Kinematics of Soft Robots by Geometric Computing

Guoxin Fang, *Student Member, IEEE*, Christopher-Denny Matte, Rob B.N. Scharff, *Student Member, IEEE*, Tsz-Ho Kwok, and Charlie C.L. Wang, *Senior Member, IEEE*

***Abstract*—Robots fabricated with soft materials can provide higher flexibility and thus better safety while interacting in unpredictable situations. However, the usage of soft material makes it challenging to predict the deformation of a continuum body under actuation and therefore brings difficulty to the kinematic control of its movement. In this paper, we present a geometry-based framework for computing the deformation of soft robots within the range of linear material elasticity. After formulating both manipulators and actuators as geometry elements, deformation can be efficiently computed by solving a constrained optimization problem. Because of its efficiency, forward and inverse kinematics for soft manipulators can be solved by an iterative algorithm with low computational cost. Meanwhile, components with multiple materials can also be geometrically modeled in our framework with the help of a simple calibration. Numerical and physical experimental tests are conducted on soft manipulators driven by different actuators with large deformation to demonstrate the performance of our approach.**

***Index Terms*—Kinematics, soft robotics, deformation prediction, geometric computing.**

I. Introduction

With the excellent behavior of continuum bodies, soft robotics have attracted a lot of attention in research. Mainly inspired by nature, designers have come up with a variety of novel designs for soft robots to achieve different tasks (see [1], [2] for a comprehensive survey). By using soft materials and specially designed structures, continuum bodies enable these robots to generate large and complex deformations with an infinite number of *Degrees-Of-Freedom* (DOFs). Highly dexterous tasks like human-interactive grasping [3] and exploration in confined regions [4] can then be realized with soft robots. In the meantime, 3D printing with multiple materials [5]–[8] has been utilized to fabricate soft robots, providing flexibility in the complexity of the geometry as well as the material properties.

Manuscript received October 19, 2019; accepted March 4, 2020. This paper was recommended for publication by Associate Editor (TBA.) and Editor A. Billard upon evaluation of the reviewers' comments. This work was supported in part by the seed fund of Industrial Design Engineering faculty at TU Delft, Natural Sciences & Engineering Research Council of Canada (NSERC) grant #RGPIN-2017-06707 and the CUHK Direct Grant (CUHK/4055094). (*Corresponding authors: Tsz-Ho Kwok and Charlie C.L. Wang.*)

G. Fang and R.B.N. Scharff are with the Department of Sustainable Design Engineering, Delft University of Technology, Delft, the Netherlands (email: G.Fang-1@tudelft.nl; R.B.N.Scharff@tudelft.nl).

C.-D. Matte and T.-H. Kwok are with the Department of Mechanical, Industrial and Aerospace Engineering, Concordia University, Montreal, Canada (email: matt_chr@live.concordia.ca; tszho.kwok@concordia.ca).

C.C.L. Wang is with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong. (email: cwang@mae.cuhk.edu.hk).

This paper has supplementary downloadable video available at http://ieeexplore.ieee.org.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

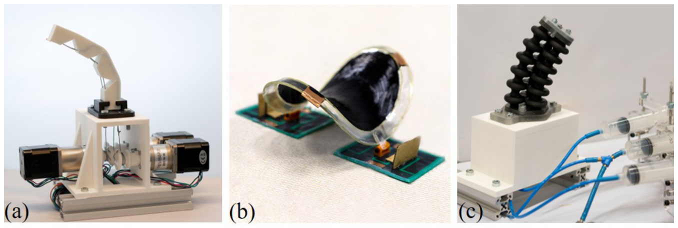

Fig. 1. Example soft robotic systems that actuation can be represented as geometric changes: (a) a soft finger actuated by stepper motor with cable length shortening, (b) a soft crawling robot driven by *dielectric elastomer actuation* (DEA) can achieve locomotion by the area change using different voltage input [9], and (c) a pneumatic driven soft manipulator controlled by syringe actuation system with the volume change in chambers.

A. Problems of Kinematics

While soft matter and 3D printing open up many opportunities in developing new soft robots, these advanced designs along with the high amount of DOFs also bring challenges to develop efficient and reliable algorithms for kinematics. Unlike robots with rigid bodies for which the position and velocity of the end-effector can be directly computed with joint parameters, it is almost impossible to explicitly formulate the kinematic function for soft manipulators. Although some reduced analytical models have been developed for specific designs, they are usually based on a particular type of soft body and therefore not general enough to model robots with complicated shapes.

A numerical approach can also be used to predict the deformation of soft robots by approximating a continuum body with discretized finite elements. With precise modeling formulation of soft materials, *Finite element analysis* (FEA) has proved its effectiveness in simulating the behavior of soft robots [10], [11]. However, when dealing with large rotational deformation, the high cost of computation by using enterprise-level FEA software (e.g. Abaqus and ComSol) can hardly meet the required efficiency in kinematics applications.

Our research is inspired by the fact that many forms of actuation in soft robotic systems can be directly transformed into geometric changes (see Fig. 1). In this paper, we tackle the problem of kinematics computing by presenting an efficient approach where soft robots with multiple materials and their actuation are systematically modeled in a geometry-oriented formulation. Comparing to other methods, our kinematic algorithm shows better convergence and keeps a good balance between the computational efficiency and the numerical accuracy. Both *forward kinematics* (FK) and *inverse kinematics* (IK) can be efficiently computed in our framework. Case studies with physical experiments have been conducted to demonstrate and verify the effectiveness of our approach.

B. Related Work

Efficient computation for simulating the deformation of soft robots under different types of actuation is a fundamental technique to solve the problems of kinematics, which is needed in many applications – e.g., adaptive grasping of soft objects in the food industry [12] and auxiliary systems for soft tissues in medical surgery [13]. Prior research works can be classified into three groups: 1) analytical methods, 2) numerical methods and 3) model-free methods (mainly using machine-learning and computer vision).

When dealing with soft robots having simple (particularly symmetric) structures, analytical methods based on mechanics or differential geometry have been commonly used. In the early stages, the backbone curve approach [14] and the constant curvature assumption [15] were applied to build the kinematics of multi-section soft robots. By using work-energy principle, Trivedi *et al.* [16] developed a geometric model for pneumatic-driven soft manipulators that has better accuracy than the constant-curvature model. Michele *et al.* conducted a series of work [17]–[19] to build forward and inverse kinematics for bio-inspired manipulators by applying the Jacobian method of statics models to compute the equilibrium status of conical shaped manipulators under cable forces. Recently, efforts have also been made to use analytical methods for soft robotic systems with high DOFs or hyper-elastic materials. For example, Panagiotis et al. [20] presented their analysis for fiber-reinforced bending pneumatic actuators. A teeth-structure soft gripper was studied by using a simplified skeleton model [21]. However, the equilibrium of a static model requires specific approximations and assumptions of shape and material properties, which can hardly be generalized to soft robots with freeform shapes fabricated by 3D printing.

While using the numerical method, the deformation of a continuum body is usually simulated by FEA with given material properties and the boundary conditions of actuation. A deformed shape can be computed in general and this method has been used to help select the optimal design parameters of soft robot to meet specific performance (e.g., providing a faster actuation behavior [22] or making the bending curvature conformable to a design surface [23]). Conversely, the trade-off between computing time and accuracy needs to be made when applying a numerical method on real examples with more than 10k elements. Commercial FEA software like Abaqus and ComSol can generate precise calculations of forward kinematics for soft robots [10], [20]; however, small time-steps are needed when confronted with situations of large deformation. For these softwares, high computation cost and slow simulation speed restrict its usage for further solving the IK problem. To speed up numerical methods, Allison and Okamura [24] presented a closed-loop control of a haptic jamming deformable surface by a mass-spring system. Hiller and Lipson [25] developed a multi-material simulation library for general static and dynamic analysis – called *Voxelyze*, where the voxel representation and beam theory were used. Based on a physics-based simulation engine SOFA [26], Duriez *et al.* [27] simulated the behavior of soft robots by progressively solving a quasi-static equilibrium function for every sample time. This method can achieve real-time computing speed with a reduced model [28]. However, the progressive computation accumulates numerical errors along time steps, which brings in the accuracy problem for the case with large rotational deformation (see the comparison given in Section V-A).

In the absence of analytical and numerical models, model-free methods based on learning or vision, have been employed to solve the challenge of computational kinematics for soft robots. Machine intelligence approaches can generate forward and inverse mappings with limited samples obtained from either physical experiments [19] or precise numerical simulations [29]. The accuracy of training-based kinematic computation however mainly relies on the quality and quantity of the training datasets. Visual servoing has been used to control the manipulation by calculating the Jacobian of deformation between the control point and an unknown elastic body [30], [31]. Similarly, Li et al. [32] employed an adaptive Kalman filter to estimate the Jacobian and only required data input from the vision tracking system. Zhang et al. [33] built a closeloop tip position control strategy for specific soft robot design by combining the numerical simulator with a visual servoing system. The vision-based methods are efficient and robust after adjusting the control law. However, the requirement of vision hardware and the complex calibration process prevents the usage of this method in many scenarios.

C. Contributions

The technical contributions of our work are summarized as follows:

- A novel method of geometric computing is presented to predict the deformation of continuum soft bodies under geometric actuations – this results in an efficient forward kinematic computation. Physical actuations are directly transformed into geometric constraints that can be intrinsically integrated into the framework.
- An image-based calibration method is introduced to enable the simulation of multiple materials in our computational framework by learning the relationship between material properties and shape parameters.
- A Jacobian-based iterative algorithm is developed to compute the IK solution with the help of our efficient deformation computing framework. The Jacobian matrix is calculated by numerical differences, which relies on a highly efficient simulator.

Our method is direct and efficient. It has been verified on 3D printed soft robots driven by different types of actuation within the deformation range of linear material elasticity. Applications of trajectory following have been conducted to demonstrate the performance of our method.

An early version of this study, which focused on predicting the deformed shape of cable and pneumatic driven soft manipulator under single actuation in 2D domain, has been presented in [34]. In this paper, our approach has been enhanced in the following aspects:

- The method is extended to support kinematic computing with multiple actuators in 3D, and has its correctness verified in the deformation range of linear elasticity.

TABLE I
LIST OF SYMBOLS

Symbol	Description	Symbol	Description
$\mathcal{M}$	Volumetric mesh represents for soft robot model	$\mathbf{V}_i^t, \mathbf{V}_i^d$	Target shape and current shape for the i-th element
Ω	Material distribution of given soft robot design	$D(\cdot,\cdot)$	Shape difference of two corresponding elements
$\mathcal{C}$	Set of geometry-defined actuation parameter $\{s, \lambda, \alpha\}$	E_i, E	Geometry-elastic energy for i-th element and whole domain
s, λ, α	Length, area and volume change ratio for actuation element	$\mathbf{N} \in \mathbb{R}^{k\times k}$	Transformation matrix to move element center to its mean
$\mathcal{V}$	Set of vertices in mesh $\mathcal{M}$	$\mathbf{R}_i \in \mathbb{R}^{3\times 3}$	Rotational matrix between two status for the i-th element
n	Number of body element	$J(\cdot)$	Objective function for inverse kinematics computing
m	Number of actuation element	R_ω	Shape parameter presenting material behavior
k	Number of vertices in single element $\mathbf{V}$	$\mathcal{F}_{dk}(\cdot)$	Forward kinematics implicit function
$\mathbf{v} \in \mathbb{R}^3$	Position vector of single vertices in set $\mathcal{V}$	$\mathcal{L}$	Desired motion trajectory
$\mathbf{V} \in \mathbb{R}^{k\times 3}$	Single element and its shape matrix $[\mathbf{v}_1\ \mathbf{v}_2\ ...\ \mathbf{v}_k]^T$	$\mathcal{P}$	Sampling point set of the trajectory $\{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_N\}$

- An algorithm of *inverse kinematics* (IK) is developed by using Jacobian-based iterations.
- The generality of our method has been further verified on additional distinct designs of soft robots.

The rest of our paper is organized as follows. Section II presents the mathematical modeling of our geometry computing approach. Both forward and inverse kinematic problems are formulated and solved with a corresponding algorithm presented in Section III. In Section IV, we study the correctness of using linear material elasticity for 3D printed soft robots and also introduce a method of physical calibration to transform multi-material properties into geometric parameters. Experimental results are given in Section V, where the effectiveness of our method has been validated on different applications and physically fabricated soft robots. Finally, our paper is concluded in Section VI.

II. Geometry-based Formulation

In this section, we present the formulation of our geometry-based modeling framework. The notations used in this paper are first presented. Then, deformation energy is defined based on the shape variation of elements. After that, bodies and actuators of soft robots are modeled as two types of elements in the formulation. Lastly, the methods for computing target shapes of different elements are presented in detail.

A. Notations

The small and capital bold letters are used to present column vectors and matrices respectively, e.g., $\mathbf{v} \in \mathbb{R}^3$ and $\mathbf{N} \in \mathbb{R}^{k\times k}$. The subscript of a variable presents its order in corresponding set, meanwhile the superscript present is for the status of meshes or elements. Particularly, the superscript d denotes the deformed (or current) shape and t means the target status. The identity matrices are denoted by $\mathbf{I}_{k\times k} \in \mathbb{R}^{k\times k}$, and $\mathbf{1}_{k\times k}$ is a matrix of $k \times k$ ones.

A volumetric mesh $\mathcal{M} = (\mathcal{V}, \mathcal{E})$ is used in our framework to represent the body of a soft robot, where $\mathcal{V}$ and $\mathcal{E}$ stand for the sets of vertices and elements in the mesh. We define the shape of each element by a $k \times 3$ matrix $\mathbf{V}_i = [\mathbf{v}_1\ \mathbf{v}_2\ \ldots\ \mathbf{v}_k]^T$ with k being the number of vertices on an element. In this paper, tetrahedron ($k = 4$) and prism elements ($k = 6$) are used to model soft robots with general 3D geometry and thin-shell structure respectively.

In our method, the status of actuation is described by a set $\mathcal{C}$ of geometric parameters:

1) Length shortening ratio s for cable actuation
2) Area stretching ratio λ for DEA
3) Volume expanding ratio α for pneumatic actuation.

This can also be extended to support other types of geometry-oriented actuation. Meanwhile, other notations used in this paper are summarized in Table I.

B. Elastic Energy Function

The general purpose of an elastic deformation simulator is to determine a new shape $\mathcal{M}^d$ for a soft body that best mimics the physical behavior of deformation under the actuation of $\mathcal{C}$ with reference to the initial shape $\mathcal{M}$ and the input material distribution Ω. When different boundary conditions (or external loads) are applied to deform an object, the elastic energy is transferred by the corresponding work and distributed internally in $\mathcal{M}$. Here the elastic energy is caused by the shape deformation, which can be evaluated from the strains (i.e., local deformations throughout $\mathcal{M}$). In this sense, the total elastic energy should be minimized when the original shape is preserved as much as possible. To mimic this physical phenomenon, we formulate the difference between $\mathbf{V}_i^d$ (current shape under deformation) and $\mathbf{V}_i^t$ (target shape) for a single element by discretized geometry-elastic energy as

$$E_i = D(\mathbf{V}_i^d, \mathbf{V}_i^t). \tag{1}$$

To measure the shape difference $D(\cdot,\cdot)$ of $\mathbf{V}_i^d$ and $\mathbf{V}_i^t$, they have to be properly aligned in terms of both position and orientation. Therefore, both shapes are centered at the origin and a rotation is applied to match $\mathbf{V}_i^t$ with $\mathbf{V}_i$, such that the above energy for the i-th element can be further defined as

$$E_i = \omega_i ||\mathbf{N}_i \mathbf{V}_i^d - \mathbf{R}_i(\mathbf{N}_i \mathbf{V}_i^t)||_F^2. \tag{2}$$

ω_i is a weight for each element which is normally set as the element's volume (ref. [35]). $||\cdot||_F$ is the Frobenius norm, $\mathbf{R}_i$ is the pure rotational matrix between two status for the i-th element. $\mathbf{N}_i$ is used to transfer an element's center to the origin and $\mathbf{N}_i = \mathbf{I}_{k_i\times k_i} - \frac{1}{k_i}\mathbf{1}_{k_i\times k_i}$.

Remark 1 Only elastic deformations are considered in our framework.

As a result, we can assume that every soft model will come back to its initial rest shape after releasing all the constraints

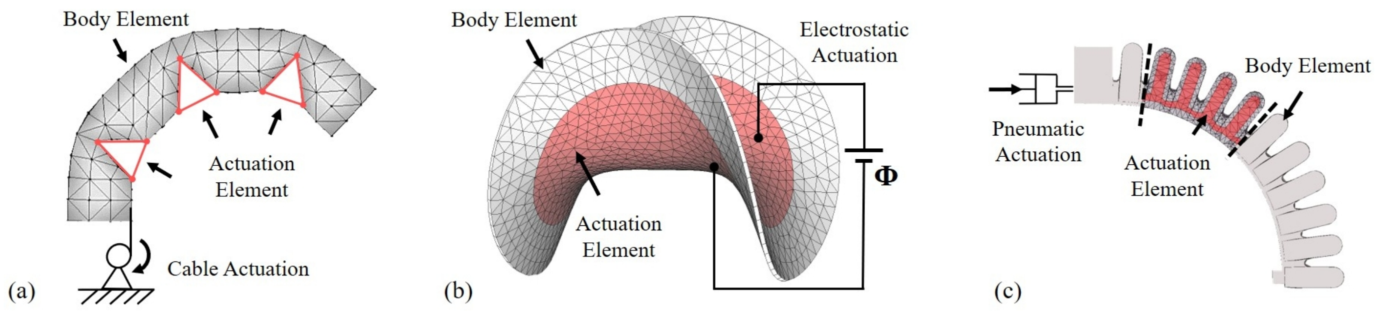

Fig. 2. Conceptual representation of our geometry-based framework for soft robotic systems with different types of actuation. The light-gray region presents the body elements and the region in red denotes actuation elements. Three different types of actuation are transformed into the shape change of actuation elements: a) cable-drive actuation is formulated as the edge-length shortening of cable elements, b) dielectric elastomer actuation is presented as the area stretching of prism actuation elements, and c) pneumatic actuation is defined as the volume expansion of internal tetrahedral chamber elements.

(i.e. actuations and handles). The energy function defined in (2) consists of three sets of variables, including:

1) Vertex positions of target shape $\mathbf{V}_i^t$,
2) Rotation matrices $\mathbf{R}_i$ for individual elements and
3) Vertex positions of current shape $\mathbf{V}_i^d$ under deformation.

How to determine these variables is presented below.

We first consider the target shape, which presents the ability of a soft body to resist deformation under actuation. It is determined commonly by the initial model $\mathcal{M}$, the set of constraints $\mathcal{C}$ and the coefficients for material properties Ω. As shown in Fig. 2, two types of elements defined in our system – body elements and actuation elements, are modelled by using the same formulation of elastic energy. However, their target shapes are defined in different ways.

- For a body element $\mathbf{V}_i$, the target shape $\mathbf{V}_i^t$ is computed with a shape blending function by combining its initial rest shape and the coefficient R_ω reflecting its material property (i.e., the stiffness). Ideally, $\mathbf{V}_i^t$ is a blended shape between a super-elastic material and a completely rigid material, where R_ω indicates the level of blending (see Section II-C for the details of blending and Section IV for coefficient calibration).
- Target shape of an actuation element $\mathbf{V}_j^t$ is determined according to the different types of actuations. All actuation elements together actually serve as the driven handles to deform a soft body. Detailed formulation can be found in Section II-D.

The final energy function is determined by integrating all the elementary elasticity together. By minimizing the integrated energy function for the whole design domain together with actuation constraints, the deformed shape of soft robots under actuation can be computed. As shown in Equation (2), $\mathbf{R}_i$ and $\mathbf{V}_i^d$ are unknown variables to be determined during the optimization computation, and the numerical method for solving this nonlinear optimization problem will be presented in Section III. We first present the details of how to compute the target shapes for body and actuation elements below.

C. Modeling for Body Elements

For the soft robots fabricated by multiple materials, regions with different materials will deform in different ways, thus the target shape should be computed disparately based on the input material distribution Ω. In this section, we propose a method to formulate soft objects with multiple materials by using linear blending method with a shape parameter.

To model the different properties of materials, a simple way is to assign different weights ω_i for each element in (2). The rigidity of an element will be preserved differently through the optimization when different weights are assigned. This mimics the deformation of multiple materials. However, handling the material difference in this way will lead to large approximation errors. In order to gain a better control and reinforce the physical property in large deformations, we control the deformation behavior of elements at the local region by altering their target shapes, $\mathbf{V}^t$, according to different material properties.

Remark 2 When the material of an element is extremely hard, it will be rigid during the deformation; respectively, an element with extremely soft material will deform to the shape which conforms to its neighbors while preserving its volume.

Based on the above remark, we came up with a method to compute two different target shapes for body element as shown in the left side of Fig. 3. Here the target shape of a rigid element $\mathbf{V}^r$ comes from the rigid transformation of its original shape. This method thoroughly preserves the initial shape $\mathbf{V}$ and keeps the same orientation as the current shape, which leads to

$$\mathbf{V}^r = \mathbf{R}\mathbf{V} \tag{3}$$

$\mathbf{R}$ is the rotation matrix between current and initial element, and can be obtained by applying SVD to the affine transformation between $\mathbf{V}^d$ and $\mathbf{V}$.

For a soft element, its target shape $\mathbf{V}^s$ comes from scaling the current shape back to its original volume (see Fig. 3) and we call this volume preservation. The shape comes from current element shape $\mathbf{V}^d$ and can be calculated as

$$\mathbf{V}^s = \mathbf{S}\mathbf{V}^d, \tag{4}$$

where $\mathbf{S} = diag(r, r, r)$ with $r = Vol(\mathbf{V})/\,Vol(\mathbf{V}^d)$.

For a material in-between, the rigid and soft target shapes are aligned by using a blending method with a shape parameter R_ω to get the target shape as shown in the right of Fig. 3. Here linear shape blending method [36] was used after centering both shapes onto the origin with $\mathbf{N}$ matrix for a general case

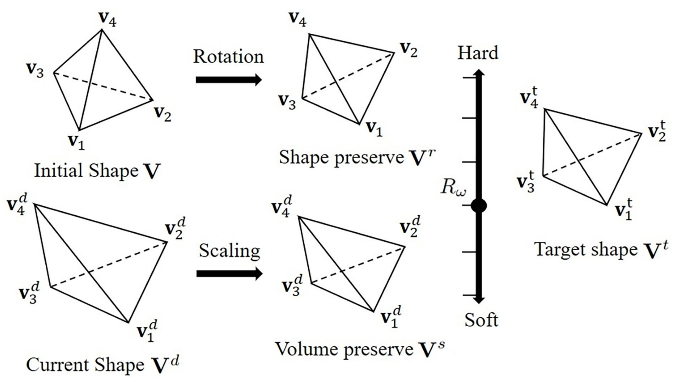
$\mathbf{v}_4$
$\mathbf{v}_3$
$\mathbf{v}_2$
$\mathbf{v}_1$
Initial Shape $\mathbf{V}$
Rotation
Shape preserve $\mathbf{V}^r$
Hard
R_ω
Soft
Target shape $\mathbf{V}^t$
Current Shape $\mathbf{V}^d$
Scaling
Volume preserve $\mathbf{V}^s$

Fig. 3. The shape blending method for controlling the material stiffness in our framework. (Top-left) The target shape for rigid material is computed by rotating the initial shape to align with the current shape. (Bottom-left) The target shape for extremely soft material is computed by scaling the current shape to preserve the volume of the initial shape. (Right) The shape blending method is applied to align the rigid and the soft materials, and merge their shapes to obtain the target shape for an intermediate material.

as:

$$\mathbf{V}^t = R_\omega \mathbf{N}(\mathbf{R}\mathbf{V}) + (1 - R_\omega)\mathbf{N}(\mathbf{S}\mathbf{V}^d). \tag{5}$$

In this way, the target shapes of elements according to different materials can be properly controlled during the deformation.

To verify the correctness of above method for controlling the relative stiffness of materials, we have tested a variety of polymer materials widely used in 3D printing. In Section IV-B, we present an image-based calibration process to determine the shape parameter – the ratio R_ω – for controlling the material behavior. Our linear shape blending method works very well when the deformation of each element is within the range of linear material elasticity. The correctness of our method will be verified in Section IV-A with the help of FEA simulation.

D. Modeling for Actuation Elements

Soft robots are deformed by applying external actuations such as cable shortening, elastomer stretching or pneumatic expansion of a chamber – these are all based on geometric metrics. When being at an equilibrium state, the geometry of an actuation must completely satisfy its given length, area or volume constraints. A straightforward method is to formulate them as hard constraints in a numerical optimization framework. However, it is hard to converge because of its high non-linearity – especially when the initial values are far away from the feasible regions.

To solve this problem of numerical computation, we formulate the deformation of an actuator as the collected function of a set of actuation elements (as shown in Fig. 2). The geometric constraints for an actuator are then converted into target shapes computed at each iterative step for these elements. The target shape of an actuation element is achieved by integrating it into the same elastic energy minimization framework. Larger weights are given to the actuation elements to make the actuation parameters satisfied effectively. As a result, the geometric actuation can be seamlessly integrated to our geometry-based simulation framework. Details of how we define the actuation elements and compute their target shape $\mathbf{V}^t$ according to the

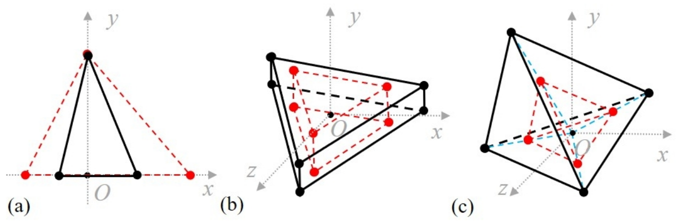

Fig. 4. An illustration of how the target shape of actuation element is computed based on the input parameter. The initial shapes are presented by red dot lines and the target shapes are displayed by black solid lines. Notice that the number of vertices k is different for different types of actuation elements. Specifically, $k = 3$ for (a) cable actuation, $k = 6$ for (b) dielectric elastomer actuation, and $k = 4$ for (c) pneumatic actuation.

input actuation parameter $\mathcal{C}$ are given below. Note that, after reshaping from the rest shape, each actuation element should be transformed to a position and orientation according to its current shape – i.e., the similar step as body element. Three different types of actuation elements are considered.

Cable-driven actuation: A typical cable-driven soft gripper with design similar to [37] is as shown in Fig. 2(a), which has three soft ‘knuckles’. A cable fixed on one side of the gripper is passed through the holes along the gripper. While pulling the cable (i.e., by shortening its length), the gripper bends towards one side. To integrate this actuation into simulation, the V-shaped ‘knuckles’ are modeled as a set of triangular elements. One edge of each triangle is aligning exactly with the cable, the deformation of which drive the simulations.

The total length L of a cable equals to the length of the gripper. It includes the inside portions L_R and the tooth length $\{l_i\}$ – i.e., $L = L_R + \sum_{i=1}^{k} l_i$, where k is the number of teeth. The shortening factor s is also given together with a cable constrain. The constraint function can be defined as

$$f_c(\mathcal{C}) = sL - (L_R + \sum_{i=1}^{k} s_i l_i) \equiv 0, \tag{6}$$

where s_i is a local shortening factor for the i-th tooth. Directly imposing this constraint to the optimization framework will lead to a computation very hard to converge.

It is more efficient to transform this function of constraint to a target shape for each actuation element. For a cable-driven actuation element, we place its rest shape into a position with its cable-driven face located in the xy-plane, the cable coincident with the x-axis and the opposite vertex on y-axis (see Fig. 4(a)). After that, the target shape can be computed by shrinking the element along x-axis by the factor s_i.

Dielectric elastomer actuation:

With voltage input, dielectric elastomers can effectively generate large deformation [38]. Driven by DEA, soft robots with specific design can perform locomotion by the areal stretching within the elastomer region. As show in Fig. 2(b), a thin-layer soft robot is modeled by prism elements where the inner red region is formed by the actuation elements. The total surface area A of the elastomeric region can be computed by $A = \sum_{i=1}^{k} a_i$ where a_i is the average area of the top and bottom triangles of a prism element. To satisfy the stretching ratio λ for a DEA, the constraint function can be defined as

$$f_d(\mathcal{C}) = \lambda A - \sum_{i=1}^{k} \lambda_i a_i \equiv 0, \tag{7}$$

where λ_i is a local expansion ratio of an actuation element. Similar to the cable-drive actuation, this constraint should also be transformed to the target shape of DEA elements.

When computing the target shape for a prism element from its rest shape, the top and bottom triangles are scaled in their own planes with the scaling ratio $\sqrt{\lambda_i}$. The center of each triangle is chosen as the center of scaling (see Fig. 4(b) for an illustration). After scaling, the triangles are shifted along their normal vectors so that the "thickness" of an actuation element is scaled to $1/\lambda_i$ to preserve the original volume.

Pneumatic actuation: A pneumatic actuator usually drives soft robots by pumping pressurized air into a bellow formed by soft materials. An example is shown in Fig. 2(c), where the left part is fixed when pumping air along the direction of white arrow into the bellows. The internal tetrahedra that fill the chamber are modeled as the actuation elements, which have been highlighted in Fig. 2(c). These actuation elements are used to model the expansion of air inside the bellows.

Given the volume u_i of each pneumatic actuation element, the total volume of a bellow is then $U = \sum_{i=1}^{k} u_i$. To achieve the volume expansion ratio α for a pneumatic-driven soft robot, the geometric constraint can be described by:

$$f_p(\mathcal{C}) = \alpha U - \sum_{i=1}^{k} \alpha_i u_i \equiv 0. \tag{8}$$

where α_i is a local expansion ratio of each element.

The target shape for a pneumatic actuation element with the volume expansion ratio α_i can be determined by scaling its rest shape with the ratio $\sqrt[3]{\alpha_i}$. The scaling is conducted at the center of tetrahedron (see Fig. 4(c)). After scaling, the target shape should be transformed to a position and orientation according to the element's current shape.

There is a remaining problem to be solved – how to determine the scaling ratios on every elements (i.e. $\{s_i, \lambda_i, \alpha_i\}$) by a global actuation parameter such as s, λ or α. We determine them proportionally to the ratios of an element's current shape w.r.t. its rest shape. The newly determined ratios must also satisfy the geometric constraints defined in (6), (7), and (8). In our implementation, a least-norm solution is employed to compute their values on all the actuation elements.

III. Algorithm for Kinematics

The kinematics of soft robots are hard to be solved analytically. In this section, we present the algorithms characterized by our geometry-based formulation to solve both the forward and the inverse problems of kinematics for soft robots. As a general framework, our algorithms for kinematics can intrinsically handle the different configurations of actuation with different material-distributions as long as the actuation can be converted into geometric inputs.

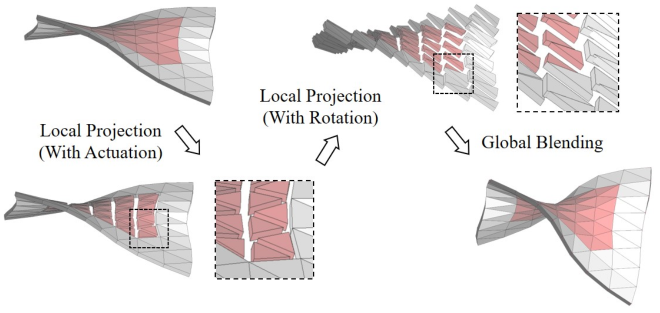
Local Projection (With Actuation)
Local Projection (With Rotation)
Global Blending

Fig. 5. An illustration of the local-global optimization process on a simple model where the whole mesh is actuated by shrinking elements in the red region.

ALGORITHM 1: ForwardKinematicComp

```
  Input: The initial shape $\mathcal{V}$, the actuation $\mathcal{C}$ and the material
         distribution $\Omega$.
  Output: The deformed shape $\mathcal{V}^d$
1 Initialize the weights $\{w_i\}$ and volumes $Vol(\mathbf{V}_i)$ for all
    elements;
2 Apply factorization to the normal equation of (10);
3 repeat
      /* Local / global optimization */
4     Compute the target shape for each element;
5     Applying SVD to obtain the rotation matrix $\{\mathbf{R}_i\}$;
6     Determine $\{\mathbf{v}_j\}$ by solving the linear system where the
        factorization can be re-used;
7     Update $\mathcal{V}^d$ by the new positions of vertices $\{\mathbf{v}_j\}$;
8 until the position change is less than $10^{-5}$ on all vertices;
9 return $\mathcal{V}^d$;
```

A. Forward Kinematics

The forward kinematics for the soft robots can simply be described as the computation of the deformed shape $\mathcal{V}^D$ from the initial shape $\mathcal{V}$ given the actuation in the form of constraints, $\mathcal{C}$. As formulated in Section II, the deformed shape of a soft body can be computed by minimizing the elastic energy after converting the actuation constraints into a set of target shapes for the actuation elements. This leads to a solution of forward kinematics in our framework by solving the unconstrained optimization problem below.

$$\min_{\mathcal{V}} E(\mathcal{M}, \mathcal{C}) = \sum_{i=1}^{m+n} w_i \, Vol(\mathbf{V}_i) \|\mathbf{N}_i \mathbf{V}_i^d - \mathbf{R}_i(\mathbf{N}_i \mathbf{V}_i^t)\|_F^2 \tag{9}$$

where the variables of optimization are the vertices $\{\mathbf{V}_i^d\}$ of a deformed shape. In this framework, the final shape of a soft body under actuation is determined by the initial shape of n body elements and the target shape of m actuation elements.

In the formulation of (9), both the local rotation $\mathbf{R}_i$ and the vertex positions $\mathbf{V}_i^d$ are unknowns to be determined. As a result, the objective function of optimization is highly nonlinear which may lead to a very slow convergence and high computation time. In order to solve it efficiently, a local / global scheme akin to [39] is employed. In the step of *local projection*, the initial shapes of actuation elements are first deformed according to the actuation parameters (as introduced in Section II-D). After that, the target shapes of all elements are independently transformed by applying the rigid transformation determined between their target shapes and the current positions (i.e., the rotation matrices as discussed in Section II-C). Then, the new positions of vertices can be computed in the *global blending* step by minimizing the energy. Letting

$$\frac{\partial E(\mathcal{M},\mathcal{C})}{\partial \mathbf{v}_j} = 0 \quad (\forall \mathbf{v}_j \in \mathcal{V}) \tag{10}$$

leads to a least-square problem that can be solved efficiently.

Through this global blending step, the incompatible positions of a vertex in different elements are "glued" together. An illustration of this local-global computation can be found in Fig. 5. Notice that, w_i, $Vol(\mathbf{V}_i)$ and $\mathbf{N}_i$ are constant during the iterations for minimizing $E(\mathcal{M},\mathcal{C})$, factorization of the normal equation defined by (10) can be pre-computed and reused to accelerate the computation of optimization. In order to well-preserve the constraints of actuation, a larger weight as $w_i = 5.0$ are employed for the actuation elements while keeping $w_i = 1.0$ for all other body elements. The pseudo-code of our algorithm can be found in **Algorithm** 1.

B. Inverse Kinematic Problem

The computation of forward kinematics is able to generate the deformed shape $\mathcal{V}^d$ from the given actuation $\mathcal{C}$. In many robotic applications, it is also demanded to obtain the needed actuation by the given deformed shape. This is an inverse kinematic problem where only a portion of the deformed shape is usually given as an input.

Remark 3 As the forward kinematics can be computed efficiently, the deformed shape $\mathcal{V}^d$ can be considered as the output of an implicit function $F_{dk}(\cdot)$, that is

$$\mathcal{V}^d = F_{dk}(\mathcal{C},\mathcal{V},\Omega) \tag{11}$$

with the initial shape $\mathcal{V}$, the actuation $\mathcal{C}$ and the material distribution Ω as the input.

Note that the material distribution Ω specifies the values of R_ω on every soft body elements. In our current work, it is given by designers after the calibration of material properties.

For articulated robots, the inverse kinematics can be described as calculating joint status. Given a subset of vertices $\bar{\mathcal{V}} = \{\mathbf{v}_p\}$ ($\bar{\mathcal{V}} \subset \mathcal{V}$), IK of soft robots can be considered as finding the proper parameters of actuation to drive the soft body into a shape that $\{\mathbf{v}_p\}$ match their desired positions – defined as $\{\mathbf{v}_p^c\}$. Different from low DOFs articulated robots where analytical IK can be obtained, IK of soft robots cannot be directly calculated as $\mathcal{C} = F_{dk}^{-1}(\bar{\mathcal{V}},\Omega)$. It needs to be solved via numerical computation (ref. [40]). This heavily relies on the efficient computation of forward kinematics. Specifically, we seek for an approximate solution that satisfies the position requirement.

Firstly, an objective function is defined below to quantify the distance between the current position and the target position of all vertices in $\bar{\mathcal{V}}$ as

$$J(F_{dk}(\mathcal{C},\mathcal{V},\Omega)) = \sum_{\mathbf{v}_p \in \bar{\mathcal{V}}} \|\mathbf{v}_p^d - \mathbf{v}_p^c\|^2, \tag{12}$$

ALGORITHM 2: InverseKinematicComp

```
   Input: The rest shape $\mathcal{V}$, the target positions $\mathcal{V}_P$ for
          investigated points, and the maximally allowed
          iterations $i_{max}$.
   Output: The actuation parameters $\mathcal{C}$
 1 Set the initial value of $\mathcal{C}$ as the rest configuration;
 2 Set the iteration time $i = 1$;
 3 Evaluate the objective function $J_0 = J(\mathcal{C})$;
 4 while $J(\mathcal{C}) > \lambda$ and $i < i_{max}$ do
 5     Compute the gradient of $J(\mathcal{C})$ as $\nabla J$;
 6     Set the step size $\Delta h = 1.0$;
 7     Compute $J_{new} = J(\mathcal{C} + \Delta h \nabla J)$;
       /* Soft line-search (line 7-17)          */
       /* Step 1: Shrinking                     */
 8     while $J_{new} \geq J_0$ do
 9         $\Delta h = \tau \Delta h$;
10         Compute and update $J_{new} = J(\mathcal{C} + \Delta h \nabla J)$;
11     end
       /* Step 2: Expanding                     */
12     Set $h = \Delta h$;
13     repeat
14         Compute and update $J_{new} = J(\mathcal{C} + (h + \Delta h)\nabla J)$;
15         if $J_{new} \leq J_{opt}$ then
16             Set $J_{opt} = J_{new}$ and $h = h + \Delta h$;
17         end
18     until $J_{new} > J_{opt}$;
       /* Best $h$ has been found               */
19     Set $\mathcal{C} = \mathcal{C} + h\nabla J$ and $i = i + 1$;
20 end
21 return $\mathcal{C}$;
```

where $\mathcal{C}$ is the set of actuation parameters that can have multiple variables. Then, the inverse kinematics of a soft robot can be defined as an optimization problem that

$$\mathcal{C}_{opt} = \arg\min_{\mathcal{C}} J(F_{dk}(\mathcal{C},\mathcal{V},\Omega)). \tag{13}$$

We use the gradient-based method to solve this optimization task, which needs to first figure out the gradients of $J(\cdot)$ with respect to $\mathcal{C} = (C_1, C_2, \ldots, C_i, \ldots)$. The analytical solution of $\frac{\partial J}{\partial C_i}$ cannot be obtained as the position $\mathbf{v}_p^d$ is only an implicit function of $\mathcal{C}$. Fortunately, we can efficiently and effectively evaluate the value of $\mathbf{F}_{dk}(\cdot)$ by our forward kinematic algorithm – i.e., we can easily get the positions of investigated vertices by computing a deformed shape according to the given actuation. As a result, numerical differences are employed to compute the gradient $\nabla J = \left[\frac{\partial J}{\partial C_i}\right]$ as

$$\frac{\partial J}{\partial C_i} = \frac{J(\ldots, C_i + \Delta C, \ldots) - J(\ldots, C_i - \Delta C, \ldots)}{2\Delta C} \tag{14}$$

where ΔC is a small constant which can be determined according to the value of $J(\cdot)$ by the strategy of [41].

Directly updating the values of $\{C_i\}$ by the gradient ∇J may lead to a computation with slow convergence. To improve it, a linear search method is applied to determine the best updating scale h so that

$$h = \arg\min J(F_{dk}(\mathcal{C} + h\nabla J, \mathcal{V}, \Omega)). \tag{15}$$

Specifically, we first determine a value of Δh so that $J(\mathcal{C} + \Delta h \nabla J) < J(\mathcal{C})$ by a *shrinking* step starting from $\Delta h = 1.0$. The shrinkage speed is controlled by a ratio $\tau \in (0, 1)$ – we

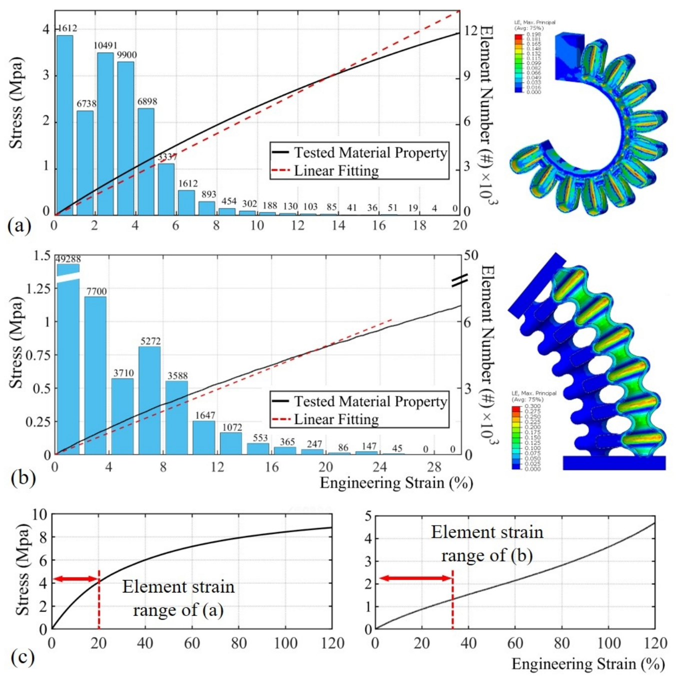

Fig. 6. Verification of small-strain assumption on two effective designs of soft robots fabricated by soft materials: (a) Ultimaker TPU 95A and (b) Aglius 30. The strain distribution of body elements (shown in the right) is generated by FEM simulation, and the histograms (left) show the statistics of strains on these two designs under large structural deformation. (c) Stress-strain curves for Ultimaker TPU 95A (left) and Aglius 30 materials (right) obtained by physical experimental tests. It can be observed that the elemental deformation mainly occurs in the range with linear material elasticity.

use $\tau = 0.1$ in all our experimental tests. After that, the scale h is further optimized by be incrementally enlarged with the step size of Δh – this is called an *expanding* step. These two steps of linear search can help us to find a 'loose' optimum along the direction of ∇J.

The terminal condition of optimization process for solving (13) is chosen to be $J(F_{dk}(\mathcal{C}, \mathcal{V}, \Omega)) \leq \lambda$ with λ being a threshold determined according to the accuracy allowed in different applications. On the other hand, it is also possible that a user-specified goal cannot be realized by a soft robot – e.g. when a desired position $\mathbf{v}_p^c$ falls outside the reachable space of a robot. Therefore, we also set a maximally allowed iterations, i_{max}, as the terminal condition in our IK algorithm. Since the line-search strategy is used to ensure the decrease of an objective function in every iteration, our method can always provide a local optimum for objective function (12). The pseudo-code of our IK computation has been given in **Algorithm** 2.

IV. Material Property: Analysis and Calibration

To formulate the deformation of soft robots made with multiple materials, we proposed a reduced model based on the linear shape-blending method presented in Section II-C. The effectiveness of our method mainly relies on two conjectures.

- For a variety of smart soft robot designs, the large deformation of continuum body is mainly generated by structural deformation instead of elemental deformation – i.e., the strains are relatively small.
- For many materials widely used for the fabrication of soft robots, the material elasticity in the range with small strain can be approximately described as a linear model.

In this section, these two assumptions are verified by both the FEM simulation and the material tests. After proving the correctness of our method, an image-based calibration process is proposed to find a shape parameter to be used in our method corresponding to the physical behavior of materials.

A. Linear Material Elasticity

Many materials used for fabricating soft robots can be largely stretched and have hyper-elastic material property, which was utilized to achieve large shape change under actuation in early years. However, recent designs of soft robots have specially designed advanced structures to realize more reliable deformation with better durability. For example, inextensible layers [8] are used to prevent the non-directional expansion so that the effectiveness of an actuator is tremendously enhanced. In these cases, extreme local stretch is no longer necessary for realizing a large global deformation. We study the range of elemental deformation on a widely applicable soft finger structure [42] and another smart design of soft manipulator [43].

Tensile tests have been conducted on two materials used in fabricating these two soft robots – Ultimaker TPU 95A and Aglius 30. The obtained stress-strain curves are shown in Fig. 6(c). The strain-stress relationship is nonlinear in general. However, when deformation occurs in a range with small strains, the relationship can be linearly approximated with small error. Specifically, when the strain is less than 20% for TPU and 30% for Aglius, a linear stress/strain curve can be obtained (see also the solid and dash lines shown in the zoom-view of Fig. 6(a) and (b)).

We conduct the FEM to further study the strains generated on these two designs of soft robots. Abaqus software is employed to generate the strain distribution when large structural deformation has been achieved on these two structures. In Fig. 6, the histograms are used to visualize the statistical distribution of strains in all elements. It can be easily found that the strains are less than 20% for most regions and all fall in the range of linear elasticity discussed above – i.e., less than 20% for TPU and 30% for Aglius.

Note that, large elemental deformation can be achieved under actuation for the materials with small Young's modulus such as silicon rubber. This material property was employed for some designs developed in early years. For these cases, the elasticity is not guaranteed to be linear for all elements, which brings modeling uncertainty although our method can still successfully predict the deformation in practice.

B. Calibration of Shape Parameter

After verifying the correctness of using the linear elasticity simplification for body elements, our shape-blending method needs to define proper shape parameters to mimic the real physical behavior. Rather than calibrating each material separately in a tensile test, an image-based method is developed to calibrate the relative properties between different materials. As shown in Fig. 7, we impose the displacement on a rectangular

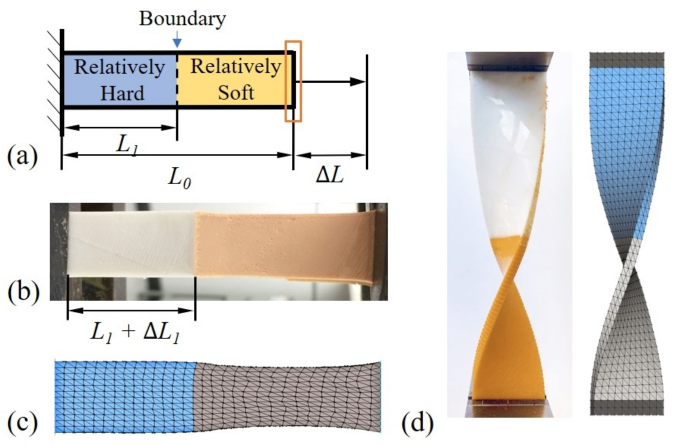

Fig. 7. Image-based calibration of the shape parameter for simulating objects with multiple materials: (a) a multi-material bar with displacement on the right, (b) a physical elongation test on 3D printed specimen using NinjaFlex and Flexible PLA materials, (c) the tensile test result generated by our simulation framework after calibrating the shape parameter R_ω, and (d) the twisting test [6] is also conducted to verify the correctness of our material elasticity and the calibration method.

TABLE II
CALIBRATED PARAMETERS FOR DIFFERENT MATERIAL COMBINATIONS

Material A	Material B	R_m	R^A_ω / R^B_ω	Actuation
TPU 95A	NinjaFlex	3.75	7.08	Cable
Tango Black	Mixed Aglius	5.68	10.30	Pneumatic

specimen at one end while fixing another end. Without loss of generality, the specimen is fabricated with two materials A and B joined with a sharp interface. Let the length of the whole specimen be L and the distance between the interface and the fixed end be L_1, where different values of $L_1 \in (0, L)$ are used for different specimens. When imposing a displacement ΔL at the free end of the bar, the displacement of the interface will be located at $\Delta L_1 \in (0, \Delta L)$ depending on the relative material properties between A and B. The relationship of two materials can be presented by an *elasticity ratio*, which is mathematically defined as

$$R_m = \frac{\epsilon_A}{\epsilon_B} = \frac{L_1(\Delta L - \Delta L_1)}{(L - L_1)\Delta L_1}, \quad (16)$$

where ϵ_A and ϵ_B are the strains in the regions of two materials with A being linked to the fixed end and B locating at the free end. Note that, for linear materials, R_m also equals to the ratio of Young's modulus (i.e., a constant when materials are given). The rest of the problem is how to find the corresponding value of the shape parameter R_ω after obtaining the elasticity ratio R_m on two materials through the physical tests. The basic idea of our calibration is to apply different values of R_ω to run the elongation tests in our geometry-based simulation by the same setup. The value of R_ω is then determined by matching our simulation results with the results of physical tests, where the bisection-search method is used. With a well calibrated parameter R_ω, the position of the material interface generated by our simulator matches well with the physical experiment accurately (see Fig. 7(c)). To further verify the generality of this parameter, we conduct a twisting test similar to the one presented in [6]. As shown in the left of Fig. 7(d), the specimen with two materials gives a symmetry torsion where the relatively soft region has a larger rotation angle. By using the same calibrated material parameter R_ω, our simulation generates a similar result (see the right of Fig. 7(d)).

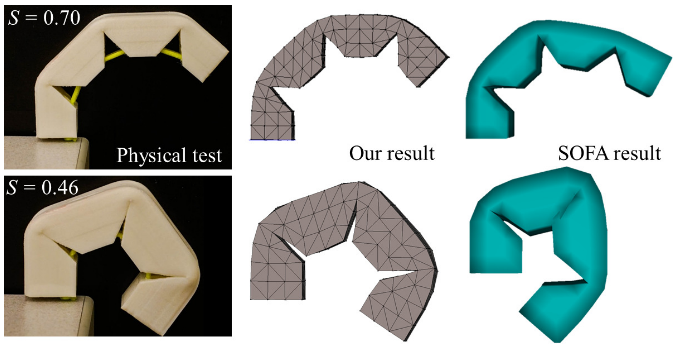

Fig. 8. Comparisons of a cable-driven gripper among the physical test (left), our simulation (middle), and the simulation by the SoftRobots plug-in for SOFA [27] (right).

We have applied this calibration method to various materials used for fabricating soft robots. For different 3D printing systems, different combinations of materials are tested and the corresponding calibrated parameters are listed in Table II. The effectiveness of our shape-blending based deformation model and the calibration method is further validated in the next section by other experimental tests taken on different robot designs.

V. RESULTS AND APPLICATIONS

We have implemented our geometric computing based kinematic algorithms for soft robots in C++ and tested on a standard PC with an Intel E5-1653 3.5GHz CPU and 16GB RAM. With the help of parallelization on multi-core CPU on the numerical solver Eigen [44], our system can support the computation of forward kinematics for models with up to 50k tetrahedra in real-time (i.e., 25 fps).

In this section, the results of forward kinematic computation for soft robots will be first presented and compared with existing numerical modeling methods. After that, the effectiveness of our inverse kinematic solver is evaluated on different soft robots with multiple actuators. The performance of our approach in these experimental tests is also presented in the supplementary video.

A. Validation of Forward Kinematics

The results generated by our forward kinematics algorithm on a deformed soft body are validated by physical tests. Moreover, our method is also compared with different simulation techniques. The models of soft robot are digitally represented by tetrahedral meshes, and their corresponding physical objects are fabricated by multi-material 3D printer (e.g., Ultimaker 3 and Object 350 Connex3). The properties of soft materials are evaluated on a Zwick Roell static testing machine.

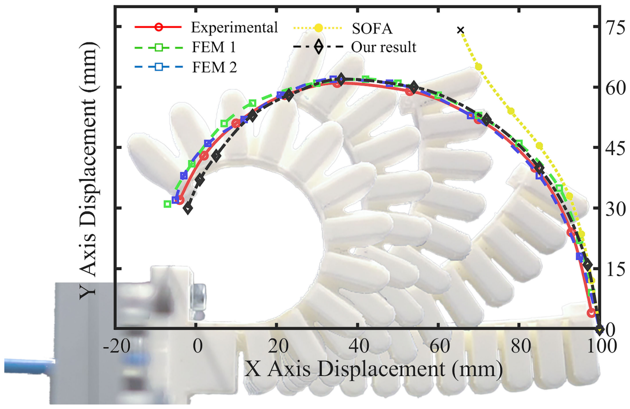

Fig. 9. Trajectories for a soft finger's tip under pneumatic actuation. The background image shows the bending results in real physical test. The results of three different numerical simulators are presented: 1) finite element analysis with linear (FEM 1) and non-linear material properties (FEM 2), 2) the SOFA simulator and 3) our method. The FEM results are generated by Abaqus.

TABLE III
COMPUTATIONAL COSTS FOR DIFFERENT METHODS OF SIMULATION

Method	Element #	t_{90° (sec.)	t_{180° (sec.)	t_{240° (sec.)
FEM 1[†]	44774	240	529	820
FEM 2	44774	288	636	1068
Our Method	45802	8.5	15.2	23.2
SOFA [27]	44900	3.8	-	-

[†]Simulations of finite element analysis use approximated linear material propriety in FEM 1 and nonlinear model in FEM 2 – both by the Abaqus software.

The first test is conducted on a cable-driven gripper with single material (Flexible PLA) as shown in Fig. 8. The top and bottom rows show two sequences of deformations at different time instants, where from left to right show the results of physical test, our simulation and SOFA [27]. Due to the reason that the 'deformations are progressively computed for each time step and the accuracy is traded off for computational speed in SOFA, its results do not match with the physical tests in large deformation. Specifically, simulation starts to vary from reality when cable length change is larger than 45%.

The second test is conducted on a pneumatic soft gripper by increasing the pressure of the air pumped into the chamber to control bending of the gripper. We quantitatively present the accuracy of our method by tracking the tip position of a soft gripper. As shown in Fig. 9, our result matches well with the analysis conducted by advanced FEA software as well as the physical experiment. When using the similar number of tetrahedra in the computation (i.e., around 45k), the computation of our framework is much faster – with 23.2 seconds vs. 13.6 minutes required to complete the simulation for bending the soft actuator up to 240 degrees by the Abaqus software. Meanwhile, we test this soft model on the SOFA platform with similar mesh size. The computing time has been reported in Table III. Noticed that the simulation speed of SOFA is faster than our method; however, the result begins to become unrealistic after being bent for more than 90 degrees (the yellow trajectory shown in Fig. 9). In contrast, our simulation can produce very realistic results for large deformation case while still having a fast speed in FK computing.

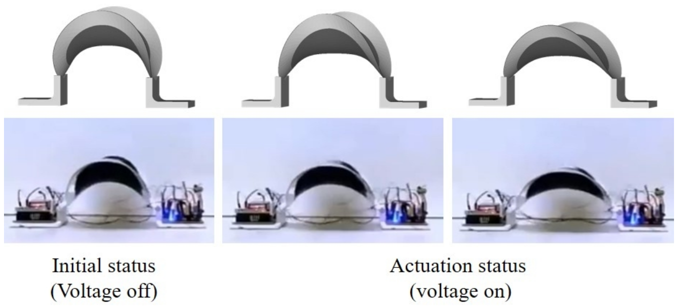

Fig. 10. Simulation result for a soft crawling robot by geometry modelling the electrostic-driven stretching behavior of DEA. (Top) The results of our forward kinematic computing. (Bottom) The locomotion behavior of a real robot [45].

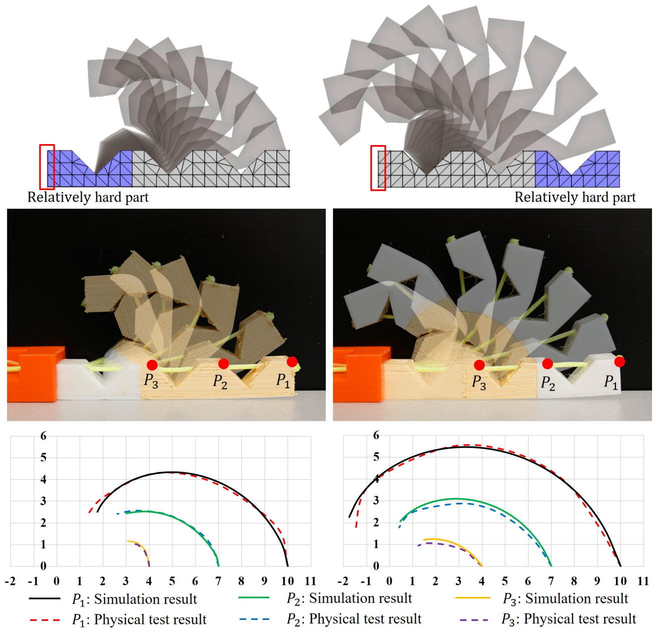

Fig. 11. Two cable-driven soft grippers (left and right) with different material distributions have different behaviors under actuation. Locations of markers determined by our simulation are well-matched with theirs in physical test.

A design of soft crawling robot [45] is studied to validate our approach on the DEA in FK computing. The actuator is fabricated by attaching bendable PET frames to a pre-stretched elastomer membrane. After releasing the constraint, the elastomer layer will shrink and drive the soft body deforming to its initial status (as shown in the left of Fig. 10). By applying the voltage to the electrodes, the elastomer region will elongate the soft body. Two rigid legs are attached to the robot and always kept on the x-y plane during simulation. In our simulation, an initial stretching ratio $\lambda_{init} = 0.7$ is used on actuation elements to first deform the planner model and get the initial shape. We use the voltage and stretch relationship in [46] to determine the parameter λ for the actuated status.

To verify the result of our forward kinematic computation for multiple materials, we test two cable-driven grippers with

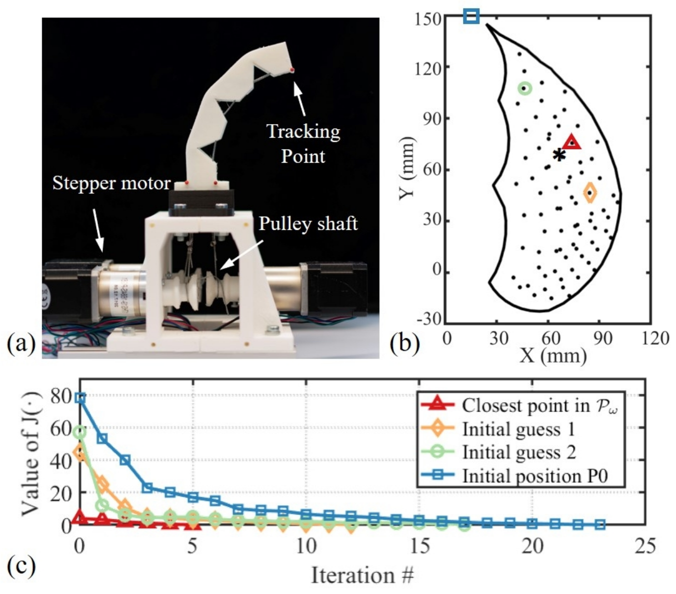

Fig. 12. A cable-driven soft finger with three tendons is used for the validation of IK computation. (a) The experiment setup. (b) The configuration space is sampled to obtain good initial values for IK computing (125 sample points are displayed). (c) The study of convergence for IK computation by evaluating the objective function $J(\cdot)$ (Eq.(12)) with the target position of the tracking point being given as the black star shown in (b). We can find that the converging speed of IK computation is greatly improved when the closest point (red triangle) in the sampled configuration space is used as initial guess.

different material compositions. The simulation and physical results are compared visually with its dynamics in Fig. 11. The deformations are also compared quantitatively by the trajectory of three corresponding markers located on the boundary of the grippers. It can be seen that both results match with the physical experiments very well.

B. Validation of IK Algorithm by Trajectory Following

To verify the accuracy and efficiency of our IK solver, we first demonstrate the behavior of our algorithm in a trajectory planning experiment by a cable-driven soft finger with three 'knuckles'. The soft finger is fixed on a solid base in our experimental setup (as shown in Fig. 12(a)) and for every 'knuckle', one iron cable is linked to its top and driven individually by its corresponding stepper motor through the pulley shaft. The design with multiple actuators enables the ability of controlling the soft finger to move in a plane.

Given a desired motion trajectory $\mathcal{L}$ for an an investigated point $\mathbf{q}$ on the soft robot $\mathcal{V}$, the task of trajectory planning can be solved by finding the parameters of actuation that drive $\mathbf{q}$ traveling along $\mathcal{L}$ accurately. To realize this, we first sample $\mathcal{L}$ into N points as $\mathcal{P}_{\mathcal{L}} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N\}$. After running the IK computation presented in **Algorithm** 2 for each target point $\mathbf{p}_k \in \mathcal{P}_{\mathcal{L}}$, we are able to determine the corresponding parameter set $\mathcal{C}_k$ in joint space for actuation (i.e., the shortening ratio for each tendon's length). For the terminal condition $J(F_{dk}(\mathcal{C}_k, \mathcal{V}, \Omega)) \leq \lambda$ that is used for IK computation, we choose $\lambda = 0.2$ mm and $i_{max} = 30$ for all sample points.

ALGORITHM 3: TrajectoryFollowing

```
Input: The rest shape $\mathcal{V}$, the sampled working space $\mathcal{P}_w$,
       the investigated point $\mathbf{q}$ and the target trajectory $\mathcal{L}$.
Result: The parameters of actuation $\mathcal{C}$.
1 Generate the set of sample points $\mathcal{P}_{\mathcal{L}} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N\}$
    on $\mathcal{L}$;
2 Find the point $\mathbf{p}_c \in \mathcal{P}_w$ that minimizes $\|\mathbf{p}_c - \mathbf{p}_1\|$;
  /* Using the corresponding $\mathcal{C}_c$ of $\mathbf{p}_c$ */
3 $\mathcal{V}^d \leftarrow$ ForwardKinematicComp($\mathcal{V}$, $\mathcal{C}_c$);
  /* Computing the actuation parameters */
4 for $k$ = 1, 2 ... $N$ do
      /* Accelerate the IK computation by
         using $\mathcal{V}^d$ and $\mathcal{C}_c$ as initial guess */
5     $\mathcal{C}_k \leftarrow$ InverseKinematicComp($\mathcal{V}$, $\mathbf{p}_k$);
6     $\mathcal{V}^d \leftarrow$ ForwardKinematicComp($\mathcal{V}$, $\mathcal{C}_k$) and $\mathcal{C}_c \leftarrow \mathcal{C}_k$;
7 end
8 return $\mathcal{C}$;
```

Implementation Details: After using different initial guesses for the IK computation, we find that its speed of convergence strongly relies on the position of initial guess (see Fig. 12(c)). Therefore, the following two strategies are conducted to speed up the computation in our trajectory planning application.

- Firstly, we generate a sample-based representation for the configuration space $\mathcal{P}_\omega$ (see Fig. 12(b)) where the sample points $\mathbf{p}_c \in \mathcal{P}_w$ are obtained by applying the FK algorithm with various combination of actuation parameters. The initial guess of IK solution (i.e. $\mathcal{C}$ in **Algorithm** 2) for the first point $\mathbf{p}_1$ on a trajectory $\mathcal{L}$ is then set as the control parameter of its closest sample point in $\mathcal{P}_w$.
- Secondly, a deformed shape is always kept during the computation and serves as the initial shape for realizing the next target point. Specifically, after obtaining the actuation parameters $\mathcal{C}_k$ for the target point $\mathbf{p}_k$, we update the shape of soft robot, $\mathcal{V}^d$, by applying the forward kinematics with $\mathcal{C}_k$ as the input. This updated shape will be used as the input for IK computation targeting on the next point $\mathbf{p}_{k+1}$.

The pseudo-code of our trajectory following algorithm and the above acceleration strategies are given in **Algorithm** 3. The computation of our method is very efficient. Firstly, a roughly sampled configuration space (e.g. 125 sample points) can be generated in 8.3 sec. for obtaining good initial values. Then, we conducted the tests on a 'L' trajectory (with $N = 55$) and a flame trajectory (with $N = 85$) as shown in Fig. 13, the computing times are 38 sec. and 46 sec. respectively.

We further conduct physical experiments (see the hardware setup shown in Fig. 12(a)) to verify the parameters of actuation generated by **Algorithm** 3. Arduino Mega 2560 and the RAMPS extension board are used to generate the modulated pulse signals that control the pull and release of cables. To generate a motion that linearly interpolates the neighboring target points, dynamic speed controller provided by Marlin firmware [47] is used to synchronize the three motors. A camera system is used to track the actual position of the investigated point $\mathbf{q}$, which is located at the top-right corner of the soft finger. The resultant trajectories of physical movement are given in Fig. 13(a) while comparing to the

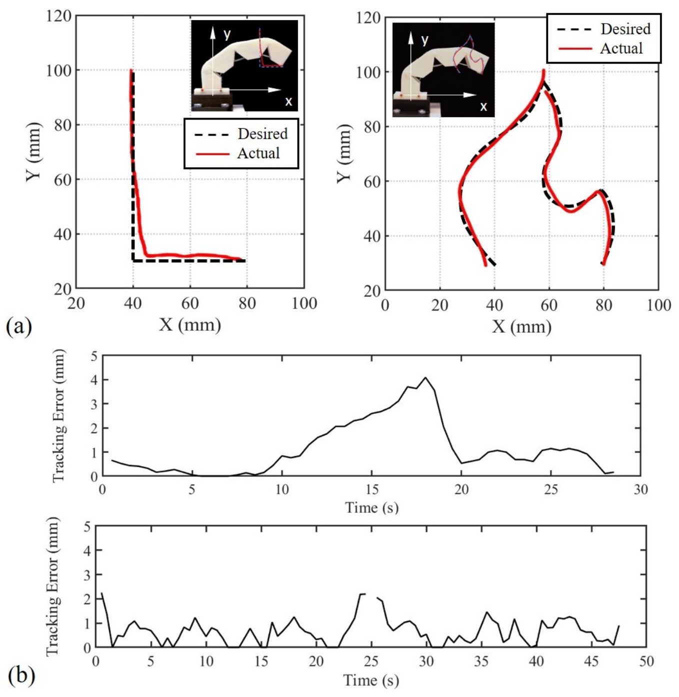

Fig. 13. The results of experimental tests for moving a marker point along desired trajectories: (a) comparison between the tracked actual movement and two target trajectories, and (b) position errors of the investigated point while moving along the 'L' trajectory (top) and the flame shape curve (bottom). The dimension of our actuator is $120\text{mm} \times 25\text{mm} \times 25\text{mm}$.

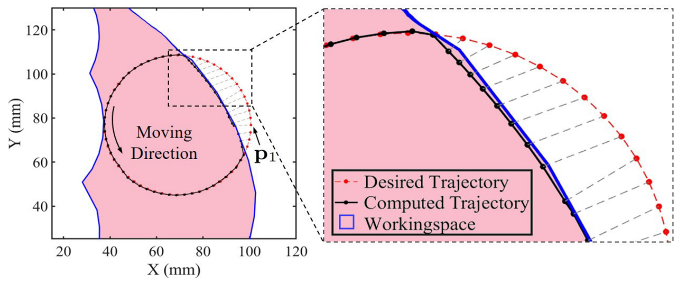

Fig. 14. The results of IK computing and trajectory following with a desired trajectory $\mathcal{L}$ that is partially out of the soft robot's working space. Waypoints (red) on $\mathcal{L}$ and their corresponding reachable points (black) determined by our IK solution are visualized by the gray dash lines.

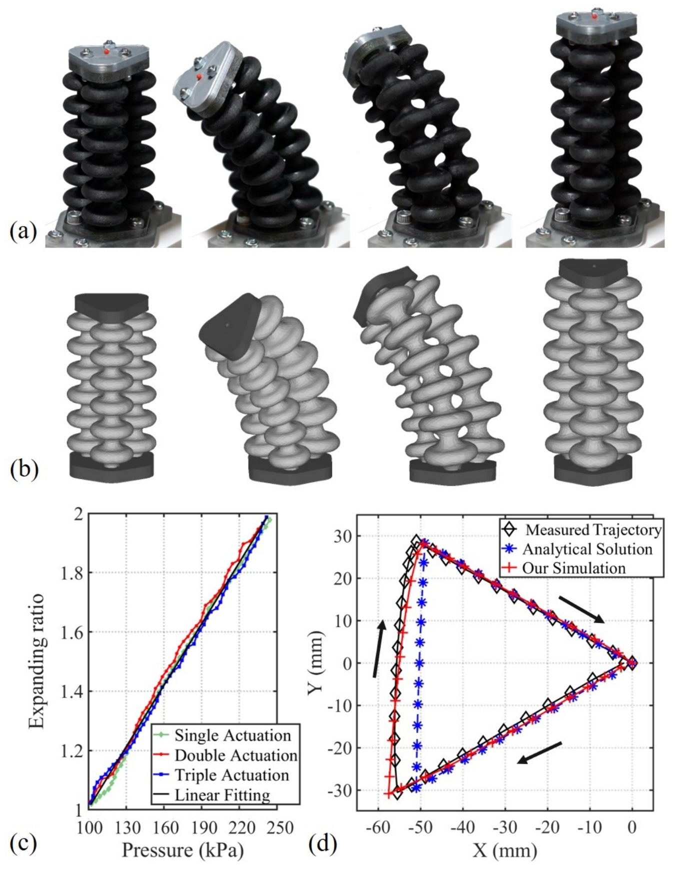

Fig. 15. The experiments taken on a multi-chamber 3D printed pneumatic-driven soft actuators, which is reproduced from [48]. (a) The physical behavior under actuation – from left to right: rest shape, expanding one chamber, expanding two chambers and expanding all chambers with the same volume-change ratio. (b) The results computed by our forward kinematic algorithm. (c) Calibrated ratio for the relationship between pressures and expanding ratios under different actuation statuses. (d) Study the trajectories for the tip point (shown as the red dot in (a)) by comparing the results of physical tests, analytic computation [43] and our forward kinematic algorithm, where the black arrow shows the direction of tip moving.

target trajectories. The errors of motion are also visualized as two error curves shown in Fig. 13(b). Besides computation error, errors in physical experiments are also generated by many other factors, including the fabrication error, the control strategy and the unpredictable friction between cables and the soft finger.

By sampling the configuration space for a soft robot, one intuitive solution of trajectory following can be realized by directly searching the closest sample points in $\mathcal{P}_\omega$ and using their corresponding parameters for actuation. However, this method needs very dense sampling rate to guarantee the required numerical accuracy that is comparable with our IK computing. Although our FK computing is very fast, the cost of this searching-based planning is still much higher than the IK-based trajectory following method presented in **Algorithm** 3. For example, computing 3375 sample points beforehand (see Fig. 8(b)) takes more than 5 minutes. Moreover, continuity is hard to be preserved on a path realized by the sample-searching method. Differently, our IK computing presented in **Algorithm** 2 can ensure the continuity by its nature of an iterative algorithm. We test it on an extreme case as shown in Fig. 14 where part of the desired trajectory falls out of the working space. The result of our algorithm is a smooth path completely inside the feasible region.

C. Kinematic Computing for Pneumatic-driven Soft Actuator

The deformation of pneumatic-driven soft robot is usually driven by changing the pressure of inflation. To conduct kinematic computation by our framework, we should be able to convert the pressure into a ratio of chamber's volume-change as

$$\alpha = V_c / V_c^0 \tag{17}$$

where V_c^0 and V_c are the volumes of chamber before and after inflation respectively. In literature, Mosadegh *et al.* [22] first introduce an experiment setup which can draw pressure-volume (PV) hysteresis curves of soft fingers. Although straightforward, this method is limited as it only supports the actuation of incompressible fluid (i.e., water). Inspired by the volumetric control system present in [48], we developed a general method to calibrate the relationship as $\alpha(P)$ with P being the pressure of inflation.

The most difficult part of this calibration process is that the current volume of deformed chamber V_c is not directly measurable. Without loss of the generality, we can consider a pneumatic-drive soft robot as shown in Fig. 1(c) – the chamber is actuated by a syringe pump module meanwhile connecting to a pressure sensor. As a result, both the inflation pressure P and the changed volume of air in the syringe ΔV_{sy} can be measured. As the system is closed, we can derive the following equation for two balanced statuses of the system based on the *Idea Gas Law*.

$$\bar{P}(V_c^0 + V_{sy}^0 + V_t) = P(V_c + V_{sy}^0 - \Delta V_{sy} + V_t), \tag{18}$$

where V_c^0, V_t and V_{sy}^0 present the initial volumes of the chamber, the tube and the syringe respectively – all can be obtained from the design. $\bar{P} = 100\text{kPa}$ is used as the standard atmospheric pressure in our calculation. This formula can be converted into

$$\alpha = \frac{V_c}{V_c^0} = \frac{\bar{P}V_c^0 + P\Delta V_{sy} - (P - \bar{P})(V_{sy}^0 + V_t)}{PV_c^0}. \tag{19}$$

Therefore, we only need to measure the value of ΔV_{sy} during the calibration process to obtain the value of α by Eq.(19).

The aforementioned method for calibrating expanding ratios is general and easy to be implemented. We have applied it to a soft robot driven by multiple pneumatic actuators (by the experiment setup shown in Fig. 1(c)). This design was originally presented in [48], and the robot has three chambers that can be actuated individually to bend its body in 3D space. The soft part of the actuator is fabricated by the Object 350 Connex3 printer with the mixture between a rigid (VeroMagenta) and a soft material (Agilus 30 Black), which has a rigidity of shore 70A hardness. In our experiments, the three chambers are pressurized one after another by $P = 100 \sim 240$ kPa, and the related hysteresis curve $\alpha(P)$ can be found in Fig. 15(c).

The forward kinematic computation for actuating multiple chambers are shown Fig. 15(b) and compared with analytic computation [43] and physical tests. The trajectories of the tip's moving are plotted in Fig. 15(d). It is easy to find that our algorithm can generate results more accurately than the analytic prediction method presented in [43], which determines the position of an investigated point by simply combining the prediction results of individual chambers. The maximum tracking error (on every waypoint) observed on our results of forward kinematic computation is less than 3mm throughout the whole trajectory. The dimension of this soft actuator is $48\text{mm} \times 48\text{mm} \times 136\text{mm}$ and the model used in kinematic computing contains $135k$ tetrahedra.

Efficiently predicting the required pressure that can generate an expected deformation on a pneumatic-driven soft robot can be very challenging for the conventional methods (such as the static force modeling [4] or the FEM analysis [7]) because of the highly non-linearity of the problem. Benefited by the efficiency of our kinematic computation, we can also solve the trajectory planning on the pneumatic-driven soft actuator. Specifically, **Algorithm** 3 is applied to compute the required volume V_c for each chamber, which is then converted into a volume-change ratio α and mapped to the required pressure P to be provided by a pump. The result is shown in Fig. 16 with the tracked trajectory plotted in the top-view while comparing with the desired trajectory. The offline IK computation conducted in the 3D space and the procedure of actuation have been provided in the supplementary video.

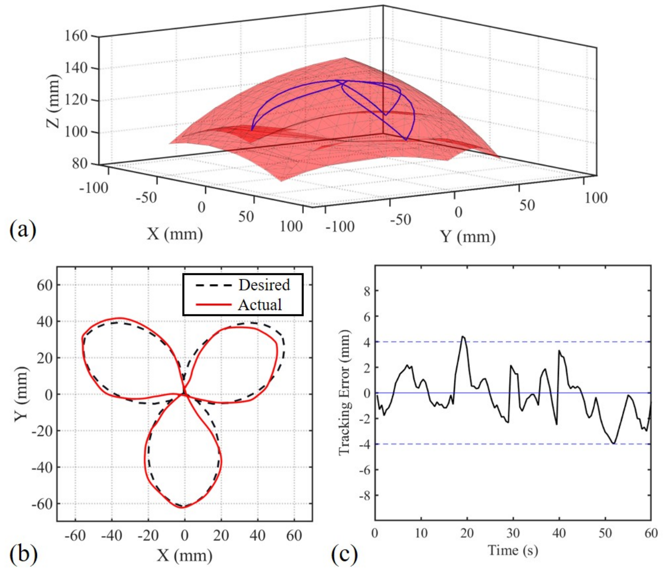

Fig. 16. Results of trajectory planning on a pneumatic-driven soft robot: (a) Configuration space for the tip point on this robot is presented as the red 3D region, where the blue curve gives the target trajectory. (b) Top view of tracked trajectory realized by our IK computation based actuation. (c) Corresponding position error for the tracked tip point.

VI. Conclusion and Discussion

In this paper, we have presented a novel framework to solve kinematic problems for soft robots based on geometric computing. In our method, both the soft body of robots and different types of actuations are modeled as geometric elements that are integrated in an energy optimization formulation. Meanwhile, the distribution of multiple materials on the body of a soft robot is formulated by giving different stiffness to different elements, where the stiffness is represented by a calibrated shape parameter in our framework. We have proposed an efficient optimize-based algorithm for solving forward kinematics and further evolved it to the computation of inverse kinematics. Our method is fast, adaptable for various actuation type and can handle soft robots with complex designs. Compared with existing kinematic solutions, our method makes a good balance between the efficiency and the accuracy in computing. In particular, it shows very excellent performance in convergence and robustness when dealing with large rotational deformation. We have conducted several physical experiments to validate the accuracy of our framework.

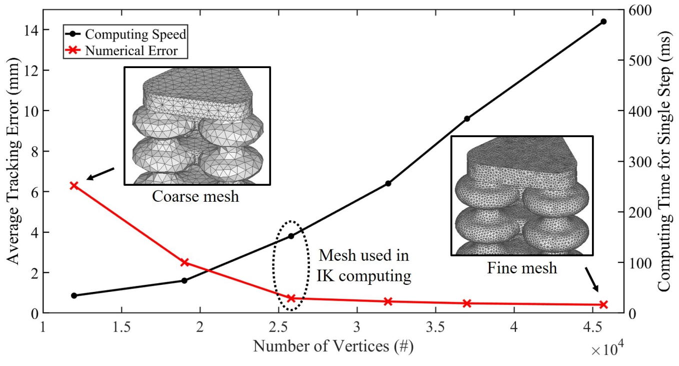

Fig. 17. The performance of our kinematic computing method on meshes with different resolutions from coarse to fine. As can be expected, a finer mesh takes more time on each iteration but can generate more accurate results. In our practice, we use a relatively fine mesh that can also give very accurate prediction (i.e., a mesh with $25.8k$ vertices as shown in the circled dash line).

As a numerical method, our method represents soft robots in a discrete form as volumetric meshes. Our framework supports different types of elements that can precisely describe the model (e.g. tetrahedron for general 3D shape and prism for thin-shell structure). The average time used to compute forward kinematics for a single step of iteration keeps a nearly linear relationship with the number of vertices in a mesh (see Fig. 17). At the same time, we also observe that accurate results can already be achieved when only a relatively fine mesh is employed to conduct the simulation. Specifically, the average tracking error is less than 1mm for the experiment presented in Section V-C) when a relatively fine mesh is employed. In real applications, we always adjust the mesh density and compare the results of FK computing to seek for a good balance between accuracy and efficiency.

There are two major limitations of for our geometric computation based framework. Firstly, the correctness of our formulation relies on the level of elemental deformation falling in the range with small strains so that gives a linear stress-strain relationship. Therefore, the current material model needs to be further extended to support cases with large local stretch – e.g., the soft robots fabricated by silicon rubber. Secondly, viscoelasticity of soft material is not considered in our framework as we only compute kinematics for quasi-static status. The actuation parameters computed by our inverse kinematic and trajectory following algorithms can perform very well when the actuation speed is relatively slow.

In our future work, the above limitations need to be solved by developing a more advanced material model. Modeling the soft robot driven by field-defined actuation (e.g. magnetic or electric field distribution [49]) is also an interesting extension of our framework. Moreover, collision responses can be incorporated into the process of kinematic computation by following the strategy of geometry-based optimization proposed in [50], which will enable the function of simulating a variety of collided interactions.

Acknowledgement

Authors of this paper would like to thank Dr. Jiawei Cao (National University of Singapore) for providing the experiment data of the DEA robot and Tessa Essers (TU Delft) for helping fabricate the soft robot model used in physical experiment.

References

[1] D. Rus and M. Tolley, “Design, fabrication and control of soft robots,” *Nature.*, vol. 521, pp. 467–75, 2015.

[2] J. Burgner-Kahrs, D. C. Rucker, and H. Choset, “Continuum robots for medical applications: A survey,” *IEEE Trans. Robot.*, vol. 31, no. 6, pp. 1261–1280, 2015.

[3] R. Katzschmann, A. D. Marchese, and D. Rus, “Autonomous object manipulation using a soft planar grasping manipulator,” *Soft Robotics.*, vol. 2, pp. 155–164, 2015.

[4] 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.

[5] M. Skouras, B. Thomaszewski, S. Coros, B. Bickel, and M. Gross, “Computational design of actuated deformable characters,” *ACM Trans. Graph.*, vol. 32, no. 4, 2013.

[6] N. Bartlett, M. Tolley, J. Overvelde, J. Weaver, B. Mosadegh, K. Bertoldi, G. M Whitesides, and R. Wood, “A 3d-printed, functionally graded soft robot powered by combustion,” *Science.*, vol. 349, 2015.

[7] D. Drotman, S. Jadhav, M. Karimi, P. deZonia, and M. T. Tolley, “3d printed soft actuators for a legged robot capable of navigating unstructured terrain,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017.

[8] R. B. N. Scharff, R. M. Doornbusch, X. L. Klootwijk, A. A. Doshi, E. L. Doubrovski, J. Wu, J. M. P. Geraedts, and C. C. L. Wang, “Color-based sensing of bending deformation on soft robots,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018.

[9] J. Cao, W. Liang, Y. Wang, H. P. Lee, J. Zhu, and Q. Ren, “Control of a soft inchworm robot with environment adaptation,” *IEEE Transactions on Industrial Electronics*, pp. 1–1, 2019.

[10] K. H. Lee, M. C. W. Leong, M. C. K. Chow, H. C. Fu, W. Luk, K. Y. Sze, C. K. Yeung, and K. W. Kwok, “Fem-based soft robotic control framework for intracavitary navigation,” in *IEEE Int. Conf. Real-time Computing and Robotics*, 2017.

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

[12] S. Tokumoto and S. Hirai, “Deformation control of rheological food dough using a forming process model,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2002, pp. 1457–1464.

[13] E. T. Roche, M. A. Horvath, I. Wamala, A. Alazmani, S.-E. Song, W. Whyte, Z. Machaidze, C. J. Payne, J. C. Weaver, G. Fishbein, J. Kuebler, N. V. Vasilyev, D. J. Mooney, F. A. Pigula, and C. J. Walsh, “Soft robotic sleeve supports heart function,” *Sci. Transl. Med.*, vol. 9, no. 373, 2017.

[14] G. S. Chirikjian and J. W. Burdick, “A modal approach to hyper-redundant manipulator kinematics,” *IEEE Trans. Robot. Autom.*, vol. 10, no. 3, pp. 343–354, Jun. 1994.

[15] B. A. Jones and I. D. Walker, “Kinematics for multisection continuum robots,” *IEEE Trans. Robot.*, vol. 22, no. 1, pp. 43–55, Feb. 2006.

[16] D. Trivedi, A. Lotfi, and C. D. Rahn, “Geometrically exact models for soft robotic manipulators,” *IEEE Trans. Robot.*, vol. 24, no. 4, pp. 773–780, Aug. 2008.

[17] M. Giorelli, F. Renda, M. Calisti, A. Arienti, G. Ferri, and C. Laschi, “A two dimensional inverse kinetics model of a cable driven manipulator inspired by the octopus arm,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 3819–3824.

[18] F. Renda, M. Giorelli, M. Calisti, M. Cianchetti, and C. Laschi, “Dynamic model of a multibending soft robot arm driven by cables,” *IEEE Trans. Robot.*, vol. 30, no. 5, pp. 1109–1122, Oct. 2014.

[19] M. Giorelli, F. Renda, M. Calisti, A. Arienti, G. Ferri, and C. Laschi, “Neural network and jacobian method for solving the inverse statics of a cable-driven soft arm with nonconstant curvature,” *IEEE Trans. Robot.*, vol. 31, no. 4, pp. 823–834, Aug. 2015.

[20] P. Polygerinos, Z. Wang, J. T. B. Overvelde, K. C. Galloway, R. J. Wood, K. Bertoldi, and C. J. Walsh, “Modeling of soft fiber-reinforced bending actuators,” *IEEE Trans. Robot.*, vol. 31, no. 3, pp. 778–789, June. 2015.

[21] Z. Wang and S. Hirai, “Soft gripper dynamics using a line-segment model with an optimization-based parameter identification method,” *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 624–631, April. 2017.

[22] 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," *Adv. funct. Mater.*, vol. 24, no. 1, pp. 2163–2170, 2014.

[23] 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," *Adv. Eng. Mater.*, vol. 18, no. 6, 2016.

[24] A. A. Stanley and A. M. Okamura, "Deformable model-based methods for shape control of a haptic jamming surface," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 2, pp. 1029–1041, 2016.

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

[26] 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," in *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*. Springer, 2012, pp. 283–321.

[27] C. Duriez, E. Coevoet, F. Largilliere, T. Morales-Bieze, Z. Zhang, M. Sanz-Lopez, B. Carrez, D. Marchal, O. Goury, and J. Dequidt, "Framework for online simulation of soft robots with optimization-based inverse model," in *Proc. IEEE Int. Conf. Simulation, Modeling, and Programming for Autonomous Robots.*, 2016, pp. 111–118.

[28] 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.

[29] K.-H. Lee, D. K. Fu, M. C. Leong, M. Chow, H.-C. Fu, K. Althoefer, K. Y. Sze, C.-K. Yeung, and K.-W. Kwok, "Nonparametric online learning control for soft continuum robot: An enabling technique for effective endoscopic navigation," *Soft Robotics.*, vol. 4, no. 4, 2017.

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

[31] D. Navarro-Alarcon and Y. H. Liu, "Fourier-based shape servoing: A new feedback method to actively deform soft objects into desired 2-d image contours," *IEEE Trans. Robot.*, vol. 34, no. 1, pp. 272–279, 2018.

[32] M. Li, R. Kang, D. T. Branson, and J. S. Dai, "Model-free control for continuum robots based on an adaptive kalman filter," *IEEE/ASME Trans. Mechatronics.*, vol. 23, no. 1, pp. 286–297, Feb 2018.

[33] Z. Zhang, T. M. Bieze, J. Dequidt, A. Kruszewski, and C. Duriez, "Visual servoing control of soft robots based on finite element model," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 2895–2901.

[34] G. Fang, C.-D. Matte, T.-H. Kwok, and C. C. Wang, "Geometry-based direct simulation for multi-material soft robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, May. 2018, pp. 4194–4199.

[35] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in *Proc. Eurographics Symposium on Geometry Processing*, 2007.

[36] M. Shapira and A. Rappoport, "Shape blending using the star-skeleton representation," *IEEE Computer Graphics and Applications.*, vol. 15, no. 2, pp. 44–50, Mar. 1995.

[37] T. Hassan, M. Manti, G. Passetti, N. d'Elia, M. Cianchetti, and C. Laschi, "Design and development of a bio-inspired, under-actuated soft gripper," in *IEEE Int. Conf. Engineering in Medicine and Biology Society*, Aug 2015, pp. 3619–3622.

[38] R. Pelrine, R. Kornbluh, Q. Pei, and J. Joseph, "High-speed electrically actuated elastomers with strain greater than 100%," *Science.*, vol. 287, no. 5454, pp. 836–839, 2000.

[39] S. Bouaziz, M. Deuss, Y. Schwartzburg, T. Weise, and M. Pauly, "Shape-up: Shaping discrete geometry with projections," *Comput. Graph. Forum.*, vol. 31, no. 5, pp. 1657–1667, Aug. 2012.

[40] S. Buss, "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods," *IEEE Trans. Robot. Autom.*, vol. 17, no. 1, May. 2004.

[41] H. P. Moreton and C. H. Séquin, "Functional optimization for fair surface design," *SIGGRAPH Comput. Graph.*, vol. 26, no. 2, pp. 167–176, 1992.

[42] R. B. N. Scharff, J. Wu, J. M. P. Geraedts, and C. C. L. Wang, "Reducing out-of-plane deformation of soft robotic actuators for stable grasping," in *IEEE Int. Conf. Soft Robotics*, April. 2019, pp. 265–270.

[43] D. Drotman, M. Ishida, S. Jadhav, and M. T. Tolley, "Application-driven design of soft, 3d printed, pneumatic actuators with bellows," *IEEE/ASME Trans. Mechatronics.*, 2018.

[44] "Eigen v3.3," http://eigen.tuxfamily.org, 2017.

[45] J. Cao, W. Liang, Q. Ren, U. Gupta, F. Chen, and J. Zhu, "Modelling and control of a novel soft crawling robot based on a dielectric elastomer actuator," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 4188–4193.

[46] T. Lu, J. Huang, C. Jordi, G. Kovacs, R. Huang, D. R. Clarke, and Z. Suo, "Dielectric elastomer actuators under equal-biaxial forces, uniaxial forces, and uniaxial constraint of stiff fibers," *Soft Matter*, vol. 8, pp. 6167–6173, 2012.

[47] "Marlin firmware," http://marlinfw.org, 2018.

[48] T. Kalisky, Y. Wang, B. Shih, D. Drotman, S. Jadhav, E. Aronoff-Spencer, and M. T. Tolley, "Differential pressure control of 3d printed soft fluidic actuators," *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 6207–6213, Sept 2017.

[49] F. Chen, K. Liu, Y. Wang, J. Zou, G. Gu, and X. Zhu, "Automatic design of soft dielectric elastomer actuators with optimal spatial electric fields," *IEEE Trans. Robot.*, vol. 35, no. 5, pp. 1150–1165, Oct. 2019.

[50] D. Harmon, D. Panozzo, O. Sorkine, and D. Zorin, "Interference-aware geometric modeling," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 137:1–137:10, Dec. 2011.