Collision-Aware Fast Simulation for Soft Robots by Optimization-Based Geometric Computing

Fang, Guoxin; Tian, Yingjun; Weightman, Andrew; Wang, Charlie C.L.

DOI
10.1109/IROS47612.2022.9981870

Publication date
2022

Document Version
Final published version

Published in
IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2022

2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 23-27, 2022, Kyoto, Japan

Collision-Aware Fast Simulation for Soft Robots by Optimization-Based Geometric Computing

Guoxin Fang*, Yingjun Tian*, Andrew Weightman, and Charlie C.L. Wang†

***Abstract*— Soft robots can safely interact with environments because of their mechanical compliance. Self-collision is also employed in the modern design of soft robots to enhance their performance during different tasks. However, developing an efficient and reliable simulator that can handle the collision response well, is still a challenging task in the research of soft robotics. This paper presents a collision-aware simulator based on geometric optimization, in which we develop a highly efficient and realistic collision checking / response model incorporating a hyperelastic material property. Both actuated deformation and collision response for soft robots are formulated as geometry-based objectives. The collision-free body of a soft robot can be obtained by minimizing the geometry-based objective function. Unlike the FEA-based physical simulation, the proposed pipeline performs a much lower computational cost. Moreover, adaptive remeshing is applied to achieve the improvement of the convergence when dealing with soft robots that have large volume variations. Experimental tests are conducted on different soft robots to verify the performance of our approach.**

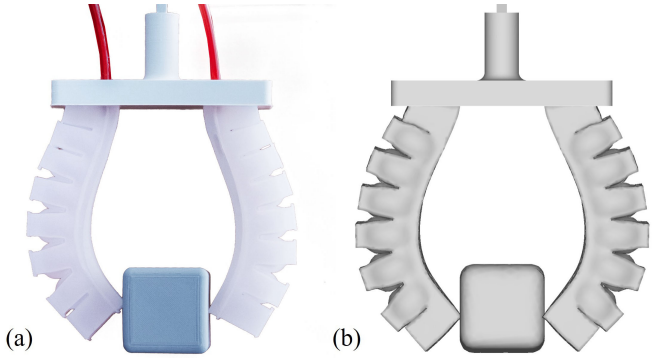

Fig. 1: Pneumatically actuated soft gripper that can effectively grasp objects by large inflation of chambers and the self-collision between neighboring chambers: (a) the physical result on a soft gripper made by silicone casting and (b) our simulation result that can well predict the shape of the soft gripper by considering both the robot-robot and the robot-obstacle collisions.

I. Introduction

Powered by the flexibility of soft materials and novel structure designs, soft robots are able to perform complex deformation in their body shapes and safely interact with other objects [1]. Thus, they have many advantages in certain tasks, such as grasping fragile objects and exploring confined environments [2], [3], which are challenging for conventional robots that have rigid bodies. When controlling rigid robots to complete tasks, robot-robot collisions and robot-environment collisions are generally prohibited. A simplified solution is to avoid contact. However, this is not the case for soft robots since the phenomenon of contact caused by collision plays an important role in allowing them to safely interact with environments and their own bodies. One should note that self-collision has been employed in modern design to achieve advanced functionality [4], [5]. For example, the soft gripper presented in Fig. 1 has the ability to interact and grasp objects with its soft body. Meanwhile, the self-collision that happens in the gap region between chambers can enhance the stiffness of the grasping and provide faster responses in bending deformation (ref. [6], [7]).

A. Challenges in Collision-Aware Simulation for Soft Robots

At present, the design process of soft robots heavily relies on the experience of engineers. Building effective computational tools is essential for accelerating the process. Studies have been conducted on using Finite-Element Analysis (FEA) simulation in the loop of design optimization (e.g., [8], [9]). Meanwhile, predicting the behavior of soft robot under actuation also plays an important role in building fast model-based controllers [10].

Efficiently simulating the deformation of soft robots while considering self-collisions and interactions with obstacles is still a very challenging problem. Incorporating the collision response into the simulation of soft robots requires to know their whole body shapes. In this case, an FEA-based simulator is more general and effective than the analytical models (e.g., [11]). Nevertheless, hyperelastic material properties and the highly nonlinear deformation presented on soft robots [12], [13] (e.g., expanding, twisting, and bending) make obtaining an accurate result from simulations time-consuming.

The challenges of building collision-aware simulation for soft robot are summarized in the following list.

1) *Complexity in deformation:* The involvement of highly nonlinear deformation makes FEA-based simulations difficult to converge in numerical iterations [14]. For example, the largely inflated chambers for pneumatic-driven soft robots can lead to highly-distorted elements for FEA, which introduces numerical instability.
2) *Fast collision checking through actuation:* Unlike articulated robots with rigid bodies, the freeform deformable body shape of a soft robot makes collision detection and self-collision detection more difficult to

All authors are with the Department of Mechanical, Aerospace and Civil Engineering, The University of Manchester, United Kingdoms
Guoxin Fang is also a PhD student at the Faculty of Industrial Design Engineering, Delft University of Technology, The Netherlands.
*Denotes equal contribution.
†Corresponding author: changling.wang@manchester.ac.uk

2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-7927-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/IROS47612.2022.9981870

conduct efficiently.

3) *Efficient collision response with hyperelastic material properties:* Once (self-)collisions are detected, a response algorithm needs to modify a soft robot's current shape to eliminate collided regions. Providing effective collision response while being able to mimic the complex physical behaviors of hyperelastic materials is challenging.

In this work, we propose a method of optimization-based geometric computing to tackle the mentioned challenges.

B. Related Work

To model the behavior of soft robot with interaction, full FEA-based simulations are most widely used due to their proofed high accuracy [8], [15]. However, the computational cost is high since a small time step needs to be used to solve the nonlinear geometry change (mainly rotational) with collision response. Reduced FEA-based numerical models that are based on voxel [16] or particle [17] approximation can achieve simulation with real-time speed. Based on off-the-shelf physics engine of rigid-body simulation, Graule *et al.* [18] built a model to predict the behavior of a continuum robot finger when interacting with a complex environment. The self-collision issue was not considered in these frameworks. A general simulator based on force-equilibrium function was developed by Duriez *et al.* [19], and it can achieve the speed of real-time simulation with the help of model reduction [20]. This work was recently enhanced by the self-collision response and has improved its accuracy when controlling cable-driven soft robots [7]. However, the (self-)collision region was defined during the pre-processing step and was fixed through actuation. Furthermore, their method based on integration along time steps accumulates the error; therefore, the simulation could become non-realistic for pneumatic-driven soft robots with large expanding and complex rotational deformation, which has been discussed in [21].

For applications such as cloth simulation and animation, research has been conducted in computer graphics (e.g., [22]–[24]) to tackle the problem of collision detection and response for deformable objects. Methods based on *bounding-volume hierarchies* (BVHs), distance fields, and spatial partitioning were utilized for fast collision detection (see [25] for a comprehensive review). In this work, BVH-based acceleration is applied to support real-time (self-)collision detection for soft robots. For collision response, the method presented in [26] was based on using constrained optimization to minimize the collided region between two objects. Penalty force-based method was also invited to accelerate the collision-response process with time integration [27], which however suffers from the difficulty of tuning parameters to avoid unrealistic results in simulations. Meanwhile, none of these works incorporates the hyperelastic properties of materials that are commonly used to fabricate soft robots. In this paper, we bring the idea from spring-mass system [28] and build the 'virtual' spring elements for collision response. When the spring energy for these elements is minimized, collision-free is achieved on the bodies of soft robots.

C. Our Method

In this work, we present an optimization-based simulator that can efficiently predict complicated deformation for soft robots and effectively provide realistic collision responses by incorporating hyperelastic material properties. Our method extends the pipeline of geometry computing previously presented in [14], [21]. The elastic energy in soft body is minimized to compute its deformed shape, and the actuation is formulated as constraints in terms of length, area or volume variation applied to some elements. After using axis-aligned bounding boxes (AABB) as BVH to support real-time (self-)collision detection, we introduce a group of 'virtual' spring elements to link collided points with their collision-free corresponding vertices. The hard-to-solve collision response constraint can then be reformulated as a quadratic energy term to be minimized. A local-global iterative solver is employed to find the collision-free shape under certain actuation. The geometry-based model [29] is adopted in our approach for hyperelastic materials, the parameters of which can be obtained from calibration.

The technical contributions of our work include the following:

- A geometry-based simulator that can effectively predict the shape of soft robots with large and complex deformation by incorporating adaptive remeshing;
- A fast collision checking method to detect both robot-robot and robot-obstacle interactions through the deformation;
- A spring-element assisted collision response model that incorporates hyperelastic material properties.

The generality and performance of our collision-aware soft robot simulator have been tested and verified on different soft robots fabricated from different materials. In all experimental tests, our simulator can successfully mimic the physical behaviors of soft robots with high accuracy. When compared with FEA-based methods, our simulator demonstrated better performance in its efficiency and numerical convergence capability. The code of our simulator is open-source at https://github.com/YingGwan/collisionAwareSOROSimulator.

II. Geometry-based Simulation for Soft Robot with Collisions

This section first presents how to formulate the problem of predicting the deformed shape of soft robot under certain actuation as a constrained optimization problem that can be solved by geometric computing. The fast collision detection is enabled by BVH based search. The collision response model will be formulated by adding 'virtual' spring elements into the constrained optimization framework.

A. Geometry-based optimization for modelling soft robots

As illustrated in Fig.2, we require an input of two surface meshes $\mathcal{S}_c$ and $\mathcal{S}_b$ that represent the initial shape of a soft

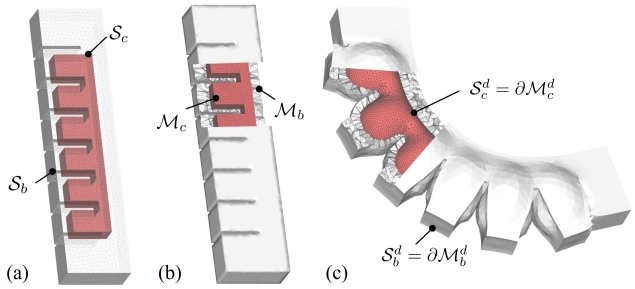

Fig. 2: (a) The input shape of a soft robot is represented by two surface meshes $\mathcal{S}_c$ (as chamber) and $\mathcal{S}_b$ (as body). (b) Volume tessellation is applied to generate tetrahedral meshes $\mathcal{M}_c$ and $\mathcal{M}_b$. (c) Inflation in the chamber region drives the soft robot being deformed to $\mathcal{M}^d = \{\mathcal{M}_c^d, \mathcal{M}_b^d\}$. Adaptive remeshing is applied to update $\mathcal{M}_c$ during the deformation.

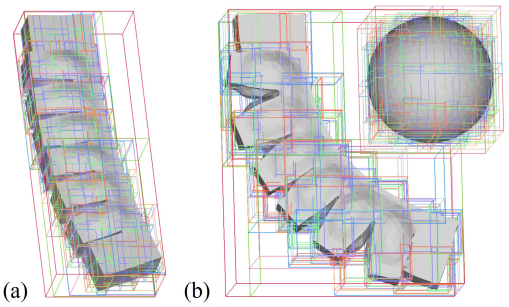

Fig. 3: (a) AABB trees are built for both soft robot and obstacle for fast collision detection. (b) The structure of BVHs can be re-used when the deformation is relatively small – i.e., only the dimensions of the bounding-boxes need to be updated.

robot in the form of its chamber and body, respectively. Volumetric tessellation (function remarked as $\mathcal{T}(\cdot)$) is applied to generate a tetrahedral mesh that interpolates these two surface meshes. Note that the tetrahedral elements are generated between $\mathcal{S}_c$ and $\mathcal{S}_b$ and also inside $\mathcal{S}_c$. $\mathcal{M}$ is segmented into two topologically connected parts $\mathcal{M}_c$ and $\mathcal{M}_b$. Elements in $\mathcal{M}$ are categorized as *chamber element* ($e \in \mathcal{M}_c$, depicts in red) and *body element* ($e \in \mathcal{M}_b$, depicts in gray). We remark the shape of each element by matrix $\mathbf{V}_e = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4] \in \mathbb{R}^{3\times4}$, where $\mathbf{v}_i$ gives the position of the i-th vertex.

Without loss of generality, when pneumatic actuation is applied, the deformation of soft robot is driven by the volume expansion in chamber elements here[1]. Meanwhile, the shape of body region will be changed as the elements are topologically connected. Here a target volume expansion ratio α is given as the input actuation parameter. In the geometry-based simulation pipeline [14], shapes of the body elements were computed by minimizing the elastic energy E_{ela}. This energy is evaluated by the shape difference between the element's *deformed shape* $\mathbf{V}^d$ and its *target shape* $\mathbf{V}^t$ (details will be presented in Sec. III-A). Meanwhile, collisions can occur either between different bodies of the soft robot (i.e., self-collision) or between the soft robot and external obstacles.

The optimization problem that computes deformed shape $\mathcal{M}^d$ is formulated as

$$\arg\min_{\mathcal{M}^d} \quad E_{ela} = \sum_{e\in\mathcal{M}_b} Vol(e)\|\mathbf{N}\mathbf{V}^d - \mathbf{R}(\mathbf{N}\mathbf{V}^t)\|^2 \tag{1}$$

$$s.t. \quad Vol(\mathcal{M}_c^d) = \alpha \sum_{e\in\mathcal{M}_c} Vol(e) \tag{2}$$

$$\mathbf{v} \cap e = \emptyset \ (\forall \mathbf{v} \in \mathcal{S}_c, e \in \mathcal{M}) \tag{3}$$

$$\mathcal{M} \cap \mathcal{O}_{obs} = \emptyset \tag{4}$$

In the definition of elastic energy (Eq.(1)), the matrix $\mathbf{N} = \mathbf{I}_{4\times4} - \frac{1}{4}\mathbf{1}_{4\times4}$ is used to transform the element to the coordinate origin. Rotation matrix $\mathbf{R}$ is computed by *single value decomposition* (SVD) and aligns $\mathbf{V}^d$ and $\mathbf{V}^t$ to the same pose. The operator $Vol(\cdot)$ in the actuation constraint (Eq.(2)) computes the volume of given element. The self-collision-free requirements are formulated as hard constraints (Eq.(3)) where each vertex located at the boundary of $\mathcal{M}_b$ (remark as $\mathbf{v} \in \partial\mathcal{M}_b = \mathcal{S}_c$) should not go inside any of the other tetrahedral elements on $\mathcal{M}$. Interaction with the obstacles are handled by the last hard constraint (Eq.(4)).

To effectively solve this nonlinear system, a projection-based local-global solver is introduced in our previous work [14]; however, we need to deal with the newly added collision constraints here (i.e., Eqs.(3) and (4)). The details of this solver incorporating collision response are presented in Sec. III. We first discuss the method for fast collision detection and our collision response model in the following section.

B. Fast (self-)collision detection with BVHs

For the purpose of collision detection, a brute-force solution is to check all tetrahedra for every vertex on the surface of a soft robot to see if any intersection happens. This method however introduces a high computational cost when dense meshes are involved. An algorithm based on BVHs is utilized to accelerate the search by avoiding unnecessary checks of collision pairs during the computation.

Firstly, the *axis-aligned bounding box* (AABB) trees [30] are constructed for both $\mathcal{M}$ and $\mathcal{O}_{obs}$. The bounding-box of a whole model is set as the tree's root, and the nodes in different levels are built to split elements into smaller and smaller bounding boxes until reaching the leaf nodes that only contain one tetrahedron (illustrated in Fig. 3). Both the self-collision and obstacle interaction can be efficiently detected by the traversal on the AABB trees, which has the complexity of $\mathcal{O}(n\log(n))$ with n being the number of elements.

AABB trees are employed in our framework instead of other BVHs since the structure of the tree can be kept through the deformation while only the geometry of bounding boxes is updated by the new positions of tetrahedra. The tightness of bounding will become loose after presenting large deformation on the soft bodies. The trees can be re-built when necessary. In all our examples presented in this

[1]For the actuation based on length or area variations, they can be formulated in a similar way. Details can be found in [14].

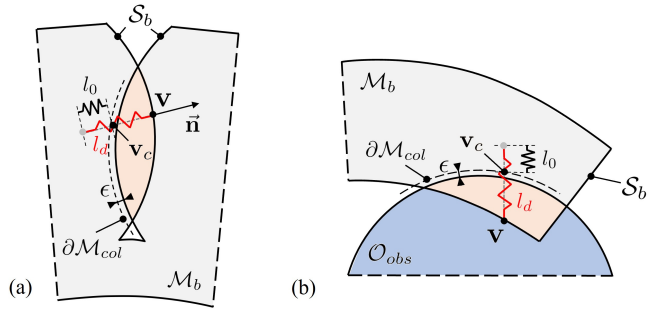

Fig. 4: Illustration of building 'virtual' spring elements for collision response. The starting point of a spring can be conceptually considered as a point through the direction of $\mathbf{v}\vec{\mathbf{v}}_c$ inside the model. (a) Self-collision happens between two chamber regions that both belong to the body mesh $\mathcal{M}_b$. (b) Interaction between $\mathcal{M}_b$ and $\mathcal{O}_{obs}$.

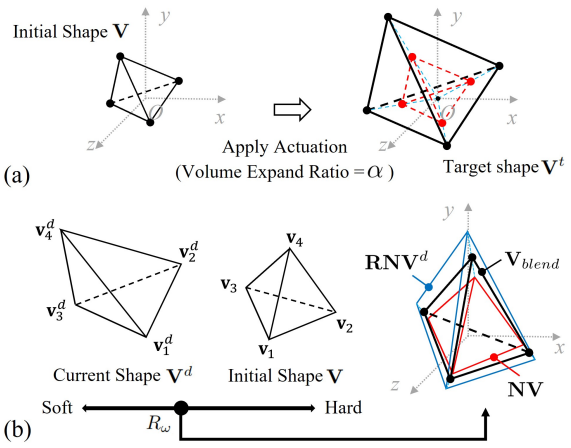

Fig. 5: Computing the target shapes for (a) a chamber element with given actuation parameter α and (b) a body element ($\mathbf{v} \in \mathcal{M}_b$) by incorporating material parameter R_ω.

paper, we only rebuild the AABB trees after remeshing. The collision detection can always be efficiently conducted on AABB trees without re-building. Based on our test, (self)-collision detection can be completed at 24fps for a mesh that contains 40k tetrahedra. In each iteration, we stored all collided vertices in the set $\mathcal{V}_{col}$. Statistics regarding the computational time for our collision detection are listed in Sec. IV.

C. Collision response by spring elements

After detecting the collided regions, we build virtual springs on every vertex $\mathbf{v} \in \mathcal{V}_{col}$ for the purpose of collision response. As illustrated in Fig. 4, when a vertex $\mathbf{v}$ goes inside the collided region (depicted in orange), we can consider adding a virtual spring between $\mathbf{v}$ and the predicted position $\mathbf{v}_c$ where it should contact the interface after removing collision. Without loss of generality, we can conceptually assume the spring's length was changed from its resting length l_0 to its deformed status l_d. To compute the collision-free shape using geometric optimization, the 'virtual' spring will drag the collided vertex $\mathbf{v}$ moving out to a collision-free position $\mathbf{v}_c$. This collision response can be formulated as minimizing the spring energy:

$$E_{col} = k(l_d - l_0)^2 = k\|\mathbf{v} - \mathbf{v}_c\|^2. \tag{5}$$

When E_{col} is minimized close to zero, the vertex would no longer collide with the model itself or other obstacles.

For the case of self-collision as illustrated in Fig. 4(a), the corresponding position $\mathbf{v}_c$ is computed by searching along the inverse normal direction $-\vec{n}(\mathbf{v})$ from $\mathbf{v}$ until intersecting with the boundary of collided region (remarked as $\partial\mathcal{M}_{col}$). In the case of robot-obstacle interaction, $\mathbf{v}_c$ is found to be the closest point on $\partial\mathcal{M}_{col}$ (see Fig. 4(b)). To avoid numerical error, $\partial\mathcal{M}_{coll}$ is computed by giving a small offset ϵ to the boundaries of robots and obstacles.

During the iteration of deformation, corresponding points need to be updated accordingly. The spring elements also need to be added or removed through the iteration to avoid unrealistic adhesion behavior. In the next section, we will present the details of adaptive collision response and a local-global solver that minimizes $E_{ela} + E_{col}$ together with the actuation constraints (i.e., Eq.(2)).

III. Projection-Based Solver with Remeshing

By incorporating collision constraints as the spring energy to be minimized, the collision-aware simulation for soft robots is reformulated as an optimization problem. A local-global iterative solver is used to solve this system. In the local step, the target shapes for chamber elements, body elements, and spring elements are computed by local projection. These target shapes are set to satisfy the corresponding objectives or constraints. In this local step, collision detection is also applied to update spring elements accordingly. After that, a global blending step is applied to assembly all elements according to their topological connections. Repeatedly applying the local projection and the global blending steps results in the collision-free shape of a soft robot. Meanwhile, progressive remeshing is conducted to ensure a stable and realistic result when simulating soft robots with very large chamber deformation.

A. Computing target shapes local projection

Here we introduce how the target shapes are computed for different types of elements.

1) Chamber element with actuation constraint: The target shapes for chamber elements with volume variation are computed by an average expansion in all verities as illustrated in Fig. 5(a). With actuation parameter α as input, the target position for the i-th vertex in the element is determined as $\mathbf{v}_i^t = \sqrt[3]{\alpha}\mathbf{v}_i$. This update ensures that the volume of the chamber region is updated to α times to its initial status that satisfies constraint (Eq.(2)).

2) Body element with hyperelastic material property: The property of soft material needs to be integrated when computing the target shape for body elements. We modify the scheme of shape blending in [14] by inviting strain-related stiffness parameter to mimic the behavior of hyperelasticity

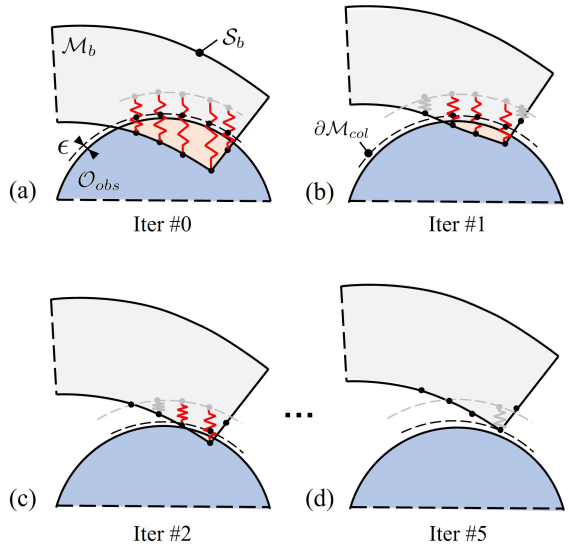

Fig. 6: Iterative updating spring elements and their correspondences $\mathbf{v}_c$ in collision response. (a) Initial status contains spring elements added on detected collision vertices. (b, c) Through the iteration, the spring elements are deactivated (shows in gray) when $\mathbf{v}$ has been pulled out of collided region $\mathcal{M}_{col}$. (d) Final collision-free status with all spring elements released.

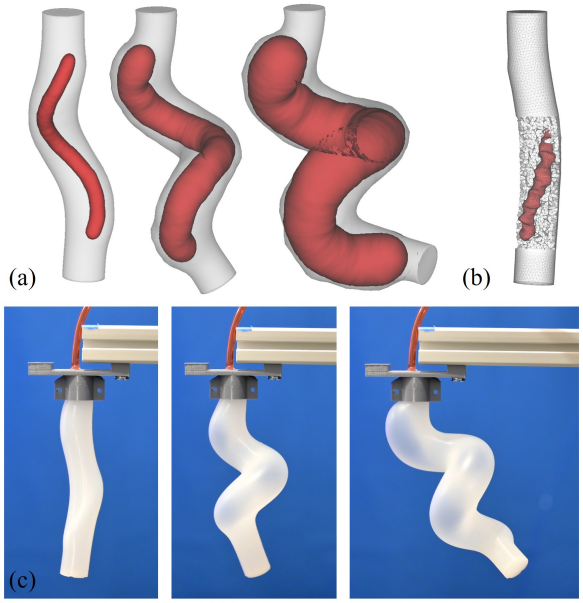

Fig. 7: Simulation results for a soft robot with twisting deformation under actuation. (a) By progressively remeshing the chamber region $\mathcal{M}_c$ (shows in red), our method can generate the results that mimic the physical behavior shown in (c) as applying different volume-based actuation parameters (from left to right: $\alpha = 5, 28$, and 115). (b) Directly applying a large actuation parameter $\alpha = 28$ leads to an unrealistic result in simulation.

in soft material. As illustrated in Fig. 5(b), the projected shape is first computed by combing the initial shape and the current shape of an element. The target shape is computed by

$$\mathbf{V}_{blend} = (1 - R_\omega(\varepsilon))\mathbf{R}\mathbf{N}\mathbf{V}^d + R_\omega(\varepsilon)\mathbf{N}\mathbf{V}, \quad (6)$$

$$\mathbf{V}^t = \frac{Vol(\mathbf{V}^t)}{Vol(\mathbf{V}_{blend})}\mathbf{V}_{blend}. \quad (7)$$

Here the parameter R_ω is a function of the strain ε (detail presented in [29]) and can be calibrated from physical test. Meanwhile, to ensure incompressibility of hyperelastic materials (i.e., with Poisson's ratio around 0.5), volume scaling is applied to this element. In our experiments, two silicon rubbers Ecoflex 00-30 and Dargon Skin 20A are modeled since they are widely used in fabricating soft robots [4], [12], [13]. Detailed data for these materials is provided in Sec. IV

3) Spring element with updated correspondence: As explained in Sec. II-C, the target shape of a spring element is set as the spring's initial status (i.e., letting the collided point $\mathbf{v}$ move to its corresponding point $\mathbf{v}_c$). In each iteration, the target shape for a spring element needs to be updated together with collision detection. This process is illustrated and explained in Fig. 6. An existing spring element will be released if the vertex $\mathbf{v}$ has been outside the collided region and is away from the boundary of $\mathcal{M}_{col}$. That is

$$\mathbf{v} \notin \mathcal{O}_{obs}, \ ||\vec{\mathbf{n}} \cdot (\mathbf{v} - \mathbf{v}_c)|| > \epsilon. \quad (8)$$

As an example, springs shown in Fig. 6(b) with a gray color are to be removed in the next iteration since they have satisfied the mentioned condition (Eq.(8)). The iteration of collision response terminates when all the springs are released (i.e., $\mathcal{V}_{col} = \emptyset$ and see Fig. 6(d) for an example).

B. Local-global solver with progressive remeshing

After defining the target shapes for all elements, the optimization problem can be transformed to an augmented form that minimizes the energy

$$\sum_{e \in \mathcal{M}_b, \mathcal{M}_c} \omega_e ||\mathbf{N}\mathbf{V}^d - \mathbf{R}(\mathbf{N}\mathbf{V}^t)||^2 + \sum_{\mathbf{v} \in \mathcal{V}_{col}} ||\mathbf{v} - \mathbf{v}_c||^2. \quad (9)$$

By adopting the strategy of local-global solver, the variables $\mathbf{V}^t$, $\mathbf{R}$, and $\mathbf{v}_c$ are computed in the *local step* by shape projection. They will be fixed in the *global step* of blending. Therefore, the energy minimization problem defined in Eq.(9) becomes a least-square problem as only the position of vertices in $\{\mathbf{V}^d\}$ are unknown variables. The local and global steps are applied in an iterative manner, and it has been proved that this solver can effectively minimize the energy in a few steps. In our implementation, the Anderson-Acceleration method [31] is applied to further improve the converging speed of numerical computations.

When directly applying a large volume-variation based actuation in the simulation solver, some unrealistic results could be obtained (see Fig. 7(b) for an example). This is caused by a very sparse density of elements in the region of chamber $\mathcal{M}_c$ in the robot's initial shape. To tackle this issue, we progressively apply actuation and remesh the chamber region $\mathcal{M}_c = \mathcal{T}(\mathcal{S}_c^d)$ when necessary. Specifically after deforming a soft robot, we apply the remeshing step if any of the chamber elements is detected either

1) having $Vol(\mathbf{V}^d)/Vol(\mathbf{V}) > \alpha_{max}$ as a significant volume change, or

ALGORITHM 1: Collision-Aware Soft Robot Simulation

Input: Soft robot model $\mathcal{S}_c$ and $\mathcal{S}_b$, actuation parameter α, obstacle model $\mathcal{O}_{obs}$.
Output: Deformed collision-free soft robot body shape $\mathcal{S}_b$.

Generate FE model $\mathcal{M} = \mathcal{M}_c + \mathcal{M}_b$ by $\mathcal{T}(\mathcal{S}_c, \mathcal{S}_b)$;
$k = 0$, $\alpha_k = \alpha_{max}, \mathcal{M}_c^k = \mathcal{M}_c$;
while $Vol(\mathcal{M}_c^k) < \alpha\, Vol(\mathcal{M}_c)$ **do**
  /* Collision-aware local-global solver */
  while $\mathcal{V}_{coll} \neq \emptyset$ *and* $i < i_{max}$ **do**
    $\mathbf{V}^t, \mathbf{R}, \mathbf{v}_c \leftarrow$ *local-step*$(\mathcal{V}_{coll}, \mathcal{M}^i, \alpha_k)$;
    $\mathcal{M}^{i+1} \leftarrow$ *global-step*$(\mathbf{V}^t, \mathbf{R}, \mathbf{v}_c)$;
    update $\mathcal{V}_{coll} \leftarrow$ *CollisionCheck*$(\mathcal{M}^i, \mathcal{O}_{obs})$;
  end
  /* Check if remeshing needed */
  for $e \in \mathcal{M}_c$ **do**
    if $Vol(e_{k+1})/Vol(e_k) > \alpha_{max}$ *or* $\mathbf{\Sigma}(e) > d_{max}$ **then**
      Remesh and update $\mathcal{M}^k = \mathcal{M}_b^k + \mathcal{T}(\mathcal{S}_c^k)$;
    end
  end
  $k = k + 1$, and $\alpha_k = k\alpha_{max}$;
end
return $\mathcal{S}_b = \partial\mathcal{M}_b^k$;

2) with $norm(\mathbf{\Sigma}) > d_{max}$ reflecting tremendous distortion in its shape.

Here $\mathbf{\Sigma}$ is the diagonal scaling matrix obtained using the SVD of the affine transformation matrix from $\mathbf{V}^d$ to $\mathbf{V}$. The threshold values $\alpha_{max} = 4.0$ and $d_{max} = 4.0$ are selected based on the experiments.

An example is presented in Fig. 7 to demonstrate the importance and effectiveness of this remeshing step, where a pneumatic-driven twisting soft robot can have the chamber region expanded more than 100 times in comparison to its initial volume. By applying progressive remeshing, the simulation result of a twisting soft robot can realistically match with the result of physical experiments (illustrated in Fig. 7(c)). The local-global solver is illustrated in **Algorithm 1**, in which the remeshing and collision response are presented.

IV. Case Studies and Experiment Results

In this section, we demonstrate the behavior of our collision-aware soft robot simulator by presenting several case studies. The proposed computational framework was implemented with C++ and run with a PC with AMD 3950x CPU and 32GB memory. The parallel computing was supported by OpenMP, and the Eigen library was used to solve the linear system. The simulation results were also compared to physical experiments to verify the performance of the approach.

A. Twisting soft robot

The first case study is conducted on a plant inspired soft robot [13] that can perform twisting deformation. The robot consists of a cylindrical body and a spiral-shape chamber. Eco-Flex 00-30 silicon rubber is used to fabricate the robot, and we model its hyperelastic material property by using first-order Mooney-Rivlin with c_1 = 0.0418 MPa and c_2 = 0.0106 MPa [13]. The simulation results are presented in Fig. 7(b) and can mimic the physical behavior well when remeshing is applied to the chamber region.

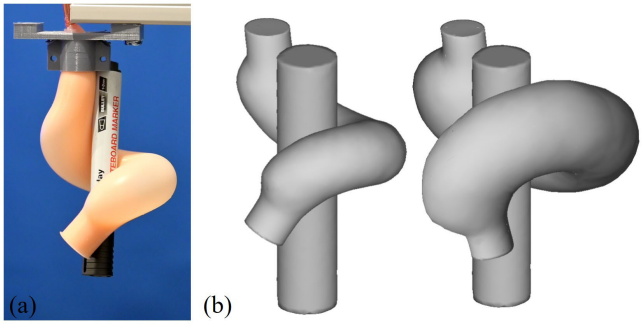

Fig. 8: Case study taken on a twisting soft robot – (a) physical experiment of using the twisting soft robot to hold a cylindrical object, and (b) our simulator can successfully model the collision behavior when the obstacle is added and when large pneumatic actuation is applied.

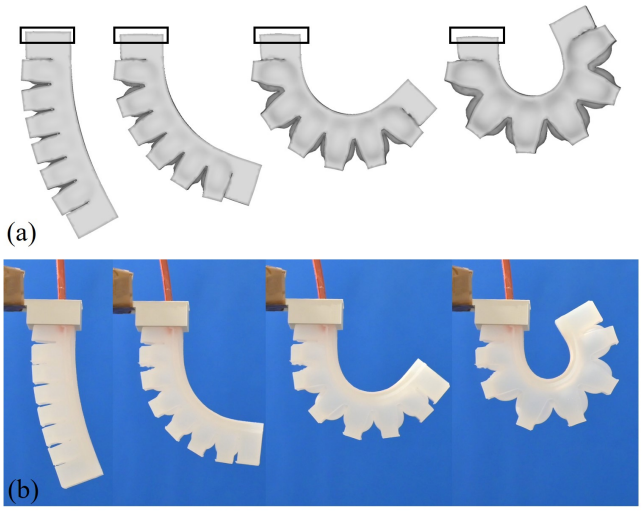

Fig. 9: (a) Simulation result for the soft finger made by silicone rubber and can perform large bending deformation. (b) The simulation result can well capture the physical behavior.

This twisting soft robot is able to firmly twine a target object. As depicted in Fig. 8, a cylindrical obstacle is tightly grasped when the soft robot is actuated into a shape around the cylinder. This behavior is also successfully simulated by our method, where larger actuation can be further added to the chamber region to produce a tighter grasping action (see the right of Fig. 8(b)). The mesh model used for simulating this soft robot contains 28.2k tetrahedra. With this size of mesh model, our simulator can achieve a computing speed of 8.09 seconds per time step. This is around 25 times faster than a FEA-based simulation (conducted by Abaqus software) that takes more than 3 minutes on a mesh with a similar density. Commercial software such as Abaqus also suffers from the convergence problem due to the distortion in elements when large inflation is applied (details can be found in Case II in the supplementary video).

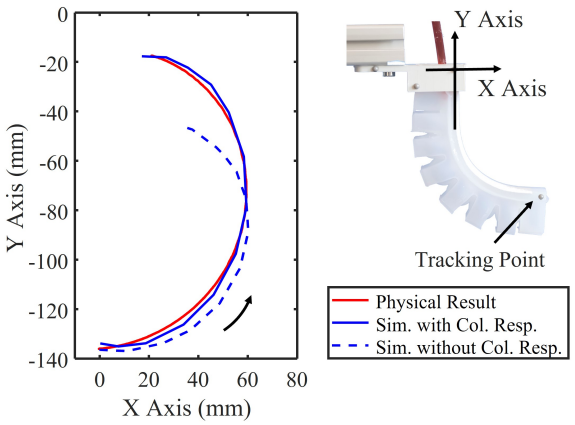

Fig. 10: The simulator can predict the tip trajectory of the soft finger more precisely after integrating the collision response model.

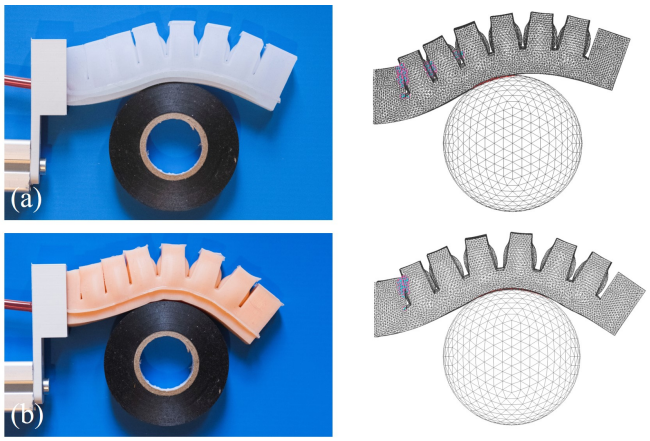

Fig. 11: Behavior of soft grippers while interacting with an obstacles with round shape. The soft grippers were fabricated using different types of silicon rubber, such as (a) Dragon Skin 20 and (b) Eco-flex 00-30. Our simulator results (depicted on the right) successfully predict both behaviors with corresponding material properties, respectively.

B. Soft finger with self-collision

The second case study is conducted on a soft finger model, where both self-collision and robot-obstacle interactions occur at the same time. The result of free-bending considering self-collision between chambers is presented in Fig. 9, and the comparison on the tracked tip position can be found in Fig. 10. Without an effective collision response, the simulator presented in our previous work [14] cannot precisely mimic the behavior of soft finger (i.e., large tracking error occurs in the tip trajectory). By using the collision-aware method presented in this paper, the error is reduced.

We also use two soft materials, Eco-flex 0030 and Dargon Skin 20, to fabricate two soft fingers for comparison. The Dargon Skin 20 material is around 3 times stiffer, and the Mooney–Rivlin model with $c_1 = 0.119$ MPa, $c_2 = 0.023$ MPa is used. As shown in Fig. 11, two soft robots perform differently when interacting with obstacles. With a softer material, the contact region becomes larger and the resultant shape is more around the obstacle's boundary. This phenomenon of different materials is well captured by our simulator with the incorporated hyperelastic material model.

C. Soft membrane driven by interactions

To verify the behavior in interactions between multiple soft robots, the proposed method is applied to simulate the behavior of a soft membrane setup as illustrated in Fig. 12. The membrane can be deformed to a free-form shape through interactions with four pneumatic-driven chambers below it [32], [33]. The size of the setup is 250×250 mm, which is fabricated using Dragon Skin 20. Our model precisely captures the shape of the outer membrane according to the given actuation parameters applied to the inner chamber array. The predicted shape is compared to the shape captured from physical setup by motion capture system. The maximum error is 5.0 mm as shown in Fig. 12, which is 2% of the hardware setup's size. The simulated motion of the membrane can also be found as case IV in the supplementary video.

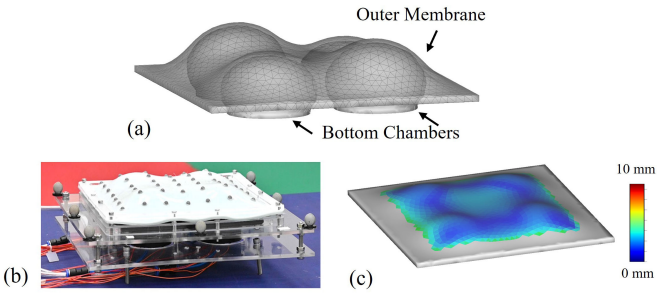

Fig. 12: Soft membrane driven by interaction with chambers at the bottom – (a) simulation result, (b) physical result on hardware, and (c) quantities analysis of the shape estimation error.

D. Discussion and Limitation

The statistics of computing costs are listed in Table I. In general, the computing speed of our simulator is around 25 times faster than the FEA-based simulation on the mesh models with similar density. In comparison to other reduced FEA-based numerical models (e.g., SOFA [19]) that have faster computing speeds, our method is more robust when large nonlinear deformations happen, which guarantees converging with a realistic simulation result. In all steps of our simulation, solving the optimization problem Eq.(9) takes more than 45% of the simulation time. Collision detection is

TABLE I: Computational cost of the collision-aware simulator†

Models	Tet. #	Mesh Gener.	Col. Check	Col. Resp.	Iter. Solver
Soft Finger	79,048	3.54 s	0.07 s	4.38 s	7.46 s
Twisting Soft Robot	28,198	2.81 s	0.03 s	1.71 s	3.54 s
Soft Membrane	34,680*	0.96 s	0.12 s	1.63 s	3.83 s

†The computing time is for single step with given actuation input.
*Contains four chamber, each one with 8,670 tetrahedrons.

run in real time (i.e., less than 0.12 seconds for all cases), and collision response consumes 25% of the total time. The time used to generate tetrahedral mesh can be further reduced by inviting a more efficient method for volume tessellation [34].

The major limitation of our current work is that the prediction of contact force is not included in the proposed collision-response model. One solution could be first computing the strain for each element based on the computed deformation, and then modelling the external force by strain-stress relationship with force equivalence function. This can be an interesting future work. Meanwhile, only the friction-less contact is considered in this work – see the soft robot sliding on the contact plan as demonstrated in Case I of the supplementary video. It's in general hard to model frictions precisely since the stick/slip condition brings a non-smooth collision response process that is hard to be computed efficiently.

V. Conclusion

In this paper, we develop a collision-aware soft robot simulator based on geometric computing. Compared with existing FEA-based approaches, the proposed method demonstrates a much faster computational speed and very robust convergence with complex deformation in a robot's body shape (e.g., expanding, twisting, and bending). Both self-collision and robot-obstacle interactions are successfully modeled. We have tested the simulator on a variety of soft robot systems fabricated from different materials with hyperelastic property, where our results can accurately simulate the physical deformation of soft robots.

References

[1] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, pp. 467–475, 2015.

[2] K. C. Galloway, K. P. Becker, B. Phillips, J. Kirby, S. Licht, D. Tchernov, R. J. Wood, and D. F. Gruber, "Soft robotic grippers for biological sampling on deep reefs," *Soft robotics*, vol. 3, no. 1, pp. 23–33, 2016.

[3] F. Connolly, P. Polygerinos, C. J. Walsh, and K. Bertoldi, "Mechanical programming of soft actuators by varying fiber angle," *Soft Robotics*, vol. 2, no. 1, pp. 26–32, 2015.

[4] A. D. Marchese, R. K. Katzschmann, and D. Rus, "A recipe for soft fluidic elastomer robots," *Soft robotics*, vol. 2, no. 1, pp. 7–25, 2015.

[5] T. Morales Bieze, A. Kruszewski, B. Carrez, and C. Duriez, "Design, implementation, and control of a deformable manipulator robot based on a compliant spine," *The International Journal of Robotics Research*, vol. 39, no. 14, pp. 1604–1619, 2020.

[6] B. Mosadegh, P. Polygerinos, C. Keplinger, S. Wennstedt, R. F. Shepherd, U. Gupta, J. Shim, K. Bertoldi, C. J. Walsh, and G. M. Whitesides, "Pneumatic networks for soft robotics that actuate rapidly," *Advanced functional materials*, vol. 24, no. 15, pp. 2163–2170, 2014.

[7] O. Goury, B. Carrez, and C. Duriez, "Real-time simulation for control of soft robots with self-collisions using model order reduction for contact forces," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3752–3759, 2021.

[8] P. Moseley, J. M. Florez, H. A. Sonar, G. Agarwal, W. Curtin, and J. Paik, "Modeling, design, and development of soft pneumatic actuators with finite element method," *Advanced engineering materials*, vol. 18, no. 6, pp. 978–988, 2016.

[9] Y. Chen, Z. Xia, and Q. Zhao, "Optimal design of soft pneumatic bending actuators subjected to design-dependent pressure loads," *IEEE/ASME Trans. Mechatron.*, vol. 24, no. 6, pp. 2873–2884, 2019.

[10] T. G. Thuruthel, E. Falotico, F. Renda, and C. Laschi, "Model-based reinforcement learning for closed-loop dynamic control of soft robotic manipulators," *IEEE Trans. Robot.*, vol. 35, no. 1, pp. 124–134, 2018.

[11] G. Zhong, W. Dou, X. Zhang, and H. Yi, "Bending analysis and contact force modeling of soft pneumatic actuators with pleated structures," *International Journal of Mechanical Sciences*, vol. 193, p. 106150, 2021.

[12] A. D. Marchese and D. Rus, "Design, kinematics, and control of a soft spatial fluidic elastomer manipulator," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 840–869, 2016.

[13] M. Yang, L. P. Cooper, N. Liu, X. Wang, and M. P. Fok, "Twining plant inspired pneumatic soft robotic spiral gripper with a fiber optic twisting sensor," *Optics Express*, vol. 28, no. 23, pp. 35 158–35 167, 2020.

[14] G. Fang, C.-D. Matte, R. B. Scharff, T.-H. Kwok, and C. C. Wang, "Kinematics of soft robots by geometric computing," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1272–1286, 2020.

[15] P. Polygerinos, Z. Wang, J. T. Overvelde, K. C. Galloway, R. J. Wood, K. Bertoldi, and C. J. Walsh, "Modeling of soft fiber-reinforced bending actuators," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 778–789, 2015.

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

[17] Y. Hu, J. Liu, A. Spielberg, J. B. Tenenbaum, W. T. Freeman, J. Wu, D. Rus, and W. Matusik, "Chainqueen: A real-time differentiable physical simulator for soft robotics," in *2019 International conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6265–6271.

[18] M. A. Graule, C. B. Teeple, T. P. McCarthy, G. R. Kim, R. C. S. Louis, and R. J. Wood, "Somo: Fast and accurate simulations of continuum robots in complex environments," in *IEEE Int. Conf. Intell. Robots Syst.* IEEE, 2021, pp. 3934–3941.

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

[20] 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, 2018.

[21] G. Fang, C. Matte, T. Kwok, and C. C. L. Wang, "Geometry-based direct simulation for multi-material soft robots," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–6, May 2018.

[22] D. Dinev, T. Liu, J. Li, B. Thomaszewski, and L. Kavan, "Fepr: Fast energy projection for real-time simulation of deformable objects," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–12, 2018.

[23] 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.

[24] J. Li, T. Liu, and L. Kavan, "Soft articulated characters in projective dynamics," *IEEE Trans. Vis. Comput. Graph.*, 2020.

[25] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M.-P. Cani, F. Faure, N. Thalmann, W. Straßer, and P. Volino, "Collision detection for deformable objects," *Comput. Graph. Forum*, vol. 24, pp. 61–81, 03 2005.

[26] J. Allard, F. Faure, H. Courtecuisse, F. Falipou, C. Duriez, and P. G. Kry, "Volume contact constraints at arbitrary resolution," *ACM Trans. Graph.*, vol. 29, no. 4, jul 2010.

[27] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically deformable models." Association for Computing Machinery, 1987.

[28] T. Liu, A. W. Bargteil, J. F. O'Brien, and L. Kavan, "Fast simulation of mass-spring systems," *ACM Transactions on Graphics*, vol. 32, no. 6, pp. 209:1–7, Nov. 2013.

[29] C.-D. Matte and T.-H. Kwok, "Simulation of Hyperelasticity by Shape Estimation," *Journal of Computing and Information Science in Engineering*, vol. 21, no. 5, 05 2021, 050903.

[30] G. van den Bergen, "Efficient collision detection of complex deformable models using aabb trees," *J. Graph. Tools*, vol. 2, no. 4, p. 1–13, jan 1998.

[31] Y. Peng, B. Deng, J. Zhang, F. Geng, W. Qin, and L. Liu, "Anderson acceleration for geometry optimization and physics simulation," *ACM Transactions on Graphics*, vol. 37, no. 4, pp. 1–14, 2018.

[32] R. Scharff, G. Fang, Y. Tian, J. Wu, J. Geraedts, and C. Wang, "Sensing and reconstruction of 3-d deformation on pneumatic soft robots," *IEEE/ASME Trans. Mechatron.*, vol. 26, no. 4, 2021.

[33] Y. Tian, G. Fang, J. S. Petrulis, A. Weightman, and C. C. L. Wang, "Soft robotic mannequin: Design and algorithm for deformation control," *IEEE/ASME Trans. Mechatron.*, pp. 1–10, 2022.

[34] F. Labelle and J. R. Shewchuk, "Isosurface stuffing: fast tetrahedral meshes with good dihedral angles," in *ACM SIGGRAPH 2007 papers*, 2007, pp. 57–es.