

A Robot-Object Unified Modeling Method for Deformable Object Manipulation in Constrained Environments

Hao Deng , Member, IEEE, Faizan Ahmad , Jing Xiong , Member, IEEE, and Zeyang Xia , Senior Member, IEEE

Abstract—Deformable object manipulation (DOM) holds significant importance in a variety of robotic applications. However, due to the absence of computationally efficient and accurate models, manipulating such objects remains a challenge. This complexity arises from the intricate laws of deformation and the high dimensionality of shape states. While prevailing solutions address DOM primarily using explicit servo-control methods in a model-free manner for task-specific local shape attainment, these methods falter when confronting more complicated tasks that demand global model-based planning. In response, we present a unified modeling method for DOM planning within constrained environments. Our approach integrates manipulating motions, object shapes, and environmental constraints into a singular physics-based deformation model, ensuring accurate computation of a unified robot-object state at each computational phase. By harnessing the alternating direction method of multipliers-based parallel numerical recipe with a learning-based sim2real parameter estimation strategy, we achieve superior computational efficiency and modeling accuracy. The detailed numerical evaluations and sim-to-real experiments show that our model outperforms the existing methods on DOM tasks with an updating rate > 25 FPS and a relative deformation error $< 10\%$. Furthermore, we demonstrate the practical utility of our model in planning a global manipulation task.

Index Terms—Deformable object manipulation (DOM), modeling, physics-based modeling.

I. INTRODUCTION

DEFORMABLE object manipulation (DOM) has given rise to an emerging set of robotic applications in recent years [1]. Notable examples include cable or tube routing in industries [2], [3], [4], soft tissue handling in surgical procedures [5], [6], [7], and the preparation of metal appliances in biomechanics [8].

Using the typical DOM task illustrated in Fig. 1 as an example, it is evident that the general objective of a DOM task is to achieve a specific spatial distribution or to mold the object into a desired morphological shape. This endeavor can be abstracted into an object manipulation and control problem in an obstacle space, as shown in Fig. 2. Here, the object's shape is both the objective of planning and a constraint for the robot(s). Therefore, a key aspect of addressing this problem is to predict the shape outcomes from potential robot actions.

Existing solutions for DOM can be categorized based on how object shape outcomes are sourced: from online observation or offline theoretical analysis. These categories are: 1) backward online visual servo control and 2) forward offline model-based planning.

The first category simplified the approach by sidestepping the complexity of the underlying physical laws. Solutions here intuitively extract shape features from specific points of interest within the object using external sensors. Based on these data, they determine how the robot(s) should move the object in order to converge to a desired shape, adhering to a predefined explicit control policy [11], [12], [13], [14], [15], [16]. However, these visual servoing methods operate on a “look and move” pattern, which makes it challenging to predict whether the desired shape is attainable due to the absence of global constraint information. As a result, this first category of solutions is suitable for local shape control in unobstructed space but struggles in more complicated scenarios, like those with obstacles, as illustrated in Fig. 2.

By contrast, solutions in the second category advocate the necessity of global planning. Such strategies account for unforeseen stationary obstacles, formulating global offline models that prioritize high autonomy and complete collision-free trajectories. However, global planning, especially when involving

Manuscript received 13 October 2023; revised 18 December 2023; accepted 15 February 2024. Date of publication 18 March 2024; date of current version 17 December 2024. This work was supported in part by the National Natural Science Foundation of China under Grant U2013205, Grant 62073309, and Grant 61773365, in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2022B1515020042, and in part by the Shenzhen Science and Technology Program under Grant JCYJ20220818101603008 and Grant JCYJ20210324115606018. Recommended by Technical Editor Markus (TE & AIM TE) Grebenstein and Senior Editor Seiichiro Katsura. (Corresponding author: Zeyang Xia.)

Hao Deng and Faizan Ahmad are with the Soft Robotics Research Center at Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China, and also with University of Chinese Academy of Sciences, Beijing 100040, China (e-mail: hao.deng@siat.ac.cn; faizanahmad@siat.ac.cn).

Jing Xiong is with the Research Center for Medical Robotics and Minimally Invasive Surgical Devices, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China (e-mail: jing.xiong@siat.ac.cn).

Zeyang Xia is with the School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zxia@sjtu.edu.cn).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TMECH.2024.3371111>.

Digital Object Identifier 10.1109/TMECH.2024.3371111

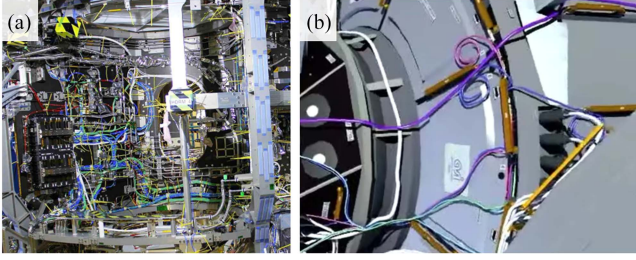


Fig. 1. Illustration of a typical type of DOM task. (a) Cable routing in aircraft manufacturing [9]. (b) Cable routing between docking adapters [10].

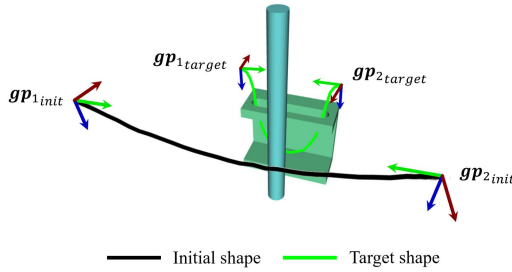


Fig. 2. General purpose of typical DOM tasks: An object manipulation and control problem in an obstacle space.

the use of search-based planners, mandates a forward prediction model. At present, constructing such models, particularly when they include deformable objects, remains a formidable challenge [17]. In light of this, our present article concentrates on resolving the fundamental modeling problem in DOM.

When a robot interacts with a deformable object, its actions both move the object and change its shape. In contrast to the 6-DOF motion of a rigid body in $SE(3)$, the geometry of the deformable object occupies a continuously dynamic space, the calculation of which depends on its critical material composition, referred to as the deformation calculation. From the technical aspect of DOM, three primary challenges emerge when constructing a global offline model: 1) the totally different continuous geometry and the more complicated underlying physics of deformation; 2) the disparity between simulations and real-world applications (sim2real); and 3) the high degrees of freedom (DoF).

To address the first challenge, extensive research spanning from mechanical engineering to computer graphics (CG) has been undertaken. The finite element method (FEM), widely employed in mechanical engineering, shows its advantage in high physical fidelity. However, the accuracy of a classic FEM is computationally expensive [14]. There have been studies [18], [19] on computational acceleration, while there is still a significant delay of several orders of magnitude. CG character animation solutions adopt considerably simplified underlying laws, such as the mass-spring-systems (MSSs) [21], the position-based-dynamics (PBD) [22], and the projective dynamics (PD) [23]. However, most CG methods have a clear limitation: they yield scenes that are visually plausible but not physically accurate. Even though the PD method attempted to introduce extra physics-based constraints implicitly, its high efficiency will

easily be lost in our DOM scenarios due to a dynamic constraint from the robot. Given how a robot interacts with a deformable object, it is crucial that deformation calculations respond almost instantaneously to robotic operation while maintaining a reasonable accuracy level. To date, there have been few reports on models that combine both rigid robots and deformable objects in one unified world.

Acquiring accurate physical parameters for real-world applications can be elusive, often necessitating additional tuning efforts to decrease the sim2real mismatches. Two main strategies have been proposed in the literature to bridge the gap. The first is an online correction strategy, which uses the measured shape data to rectify cumulative errors [24], [25]. This method allows for direct adjustments to particles of interest within the object according to the more accurate online tracking data. However, its applicability is limited when transitioning to new DOM tasks because the model parameters remain static. The other strategy uses a learning-based approach for parameter identification. Online measurement data are fed into a learning-based agent, which then estimates any uncertainties or property mismatches. The updated model with tuning parameters can then cater to a broader range of scenarios. However, given the challenges previously discussed, efficient data collection for training—whether from simulations or physical robot interactions—remains an impediment. The training process for these agents can, thus, be tedious, hindered by the scarcity of accessible DOM dataset [24].

Even if both of the above challenges are surmounted, the model established is not readily applicable for DOM task planning. The object's state derived from the deformation model can encompass DoFs as high as the tens of thousands, and such extensive DoFs inevitably lead to the “curse of dimensionality” for motion planners [26]. Regarding this challenge, the study in [17], [27], and [28] purposed a reduced state representation. Yet, the conundrum of balancing calculation accuracy, computational efficiency, and MOR DoF-state output remains an open question.

In summary, a forward model-based prediction with global planning offers a distinct edge in DOM tasks, especially in constrained environments. However, constructing a global model that encompasses deformable objects is still challenging. This difficulty has led to a distinct divide between the description and modeling of rigid robots and deformable objects. We witness that many practical DOM tasks are still directly performed manually on a physical system or through a repetitive trial-and-error pattern.

Therefore, we introduce a unified modeling framework for both rigid and deformable entities in this article. Within this structure, robot actions, object states, and environmental constraints are all integrated into a singular physical model, facilitating the computation of unified robot-object states. Building upon this unified model, a global searching-based offline planning can be performed for general DOM tasks. Our purposed framework is substantiated through rigorous numerical and sim-to-real experiments.

The principal advancements offered by our research are as follows.

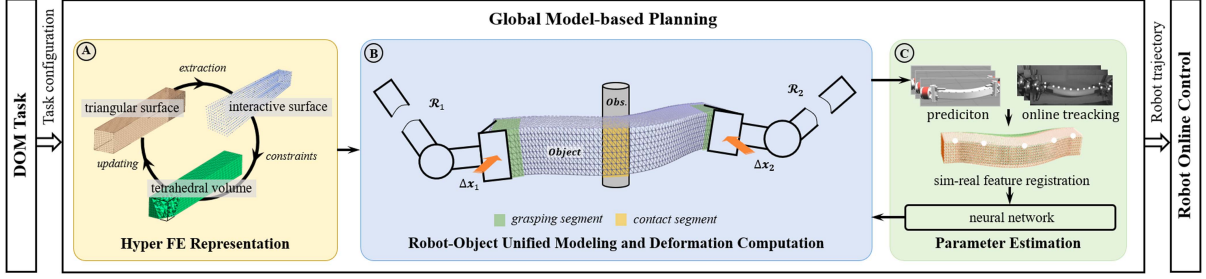


Fig. 3. Overall framework with our purposed unified modeling method for DOM. A general DOM task can be accomplished via conducting a global model-based planning to generate valid trajectory and further sending to robot control for execution. Here, for better illustration, we use a cuboid object as an example.

- 1) Introduction of a general framework for concurrently describing the states of rigid robots and deformable objects. Within this framework, we propose a hyper-FE-based representation for the deformable component. This approach ensures both accurate state calculation and the MOR of the resulting state, laying the groundwork for subsequent planning.
- 2) Deployment of a parallel numerical approach, termed the alternating direction method of multipliers (ADMM)-PD, which leverages implicit underlying physics to resolve the dynamic model both efficiently and accurately.
- 3) Development of a parameter estimation strategy to minimize any model-reality mismatch.

The rest of this article is organized as follows. Section II delves into the overall framework we have advanced for global model-based planning as well as its mathematical preliminaries. Section III depicts the three main modules of our unified modeling method: a hyper-FE model representation, a parallel numerical recipe, and a learning-based parameter estimation strategy. Performance validations are demonstrated in Section IV, followed by selected practical case study in Section V. Finally, Section VI concludes this article.

II. FRAMEWORK AND PRELIMINARIES

A. General Framework of Global Model-Based Planning

In this article, we tackle a general DOM task to obtain a deterministic model for planning. As illustrated in Fig. 3, the model-based planning is incorporated into unified robot-object state prediction in an offline phase. The term “unified” specifically refers to a model containing the robot, deformation object, and all environmental constraints synchronously. For a deformable object, we aim to build a three-layer hyper-FE model (block A in Fig. 3) from the initial CAD data and necessary material properties. Then the hyper-FE model will be used in a physics-based model (block B in Fig. 3) to compute deformation. After that, we also conduct a parameter estimation (block C in Fig. 3) for a new object in an online phase. In this phase, the error between the offline computation and online tracking shape is fed to a neural network for parameter estimation.

B. Mathematical Expressions of Unified Modeling

For generalization and simplicity, we select the DOM, where a dual-arm robot manipulates an object with a known obstacle.

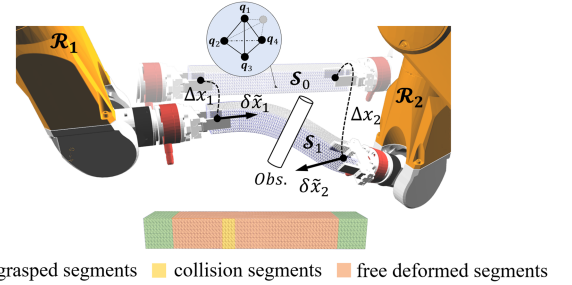


Fig. 4. Description of a DOM task: A dual-arm robot manipulates an object with a known obstacle.

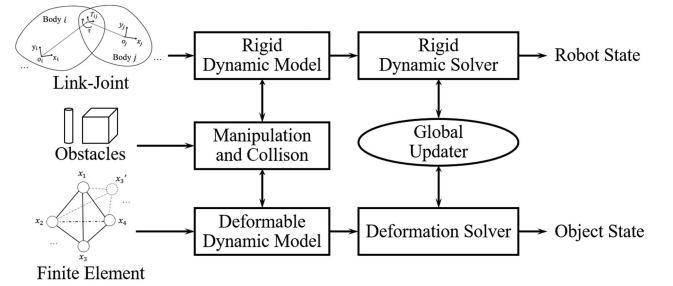


Fig. 5. Unified rigid-deformable object modeling strategy in which two different mechanisms are coupled through mutual interactions.

Consider that a deformable object is physically grasped by a dual-arm-manipulator at ends and interacts with an arbitrary obstacle in a region contact manner, as illustrated in Fig. 4. Here, we use a cuboid object as an example, and note that the DOM can be extended to multirobots with more obstacles.

During the entire manipulation process, contact regions are grasped rigidly and the obstacle is known. Let a manipulating motion Δx be defined as a sequence of each end-effector transition poses $\{\delta \tilde{x}\}$ in $SE(3)$, and ΔS represents the shape change of the object from its initial state S_0 to a desired state S_1 , respectively. A general DOM task can be described as

$$\Delta x := \{\delta \tilde{x}\} \Leftrightarrow \Delta S := S_0 \rightarrow S_1. \quad (1)$$

To couple the rigid mechanism on the left side of (1) and the deformable mechanism on the right side of (1) in a unified description world, we then proposed a unified rigid-deformable object modeling strategy as illustrated in Fig. 5. In which, two different dynamic models are constructed for the rigid robot and deformable object. With known environment configuration, the robot, object, and obstacles can interact mutually through

robotic manipulation and collision. Each dynamic solver can be triggered by the global updater to calculate the corresponding state.

In terms of the rigid robot, classic robotics provides complete methods for modeling and dynamic state solving. On this basis, our following work will focus on the modeling of deformable object and their interaction.

Some fundamental expressions are described as follows.

1) Robot State: Consider a serial rigid manipulator $\mathcal{R}$ with k DOFs. The robot state can be represented by the pose of its end-effector(s) as $\mathbf{x} = \mathbf{x}_k(\boldsymbol{\theta}_k)$, where $\boldsymbol{\theta}_k$ represents the joint states and all are known for control propose.

2) Object State: Consider a deformable object consisting of n vertices and m tetrahedrons from standard FE discretization. The object state based on the work in [29] can be represented by the positions of vertices $\mathbf{q}$, and calculated by solving a nonlinear optimization problem

$$\mathbf{q} = \arg \min_{\mathbf{q}} \left(\frac{1}{2h^2} \left\| \mathbf{M}^{\frac{1}{2}} (\mathbf{q} - \mathbf{s}) \right\|_F^2 + \mathbf{E}(\mathbf{q}) \right) \quad (2)$$

where $\mathbf{s} = \mathbf{q} + h\mathbf{v} + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}$, $\mathbf{v} = \dot{\mathbf{q}}$ is the velocities, $\mathbf{M}$ is a constant diagonal mass matrix, $\mathbf{f}_{\text{ext}}$ sums all external force contributions, h is the time step size for the implicit Euler time integration scheme, $\|\cdot\|_F$ is the Frobenius norm, and $\mathbf{E}(\mathbf{q})$ is the potential term that determines the internal relationship between vertices.

3) Robot-Object Unified State: The overall state of vertices within all segments as shown in Fig. 4 can be expressed as

$$\mathcal{S} := \left[\underbrace{\mathbf{q}_{11}; \cdots; \mathbf{q}_{1k_1}}_{\text{robot}} \mid \underbrace{\mathbf{q}_{21}; \cdots; \mathbf{q}_{2k_2}}_{\text{contact}} \mid \underbrace{\mathbf{q}_1; \cdots; \mathbf{q}_{n-k_1-k_2}}_{\text{free}} \right] \quad (3)$$

where the term robot represents the grasped segments with k_1 vertices, the term contact represents the collision segments with k_2 vertices, and the term free contains all other vertices in free deformed segments. The robot and contact terms are referred to the external constraints of object deformation.

If we define a linear operator $\mathcal{T}(\cdot)$ that maps the poses of end-effectors into the locations of vertices in the grasped segments, $\mathcal{Q}(\cdot)$ that maps the poses of obstacles into the locations of vertices in the collision segments, respectively, and use notation $\mathbf{X}$ to indicate the locations of free vertices, we can rewrite the overall state of the object as

$$\mathcal{S} := [\mathcal{T}(\mathbf{x}), \mathcal{Q}(\mathbf{x}_{\text{obs}}), \mathbf{X}] \quad (4)$$

where $\mathbf{x}_{\text{obs}}$ is the pose of the obstacle in SE(3) and is constant.

Here, we define a combination of robot, object, and obstacle, as a unified robot-object state.

C. Problem Formulation of Deformable Dynamic Model

A forward model-based prediction can be described as: given $\{\mathbf{x}, \dot{\mathbf{x}}, \mathbf{x}_{\text{obs}}\}$ as an input, the task is to estimate a mapping from the states of end-effectors $\{\mathbf{x}, \dot{\mathbf{x}}\}$ to the motion of free vertices $\{\mathbf{X}, \dot{\mathbf{X}}\}$. Note that this process is dynamic. The problem can then be formulated as calculating all vertex states of the deformable object $\{\mathbf{q}^t\}$ under robot manipulation $\{\boldsymbol{\theta}^t\}$ and the

environmental constraint $\{\mathbf{x}_{\text{obs}}\}$

$$\mathbf{q} = \arg \min_{\mathbf{q}} \left(\underbrace{\frac{1}{2h^2} \left\| \mathbf{M}^{\frac{1}{2}} (\mathbf{q} - \mathbf{s}) \right\|_F^2}_{\text{inertia}} + \underbrace{\mathbf{E}(\mathbf{q})}_{\text{potential}} \right) \quad \text{s.t.} \quad \mathbf{G}\mathbf{q} = \mathcal{T}(\mathbf{x}(\boldsymbol{\theta})) \quad \text{and} \quad \mathbf{L}\mathbf{q} = \mathcal{Q}(\mathbf{x}_{\text{obs}}) \quad (5)$$

where $\mathbf{G}$ is a $k_1 \times n$ matrix selecting all grasped vertices, and $\mathbf{L}$ is a $k_2 \times n$ matrix selecting the vertices involved in a collision.

Remark 1: Solving the problem in (5) involves finding the equilibrium between the inertia and potentials terms, taking constraint inputs into account.

III. ROBOT-OBJECT UNIFIED MODELING METHOD

In (5), all variables of $\mathbf{q}$ are high-dimensional, and $\mathbf{E}(\mathbf{q})$ is nonlinear, making it challenging to find the solution in a computationally efficient manner. In this section, we will depict our proposed method to address this issue via robot-object unified modeling. First, we introduce a hyper-FE model representation to achieve a tradeoff between accuracy and dimensionality reduction in state description. Then, the ADMM-based parallel numerical solution is introduced for efficient computation of robot-driven deformation. Three types of constraints for constructing $\mathbf{E}(\mathbf{q})$ are formulated using the same projection technique in PD. After that, a learning-based parameter estimation is adopted to minimize model-reality mismatches, enhancing the accuracy.

A. Representation Using Hyper-FE Model

We aim to reduce DOFs based on the geometry of various regions, and propose a new hyper-FE model consisting of three coupled layers. As shown in block A of Fig. 3. The triangular surface layer and a tetrahedral volume layer are first generated through standard meshing from the CAD file of an object in its initial state. Subsequently, an interactive surface layer with equidistant features sampled over the surface layer is extracted using the farthest point sampling method [30].

Consider a generated hyper-FE model containing t triangular faces with n_1 vertices, m tetrahedrons with n vertices, and the interactive surface with n_2 sampled vertices. By controlling the fineness of meshing and sampling, we ensure that $n_2 \ll n_1 < n$ for reducing dimensionality. Denoting the states of vertices within the interactive surface, triangular surface, and tetrahedral volume as $\{\mathbf{q}_f^t\}$, $\{\mathbf{q}_s^t\}$, and $\{\mathbf{q}^t\}$, respectively, we have $\mathbf{q}_f^t \subseteq \mathbf{q}_s^t \subseteq \mathbf{q}^t$. Therefore, $\{\mathbf{q}_s^t\}$ will be updated from $\{\mathbf{q}^t\}$, which can be calculated under constraints from $\{\mathbf{q}_f^t\}$. The updating procedures of the hyper-FE model are summarized in Algorithm 1.

Remark 2: The significance of our proposed hyper-FE model is to use a specific DOFs-reduction but intercoupled model to achieve a tradeoff between accuracy and dimension reduction.

B. Computation of Robot-Driven Deformation

1) ADMM-Based Parallel Numerical Solution: The ADMM is renowned for its highly parallelizable performance in solving

Algorithm 1: Hyper-FE model Updating Scheme.**Input:** A hyper FE-model $\mathcal{H}$ and motion mapping $\mathcal{T}$

- 1: Initialize $\{q_f^t\}$, $\{q_s^t\}$ and $\{q^t\}$ from $\mathcal{H}$
- 2: **while** model_loop **is** running **do**
- 3: modify $\{q_s^t\}$ from *collision_checker*
- 4: extract and update $\{q_f^t\}$
- 5: receive motion $\mathcal{T}$ to update $\{q^t\}$
- 6: establish all constraints
- 7: compute new $\{q^t\}$ from the theoretical model
- 8: update $\{q_s^t\}$
- 9: **end while**

Output: Updated hyper FE-model $\mathcal{H}$

optimization problems of a specific form [31]

$$\arg \min_{\mathbf{q}, \mathbf{z}} f(\mathbf{q}) + g(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{A}\mathbf{q} + \mathbf{B}\mathbf{z} = \mathbf{C}. \quad (6)$$

In our scenario

$$f(\mathbf{q}) = \frac{1}{2h^2} \left\| \mathbf{M}^{\frac{1}{2}} (\mathbf{q} - \mathbf{s}) \right\|_F^2. \quad (7)$$

For the term $E(\mathbf{q})$, by comparing (6) and (7), we find that if a projection matrix $\mathbf{S}$ exists such that $\mathbf{S}\mathbf{q} = \mathbf{z}$, then $g(\mathbf{z}) := E(\mathbf{q})$ and $\mathbf{W}(\mathbf{S}\mathbf{q} - \mathbf{z}) = 0$, where $\mathbf{W}$ is a user defined weighting matrix. Consequently, $\mathbf{A} = \mathbf{W}\mathbf{S}$, $\mathbf{B} = -\mathbf{W}$, $\mathbf{C} = 0$.

The remaining task involves constructing $E(\mathbf{q})$ for different constraint types and performing a local-global iteration process.

2) Constraints of Deformation: The unified modeling in Section II-B identifies three kinds of constraints determining the object deformation. Our objective is to model each constraint and obtain matrix $\mathbf{S}$ for constructing of $g(\mathbf{z})$.

a) Strain energy constraints: Material deformation is classified into elasticity or elastoplasticity depending on the deformed shape relative to a reference shape postexternal constraints removal. Consider the j th tetrahedral element, illustrated in Fig. 4. Defining the states of its four vertices as q_i^t , where t denotes time and $i = 1, 2, 3, 4$, we derive its deformation gradient as $\mathbf{F} = \mathbf{D}_s \mathbf{D}_m^{-1}$, with $\mathbf{D}_s := (q_1^t - q_4^t, q_2^t - q_4^t, q_3^t - q_4^t)$ as the deformed shape matrix at time t and $\mathbf{D}_m := (q_1^0 - q_4^0, q_2^0 - q_4^0, q_3^0 - q_4^0)$ as the reference shape matrix.

For elastic materials, strain energy is stored in all deformed elements and calculated using constitutive models. We opt for Neo-Hookean model for its simplicity and low computational cost [32]. The strain energy density over the j th element is expressed as

$$\Psi_j = \frac{\mu}{2} (\mathbf{I}_1 - \log(\mathbf{I}_3) - 3) + \frac{\lambda}{8} \log^2(\mathbf{I}_3) \quad (8)$$

with $\mathbf{I}_1 = \text{tr}(\Sigma^2) = \sum \sigma_i^2$, $\mathbf{I}_3 = \det(\Sigma^2) = \prod \sigma_i^2$, and μ and λ as the Lamé coefficients, calculated from material properties of Young's modules and Poisson's ratio, respectively. Here, σ_i , $i = 1, 2, 3$ is the singular value of $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^T$.

As the strain energy density remains constant over each element, the elastic strain energy constraints are summed as

$$E_s(\mathbf{q}) = \sum_{j=1}^m V_j \Psi_j \quad (9)$$

where V_j represents the volume of each tetrahedron in its reference shape.

For elastoplastic materials, the object will be shifted to a new reference shape once the plastic strain begins to develop. Based on the plastic flow theory [33], wherein a total strain is divided into elastic and plastic parts expressed as $\mathbf{F}^t = \mathbf{F}_e^t \mathbf{F}_p^t$, the remaining elastic portion is calculated as follows.

We first assume that the yield surface remains unchanged during a pure plastic phase from step t to $t+1$. Using the current $\mathbf{F}_p^t$, we obtain a next nominal elastic part: $\bar{\mathbf{F}}_e^{t+1} = \mathbf{F}^{t+1} (\mathbf{F}_p^t)^{-1}$. The singular value decomposition of this part is expressed as $\mathbf{U}_e^{t+1} \bar{\Sigma}_e \mathbf{V}_e^{t+1T}$. Then, a clamp operation on each entry of $\bar{\Sigma}_e$ formulated as $\bar{\Sigma} = (\bar{\Sigma}_e - \mathcal{C}(\bar{\Sigma}_e))^\gamma$ helps obtain a local plastic part: $\mathbf{F}_p^{t+1} = \mathbf{U}_e^{t+1} \bar{\Sigma} \mathbf{V}_e^{t+1T}$, where $\gamma = \nu(1 - \kappa\alpha)$ represents the ratio indicating the portion of the exceeded elastic part transitioning to plasticity. The variables ν , κ , and α signify the flow rate, hardening rate, and a tuning variable associated with the time integration step, respectively. $\mathcal{C}(\cdot)$ denotes a clamp operation governing the singular values within the range of yield criterion $\mathcal{C}$. The remaining elastic part then defined as: $\mathbf{F}_e^{t+1} = \mathbf{F}^{t+1} (\mathbf{F}_p^{t+1})^{-1}$.

With the initial condition $\mathbf{F}_p^{t=0} = \mathbf{I}$ are given, $\mathbf{F}^t$ and $\mathbf{F}^{t+1}$, $\mathbf{F}_e^{t+1}$ is calculated through iterations and an elastic strain energy model in (9) can be established.

b) Robotic manipulation constraints: These constraints are considered “hard” compared to material strain constraints, as they directly influence the movement of the grasped vertices. To formulate this hard constraint in terms of energy, we assume that each involved vertex is anchored to a certain target position by a zero-length spring with an infinite stiffness.

A $k_1 \times n$ matrix $\mathbf{S}_p$, selecting all k_1 vertices involved, is used to express a positional constraint as

$$E_p(\mathbf{q}, \mathbf{p}) = \frac{k}{2} \|\mathbf{S}_p \mathbf{q} - \mathbf{p}\|_F^2 \quad (10)$$

where $\mathbf{p} \in \mathbb{R}^3$ denotes target positions to fix or kinematically control the object. The $k = \infty$ when any vertex is out of target and $\mathbf{S}_p$ is dynamically assembled upon determining the grasping points (GPs). The GPs from the object are detected by collision checking between the gripper and object when robotic manipulation applies.

c) Collision constraints: In the updating loop, the collision detection algorithm takes the current triangular surface state $\{q_s^t\}$ as input and computes any penetration with an arbitrary obstacle in the global environment. To prevent penetration, repulsive forces are applied to each vertex on the contact surface. However, to generate such forces is computationally expensive. In our work, any occurring collision is converted into positional constraint for all vertices within the corresponding contact area. For a given collision constraint arising from an arbitrary contact j , a $k_2 \times n$ matrix $\mathbf{S}_c$ selects the involved k_2 vertices. Here,

$p \in \mathbb{R}^3$ denotes the target positions of vertices where the obstacle and object can be geometrically separated via penalty method. The k becomes infinite when a vertex is in the penetrating area, which can be detected through collision checking between the object and the environment. We then apply (10) to each collision constraint.

We process the collision handling and robotic manipulation reception in Steps 3 and 5 of Algorithm 1, respectively.

Remark 3: One significance of the proposed unified model that couples all components in one world is its ability to detect any interaction resulting from valid manipulation and obstacle collision. This capability is crucial for forward-prediction in global planning.

3) Computation of Robot-Driven Deformation: The iteration process of ADMM is performed as follows:

$$q^{t+1} = \arg \min_q \left(f(q) + \frac{1}{2} \|W(Sq - z^t + \bar{u}^t)\|_F^2 \right) \quad (11)$$

$$z^{t+1} = \arg \min_z \left(g(z) + \frac{1}{2} \|W(Sq^{t+1} - z + \bar{u}^t)\|_F^2 \right) \quad (12)$$

$$\bar{u}^{t+1} = \bar{u}^t + (Sq^{t+1} - z^{t+1}) \quad (13)$$

where $\bar{u} = W^{-1}u$ and u are the scaled dual variable [31].

Locally solving (11) and (12) requires each gradient to vanish at q^{n+1} or z^{n+1} , leading to

$$q^{t+1} = \mathcal{A}^{-1} \left(Ms + h^2 S^T W^T W (z^t - \bar{u}^t) \right) \quad (14)$$

$$\nabla g(z^{t+1}) = W^T W (Sq^{t+1} - z^{t+1} + \bar{u}^t) \quad (15)$$

where $\mathcal{A} := M + h^2 S^T W^T W S$ can be precomputed.

We summarize the entire procedure for solving problem (6) using the ADMM numerical solution in Algorithm 2. A notable aspect is that the z -update in (15) and $\bar{u}$ -update in (13) can be implemented in parallel by substeps z_j and $\bar{u}_j$, respectively, on a multithread CPU or GPU, enhancing computational performance.

C. Parameter Estimation

For a given deformable object, its shape can be observed by a set of feature vertices and denoted as $\mathcal{S} := [q_1; \dots; q_{ca}]$. Let Ξ represent a vector of material parameters for our constructed model $\mathcal{M}$. The theoretically computed shape is denoted as $\hat{\mathcal{S}} = \mathcal{M}(\Xi)$, and the real shape, obtained from online measurement, is represented as $\mathcal{S}_{\text{true}}$. Hence, our task is to find a set of parameters Ξ_{est} that best fits the predicted shape to the real shape. This task can be equivalent to finding a solution to the problem expressed as

$$\Xi_{\text{est}} = \arg \min_{\Xi} \|\mathcal{M}(\Xi) - \mathcal{S}_{\text{true}}\|^2 \quad (16)$$

where Ξ comprises the parameters (μ, λ, ρ) , with ρ representing the element density.

To solve this problem, we have employed a cross-training method based on convolutional neural network (CNN) to estimate these material parameters. During each training round,

Algorithm 2: Robot-Driven State Update Using ADMM.

Input: Current positions q^t and velocities v^t

```

1: Let  $s = q^t + hv^t + h^2 M^{-1} f_{\text{ext}}$ 
2: Locally modify  $s$  according to  $\{q_s^t\}$ 
3: Initialize  $q = s$  and  $\bar{u} = 0$ 
4: for  $i = 1$  to  $\text{numIterations}$  do
5:   for each constraint term  $j$  in  $\text{constraints}$  do
6:     update  $z_j$  using Eq. 15
7:     update  $\bar{u}_j$  using Eq. 13
8:   end for
9:   Update  $q$  using Eq. 14
10:  if convergence criterion then
11:    Terminate the iterations
12:  end if
13: end for
14: Let  $\hat{q} = q$ 

```

Output: Updated positions q^{t+h} and velocities v^{t+h}

the estimated outputs are fed into a simulator to regenerate a new set of predictions. A performance assessment module then evaluates whether the problem in (16) has converged.

The key modules involved in solving the above problem are described as follows:

1) Neural Network Design: We have constructed a 5-layer CNN as the nonlinear mapping module. In the network, ReLU is used as the activation function for the convolutional layers, and tanh is used for the full-connected layer. A max-pooling operation is applied after each convolution. The model is trained for 1000 iterations with a batch size of 8, using the Adam optimizer [34] with a learning rate of 10^{-3} and a weight decay of 10^{-5} . A softmax heatmap activation is applied to the output of 3 neurons as a fully connected layer. Our network is trained using the mean square error (MSE) as the loss function, defined as: $\text{MSE} = \frac{1}{n} \sum (Y_i - \hat{Y}_i)^2$ where Y_i and $\hat{Y}_i$ represent the measured and estimated shape features, respectively, and n is the number items in the training dataset. Here, Y_i is from the online measured set, and $\hat{Y}_i$ is obtained from $\{q_f^t\}$ in the hyper-FE model.

2) Data Augmentation: Collecting a training dataset manually from the physics-based simulator can be time-consuming, despite the high computational efficiency of our proposed modeling method. To train a CNN typically requires a huge amount of data, prompting us to implement data augmentation techniques to enlarge the simulation dataset.

Since our shape coding uses the relative features from the hyper-FE model, we employ transformation data augmentation by generating new observation of object shape in different known coordinates. It is implemented as follows: for a set of collected feature vertices $\{q_i\}^{\mathcal{W}}$ in their original world frame $\mathcal{W}$, a new coordinate $\mathcal{W}_i$ is generated by randomly transforming relative to world frame $\mathcal{W}$. The feature vertices are transformed into a new set of data $\{q_i\}^{\mathcal{W}_i}$ and then sent to the CNN for training. Noted that such data augmentation operation allows for generating infinite new data without changing the

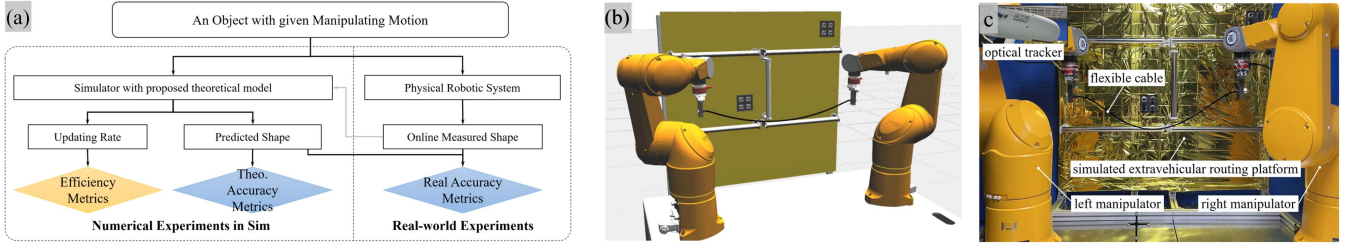


Fig. 6. Validation experiment design and setup. (a) Numerical experiments in simulation and real-world physical experiments are designed to evaluate the efficiency and accuracy of our proposed method. (b) Developed simulator with our method integrated, where the dynamic engine can be easily configured. (c) Real-world physical system for DOM online manipulation.

actual object shape and its corresponding manipulation action. Thus, our model can be trained with a limited amount of real simulation, mitigating the over-fitting problem.

3) Data Collection for Training CNN and Estimation: During the first-time online phase of a DOM task, a script is executed to manipulate the object with a given set of actions while collecting the motion sequence of the object. Using the same script, an offline data collection in the simulation is also executed. With the collected data, we obtain the estimated material parameters. In subsequent offline phases, the simulator receives the updated set of parameters to perform more accurate deformation computations.

IV. VALIDATION EXPERIMENTS AND RESULTS

In this section, we introduce our experimental design and setup, followed by the experiments conducted in simulation and a real-world physical system to evaluate the performance of our purposed modeling method.

A. Experiment Design and Setup

To quantitatively evaluate our purposed method, we have designed two groups of experiments, including numerical experiments in simulation for performance comparison with existing methods, and real-world experiments using a physical robotic system to assess the overall modeling accuracy. The design is illustrated in Fig. 6(a).

In the numerical experiments, we define a manipulating motion for each object, then compute the corresponding robot-driven deformation in the developed simulator using our purposed method and selected existing ones, respectively. Similarly, for real-world experiments, we execute the object motion in a physical robotic system as defined in numerical experiments. The shapes throughout the entire manipulating process are tracked and recorded in a time sequence using an optical measurement system.

Three types of deformable objects are used in our experiments, including elasticity, plasticity, and elastoplasticity. The detailed configuration of these objects is listed in Table II of the Appendix. We sourced parts of the theoretical material parameters from existing data [35] and others from product property manuals.

To implement our experimental design, we first developed a DOM simulator [see Fig. 6(b)] based on Gazebo with ROS [36], and realized our purposed method as a plugin for Gazebo.

All the simulations discussed in this article were executed on an 8-core 16-thread Intel i7 processor with TBB [37] support, without the use of additional GPU acceleration. After that, we developed a dual-arm physical robotic system [see Fig. 6(c)], which features two manipulators (dual-Staubli TX90) equipped with a pair of pneumatic grippers. An optical measurement system (NDI Polaris Vega ST) is used to track real-time positions of laser-reflective feature markers. Additionally, a ceiling-mounted RGB camera is used to capture a global view of the scene. All the hardware components are integrated via high-speed Ethernet under ROS with ROS-Industrial [38].

To evaluate the results from both the simulation and the physical system, we use the following evaluation metrics:

a) Metrics for computational efficiency: The updating rate, indicative of computational efficiency, represents the maximal number of iteration loops per second for deformation computation. It is calculated from the statistical elapsed time of the global cycle from steps 2 to 9 in Algorithm 1.

In this article, we adopt a baseline for computational efficiency aligned with standards prevalent in animation, specifically, a minimum updating rate of 24 frames per second (FPS).

b) Metrics for modeling accuracy: We evaluate the modeling accuracy from two perspectives: *theoretical prediction accuracy* and the *sim2real measurement accuracy*. The relative deformation error is defined as

$$e_s = \frac{\|\hat{\mathbf{S}} - \mathbf{S}_{true}\|_2}{\|\mathbf{S}_{true}\|_2} \times 100\% \quad (17)$$

where $\hat{\mathbf{S}}$ and $\mathbf{S}_{true}$ are the evaluated and true shape.

For the assessment of *theoretical prediction accuracy*, we consider results from standard FEM software used in engineering practices as the true values, which can be easily calculated from the same mesh and material properties. Similarly, for the assessing *sim2real measurement accuracy*, the true values are obtained from the online measurement. In this article, with reference to the relative deformation error associated with visual servoing enhanced by learning methods, as reported in [15] and [39], we establish a 10% baseline for *sim2real measurement accuracy*.

For a fair performance comparison of our proposed method with existing methods, obstacles will not be considered in the following experimental implementation. However, the capability to handle obstacle constraints will be demonstrated in a case study in Section V.

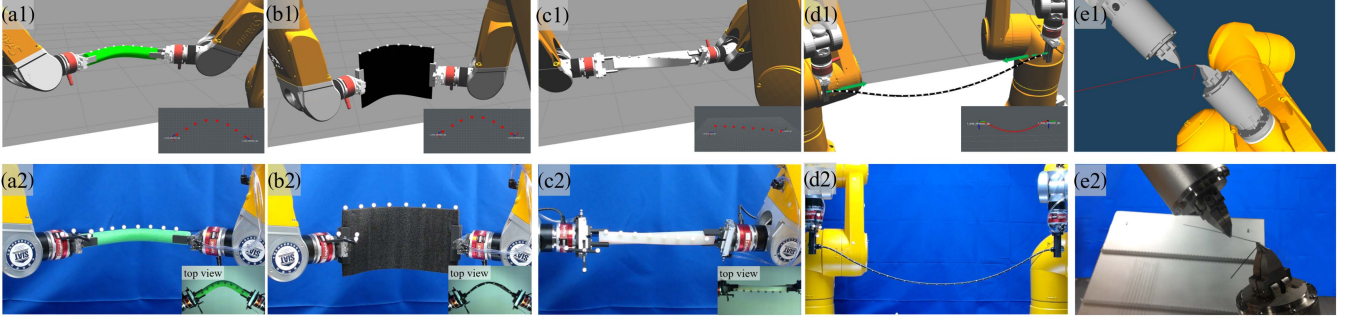


Fig. 7. Performance evaluation experiments for comparison with existing methods in simulation and real-world physical system. (a1) and (a2) Bending cylinder at each end about its vertical axis crossing the centroid. (b1) and (b2) Bending a sheet at each short-edge about its minor centerline. (c1) and (c2) Twisting a cuboid stick at each end about its center axis. (d1) and (d2) Extrusion a flexible cable at each end along its horizontal direction. (e1) and (e2) Bending a metal wire at a point about a vertical axis.

TABLE I
RESULTS OF COMPUTATIONAL EFFICIENCY AND THEORETICAL PREDICTION ACCURACY VALUATION

Object	Updating rate (FPS)					Largest relative deformation error (%) ^a				
	Vega	PBD	GN-PBD ^b	PD	Ours	Vega	PBD	GN-PBD	PD	Ours
cylinder	8.08	40.05	60.07	10.47	25.18	1.21	6.62	5.33	1.56	1.38
board	8.72	40.62	60.86	10.02	25.76	3.78	24.36	19.54	4.83	4.67
cuboid	8.46	41.73	62.95	11.12	26.11	1.23	13.55	15.24	2.52	2.78
cable	8.65	42.25	63.58	12.35	28.45	4.55	29.82	28.56	6.73	6.63
wire	7.83	40.11	60.12	10.36	24.52	5.95	32.88	29.21	8.35	7.47

^aRelative displacement errors are calculated using the standard FEM results as the true value. ^bTraining time is excluded. The bold values indicate the best two groups of results for each metrics.

B. Experiment Implementation and Results

1) *Numerical Experiments*: Following the flow outlined in Fig. 6(a), we compare our approach with four widely used FE-based models from previous published work: Vega [40], PBD, GN-PBD [41], and PD, each implemented as PhysicsPlugin and preconfigured.

Five objects with given manipulating motions are defined as follows. 1) A cylindrical stick undergoing a $\pi/6$ bending action at each end about its vertical axis crossing the centroid. 2) A sheet experiencing a $\pi/6$ bending action at each short-edge about its minor centerline. 3) A cuboid stick with a $\pi/6$ twisting action at each end about its central axis. 4) A flexible cable extended by 10 mm at each end along its horizontal direction. 5) A metal wire with a $\pi/2$ bending at a given point about a vertical axis. Eight feature markers are generated on each object's contour edge using equidistant sampling.

As illustrated in Fig. 7(a1)–(e1), we fetched the elapsed time from iteration loops and recorded the positions of 8 feature points (white solid balls) as predicted shape in the simulation. In addition, for the metal wire, we calculated the final formed angle using the method in [8]. We then evaluated the *computational efficiency* and *theoretical prediction accuracy* by calculating the updating rate and largest relative deformation error, as summarized in Table I.

a) *Result of computational efficiency*: For objects consisting of 11–15 k tetrahedra, our method achieves an updating rate of approximately 26 FPS. This is compared to

8 FPS for Vega, 41 FPS for PBD, 62 FPS for GN-PBD, and 11 FPS for PD. Only our method and PBDs satisfy the >24 FPS updating rate requirement.

b) *Result of theoretical prediction accuracy*: Our method exhibits theoretical modeling accuracies with errors of 1.38%, 4.67%, 2.78%, 6.63%, and 7.47% for each type of object, outperforming PBD (6.62%, 24.36%, 13.55%, 29.82, and 32.88%) and GN-PBD (5.33%, 19.54%, 15.24%, 28.56, and 29.21%). PD's accuracies of 1.56%, 4.83%, 2.52%, 6.37%, and 8.35% are comparable to ours, while Vega achieves the highest accuracies of 1.21%, 3.78%, 1.23%, 4.55%, and 5.95%.

Our method outperforms existing ones in achieving both fine computational efficiency and high prediction accuracy. In particular, a nearly real-time updating rate enables our model to handle dynamic DOM tasks, which is impossible for existing online visual servoing control methods that rely on slow motion to ensure a quasi-static condition.

2) *Real-World Experiments*: In the real-world experiments, we applied the same robot trajectories used in the simulations to a real-world physical robotic system, as illustrated in Fig. 7(a2)–(e2). We attached a set of laser reflective markers along the contour edge line of the first four objects. The real-time location feedback from the optical measurement system ensures the consistency of distribution with the simulation. This setup allowed us to track their 3-D transformation data in a time sequence. For the metal wire, we also measured the final formed angle using

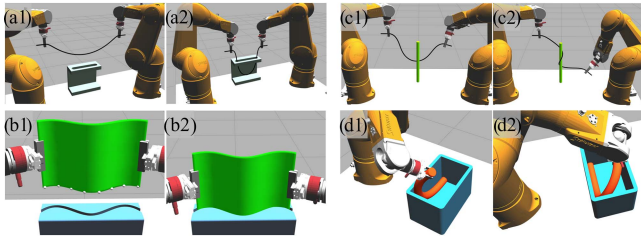


Fig. 8. Results of model-based planning with target shape constraint and obstacle constraint. (a1) and (a2) To insert a cable through a narrow hole. (b1) and (b2) To assembly a sheet into a groove. (c1) and (c2) To wrap a cable around a rigid rod. (d1) and (d2) To pack a flexible rod into a rigid box.

a device in [42]. After coordinate registration, we obtained the online measured shape states and calculated the largest relative deformation error using the theoretical parameters and estimated parameters, respectively.

a) Result of real measurement accuracy: Using the theoretical parameters, the sim2real modeling accuracies are with the errors of 15.96%, 33.75%, 25.81%, 36.08%, and 44.21% for each respective object. After parameter estimation and testing with new manipulating motions, the accuracies improve to 6.65%, 8.68%, 8.12%, 9.22%, and 9.63%. Here, we observed a slight increase in error when estimating the metal wire. This may result from measuring the final angle formed using our developed device rather than optical measurement. Online measurement of such metal wire remains a challenge. Generally, we achieved a relative deformation error of $< 10\%$ in DOM tasks.

V. PRACTICAL CASE IMPLEMENTATION

In this section, we delve deeper into showcasing the practicality of our method for DOM tasks with constraints.

Since our method can model the unified states of robots and objects while efficiently managing interactions between deformable objects and arbitrary environmental obstacles, it is fully capable of complex tasks in clutter. We first designed two types of subtasks to show the model-based planning with the target shape constraint and obstacle constraint, respectively, in simulation. Then, a complete routing task with both target shape and obstacle constraints was planned and conducted in both simulation and the real-world physical system. The detailed results are as follows.

a) Target shape constraint: We created two simulation scenarios, where a flexible cable is to be manipulated to pass through a hole [see Fig. 8(a1) and (a2)] in a base and a sheet is to be manipulated to match a groove for assembling [see Fig. 8(b1) and (b2)]. By integrating the model predictive control (MPC) method [43], our method could predict the intermediate desired shapes of the objects, similar to existing visual servoing control methods, successfully achieving the final target.

b) Obstacle constraint: In another two scenarios, where a rod was introduced as an obstacle, around which a flexible cable needed to be wrapped [see Fig. 8(c1) and (c2)] and a box was introduced as an obstacle, in which a flexible rod needed to be packed [see Fig. 8(d1) and (d2)]. The simulation results show

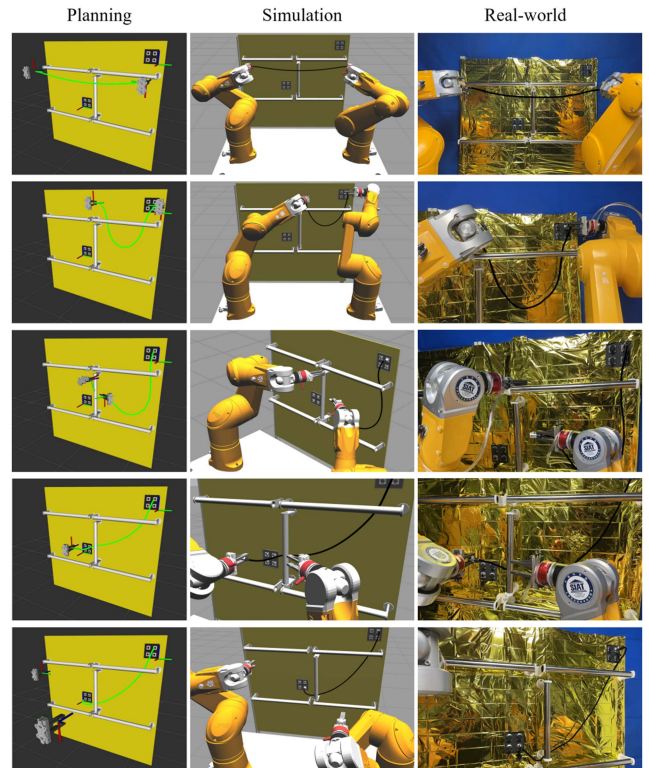


Fig. 9. Results of model-based planning with both obstacle and shape constraints for routing an avionics cable to connect two docking adapters.

that our method can handle complex manipulating motions while considering interaction constraints imposed by obstacles.

c) Both obstacle and shape constraints: As illustrated in Fig. 9, we demonstrated a more complicated robotic cable routing task. A dual-arm robot was tasked to route an avionics cable, connecting two docking adapters, and navigating several bulk-head handrails. To solve this, we integrated our model with the sampling-based methods, RRT-Connect [44] and MPC within the MoveIt framework for global planning. The execution of this intricate task in both our developed simulator and the real-world physical robotic system underscored the efficacy of our proposed method. Our approach proved adept at model-based planning for DOM task in constraint environments, heralding a new frontier in endowing robots with higher autonomy.

VI. CONCLUSION

In this article, we addressed a general manipulation problem for deformable objects in constrained environments using a dual-arm manipulator and particularly focused on modeling for task planning. To address this problem, we proposed a unified robot-object modeling method, achieving fine computational efficiency and high modeling accuracy for robot-driven deformation computation. Our method is based on a hyper-FE-model representation and a local-global parallel updating strategy. An ADMM-based parallel numerical solution and learning-based

TABLE II
DETAILED CONFIGURATION OF OBJECTS USED IN THIS ARTICLE

Object	Dimension (mm)	Material Parameters ^b				FE Info.	
		Density (kg/m ³)	Young's modules (MPa)	Poisson's ratio	Tensile strength (MPa)	Vertex num.	Element num.
cylinder	$\Phi 38 \times 350$	54	2	0.32	1.32	3986	15645
sheet	$370 \times 8.5 \times 185$	38	1	0.20	0.24	4441	14783
cuboid	$325 \times 28.5 \times 28.5$	1800	22	0.49	5.5	3747	14595
cable	$\Phi 7 \times 1000$	1600	4100	0.29	65	4007	13796
wire	$0.04 \times 0.06 \times 100$	7960	194020	0.30	520	3727	11331

^bParts of the initial material parameters are from the data for engineering materials in [35] and others are from product property manuals.

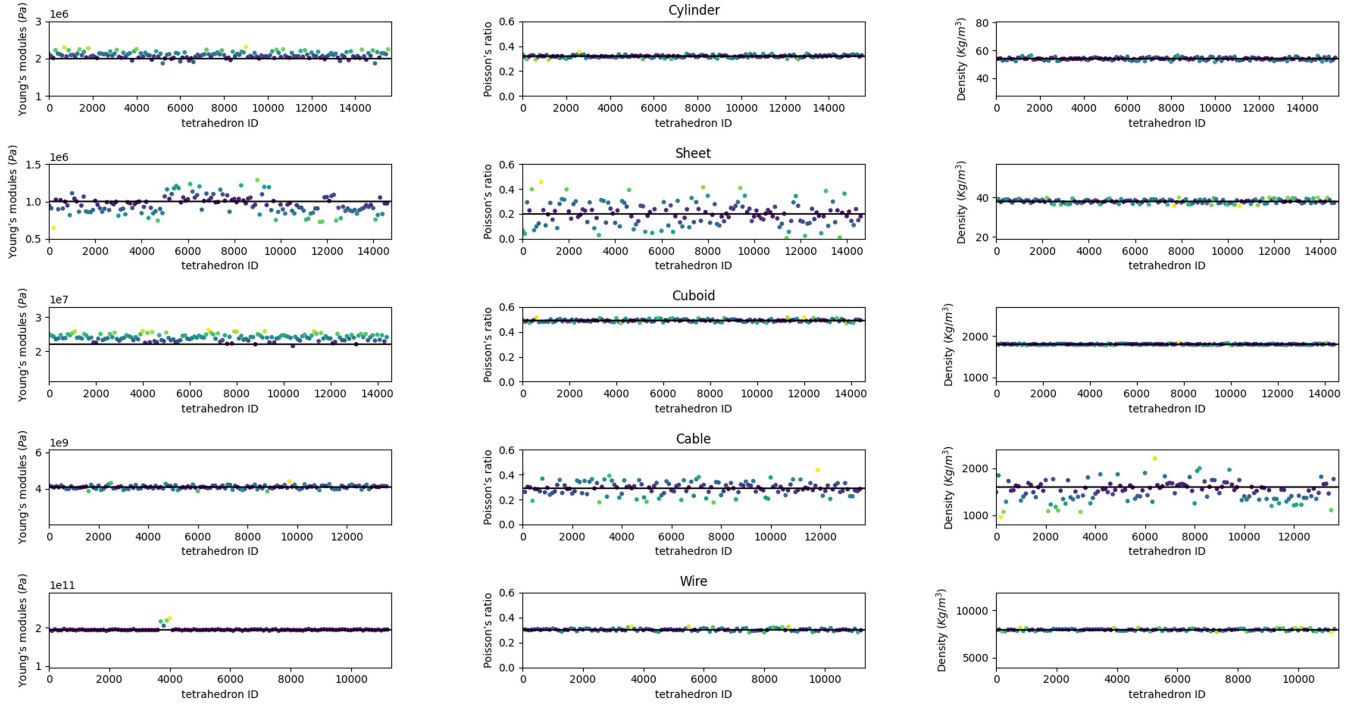


Fig. 10. Theoretical and estimated material properties using the tetrahedron ID as x-coordinate and the material property as y-coordinate. Each straight-line $y = \mu$, $y = \lambda$, and $y = \rho$ indicates the initial property and any deviation from the straight-line indicates the theoretical error compared to its estimated value.

parameter estimation are utilized for better efficiency and improved modeling accuracy. Numerical evaluations and real-world experiments show that our proposed model outperformed existing methods and can be implemented for practical DOM tasks.

The concept of our research follows the classic roadmap of robot manipulation, which roughly includes modeling, offline planning, online execution, and control. This article focused on the basis of modeling, and we would like to continuously work on problems of object-in-the-loop manipulation planning and DOM manipulation from simulation to the real-world in the future. In addition, the perception of deformable objects is still a very challenging topic, which is beyond the scope of this article. Recently, some marker-free approaches have been reported. These approaches can serve as the foundation of our

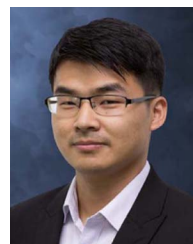
method in terms of parameter estimation and modeling accuracy evaluation. Moreover, we emphasize that our modeling method is general for any DOM task and does not require a quasi-static condition. Thus, it can receive relatively fast manipulation and predict a dynamic material response. With the advantages of both efficiency and accuracy, our method can be integrated with neural networks to achieve performance improvements.

APPENDIX DETAILED CONFIGURATION OF OBJECTS

The detailed configuration of all used objects are listed in Table II. And a fine estimation material parameter distribution for each element in our experiments is given in Fig. 10.

REFERENCES

- [1] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Sci. Robot.*, vol. 6, no. 54, 2021, Art. no. eabd8803.
- [2] T. Matsuno, D. Tamaki, F. Arai, and T. Fukuda, "Manipulation of deformable linear objects using knot invariants to classify the object condition based on image sensor information," *IEEE-ASME Trans. Mechatron.*, vol. 11, no. 4, pp. 401–408, Aug. 2006.
- [3] S. Jin, W. Lian, C. Wang, M. Tomizuka, and S. Schaal, "Robotic cable routing with spatial representation," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 5687–5694, Apr. 2022.
- [4] M. Saha and P. Isto, "Manipulation planning for deformable linear objects," *IEEE Trans. Robot.*, vol. 23, no. 6, pp. 1141–1150, Dec. 2007.
- [5] L. Han, H. Wang, Z. Liu, W. Chen, and X. Zhang, "Vision-based cutting control of deformable objects with surface tracking," *IEEE-ASME Trans. Mechatron.*, vol. 26, no. 4, pp. 2016–2026, Aug. 2021.
- [6] B. Yang, S. Xu, H. Chen, W. Zheng, and C. Liu, "Reconstruct dynamic soft-tissue with stereo endoscope based on a single-layer network," *IEEE Trans. Image Process.*, vol. 31, pp. 5828–5840, 2022.
- [7] P. Boonvisut and M. C. Çavuşoğlu, "Estimation of soft tissue mechanical parameters from robotic manipulation data," *IEEE-ASME Trans. Mechatron.*, vol. 18, no. 5, pp. 1602–1611, Oct. 2013.
- [8] Z. Xia, H. Deng, S. Weng, Y. Gan, J. Xiong, and H. Wang, "Development of a robotic system for orthodontic archwire bending," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 730–735.
- [9] "European service module will serve as the powerhouse for NASA's new Orion spacecraft," Airbus, 2022. Accessed: Nov. 8, 2022. [Online]. Available: <https://www.nasa.gov/exploration/systems/orion/gallery/index.html>
- [10] "SS Robots position international docking adapter," NASA, Aug. 18, 2016. Accessed: Nov. 8, 2022. [Online]. Available: <https://spaceflight101.com/iss/iss-robots-position-ida-2-for-installation-spacewalk/>
- [11] D. Berenson, "Manipulation of deformable objects without modeling and simulating deformation," in *Proc. IEEE-RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 4525–4532.
- [12] D. Navarro-Alarcon et al., "Automatic 3-D manipulation of soft objects by robotic arms with an adaptive deformation model," *IEEE Trans. Robot.*, vol. 32, no. 2, pp. 429–441, Apr. 2016.
- [13] J. Zhu, B. Navarro, R. Passama, P. Fraithe, A. Crosnier, and A. Cherubini, "Robotic manipulation planning for shaping deformable objects with environmental contacts," *IEEE Robot. Autom. Lett.*, vol. 5, no. 1, pp. 16–25, Jan. 2020.
- [14] J. Qi et al., "Contour moments based manipulation of composite rigid-deformable objects with finite time model estimation and shape/position control," *IEEE-ASME Trans. Mechatron.*, vol. 27, no. 5, pp. 2985–2996, Oct. 2022.
- [15] M. Yu, K. Lv, H. Zhong, S. Song, and X. Li, "Global model learning for large deformation control of elastic deformable linear objects: An efficient and adaptive approach," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 417–436, Feb. 2023, doi: [10.1109/TRO.2022.3200546](https://doi.org/10.1109/TRO.2022.3200546).
- [16] Z. Hu, T. Han, P. Sun, J. Pan, and D. Manocha, "3-D deformable object manipulation using deep neural networks," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4255–4261, Oct. 2019.
- [17] P. Mitrano, D. McConachie, and D. Berenson, "Learning where to trust unreliable models in an unstructured world for deformable object manipulation," *Sci. Robot.*, vol. 6, no. 54, pp. 1–12, 2021.
- [18] F. Largilliere, V. Verona, E. Coevoet, M. Sanz-Lopez, J. Dequidt, and C. Duriez, "Real-time control of soft-robots using asynchronous finite element modeling," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 2550–2555.
- [19] P. Ramuhalli, L. Udpa, and S. S. Udpa, "Finite-element neural networks for solving differential equations," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1381–1392, Nov. 2005.
- [20] A. Dziekonski, P. Sypek, A. Lamecki, and M. Mrozowski, "Communication and load balancing optimization for finite element electromagnetic simulations using multi-GPU workstation," *IEEE Trans. Microw. Theory Techn.*, vol. 65, no. 8, pp. 2661–2671, Aug. 2017.
- [21] T. Liu, A. W. Bargteil, J. F. O'Brien, and L. Lavan, "Fast simulation of mass-spring systems," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 1–7, 2013.
- [22] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, "Position based dynamics," *J. Vis. Commun. Image Representation*, vol. 18, no. 2, pp. 109–118, 2017.
- [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, pp. 1–11, 2014.
- [24] F. Liu, Z. Li, Y. Han, J. Lu, F. Richter, and M. C. Yip, "Real-to-sim registration of deformable soft tissue with position-based dynamics for surgical robot autonomy," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 12328–12334.
- [25] M. Afshar et al., "Accurate tissue deformation modeling using a Kalman filter and ADMM-based projective dynamics," *IEEE-ASME Trans. Mechatronics*, vol. 27, no. 4, pp. 2194–2203, Aug. 2022.
- [26] P. Mitrano and D. Berenson, "Data augmentation for manipulation," in *Proc. Robot.: Sci. Syst.*, 2022, pp. 1–10.
- [27] 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.
- [28] A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle, "A material point method for snow simulation," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 1–10, 2013.
- [29] M. Overby, G. E. Brown, J. Li, and R. Narain, "ADMM $\supseteq$ projective dynamics: Fast simulation of hyperelastic models with dynamic constraints," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 10, pp. 2222–2234, Oct. 2017.
- [30] C. Moenning and N. A. Dodgson, "Fast marching farthest point sampling for point clouds and implicit surfaces," *Comp. Lab., Univ. Cambridge, Cambridge, U.K., Tech. Rep. UCAM-CL-TR-565*, 2003.
- [31] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
- [32] E. Sifakis and J. Barbic, "FEM simulation of 3D deformable solids: A practitioner's guide to theory, discretization and model reduction," in *Proc. ACM SIGGRAPH Courses*, 2012, pp. 1–50.
- [33] J. Pan and J. R. Rice, "Rate sensitivity of plastic flow and implications for yield-surface vertices," *Int. J. Solids Struct.*, vol. 19, no. 11, pp. 973–987, 1983.
- [34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, 2014, pp. 1–15.
- [35] M. Ashby, "Material property data for engineering materials," in *Department of Engineering*, 5th ed., Univ. Cambridge, Cambridge, U.K., Oct. 2021.
- [36] W. Qian et al., "Manipulation task simulation using ROS and Gazebo," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2014, pp. 2594–2598.
- [37] A. Kukanov and M. J. Voss, "The foundations for scalable multi-core software in Intel threading building blocks," *Intel Technol. J.*, vol. 11, no. 4, 2007, pp. 309–322.
- [38] C. Vavra, "Open-source robotic software platform, ROS-Industrial, adds efficiency, reach," *Control Eng.*, vol. 62, no. 1, pp. 32–34, 2015.
- [39] R. Lagneau, A. Krupa, and M. Marchal, "Automatic shape control of deformable wires based on model-free visual servoing," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5252–5259, Oct. 2020.
- [40] F. S. Sin, D. Schroeder, and J. Barbič, "Vega: Non-linear FEM deformable object simulator," *Comput. Graph. Forum*, vol. 32, no. 1, pp. 36–48, 2013.
- [41] H. Shao et al., "Accurately solving rod dynamics with graph learning," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2021, pp. 4829–4842.
- [42] H. Deng et al., "Motion planning and control of a robotic system for orthodontic archwire bending," in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, 2015, pp. 3729–3734.
- [43] E. F. Camacho and C. B. Alba, *Model Predictive Control*. Berlin, Germany: Springer, 2013.
- [44] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2000, pp. 995–1001.



Hao Deng (Member, IEEE) received the B.S. degree in mechanical engineering and automation from the Wuhan University of Technology, Wuhan, China, in 2012, the M.S. degree in mechatronic engineering from the Harbin Institute of Technology, Shenzhen, China, in 2015, and the Ph.D. degree in pattern recognition and intelligent systems from the University of Chinese Academy of Sciences and Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, in 2023.

He is currently a Postdoctoral Fellow with the Soft Robotics Research Center, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences. His research interests include manipulation planning and control.



Faizan Ahmad received the B.S. degree in electrical engineering from the National University of Computer and Emerging Sciences, Lahore, Pakistan, in 2013, and the M.S. degree in communication and information systems from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2020. He is currently working toward the Ph.D. degree in pattern recognition and intelligent systems under the supervisions of Prof. Zeyang Xia and Prof. Jing Xiong with Soft Robotics Research Center,

Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China.

His research interests include medical imaging processing, deep learning, augmented reality, and medical robotics.



Jing Xiong (Member, IEEE) received the B.S. and Ph.D. degrees from Tsinghua University, Beijing, China, in 2004 and 2010, respectively, both in mechanical engineering.

She is currently a Full Professor with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China, where she is the Vice Director of Research Center for Medical Robotics and Minimally Invasive Surgical Devices. She was the PI of over ten research grants, including two

National Natural Science Funds, and is the founder of two industrial joint labs. She has authored or coauthored more than 70 peer-reviewed papers, and applied for over 100 patents. Her research interests include medical robotics, intelligent robotics, and image-guided therapy.

Dr. Xiong was the recipient of the Wu Wenjun Artificial Intelligence Science and Technology Award (Natural Science) in 2017. She was the Program Co-Chair of IEEE RCAR 2019, and the Organizing Co-Chair of IEEE CYBER 2020.



Zeyang Xia (Senior Member, IEEE) received the B.S. degree from Shanghai Jiao Tong University, Shanghai, China, in 2002, and the Ph.D. degree from Tsinghua University, Beijing, China, in 2008, both in mechanical engineering.

He is currently a tenured Full Professor with the School of Mechanical Engineering, Shanghai Jiao Tong University. Before that, he was a tenured Full Professor with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China, where

he was the founding Director of the Soft Robotics Research Center and the Director of International Laboratory for Soft Medical Robotics. He was the PI of over 20 research grants, including the National Key Research and Development Project and four National Natural Science Funds. He has authored or coauthored two monographs and authored or coauthored more than 110 peer-reviewed papers, and applied for over 110 patents. His research interests include robotics and biomechanics, specifically medical robotics, soft robotics, humanoid robotics, and biomechanics.

Dr. Xia was the recipient of the Guangdong Natural Science Funds for Distinguished Young Scholar in 2015, the Wu Wenjun Artificial Intelligence Science and Technology Award (Natural Science) in 2017, and Xiong Youlun Excellent Young Scholars Award in 2019, respectively. He is the Technical Editor of IEEE/ASME TRANSACTIONS ON MECHATRONICS. He was the General Chair of IEEE RCAR 2019 and key organizing members of several other IEEE research conferences. He is the Senior Member of CIE/CCF and has been a Fellow of IET since 2019.