

DisMech: A Discrete Differential Geometry-based Physical Simulator for Soft Robots and Structures

Andrew Choi¹, Ran Jing², Andrew Sabelhaus², and Mohammad Khalid Jawed^{†,3}

Abstract—Fast, accurate, and generalizable simulations are a key enabler of modern advances in robot design and control. However, existing simulation frameworks in robotics either model rigid environments and mechanisms only, or if they include flexible or soft structures, suffer significantly in one or more of these performance areas. To close this “sim2real” gap, we introduce DisMech, a simulation environment that models highly dynamic motions of rod-like soft continuum robots and structures, quickly and accurately, with arbitrary connections between them. Our methodology combines a fully implicit discrete differential geometry-based physics solver with fast and accurate contact handling, all in an intuitive software interface. Crucially, we propose a gradient descent approach to easily map the motions of hardware robot prototypes to control inputs in DisMech. We validate DisMech through several highly-nuanced soft robot simulations while demonstrating an order of magnitude speed increase over previous state of the art. Our real2sim validation shows high physical accuracy versus hardware, even with complicated soft actuation mechanisms such as shape memory alloy wires. With its low computational cost, physical accuracy, and ease of use, DisMech can accelerate translation of sim-based control for both soft robotics and deformable object manipulation.

Index Terms—deformable simulation, soft robots, real2sim

I. INTRODUCTION

DEFORMABLE materials are ubiquitous, from knots to clothing to new types of machines. Despite the prevalence of deformable materials in everyday life, there is a lack of physically accurate and efficient continuum mechanics simulations for complicated and arbitrarily-shaped robots, particularly in hardware. As opposed to their rigid-body counterparts, deformable structures such as rods and shells possess infinite degrees of freedom and are capable of undergoing highly nonlinear geometric deformations from even minute external forces, requiring specialized simulators.

Accurate and efficient soft physics simulators serve two key purposes in the robotics community: they allow for 1) training traditional rigid manipulators to intelligently handle deformable objects [1] with sim2real realization [2], [3] and 2) for the

modelling, training, and development of controllers for soft robots themselves [4]. As the demands for efficient large-scale robot learning necessitate viable sim2real strategies, the need for such simulators becomes even more pressing. With this in mind, we introduce DisMech, a full end-to-end discrete differential geometry (DDG)-based physical simulator for both soft continuum robots and structures. Based on the Discrete Elastic Rods (DER) [5], [6] framework, DisMech is designed to allow users to create custom geometric configurations composed of individual elastic rods. Such configurations can be expressed quickly through an elegant API, allowing for rapid prototyping of different soft robot builds. Actuation of the soft robot is then readily achievable by manipulating the natural curvatures of the individual elastic rods (i.e., limbs), enormously simplifying sim2real control tasks.

As opposed to previous simulation frameworks focusing on soft physics [7], [8], [9], [10], DisMech’s equations of motion are handled fully implicitly, allowing for far more aggressive time step sizes than previous formulations. This allows our simulations to reach an order of magnitude speed increase over previous state-of-the-art simulators while maintaining physical accuracy as we show later for several canonical validation cases. Though our prior work implemented a DDG-based simulation for soft robot locomotion [4], this proof-of-concept operated solely in 2D, lacked elastic contact and self-contact, and was specialized to only one simple robot morphology. In comparison, DisMech is a generalizable DDG-based framework for arbitrary soft rod-like robots in contact-rich 2D and 3D environments, and includes an algorithm to map hardware motions to intuitive DDG control inputs. This article therefore compares a DDG simulation against the contemporary suite of soft robot simulators for the first time.

This article’s contributions are:

- 1) We introduce a methodology and open-source implementation¹ of DisMech as a fully implicit simulator supporting soft physics, frictional contact, and intuitive control inputs. To the best of our knowledge, DisMech is the first general purpose DDG-based 3D simulation framework for easy use by the robotics community.
- 2) We numerically validate DisMech through several complicated simulations, and compare with the state-of-the-art framework *Elastica* [10], showing comparable accuracy while obtaining an order of magnitude speed increase.
- 3) We demonstrate generalizability of DisMech for design simulations of dynamic soft robots, including a four-legged

Manuscript received: November 16, 2023; Accepted January 22, 2024. This paper was recommended for publication by Editor Aniket Bera upon evaluation of the Associate Editor and Reviewers’ comments.

¹Andrew Choi is with the Computer Science Department at the University of California, Los Angeles (UCLA) (email: asjchoi@cs.ucla.edu).

²Ran Jing and Andrew Sabelhaus are with the Department of Mechanical Engineering at Boston University (email: {rjing}{asabelha}@bu.edu).

³Mohammad Khalid Jawed is with the Department of Mechanical & Aerospace Engineering at UCLA (email: khalidjm@seas.ucla.edu).

[†] Corresponding author.

This research was funded in part by the National Science Foundation under award numbers OAC-2209782, OAC-220978, CAREER-2047663, and IIS-1925360.

Digital Object Identifier (DOI): see top of this page.

¹<https://github.com/StructuresComp/dismech-rods>

TABLE I
ROBOTICS SIMULATION FRAMEWORK COMPARISON

Framework	Model	Continuum Physics	Physics Complexity	Numerical Integration Scheme(s)	Solver Type
Gazebo (ODE) [13]	Rigid-body physics	✗	low	Semi-implicit Euler	Explicit
MuJoCo [14]	Rigid-body physics	✗	low	Semi-implicit Euler / 4 th Runge-Kutta	Explicit or Implicit
Bullet [15]	Rigid-body physics	✗	low	Semi-implicit Euler	Explicit
SoMo [16]	Rigid-body physics	✓—	low	Semi-implicit Euler	Explicit
SOFA [17], [9]	FEA	✓	high	Several Options	Explicit or Implicit
SoRoSim [8]	Cosserat Rod + GVS	✓	medium-high	Variable Step Methods	Explicit or Implicit
Elastica [10], [18]	Cosserat Rod	✓	medium-high	Verlet Position	Explicit
DisMech	Kirchhoff Rod + DDG	✓	medium-high	Backward Euler / Implicit Midpoint	Explicit or Implicit

spider and an active entanglement gripper [11].

- 4) Finally, we propose a generalizable gradient descent algorithm to map hardware robot data to DisMech’s natural curvature parameters, validating their use as control inputs in “real2sim” of a real world soft manipulator [12].

II. RELATED WORK

Though soft robotics research has been a popular and growing field [19], the presence of physical simulators capable of robustly capturing continuum mechanics is scarce. Popular simulation frameworks used for sim2real training of robots include Gazebo [13], MuJoCo [14], and Bullet/PyBullet [15], but such simulators focus primarily on rigid body dynamics and fail to replicate physically accurate elasticity.

One avenue to remedy this involve frameworks such as SoMo [16], which models deformable structures by representing them as rigid links connected by springs. Frameworks such as this simply act as wrappers built on top of preexisting rigid-body physics simulators (Bullet). Although plausible results have been shown for soft grippers and control [7], the underlying rigid-body physics is unable to simulate more complicated deformation modes such as twisting and elastic buckling [20]. With this, accurate soft robot simulation requires simulation frameworks more dedicated to soft physics.

Another avenue for simulating the continuum mechanics of soft robots involves the classical finite element method (FEM) [17], [9]. Although such methods are accurate, their high fidelity mesh representation often results in high computational costs, requiring separate model-order reduction (MOR) techniques for online control [21], [22]. These limitations focus the use of FEM/MOR for offline tasks such as reinforcement learning (RL) [23] and through asynchronous techniques [24].

Finally, there are simulation frameworks that simulate elastic rods using either the unstretchable and unsharable Kirchhoff rod model [25] or fully deformable Cosserat rod model [10]. Using the Kirchhoff rod model, the computer graphics community has developed discrete differential geometry-based frameworks such as Discrete Elastic Rods (DER) [5], [6]. Originally meant for realistic animation, DER has shown surprisingly great performance in accurately capturing the nonlinearity of rods and has been rigorously physically validated in cases such as rod deployment [26], knot tying [27], [28], and buckling [20], [29]. Using a DER-inspired framework, Huang et al. showed

success at simulating and modeling soft robot locomotion [4], albeit with the omission of twisting as locomotion occurred solely along a 2D plane.

Other frameworks have opted for using the Cosserat rod model as it incorporates the influence of shearing effects. One such example involves SoRoSim [8] which models soft links using a Geometric Variable Strain (GVS) model of Cosserat rods. This allows for the modeling of soft links with less degrees of freedom (DOFs) compared to traditional lumped mass models [5], [10]. Despite this mathematical elegance, the framework can be difficult for users to setup due to the prohibitively unintuitive representation of strains as the DOF.

Arguably the most prominent soft physics simulator based on Cosserat rod theory is Elastica [10], [18] This framework also models soft links as Cosserat rods but uses the more traditional lumped mass model similar to DER. This framework has been extensively physically validated and its potential for reinforcement learning methods has also been demonstrated [30]. Despite these results, Elastica uses an explicit integration scheme requiring prohibitively small time step sizes to maintain numerical stability. This is especially true when simulating stiff materials as we will show later in Sec. IV.

To circumvent this computational limitation, we formulate DisMech, a fully implicit DER-based physical simulator. Despite DER’s non-consideration of shearing, we argue that shearing effects are negligible so long as the radii of the rod is small enough. Though this disqualifies DisMech from accurately modeling structures such as muscle fibers, the DDG-based framework of DisMech allows for an intuitive implicit formulation resulting in order of magnitude speed increases while maintaining highly accurate results for a large array of soft structures. In addition to computational benefits, DisMech’s framework also allows for the natural incorporation of shell structures via the DDG-based framework Discrete Shells [31], something that is rather nontrivial to accomplish in frameworks such as SoRoSim and Elastica. A summary of popular robotics simulation frameworks can be seen in Table I.

III. METHODOLOGY

In this section, we formulate the DisMech framework involving its soft physics, frictional contact, and control input mapping. As mentioned previously, DisMech’s simulation of elasticity is based off of the Discrete Elastic Rods (DER) [5],

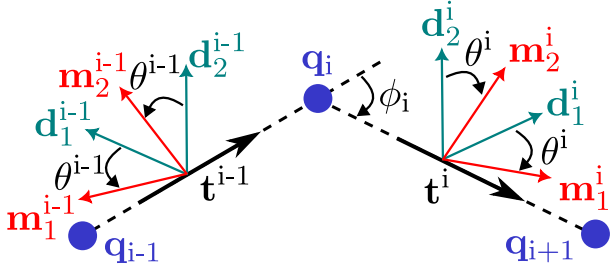


Fig. 1. Discrete rod schematic with relevant notations describing the rod's discrete bending and twisting at node \mathbf{q}_i .

[6] framework. In this framework, elastic rods are represented entirely by their centerline which are discretized into N nodes $\mathbf{q}_i \in \mathbb{R}^3$, resulting in a total of $N - 1$ edges $\mathbf{e}^i = \mathbf{q}_{i+1} - \mathbf{q}_i$. Note that node-relevant quantities are denoted with subscripts while edge-relevant quantities are denoted with superscripts. To capture the influence of bending and twist, each edge possesses two orthonormal frames: a reference frame $\{\mathbf{d}_1^i, \mathbf{d}_2^i, \mathbf{t}^i\}$ and a material frame $\{\mathbf{m}_1^i, \mathbf{m}_2^i, \mathbf{t}^i\}$ as shown in Fig. 1. The material frame describes the rotation of the elastic rod centerline and is obtainable by rotating the reference frame by a signed angle θ^i along the shared director \mathbf{t}^i . The reference frame is arbitrarily initialized at the start of the simulation and is updated between time steps via time parallel transport [6]. With N Cartesian coordinate positions and $N - 1$ twist angles, this results in a $4N - 1$ size DOF vector

$$\mathbf{q} = [\mathbf{q}_0, \theta^0, \mathbf{q}_1, \theta^1, \dots, \mathbf{q}_{N-2}, \theta^{N-2}, \mathbf{q}_{N-1}]^T. \quad (1)$$

A. Elastic Energies

We now define the constitutive laws of the elastic energies whose gradients and Hessians will be used to generate the inner elastic forces and Jacobians, respectively. With DER based on the Kirchhoff rod model [25], the three main modes of deformation are stretching, bending, and twisting.

1) *Stretching Energy*: The stretching strain of an edge \mathbf{e}^i is expressed simply by the ratio of elongation / compression with respect to its undeformed state. Moving forwards, all quantities with a $\bar{\cdot}$ represent those respective quantities in their natural undeformed state. With this, stretching strain can be defined as

$$\epsilon^i = \frac{\|\mathbf{e}^i\|}{\|\bar{\mathbf{e}}^i\|} - 1. \quad (2)$$

Stretching energy is then be computed as

$$E_s = \frac{1}{2}EA \sum_{i=0}^{N-2} (\epsilon^i)^2 \|\bar{\mathbf{e}}^i\|, \quad (3)$$

where E is Young's modulus and A is the cross-sectional area.

2) *Bending Energy*: Bending deformations occur between two adjacent edges and therefore, their respective strain involves computing the curvature at each interior node. The bending strain can be evaluated through a curvature binormal which captures the misalignment between two edges:

$$(\kappa \mathbf{b})_i = \frac{2\mathbf{t}^{i-1} \times \mathbf{t}^i}{1 + \mathbf{t}^{i-1} \cdot \mathbf{t}^i}. \quad (4)$$

Using this curvature binormal, the integrated curvature vector at node \mathbf{q}_i can be evaluated using the material frames of each adjacent edge as

$$\begin{aligned} \kappa_{1,i} &= \frac{1}{2}(\kappa \mathbf{b})_i \cdot (\mathbf{m}_2^{i-1} + \mathbf{m}_2^i), \\ \kappa_{2,i} &= -\frac{1}{2}(\kappa \mathbf{b})_i \cdot (\mathbf{m}_1^{i-1} + \mathbf{m}_1^i), \\ \boldsymbol{\kappa}_i &= [\kappa_{1,i}, \kappa_{2,i}] \end{aligned} \quad (5)$$

Note that $\boldsymbol{\kappa}_i = 2 \tan(\phi_i/2)$ where $\phi_i = [\phi_{1,i}, \phi_{2,i}]$ is the turning angle vector (Fig. 1). With this, we compute the bending energy of the rod as

$$E_b = \frac{1}{2}EI \sum_{i=1}^{N-2} (\boldsymbol{\kappa}_i - \bar{\boldsymbol{\kappa}}_i)^2 \frac{1}{V_i}, \quad (6)$$

where $I = \pi h^4/4$ is the moment of inertia; h is the rod radius, and $V_i = (\|\mathbf{e}^i\| + \|\mathbf{e}^{i-1}\|)/2$ is the Voronoi length. Later, we will show how to manipulate the natural curvature $\bar{\boldsymbol{\kappa}}$ to intuitively actuate the robot's limbs.

3) *Twisting Energy*: Like bending, twisting is also a deformation mode that occurs between two adjacent edges. Therefore the twisting strain at an interior node \mathbf{q}_i is computed as

$$\tau_i = \theta^i - \theta^{i-1} + \beta^i, \quad (7)$$

where β^i is the signed angular difference between the two consecutive reference frames of the i and $i - 1$ -th edges.

Finally, twisting energy is then defined as

$$E_t = \frac{1}{2}GJ \sum_{i=1}^{N-2} (\tau_i - \bar{\tau}_i)^2 \frac{1}{V_i}, \quad (8)$$

where G is the shear modulus and $J = \pi h^4/2$ is the polar second moment of area.

4) *Elastic Forces*: With each of the elastic energies defined, we can now compute the inner elastic forces (for nodal positions \mathbf{q}_i) and elastic moments (for twist angles θ^i) as the negative gradient of elastic energy:

$$\mathbf{F}^{\text{int}} = -\frac{\partial}{\partial \mathbf{q}} (E_s + E_b + E_t). \quad (9)$$

B. Contact and Friction

In addition to the inherent elasticity of the rod, we must also model external forces such as contact and friction. We do so by using Implicit Contact Model (IMC) [27], [28], an implicit penalty energy method that integrates seamlessly into the DER framework. Similar to the elastic energies, IMC defines an artificial contact energy

$$E_c = \begin{cases} (C - D)^2 & D \in (0, C - \delta), \\ \left(\frac{1}{K_1} \log(1 + e^{K_1(C-D)})\right)^2 & D \in (C - \delta, C + \delta), \\ 0 & D \geq C + \delta, \end{cases} \quad (10)$$

where $K_1 = 15/\delta$ is a stiffness parameter; δ is a user-defined contact distance tolerance; D is the minimum distance between two edges, and C is the distance from the centerlines at which

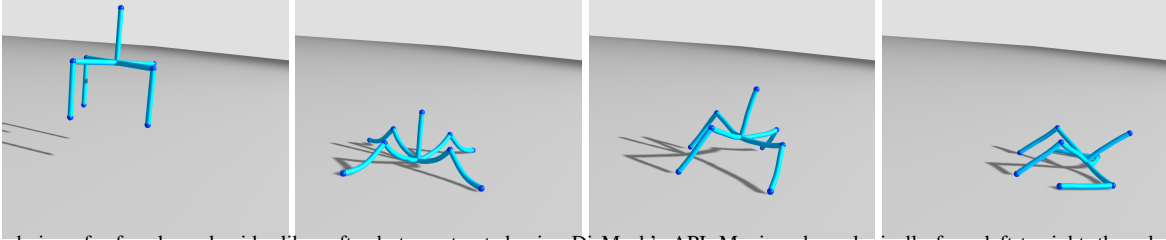


Fig. 2. Rendering of a four-legged spider-like soft robot constructed using DisMech’s API. Moving chronologically from left to right, the robot is dropped from a height where it then makes contact with an incline plane. After some initial sliding, the influence of sticking friction results in the robot’s eventual equilibrium position in the rightmost column.

contact would occur (e.g., self-contact occurs at $C = 2h$). By employing chain rule, we can obtain contact forces promoting non-penetration as

$$\mathbf{F}^{\text{con}} = -\frac{\partial E_c}{\partial D} \frac{\partial D}{\partial \mathbf{q}}. \quad (11)$$

Coulomb friction forces for an edge i are then computed using the above contact forces (i.e., the normal forces), as

$$\gamma = \frac{2}{1 + e^{-K_2 \|\mathbf{u}\|}} - 1, \quad (12)$$

$$\mathbf{F}^{\text{fr},i} = -\mu \gamma \hat{\mathbf{u}} \|\mathbf{F}^{\text{con},i}\|, \quad (13)$$

where $K_2 = 15/\nu$ is a stiffness parameter; ν is a user-defined slipping tolerance; \mathbf{u} is the tangential relative velocity between two contacting bodies (note that $\hat{\cdot}$ indicates unit vector); μ is the friction coefficient, and $\gamma \in [0, 1]$ is a smooth scaling factor that models the transition from sticking and sliding modes between the contacting bodies. This results in an external force vector of $\mathbf{F}^{\text{ext}} = \mathbf{F}^{\text{con}} + \mathbf{F}^{\text{fr}} + \mathbf{F}^{\text{misc}}$, where \mathbf{F}^{misc} refers to other miscellaneous external forces such as gravity, viscous damping, hydrodynamic forces, etc.

C. Time Stepping and Numerical Integration Schemes

With the internal and external forces defined, we can now construct the equations of motion (EOM) using Newton’s second law as

$$\mathbf{M}\ddot{\mathbf{q}} = \mathbf{F}^{\text{int}} + \mathbf{F}^{\text{ext}}, \quad (14)$$

where \mathbf{M} is the diagonal mass matrix and $\ddot{\mathbf{q}}$ is the second derivative of the DOFs with respect to time. Given this, we can then format the EOM to be solved for implicitly as

$$\frac{\mathbf{M}}{\Delta t} \left(\frac{\mathbf{q}(t_{i+1}) - \mathbf{q}(t_i)}{\Delta t} - \dot{\mathbf{q}}(t_i) \right) - \mathbf{F}^{\text{int}} - \mathbf{F}^{\text{ext}} = 0, \quad (15)$$

where Δt is the time step size. We solve for $\mathbf{q}(t_{i+1})$ in Eq. 15 iteratively using Newton’s method using the elastic energy Hessian, contact energy Hessian, and friction Jacobian (obtainable via chain rule [28]):

$$\mathbf{J} = \frac{\partial^2}{\partial \mathbf{q}^2} (E_s + E_b + E_t) + \frac{\partial^2 E_c}{\partial \mathbf{q}^2} + \frac{\partial \mathbf{F}^{\text{fr}}}{\partial \mathbf{q}}. \quad (16)$$

Once $\mathbf{q}(t_{i+1})$ is known, the velocity at time $t+1$ can be trivially solved for algebraically as $\dot{\mathbf{q}}(t_{i+1}) = (\mathbf{q}(t_{i+1}) - \mathbf{q}(t_i))/\Delta t$.

Currently, DisMech has been outfitted to support both explicit (e.g., Elastica’s Verlet position [10]) and implicit numerical

integration schemes. For implicit schemes, we offer both backward Euler and implicit midpoint. As backward Euler results in artificial damping [32], we opt to use implicit midpoint for scenarios where energy loss is unwanted and backward Euler for scenarios where stability is preferred. To further maintain numerical stability, DisMech is outfitted with both a line search method [28] as well as adaptive time stepping as optional features when an implicit scheme is chosen.

D. Elastic Joints

To allow for creating connections and grid-like structures, we introduce the concept of “elastic joints”, i.e., nodes along a rod that have one or more other rods connected to them. Such connections experience the same stretching, bending, and twisting energy constraints formulated in Sec. III-A as individual rods. Bending and twisting forces at joints are computed for every possible edge combination. Examples of elastic joints can be seen in Figs. 2 and 5 as blue spheres.

E. Actuation via Natural Curvatures

Given how the material frames are first setup, we can actuate a soft robot by manipulating its natural curvature $\bar{\kappa}$ or equivalently its natural turning angle $\bar{\phi}$. This in effect results in a bending energy differential in Eq. 6 which produces contractions and/or relaxations of the elastic rod along the respective material directors. Later in Sec. V-B, we show how this actuation can be framed as a control input, using a gradient descent method to calculate $\bar{\kappa}$ values for real2sim open-loop control of a hardware robot prototype.

IV. THEORETICAL VALIDATION

In this section, we perform physical validation of our model through several canonical experiments. We compare our results with both theory (when appropriate) and the state-of-the-art framework Elastica [10]. Efficiency metrics such as time step size and computational time for all experiments are listed in Table II while simulation results are plotted in Fig. 3. For Elastica, the largest time step size before numerical instability arose was used while for DisMech, the time step size that had the best tradeoff between number of iterations and computational time was used. For all experiments, we assume a Poisson ratio of 0.5 and a gravitational acceleration of $g = 9.8\text{m/s}^2$ for those involving gravity. All simulations were run on a workstation containing an Intel Core i7-9700K (3.60GHz \times 8) processor and 32GB of RAM.

TABLE II
SIMULATION PERFORMANCE COMPARISON

Simulation Framework	Numerical Integration	Metrics [s]	Sim Validation Experiments					
			Parameters	Cantilever (a1)	Cantilever (a2)	Helix (b1)	Helix (b2)	Friction (c)
			N	201	201	100	100	26
			h [m]	0.020	0.020	0.005	0.001	0.025
			E [Pa]	1×10^5	1×10^7	1×10^7	1×10^9	1×10^5
			T [s]	100	20	10	2	1.5 (44 sims)
Elastica	Verlet	Max Δt	—	2.5×10^{-4}	3×10^{-5}	4×10^{-5}	1×10^{-6}	1×10^{-4}
	Position	Comp. Time	—	23.80	40.01	11.77	89.24	7.42
DisMech	Implicit	Best Δt	—	5×10^{-1}	5×10^{-2}	6×10^{-3}	3×10^{-3}	5×10^{-3}
	Midpoint	Comp. Time	—	0.661	1.24	7.30	3.18	1.59

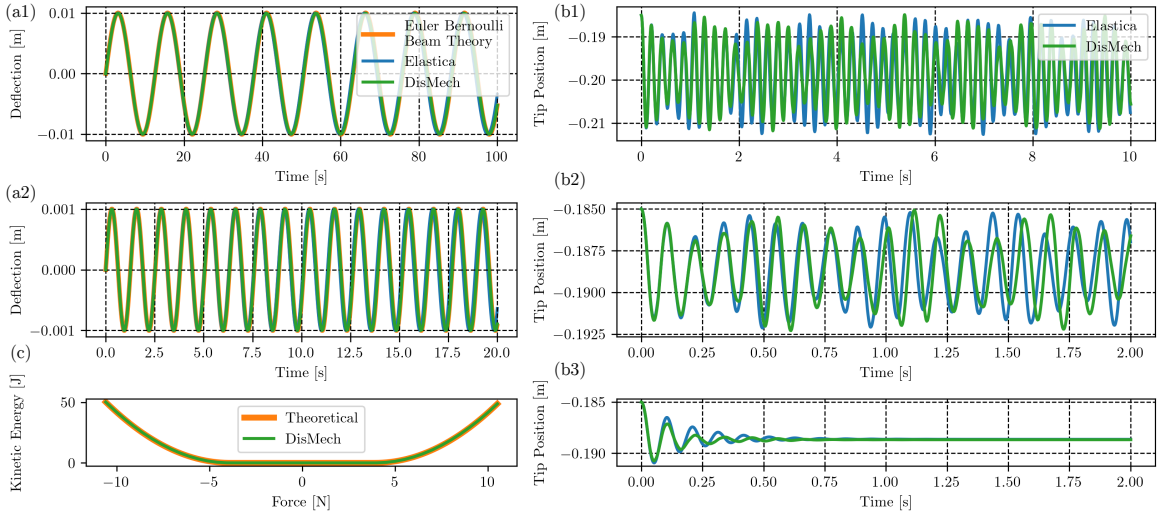


Fig. 3. Simulation results for DisMech compared with Elastica and theory (when applicable). Each plot refers to a specific experiment shown in Table II. Plots (a1) and (a2) showcase the deflection of the cantilever beam experiment for different material properties. Likewise, plots (b1) and (b2) showcase the tip position of the helical rod experiment for different material and geometric properties. As the helix experiment does not have an analytical solution to compare against, we show in plot (b3) that both DisMech and Elastica reach the same static equilibrium point when damping is introduced into the system as a sanity check. Finally, plot (c) showcases the kinetic energy of the rod as a function of an enacted external force for a friction coefficient of $\mu = 0.4$.

A. Dynamic Cantilever Beam

The first validation experiment we conduct is comparing the deflections of a rod modelled as a cantilever beam against Euler-Bernoulli beam theory. In absence of external loads, we can refer to the free vibration equation

$$w(s, t) = \text{Re} [\hat{w}(s)e^{-i\omega t}], \quad (17)$$

where w is the z -direction deflection at arc length $s \in [0, L]$ at time t ; ω is the frequency of the vibration, and $\hat{w}(x)$ is the natural frequency of the beam. We use the analytical solution available for cantilever beams [33] for an initial tip velocity of 5 mm/s. With this, we conduct two simulations to showcase the effect of material stiffness on time step size. For both experiments, we use a rod radius $h = 2$ cm, density $\rho = 500$ kg/m³, and rod length $L = 1$ m. A discretization of $N = 201$ nodes is also used. For the first and second experiments, we then use a Young's modulus E of 0.1 MPa and 10 MPa and a sim time T of 100 s and 20 s, respectively.

Deflections in Fig. 3(a1-a2) show that both Elastica and DisMech have excellent agreement with theory (Eq. 17), and neither suffer from artificial damping. However, DisMech achieves these results with time step sizes three orders-of-magnitude larger than Elastica, corresponding to an order of magnitude speed increase.

B. Oscillating Helix under Gravity

The next experiment consists of a suspended helical rod oscillating under gravity. We use the same experiment setup as [32] with density $\rho = 1273.52$ kg/m³, helix radius of 2 cm, pitch of 5 cm, and a contour length of 0.5 m (resulting in an axial length of ≈ 0.185 m). A discretization of $N = 100$ is used. Like the previous section, we also conduct two experiments with a Young's modulus E of 10 MPa and 1 GPa, radius h of 5 mm and 1 mm, and sim time T of 10 s and 2 s, respectively.

As an analytical solution for comparison is not available, we simply plot the oscillating tip positions as shown in Fig. 3(b1-b2). As shown, both simulation frameworks produce results with identical frequencies for the first experiment and

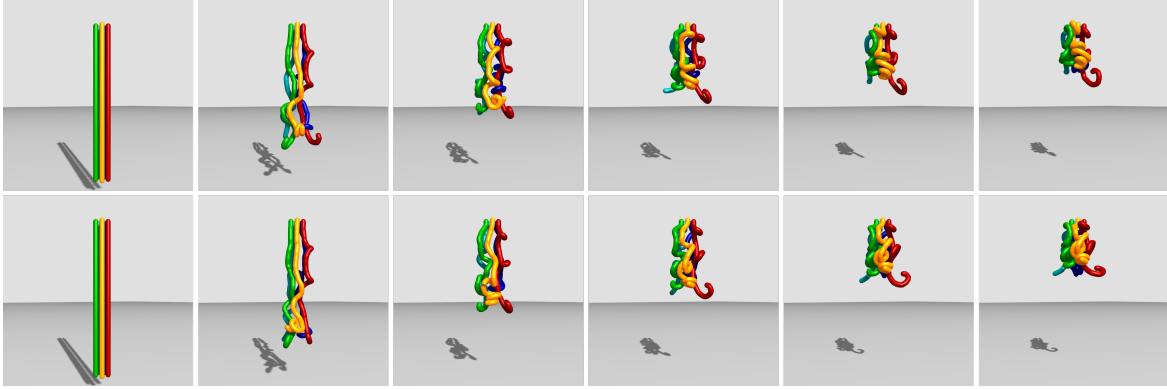


Fig. 4. Renderings of an active entanglement gripper [11]. The top row showcases contact-only entanglement whereas the bottom row showcases entanglement with friction $\mu = 0.5$. Note the influence of stiction causing more distorted helices in the latter example.

near identical frequencies for the second. To confirm that the difference in results is not a result of improper system representation, we also show that both DisMech and Elastica reach the same static equilibrium point in Fig. 3(b3) after introducing damping. Therefore, we presume that these slight frequency differences are numerical artifacts arising from differences in integration schemes and/or time step size. As before, we also report being able to take time step sizes of up to three orders-of-magnitude, correlating to an order of magnitude speed increase.

C. Friction Validation

For the final experiment, we test axial friction on a rod. We use a rod with radius $h = 2.5$ cm, density 509.3 kg/m³, length $L = 1$ m, and Young’s modulus $E = 0.1$ MPa. A discretization of $N = 26$ is used. Floor contact was simulated using IMC using a friction coefficient $\mu = 0.4$, distance tolerance $\delta = 5e-4$ m, and slipping tolerance $\nu = 1e-3$ m/s. With this, we can compute the kinetic energy of the rod when experiencing a constant uniform push/pull force \mathbf{F}^p as

$$E_k = \begin{cases} 0 & \|\mathbf{F}^p\| \leq |\mu mg|, \\ \frac{t^2}{2m} (\|\mathbf{F}^p\| - \mu mg)^2 & \|\mathbf{F}^p\| > |\mu mg|, \end{cases} \quad (18)$$

where m is the mass of the rod and t is the time after starting force exertion from a resting configuration. We compute results for 44 simulations in parallel for a range of forces $\|\mathbf{F}^p\| \in [-10.6, 10.5]$ N. When observing results in Fig. 3(c), we see excellent agreement between DisMech’s simulated kinetic energy and theory for a sim time of $T = t = 1.5$ s. For further more rigorous validation of the contact used by DisMech, we refer readers to Refs. [27] and [28].

V. PRACTICAL DEMONSTRATIONS FOR FLEXIBLE ROBOTS

Three demonstrations below showcase the accuracy and ease-of-use of DisMech over competing frameworks in simulating complicated flexible and soft robots. We also provide an algorithm that maps hardware robot motions to DisMech’s control inputs, emphasizing its generalizability.

A. Arbitrary Robot Prototyping

1) *Spider Robot*: We showcase the ease-of-use and environmental contact capabilities of DisMech by simulating a four-legged spider-like soft robot composed of interconnected rods (Fig. 2). Individual limbs are created as rods, with joints between them, using a single line of code each. We simulate dropping the robot onto a floor (no actuation) with $\mu = 0.4$, and replicate an incline using gravity $\mathbf{g} = [0.707, 0.707, -9.8]$ m/s². We can observe visually plausible results of the robot colliding with the floor, rebounding, and then after an initial sliding period, coming to static equilibrium via stiction.

2) *Active Entanglement Gripper*: Next, we further showcase the generality of DisMech by simulating an active entanglement gripper [11], a highly nontrivial frictional contact case. To do so, we simulate five fingers (rods) placed equidistantly along a circular perimeter, each with length $L = 0.3$ m, radius $h = 5$ mm, density $\rho = 1200$ kg/m³, and Young’s modulus $E = 0.3$ MPa. Each rod is discretized using $N = 60$. Contact is simulated using $\delta = 5e-4$ m and $\nu = 1e-3$ m/s. To simulate rapid entanglement, each edge is actuated via a random ϕ value uniformly sampled from a range of $[0, 40]^\circ$.

Results for both contact-only and frictional contact ($\mu = 0.5$) scenarios show convincing, visually plausible results (Fig. 4). When comparing between the two, friction causes the rods to stick to each other during the coiling, resulting in more distorted helices. Simulations such as these open up many opportunities for generating control policies.

B. Real2Sim Open-Loop Control

Finally, we provide a method to use DisMech to simulate an open-loop trajectory of a real soft robot in hardware, as a form of open-loop control. Here, we use a shape memory alloy (SMA) actuated double limb soft manipulator [12] (Fig. 5). The robot’s limbs are $76 \times 49 \times 9$ mm, and are constructed from a silicone polymer (Smooth-On Smooth-Sil) with a density of 1240 kg/m³ and a Young’s modulus of 1.793 MPa. Despite the manipulator being quite wide, we can still represent the manipulator as a rod since manipulation occurs primarily in 2D [2]. For the rod radius, we used 10 cm to compensate for the ridges of the limbs. Two rods are initialized with the

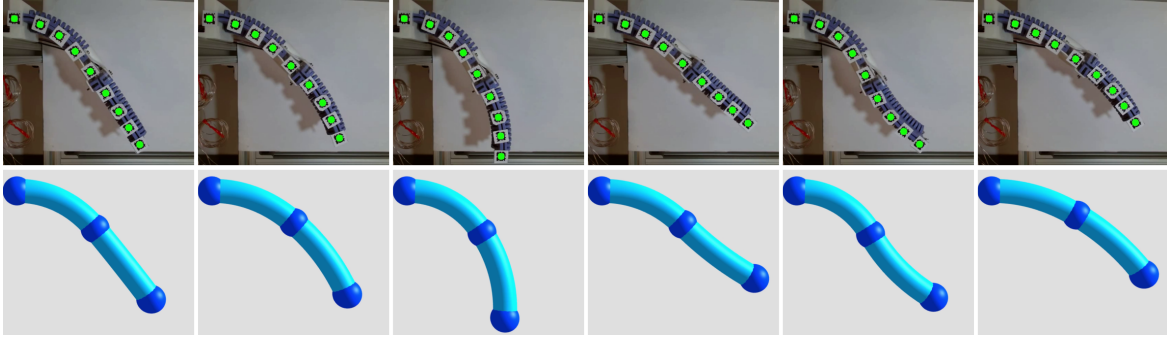


Fig. 5. Snapshots showcasing real2sim realization of a SMA actuated dual soft limb manipulator [12]. Using our gradient descent approach, we showcase excellent agreement between the real and simulated curvatures for a wide variety of geometric configurations.

Algorithm 1: Real2Sim via Gradient Descent

Input: $\mathbf{D} \leftarrow$ geometric configuration of robot

Output: $\tau_{\bar{\kappa}} \leftarrow$ trajectory of $\bar{\kappa}$ s

```

1 Func SolveNaturalCurvatures ( $\mathbf{D}$ ):
2    $\kappa^* \leftarrow$  ComputeCurvature( $\mathbf{D}$ ) // Eq. 5
3    $\tau_{\bar{\kappa}} \leftarrow [ ]$  // trajectory
4    $\bar{\kappa} \leftarrow [0, 0]$ 
5   for  $\kappa_1^*, \kappa_2^* \in \kappa^*$  do
6      $\alpha \leftarrow 0.1$  // step size
7      $\lambda_{\text{prev}} \leftarrow \infty$ 
8     while True do
9        $\kappa_1, \kappa_2 \leftarrow \mathcal{F}(\bar{\kappa})$  // DisMech Sim
10       $\lambda \leftarrow |\kappa_1^* - \kappa_1| + |\kappa_2^* - \kappa_2|$  // Eq. 19
11      if  $\lambda <$  tolerance then
12        break
13      if  $\lambda >$   $\lambda_{\text{prev}}$  then
14         $\alpha \leftarrow 0.5\alpha$ 
15         $\lambda_{\text{prev}} \leftarrow \lambda$ 
16         $\nabla_{\kappa} \lambda \leftarrow$  finite diff with  $\epsilon$  // Eq. 20
17         $\bar{\kappa} \leftarrow \bar{\kappa} - \alpha \nabla_{\kappa} \lambda$  // Eq. 21
18       $\tau_{\bar{\kappa}}.append(\bar{\kappa})$ 
19 return  $\tau_{\bar{\kappa}}$ 

```

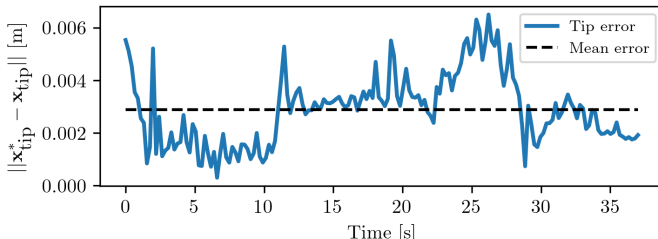


Fig. 6. Tip error between the target tip position $\mathbf{x}_{\text{tip}}^*$ and the simulated tip position \mathbf{x}_{tip} when carrying out our gradient descent natural curvature actuation.

aforementioned parameters to represent each limb, which are then connected via an elastic joint.

1) *Solving Natural Curvature via Gradient Descent:* To achieve real2sim realization, we must first calculate the appropriate $\bar{\kappa}$ values that generate geometric configurations of the robot corresponding to hardware data (Sec. III-E). Given the nonlinearity of the robot’s geometry, coupled with

deformations produced by gravity, solving for the appropriate natural curvatures analytically is nontrivial. Therefore, we propose a gradient descent-based approach to solve for $\bar{\kappa}$.

In our approach, we assume that the movement of the limbs is quasistatic and that each limb has more-or-less a constant curvature along its length. We use the AprilTag library as in our prior work [12] to extract the arm’s position from video (Fig. 5, green dots). Using these positions, we can then compute the target curvatures κ^* using Eq. 5.

We then define a loss

$$\lambda(\bar{\kappa}_1, \bar{\kappa}_2) = |\kappa_1^* - \kappa_1| + |\kappa_2^* - \kappa_2|, \quad (19)$$

where κ_1 and κ_2 are the resulting simulated curvatures of the first and second limbs when changing the natural curvatures to values $\bar{\kappa}_1$ and $\bar{\kappa}_2$, respectively. The gradient of λ with respect to $\bar{\kappa}$ can then be obtained using a forward finite difference approach:

$$\nabla_{\bar{\kappa}} \lambda = \begin{bmatrix} \frac{\partial \lambda}{\partial \bar{\kappa}_1} \\ \frac{\partial \lambda}{\partial \bar{\kappa}_2} \end{bmatrix} = \frac{1}{\epsilon} \begin{bmatrix} \lambda(\bar{\kappa}_1 + \epsilon, \bar{\kappa}_2) - \lambda(\bar{\kappa}_1, \bar{\kappa}_2) \\ \lambda(\bar{\kappa}_1, \bar{\kappa}_2 + \epsilon) - \lambda(\bar{\kappa}_1, \bar{\kappa}_2) \end{bmatrix}, \quad (20)$$

where ϵ is a small input perturbation. Using this finite difference gradient, we can then iteratively solve for the correct $\bar{\kappa}$ by updating

$$\bar{\kappa} = \bar{\kappa} - \alpha \nabla_{\bar{\kappa}} \lambda \quad (21)$$

until λ reaches below a set tolerance, where α is a step size. The full pseudocode for this approach can be seen in Alg. 1.

We demonstrate excellent real2sim realization of the soft limb actuator through a DisMech model both visually (Fig. 5) and numerically (Fig. 6), where we achieve an average tip position error of just 2.9 mm (1.5% of the robot’s length). Future work, using more data, will derive separate actuator models (e.g. SMA voltages to natural curvatures), which in turn will allow for feedback controller design via sim2real.

VI. CONCLUSION

In this work, we introduced DisMech, a fully generalizable, implicit, discrete differential geometry-based physical simulator capable of accurate and efficient simulation of soft robots and structures composed of elastic rods. We physically validated DisMech through several simulations, using both representative

examples and new and complicated robots, showcasing an order of magnitude computational gain over previous methods. In addition, we also introduced a method for intuitively actuating soft continuum robots by bending energy manipulation via natural curvature changes, and demonstrated this actuation's use as a control input in real2sim of a soft manipulator.

Future work will involve integrating shells [31] into DisMech to allow for more complex deformable assemblies. In addition to this, the incorporation of shearing into DisMech will be a key research area to allow it to simulate all deformation modes similar to Cosserat rod-based frameworks [10], [8]. In fact, shearing has previously been integrated into a DDG-based framework when simulating a Timoshenko beam [34], albeit for just a 2D framework. Finally, efficient pipelines for real2sim2real modelling and training through reinforcement learning using DisMech will be a key research area moving forward, whether it be for soft robot control or deformable object manipulation.

REFERENCES

- [1] X. Lin, Y. Wang, J. Olkin, and D. Held, "Softgym: Benchmarking deep reinforcement learning for deformable object manipulation," *arXiv*, 2021.
- [2] A. Choi, D. Tong, D. Terzopoulos, J. Joo, and M. K. Jawed, "Deep learning of force manifolds from the simulated physics of robotic paper folding," *arXiv*, 2023.
- [3] D. Tong, A. Choi, L. Qin, W. Huang, J. Joo, and M. K. Jawed, "Sim2real neural controllers for physics-based robotic deployment of deformable linear objects," *The International Journal of Robotics Research*, vol. 0, no. 0, p. 02783649231214553, 2023.
- [4] W. Huang, X. Huang, C. Majidi, and K. Jawed, "Dynamic simulation of articulated soft robots," *Nature Communications*, vol. 11, 05 2020.
- [5] M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, and E. Grinspun, "Discrete elastic rods," in *ACM SIGGRAPH '08 Conference*, pp. 63:1–12, 2008.
- [6] M. Bergou, B. Audoly, E. Vouga, M. Wardetzky, and E. Grinspun, "Discrete viscous threads," *ACM Trans. Graph.*, vol. 29, jul 2010.
- [7] M. A. Graule, T. P. McCarthy, C. B. Teeple, J. Werfel, and R. J. Wood, "Somogym: A toolkit for developing and evaluating controllers and reinforcement learning algorithms for soft robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4071–4078, 2022.
- [8] A. T. Mathew, I. M. B. Hmida, C. Armanini, F. Boyer, and F. Renda, "Sorosim: A matlab toolbox for hybrid rigid-soft robots based on the geometric variable-strain approach," *IEEE Robotics & Automation Magazine*, pp. 2–18, 2022.
- [9] E. Coevoet, T. Morales-Bieze, F. Largilliere, Z. Zhang, M. Thieffry, M. Sanz-Lopez, B. Carrez, D. Marchal, O. Goury, J. Dequidt, and C. Duriez, "Software toolkit for modeling, simulation, and control of soft robots," *Advanced Robotics*, vol. 31, pp. 1–17, 11 2017.
- [10] M. Gazzola, L. Dudte, A. McCormick, and L. Mahadevan, "Forward and inverse problems in the mechanics of soft filaments," *Royal Society open science*, vol. 5, no. 6, p. 171628, 2018.
- [11] K. Becker, C. Teeple, N. Charles, Y. Jung, D. Baum, J. C. Weaver, L. Mahadevan, and R. Wood, "Active entanglement enables stochastic, topological grasping," *Proc Natl Acad Sci U S A*, vol. 119, p. e2209819119, Oct. 2022.
- [12] J. C. Pacheco Garcia, R. Jing, M. L. Anderson, M. Ianus-Valdivia, and A. P. Sabelhaus, "A comparison of mechanics simplifications in pose estimation for thermally-actuated soft robot limbs," in *Smart Materials, Adaptive Structures and Intelligent Systems*, ASME, 09 2023.
- [13] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, pp. 2149–2154 vol.3, 2004.
- [14] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.
- [15] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning." <http://pybullet.org>, 2016–2021.
- [16] M. A. Graule, C. B. Teeple, T. P. McCarthy, R. C. St. Louis, G. R. Kim, and R. J. Wood, "Somogym: Fast and accurate simulations of continuum robots in complex environments," in *2021 IEEE International Conference on Intelligent Robots and Systems (IROS)*, pp. 3934–3941, IEEE, 2021.
- [17] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, I. Peterlik, and S. Cotin, *SOFA: A Multi-Model Framework for Interactive Physical Simulation*, pp. 283–321. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.
- [18] X. Zhang, F. Chan, T. Parthasarathy, and M. Gazzola, "Modeling and simulation of complex dynamic musculoskeletal architectures," *Nature Communications*, vol. 10, no. 1, pp. 1–12, 2019.
- [19] C. Majidi, "Soft robotics: A perspective—current trends and prospects for the future," *Soft Robotics*, vol. 1, no. 1, pp. 5–11, 2014.
- [20] D. Tong, A. Choi, J. Joo, A. Borum, and M. Khalid Jawed, "Snap Buckling in Overhand Knots," *Journal of Applied Mechanics*, vol. 90, p. 041008, 01 2023.
- [21] M. Thieffry, A. Kruszewski, C. Duriez, and T.-M. Guerra, "Control Design for Soft Robots Based on Reduced-Order Model," *IEEE Robotics and Automation Letters*, vol. 4, pp. 25–32, Jan. 2019.
- [22] J. I. Alora, M. Cenedese, E. Schermerling, G. Haller, and M. Pavone, "Data-Driven Spectral Submanifold Reduction for Nonlinear Optimal Control of High-Dimensional Robots," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2627–2633, May 2023.
- [23] P. Schegg, E. Ménager, E. Khairallah, D. Marchal, J. Dequidt, P. Preux, and C. Duriez, "SofaGym: An open platform for reinforcement learning based on soft robot simulations," *Soft Robot*, vol. 10, pp. 410–430, Dec. 2022.
- [24] F. Largilliere, V. Verona, E. Coevoet, M. Sanz-Lopez, J. Dequidt, and C. Duriez, "Real-time control of soft-robots using asynchronous finite element modeling," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2550–2555, 2015.
- [25] G. Kirchhoff, "Über das gleichgewicht und die bewegung eines unendlich dünnen elastischen stabes," *J. Reine Angew. Math.*, vol. 56, pp. 285–313, 1859.
- [26] M. K. Jawed, F. Da, J. Joo, E. Grinspun, and P. M. Reis, "Coiling of elastic rods on rigid substrates," *Proceedings of the National Academy of Sciences*, vol. 111, no. 41, pp. 14663–14668, 2014.
- [27] A. Choi, D. Tong, M. K. Jawed, and J. Joo, "Implicit Contact Model for Discrete Elastic Rods in Knot Tying," *Journal of Applied Mechanics*, vol. 88, 03 2021.
- [28] D. Tong, A. Choi, J. Joo, and M. K. Jawed, "A fully implicit method for robust frictional contact handling in elastic rods," *Extreme Mechanics Letters*, vol. 58, p. 101924, 2023.
- [29] D. Tong, A. Borum, and M. K. Jawed, "Automated stability testing of elastic rods with helical centerlines using a robotic system," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1126–1133, 2022.
- [30] N. Naughton, J. Sun, A. Tekinalp, T. Parthasarathy, G. Chowdhary, and M. Gazzola, "Elastica: A compliant mechanics environment for soft robotic control," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3389–3396, 2021.
- [31] E. Grinspun, A. N. Hirani, M. Desbrun, and P. Schröder, "Discrete shells," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, (Goslar, DEU), p. 62–67, Eurographics Association, 2003.
- [32] W. Huang and M. K. Jawed, "Newmark-Beta Method in Discrete Elastic Rods Algorithm to Avoid Energy Dissipation," *Journal of Applied Mechanics*, vol. 86, p. 084501, 06 2019.
- [33] S. M. Han, H. Benaroya, and T. Wei, "Dynamics of transversely vibrating beams using four engineering theories," *Journal of Sound and Vibration*, vol. 225, no. 5, pp. 935–988, 1999.
- [34] X. Li, W. Huang, and M. K. Jawed, "A discrete differential geometry-based approach to numerical simulation of timoshenko beam," *Extreme Mechanics Letters*, vol. 35, p. 100622, 2020.