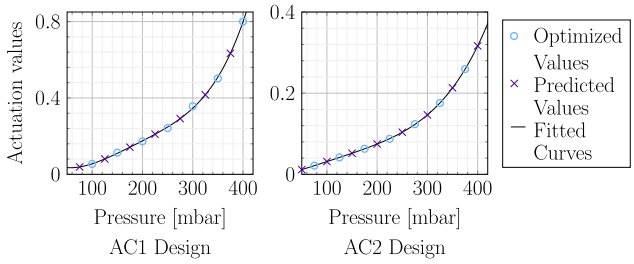


Fig. 10. Pressure-to-actuation maps for the AC1 and AC2 designs.

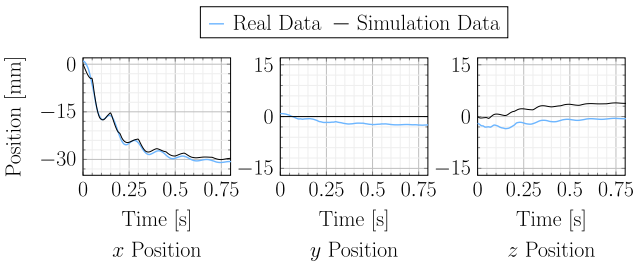


Fig. 11. Detailed simulation results for the Muscle Model AC2. We optimize for the x position. The difference we observe for the y and z position probably stems from the small manufacture inaccuracies of the arm or experimental setup.

muscle. Muscle Models use considerably less memory and are faster to simulate than meshing the true ribbing of the actuator. Furthermore, they do not need to resolve complex internal self-collisions.

We consider two experiments for each arm design, each of which leverages the differentiable character of DiffPD to optimize the muscle actuation signals a. In the first experiment, we optimize a single actuation signal, applied at every time step, such that the final pose of the virtual arm matches a static physical arm pneumatically actuated with a fixed constant pressure. In the second experiment, we optimize an actuation sequence to match our simulations with the full trajectory of the physical arm . This actuation sequence applies actuation signals at a fixed frequency f; if our simulation runs for T_{total} seconds, the actuation sequence has $\lfloor \frac{T_{total}}{f} \rfloor$ decision variables.

In the single-actuation, we optimize the loss-function $\mathcal{L} = \frac{1}{T \cdot M} \sum_{t=1}^{T} \sum_{i=1}^{M} \|\boldsymbol{q}_{\text{sim},i}(t) - \boldsymbol{q}_{\text{real},i}(T)\|^2$ where $\boldsymbol{q}_i$ is the position of motion marker i; this prioritizes that the soft arm achieves an accurate static response. Note that $\mathcal{L}$ is dependent on a since the final pose $\boldsymbol{q}_{\text{sim}}$ depends on a. By repeating optimization for successive pressure values, we extract pressure-to-actuation maps (Fig. 10). By fitting a curve through the optimized actuation values, we can predict interpolated values for which we did not optimize. For these predicted actuation values there is high agreement between the physical arm and simulated models, with an average relative position error of 13.07%.

In the multi-actuation example, we optimize the similar loss $\mathcal{L} = \frac{1}{T \cdot M} \sum_{t=1}^{T} \sum_{i=1}^{M} \|\boldsymbol{q}_{\text{sim},i}(t) - \boldsymbol{q}_{\text{real},i}(t)\|^2$ (note the term $\boldsymbol{q}_{\text{real},i}(t)$ cf $\boldsymbol{q}_{\text{real},i}(T)$); this promotes dynamic position tracking over time. In our experiments, the actuation frequency is 20 Hz. Fig. 11 shows that the optimized actuation sequence accurately matches the tracked physical arm.

These results demonstrate that Muscle Models can be accurately and predictively mapped to real pressure actuation.

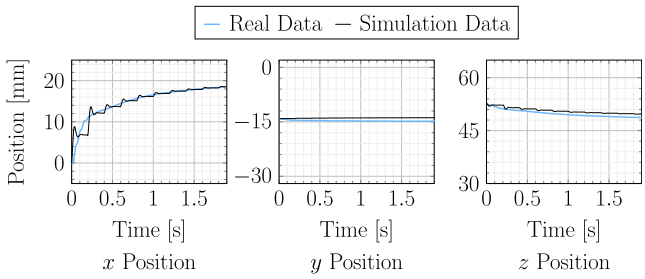


Fig. 12. Detailed simulation results for the Muscle Model of the fish. We optimize trajectories to match target x and z positions.

C. Pneumatically Actuated Fish Tail

To validate our Muscle Models on a separate geometry, we model the pneumatically actuated soft robotic fish tail (Section V-A). As with the arm, our Muscle Models' actuations are optimized to follow the dynamic behavior of the real-world robot (Fig. 12). We optimized for actuation sequences with 5 Hz control signals. Despite the step-wise behavior of the simulation, our optimized solution is still able to very closely match the motions of this second robotic morphology.

VII. Discussion and Future Work

In this paper, we investigated several traits of FEM modeling, and provided recommendations on how to better match physical reality and overcome the sim-to-real gap.

Our first recommendation resulting from the experiments is with regards to meshing. The linear tetrahedral mesh elements produced inaccurate behavior in the incompressible regime. This problem, named *locking problem*, is well known in the FEM community. On the other hand, the steady state results of the DiffPD simulations using the Hex Mesh are satisfying and allow for parameter calibration. Therefore, we recommend to use a Hex Mesh for problems in the incompressible regime.

Our second recommendation is with respect to damping. Although the steady state results are simple to match precisely once system identification of the Young's modulus is achieved, the dynamic behavior of vanilla FEM simulation accumulates significant error. DiffPD, using the implicit Euler method, suffers from numerical damping as derived analatytically. Depending on the exact use of FEM simulations, it is crucial to be aware of numerical damping, and a linear damping model is recommended to fit the material's response.

Our third recommendation is with respect to actuator modeling. Complex structures, such as the ribbed pressure chambers of fluidic actuators, require a fine mesh to be modeled correctly. These complex geometries can generally not be modeled with a Hex Mesh, due to the regular shape of its elements. Although a linear Tet Mesh is able to represent the geometry of the pressure chambers of the arm correctly, it fails to model the deformation of the arm correctly and is computationally costly. Muscle Models, correctly designed, can lead to good results in accuracy and computational efficiency. The differentiability of DiffPD allows efficient optimization of the actuator behavior.

We see four areas of further investigation. First, to model a more complex geometry without experiencing element locking, higher-order tetrahedral elements should be considered. Second, since the Muscle Models show promise as a means of modeling

fluidic actuation, it would be useful to develop more general, data-driven rules to obtain pressure-to-actuation maps for other geometries. Third, while our methodology and its principles are generally applicable to any finite-element-based simulator, DiffPD was primarily employed in this work. Further validation with other FEM simulators would be valuable. Finally, investigating more nonlinear materials or dynamics, in which Neo-Hookean responses play a role, would be helpful for future applications.

REFERENCES

[1] T. Du *et al.*, "DiffPD: Differentiable projective dynamics," *ACM Trans. Graph.*, vol. 41, no. 2, pp. 1–21, Nov. 2021.

[2] C. Della Santina, R. K. Katzschmann, A. Biechi, and D. Rus, "Dynamic control of soft robots interacting with the environment," in *Proc. IEEE Int. Conf. Soft Robot.*, 2018, pp. 46–53.

[3] I. R. J. Webster and B. A. Jones, "Design and kinematic modeling of constant curvature continuum robots: A review," *Int. J. Robot. Res.*, vol. 29, no. 13, pp. 1661–1683, 2010.

[4] D. A. Haggerty, M. J. Banks, P. C. Curtis, I. Mezić, and E. W. Hawkes, "Modeling, reduction, and control of a helically actuated inertial soft robotic arm via the koopman operator," 2020, *arXiv: 2011.07939*.

[5] D. Bruder, B. Gillespie, C. D. Remy, and R. Vasudevan, "Modeling and control of soft robots using the koopman operator and model predictive control," 2019, *arXiv: 1902.02827*.

[6] M. Han, J. Euler-Rolle, and R. K. Katzschmann, "DeSKO: Stability-assured robust control with a deep stochastic koopman operator," in *Proc. Int. Conf. Learn. Representations*, 2021.

[7] S. Tonkens, J. Lorenzetti, and M. Pavone, "Soft robot optimal control via reduced order finite element models," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 12 010–12 016.

[8] C. Duriez *et al.*, "Framework for online simulation of soft robots with optimization-based inverse model," in *Proc. IEEE Int. Conf. Simul., Model., Program. Auton. Robots*, Institute Electrical Electronics Engineers Inc., 2017, pp. 111–118.

[9] O. Goury and C. Duriez, "Fast, generic, and reliable control and simulation of soft robots using model order reduction," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1565–1576, Dec. 2018.

[10] S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly, "Projective dynamics: Fusing constraint projections for fast simulation," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 1–11, 2014.

[11] H. Wang, "A chebyshev semi-iterative approach for accelerating projective and position-based dynamics," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 1–9, Nov. 2015.

[12] H. Wang and Y. Yang, "Descent methods for elastic body simulation on the GPU," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 1–10, 2016.

[13] M. Fratarcangeli, V. Tibaldo, and F. Pellacini, "Vivace: A practical gauss-seidel method for stable soft body dynamics," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 1–9, Nov. 2016.

[14] D. Chen, D. I. Levin, W. Matusik, and D. M. Kaufman, "Dynamics-aware numerical coarsening for fabrication design," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–15, 2017.

[15] Y. Hu *et al.*, "ChainQueen: A real-time differentiable physical simulator for soft robotics," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 6265–6271.

[16] D. Hahn, P. Banzet, J. M. Bern, and S. Coros, "Real2Sim: Visco-elastic parameter estimation from dynamic motion," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 1–13, Nov. 2019.

[17] Y. Hu *et al.*, "DiffTaichi: Differentiable programming for physical simulation," Oct. 2019, *arXiv: 1910.00935*.

[18] M. Geilinger, D. Hahn, J. Zehnder, M. Bächer, B. Thomaszewski, and S. Coros, "ADD: Analytically differentiable dynamics for multi-body systemswith frictional contact," *ACM Trans. Graph.*, vol. 39, no. 6, p. 15, Nov. 2020.

[19] A. D. Marchese and D. Rus, "Design, kinematics, and control of a soft spatial fluidic elastomer manipulator," *Int. J. Robot. Res.*, vol. 35, no. 7, pp. 840–869, 2016.

[20] A. D. Marchese, R. Tedrake, and D. Rus, "Dynamics and trajectory optimization for a soft spatial fluidic elastomer manipulator," *Int. J. Robot. Res.*, vol. 35, no. 8, pp. 1000–1019, 2016.

[21] B. Mosadegh *et al.*, "Pneumatic networks for soft robotics that actuate rapidly," *Adv. Funct. Mater.*, vol. 24, no. 15, pp. 2163–2170, 2014.

[22] E. B. Joyee and Y. Pan, "A fully three-dimensional printed inchworm-inspired soft robot with magnetic actuation," *Soft Robot.*, vol. 6, no. 3, pp. 333–345, 2019.

[23] J. Allard *et al.*, "Sofa-an open source framework for medical simulation," in *MMVR 15-Medicine Meets Virtual Reality*, vol. 125. Amsterdam, The Netherlands: IOS Press, 2007, pp. 13–18.

[24] C. Duriez, "Control of elastic soft robots based on real-time finite element method," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 3982–3987.

[25] Z. Zhang, J. Dequidt, A. Kruszewski, F. Largilliere, and C. Duriez, "Kinematic modeling and observer based control of soft robot using real-time finite element method," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 5509–5514.

[26] E. Coevoet, A. Escande, and C. Duriez, "Optimization-based inverse model of soft robots with contact handling," *IEEE Robot. Automat. Lett.*, vol. 2, no. 3, pp. 1413–1419, Jul. 2017.

[27] R. K. Katzschmann *et al.*, "Dynamically closed-loop controlled soft robotic arm using a reduced order finite element model with state observer," in *Proc. IEEE Int. Conf. Soft Robot.*, Institute Electrical Electronics Engineers Inc., 2019, pp. 717–724.

[28] J. M. Bern, G. Kumagai, and S. Coros, "Fabrication, modeling, and control of plush robots," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2017, pp. 3739–3746.

[29] J. M. Bern, P. Banzet, R. Poranne, and S. Coros, "Trajectory optimization for cable-driven soft robot locomotion," in *Proc. Robot., Sci. Syst.*, 2019, p. 52, vol. 1, no. 3.

[30] S. Kriegman *et al.*, "Scalable sim-to-real transfer of soft robot designs," in *Proc. 3rd IEEE Int. Conf. Soft Robot.*, 2020, pp. 359–366.

[31] T. Du, J. Hughes, S. Wah, W. Matusik, and D. Rus, "Underwater soft robot modeling and control with differentiable simulation," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4994–5001, Jul. 2021.

[32] J. Hiller and H. Lipson, "Dynamic simulation of soft multimaterial 3D-printed objects," *Soft Robot.*, vol. 1, no. 1, pp. 88–101, 2014.

[33] E. Sifakis, "FEM simulation of 3D deformable solids: A practitioner's guide to theory, discretization and model reduction," in *Proc. ACM SIGGRAPH*, 2012, pp. 1–35.

[34] S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly, "Projective dynamics: Fusing constraint projections for fast simulation," *ACM Trans. Graph.*, vol. 33, no. 4, Jul. 2014, Art. no. 154.

[35] H. Ammari, W. Wei, and Y. Sanghyeon, "Numerical methods for ordinary differential equations," ETH Zürich, Tech. Rep., 2018.

[36] Y. J. Chen, U. M. Ascher, and D. K. Pai, "Exponential rosenbrock-euler integrators for elastodynamic simulation," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 10, pp. 2702–2713, Oct. 2018.

[37] I. Babuska and M. Suri, "Locking effects in the finite element approximation of elasticity problems," *Numerische Mathematik*, vol. 62, pp. 439–463, 1992.

[38] M. A. Puso and J. Solberg, "A stabilized nodally integrated tetrahedral," *Int. J. Numer. Methods Eng.*, vol. 67, no. 6, pp. 841–867, Aug. 2006.

[39] S. Min, J. Won, S. Lee, J. Park, and J. Lee, "Softcon: Simulation and control of soft-bodied animals with biomimetic actuators," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 1–12, 2019.

[40] P. Rothemund *et al.*, "A soft, bistable valve for autonomous control of soft actuators," *Sci. Robot.*, vol. 3, no. 16, Mar. 2018, Art. no. eaar7986.

[41] F. Connolly, C. J. Walsh, and K. Bertoldi, "Automatic design of fiber-reinforced soft actuators for trajectory matching," *Proc. National Acad. Sci. USA*, vol. 114, no. 1, pp. 51–56, 2017.