2024 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)
July 15-19, 2024. Boston, MA, USA

Bridging Mechanical Behavior Differences of Deformable Soft Objects in Simulation and Experiments Using a Data-Driven Model

Fan-Chien Kao[1], Zhi-Ren Chen[1], Chung-Shan Shih[2], Shao-Huang Lu[2], and Pei-Chun Lin[1*]

Abstract—**The complexity of deformable objects poses a challenge when attempting to replicate real-world behavior in simulation, which impedes the use of simulation as a testing environment for empirical applications. This study aims to create a data-driven model for seamlessly translating real-world deformable objects into simulation environments. Compressed soft balls are studied as an example of this strategy. Using machine learning, the model refines simulation parameters based on experimental data, such as forces and contours, allowing for highly realistic simulations and applications in areas such as manipulator manipulation interactions and reinforcement learning for task strategies.**

I. Introduction

Having high-fidelity simulations for empirical systems is essential for factory automation. Using simulation as a tool can significantly reduce initial setup and optimization efforts during the process, and it can even incorporate real-time control of the empirical system for better performance. Along with the prevalence of artificial intelligence, such as machine learning, the importance of simulation has increased significantly. It provides an alternative environment for (virtual) testing and data generation, and this learning-based iteration process is critical to improving factory automation's intelligence.

A physics–data hybrid model can combine the advantages of physics-based and data-driven models. The physics-based model can capture the primary behavior of the empirical system governed by physics laws, and the data-driven model can capture other unmodeled behaviors of the practical systems. Numerous related research efforts and practical applications are experiencing significant growth in this field. Among these, tasks involving the operation of robotic arms have substantial potential. For example, reinforcement learning has been implemented on empirical manipulators [1]. However, when dealing with deformable objects as the target of manipulation, the complexity of manipulation tasks escalates. Deformable objects are unpredictable and variable, and learning strategies in such scenarios are time-consuming and uncertain. Consequently, there is a pronounced need to create task environments for deformable objects within simulation engines. This necessitates the development of task-specific strategies, mitigating safety concerns associated with empirical systems while saving time and adhering to economic considerations.

In the research field that combines reinforcement learning with robotic manipulators, reinforcement learning in simulations has seen significant growth with the development of the OpenAI Gym framework [2], an open-source package that provides training environments. A study demonstrated the feasibility of utilizing this simulation software for reinforcement learning by implementing reinforcement learning for robotic arms to perform goal-oriented tasks in the PyBullet environment [3]. Several other research later expanded the scope of reinforcement learning strategies to interactive tasks involving rigid objects and deformable ones, such as soft wires and fabrics [4][5].

The complexity of the task has increased from essential displacement or shape arrangement to research that can achieve the goal of threading soft wires into holes [6]. Furthermore, pursuing non-rigid target strategy learning has led to specialized research on optimizing strategy reinforcement learning outcomes through various algorithms [7], with findings that demonstrate the developmental potential and feasibility of strategy learning for deformable objects. However, these studies focus on only one-dimensional strings or two-dimensional woven fabrics as deformable objects. More complex targets include bags, which are still treated as 2D tasks, similar to fabrics [8]. Previous studies have predominantly addressed only a few key points, with crucial point positions serving as task objectives and criteria for assessing achievement. The methods for collecting and analyzing data from 3D deformable objects have precedents in the field of physical deformable objects. Besides point-based approaches, techniques such as [9], where feedback from forces applied by a gripper is utilized as a control basis between the manipulator and deformable objects, and [10], which leverages RGB-D vision to process three-dimensional deformation data, demonstrate that there are viable approaches to handling 3D deformable objects in the simulation domain.

Reinforcement learning for three-dimensional deformable objects is a relatively unexplored area of research. The challenges become more complex when the learning objectives are extended to three dimensions. Predicting the behavior of deformable objects becomes inherently more complicated, and the accuracy of simulations tends to decrease due to disparities between mathematical representations and real-world phenomena. One significant challenge arises from

This work is supported by Delta Electronics Inc. under contract 111HT907014 and 112HT907004.

[1] F. C. Kao, Z. R. Chen, and P. C. Lin are with the Department of Mechanical Engineering, National Taiwan University (NTU), No.1 Roosevelt Rd. Sec.4, Taipei 106, Taiwan. (Corresponding email: peichunlin@ntu.edu.tw).

[2] C. S. Shih and S. H. Lu are with Delta Research Center, Delta Electronics Inc., No. 186, Ruiguang Rd., Taipei 114, Taiwan.

the mathematical model simulation engines used to compute the behavior of deformable objects. These models often involve parameters that are not directly transferable to real-world scenarios. Consequently, accurately selecting parameters for deformable objects in simulations becomes daunting.

This study aims to address the sources of errors commonly encountered in simulations. It seeks to leverage data-driven approaches to train a model capable of mitigating various sources of error and aligning simulated behavior with real-world performance. Such an alignment will ensure that real-world deformable objects can be accurately represented in the simulation environment, leading to improved fidelity in reinforcement learning and training outcomes. Additionally, the study introduces potential indicators for controlling and evaluating the accuracy of three-dimensional deformable objects in simulations. Ultimately, the goal is to expand the application of deformable objects in operational tasks and reinforcement learning strategies to encompass the more intricate realm of three-dimensional deformable objects.

The remainder of this paper is organized as follows. Section II describes the selected simulation engine and its mathematical modeling for deformable objects. It analyzes and consolidates the influence of the parameters on the performance of deformable objects. Section III outlines the task specifics for data collection and the configuration of the virtual and real task environments. The physical behavior of the target object is examined, and how to organize the data for training is discussed. The latter part of the section focuses on data-driven model training, covering the processing of training data for the target object's static and dynamic behaviors and the training of the fitting model. We further scrutinize the prediction of parameters and their feedback into the results, propose possible sources of errors, and suggest directions for correction. Section IV presents the results of the experimental data and how this fitting method can be integrated with reinforcement learning tasks involving robotic arms. Potential areas for further expansion are also explored.

II. Simulation Software

The simulation engine chosen for the study presented in this paper is PyBullet [11], a Python module for physics simulation for games, robotics, and machine learning. Pybullet is a physics engine that simulates collision detection and soft and rigid body dynamics. Pybullet is a suitable choice among the various commercial simulation software. Compared to others that can deal with either kinematics or rigid body dynamics, Pybullet is among the few that can manage soft body contact. Soft body support includes cloth, rope, and deformable objects.

The reason for using Pybullet as the simulation engine is that in addition to implementing deformable objects, another reason is that it can be combined with the OpenAI Gym to perform reinforcement learning, conducting strategy training for deformable object tasks in simulation scenes. The task of assembling the flexible wire into the slot is shown in Figure 1(a), and the scene of wheels moving on soft ground is shown in Figure 1(b). After the task environment is established and the parameters of each object are properly set in the above scenarios, reinforcement learning can be executed directly in Pybullet. This is a safer, time-saving, and resource-saving method compared to training in the real world.

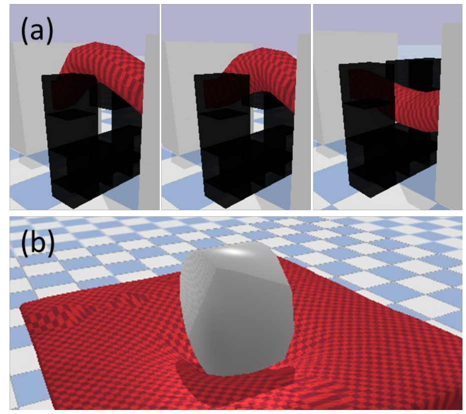

Fig.1 The soft body contact performance scenarios in Pybullet: (a) assembling the flexible wire into the slot and (b) a wheel on the soft ground.

A. Mathematical Model of Soft Bodies

In PyBullet, the mathematical model used to describe the soft body primarily relies on the neo-Hookean model [12], a hyperelastic material model similar to Hooke's law. The neo-Hookean model derives its foundation from the principles of statistical thermodynamics, which govern the intricate behavior of cross-linked polymer chains. This robust and versatile model finds its ideal application in materials sharing traits akin to plastics and rubber, thus extending its relevance to a diverse spectrum of deformable substances.

The PyBullet function responsible for generating soft bodies configures three key parameters: the first Lamé parameter λ, the second Lamé parameter μ, and a damping parameter that influences only the dynamic behavior of the soft body. In continuum mechanics, Lamé parameters are two material-dependent quantities that arise in strain–stress relationships [13]. Equation (1) is the strain–stress definition of Hooke's law in 3D in homogeneous and isotropic materials.

$$\boldsymbol{\sigma} = 2\mu\boldsymbol{\varepsilon} + \lambda\, \mathrm{tr}(\boldsymbol{\varepsilon})\, I \tag{1}$$

where $\boldsymbol{\sigma}$ is the stress tensor, $\boldsymbol{\varepsilon}$ is the strain tensor, I is the identity matrix, and tr is the trace function. The derivation of the neo-Hookean model starts with the strain energy density function as Equation (2) in 2D of a compressible neo-Hookean material:

$$W = C_1(I_1 - 3 - 2\ln J) + D_1(J - 1)^2 \tag{2}$$

where C_1 and D_1 are material constants, I_1 is the first invariant, and J is the Jacobian matrix of the deformation gradient. For consistency with linear elasticity, applying an alternative formulation and alternative definitions of parameters, $\mu = 2\,C_1$ and $\lambda = 2\,D_1$ [13], the strain energy density function turns into Equation (3):

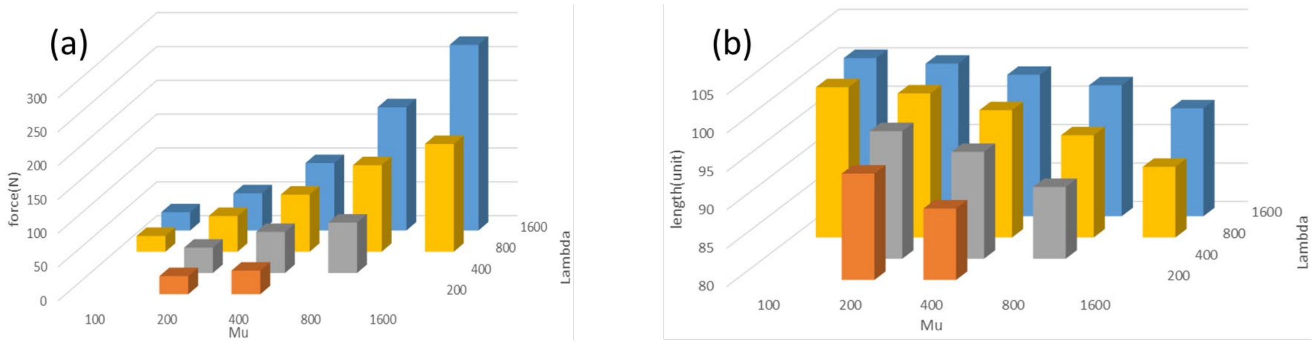

Fig. 2 The soft ball compression experiment simulated in PyBullet. (a) The steady-state forces and (b) the cross-section diameter of soft balls under compression of 30 units in length using different combinations of λ and μ.

$$W = \mu\,(J^{-2/3} I_1 - 3)/2 + (\mu/12 - \lambda/8)\,(J^2 + 1/J^2 - 2) \quad (3)$$

The first Lamé parameter λ has no direct physical definition but relates to the material's compressibility. It can be calculated using Young's modulus (E) and Poisson's ratio (ν). The second Lamé parameter μ is known as the shear modulus (G) in the context of elasticity. Given that the neo-Hookean model is one of the stress-strain relationships, these two parameters are derived from the strain energy density function of neo-Hookean. Their impact on the physical properties of deformable objects primarily pertains to static performance aspects. The physical representation addressed in this paper also focuses on data related to the performance of static states.

B. Simulation Performance of Soft Bodies Under Variable Parameters

The influence of soft body parameters on performance was simulated and studied through simulations with multiple parameter combinations. This approach served as the foundation for Section III (Experiment) in this paper, where ML-generated fitting models are used to estimate the effects of parameters.

The primary observational metrics for evaluating the performance of soft bodies consist of body contours and applied forces. A homogeneous soft ball with a diameter of 90 units was employed as the test object. A downward force of 30 units was applied to the soft ball until it reached a steady state. This procedure was repeated to collect data under various combinations of μ and λ. Due to the computational constraints of the simulation engine, minimal parameter values may result in the collapse of the ball or anomalous shrinkage upon compression. The lower bounds for both parameters are approximately in the range of over one hundred.

Figure 2(a) reveals that parameters λ and μ significantly impact force values, showing a positive correlation. The relationship with contour variation differed. The length of the cross-section diameter after deformation was used as an indicator of contour change. As depicted in Figure 2(b), λ exhibited a positive correlation, while larger values for the shear modulus μ, resulted in reduced deformation effects. Moreover, the scale of influence of μ was significantly smaller than the force, with extreme parameter combinations resulting in differences of approximately ten units in length, less than 11.8% of the average cross-section diameter after deformation, as shown in Figure 3.

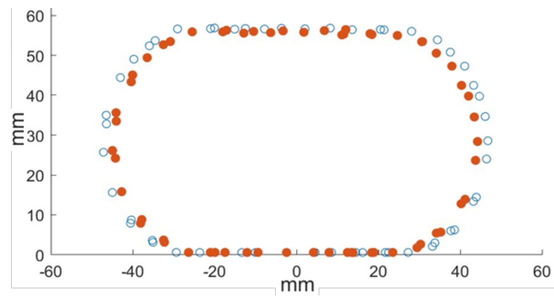

Fig. 3 The contour of the soft ball simulated in PyBullet. Blue: (λ, μ) = (1600, 100), diameter = 100.5988; Orange: (λ, μ) = (800, 1600), diameter = 89.1987.

In Pybullet, NeoHookeanDamping is a parameter relevant to the dynamic properties of soft bodies. This parameter is not mentioned in the neo-Hookean model and appears to be a custom parameter within the physics engine used to determine the dynamic behavior of soft bodies. To assess whether this custom parameter also affects static performance, the two Lamé parameters λ and μ that determine the static behavior constant were maintained while adjusting the value of NeoHookeanDamping to determine its impact on the dynamic behavior of soft bodies as compressing homogeneous soft balls until equilibrium.

Regardless of the value of NeoHookeanDamping, the soft ball's force exhibited maximum instantaneous force at the moment of compression. The soft ball then tended to reach a steady state according to its respective damping performance, corresponding to the behavior of a serially connecting damper. In the case of (λ, μ) = (200, 200), the comparison result indicates that, in general, NeoHookeanDamping only affected the force variations during the approach to equilibrium, and the final force, upon reaching equilibrium, remained unchanged at 27.8N. At more extreme values of NeoHookeanDamping , such as 0.003, the simulation visually exhibited under-damped oscillatory behavior, resembling a water balloon. Regarding force performance, the soft ball in its equilibrium state reduced the force by 4.9% compared to when NeoHookeanDamping are set to 0.03 and 0.3.

III. EXPERIMENTS

The objective of the experiment was to train a fitting model that could predict the μ and λ parameters of real-world deformable objects, allowing their performance in the simulation to match their real-world behavior closely. This alignment will ensure that subsequent behaviors or training outcomes in the simulation closely mirror real-world performance.

The physical target objects for this experiment consisted of two polyurethane foam balls, both with a diameter of 90 mm. One of these balls exhibited a relatively higher bounce than the other, but the specific elastic-related parameters were unknown. Data about contour and force information necessary for machine learning were collected separately within task scenarios designed to resemble real-world and simulation environments.

A. The Experimental Environment

In the empirical experiments, a six-axis manipulator (TM-14, Techman Robot Inc.) was utilized to manipulate the soft balls. A six-axis force/torque sensor (Axia80-M8, ATI) was mounted on the arm's end effector to collect contact force information between the arm and the target objects during motion, as shown in Figure 4(a). The contour data of the balls were obtained using a camcorder (RX100 VII, SONY), with image processing conducted using OpenCV to extract contour information and convert it into point data.

B. The Simulation Environment

The simulation experiment was implemented in the Pybullet environment mentioned in Section II. The target objects were scaled representations of physical balls imported as soft balls using the method described in Section II. In the simulation, a simple robot with a revolute-prismatic structure interacted with the target objects, allowing for the direct retrieval of force information at the contact point. PyBullet facilitated the direct extraction of mesh files from the soft balls. The points were projected through a vertical cross-section at the center to yield the contours of the balls.

C. Training and Testing Data Collection

Compression force was applied along the radial direction of the soft balls, varying the compression displacement from 0 to 30 mm in 5 mm increments. The steady-state force values and the contour deformation at the great circle were recorded and compared. The simulation explored a broader set of parameter combinations of λ and μ to gather force information. The same operations were performed on balls with unknown λ and μ values for the physical balls. Similar to the simulation data, steady-state force values and lateral contour deformation width were obtained. The critical distinction is that the physical balls lack the λ and μ parameters, the sought-after objectives for the testing data. The results from all simulations were combined to form a training dataset for machine learning. λ and μ parameters become the target variables for the testing data.

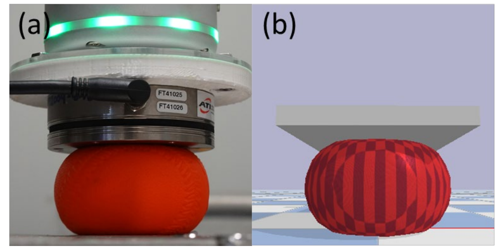

Fig. 4 The setup of the soft ball in compression: (a) in experiments and (b) in PyBullet simulation.

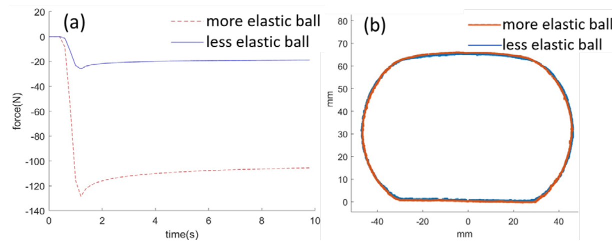

Fig. 5 The experiments of the compressed two physical soft balls under compression of 30 mm: (a) The time sequence of the balls. (b) The contour at of the balls.

D. The Performance of the Physical Balls

Before applying the fitting model to the physical balls, we initially observed differences in their force and contour data, as shown in Figure 5. At the same compression distance, the more elastic ball consistently exhibited forces approximately 5.5 times greater than the less elastic ball. However, regardless of the compression distance, their contour changes aligned perfectly. Upon examining the force-time curves as they approached a steady state, when the forces were proportionally scaled to match each other, the slopes of the force curves also appeared to align closely. This observation suggests that the damping values of the two balls were similar.

E. The Prediction of Static Parameters

The fitting model was trained using the above data, employing a neural network architecture within a machine learning framework. After attempts to change the architecture, we utilized two hidden layers with [32, 8] neurons and a Sigmoid activation function in each layer. The number of epochs is 5×10^3 for each learning. During the conceptualization phase, we assumed that there would be a continuous relationship between the two parameters and the performance of force and contour metrics, indicating that the neural network should have the potential for successful learning. However, two main challenges emerged during the training process. First, it was unexpected that the physical balls with such disparate elastic properties would exhibit perfect contour alignment, making it difficult to discern which parameter adjustments would enhance contour prediction intuitively. Second, the cases with lower force values encountered boundary issues in the simulation, resulting in data scarcity and making it challenging for the model to learn effectively. Consequently, the predictions for the less elastic ball were less accurate, while the more elastic ball yielded better results.

The optimal prediction for the Lamé parameters, obtained from the model trained using machine learning, yielded λ = 709.42 and μ = 611.32 for the more elastic ball. This parameter set was subsequently integrated into the Pybullet simulation environment to instantiate a virtual replica of the physical ball to conduct the experimental procedures and data collection mentioned in Part B. When subjected to a compression of 30 mm, the cross-sectional diameter value in the simulation was 94.807, 1.67% larger than the physical value, measuring 93.246 mm, as shown in Figure 6.

In terms of force, during the simulation's steady-state phase, the force was 4.75% lower than the physical value when the compression distance was 30 mm, measuring 113.428 N. A graphical representation of the relationship between steady-state force and compression level is shown as the orange line in Figure 7.

F. *The Prediction Incorporating Dynamic Performance Correction to Static Parameters*

To achieve a more comprehensive alignment between the simulated counterpart and the physical prototype in terms of performance, we addressed the previously fixed parameter NeoHookeanDamping, which has been maintained at its default value of 0.03. Setting the λ and μ results obtained from Section III.E as constants, the continuous force data for soft spheres subjected to compression under various NeoHookeanDamping values were collected.

The F-t diagram of Figure 8 presents the results. Given that NeoHookeanDamping affects the dynamic performance of objects, we compiled continuous force data into training data. We attempted to train the fitting model directly using the force at each time point as a machine-learning feature. However, as shown in Figure 8, numerical differences exist between the physical and simulated balls with all parameters. Still, the numerical change trend is similar to several of the groups. So, we divided each stroke force data by the steady-state value to standardize and differentiate. We processed the data into the change amplitude of the force at each time and used it as a machine learning feature to train the fitting model. The resulting predicted value is 0.0194. Compared with the initial default value of 0.03, the simulated ball behaves closer to a compressible but less elastic clay-based material in the simulation. The simulated soft ball under this condition shows performance closer to that of a foam ball, including free fall—the rebound height and frequency and the performance of oblique throws.

Then, after changing the fixed value of NeoHookeanDamping to 0.0194, we re-performed the same steps in Section III.E to use machine learning to generate a parameter prediction model. The best prediction of the Lamé parameter obtained is λ = 815.43 and μ = 675.97. the relationship between steady-state force and compression level is shown as the gray line in Figure 7. After correcting the dynamic performance, the minimum error dropped to 2.58% when the compression distance was 30 mm, measuring 116.359 N. The force values are also closer to the physical ball at other compression levels than the first prediction.

The whole process of parameters' prediction is shown in Figure 9. These attempts confirmed that dynamic behavior in Pybullet is closely related to damping and force trends. With more precise dynamic parameter settings, the prediction of static parameters can be more accurate. However, revising predictions back and forth in three steps is somewhat time-consuming. In the future, efforts will be made to mix static and dynamic data for training and find a model that can capture the connection between 3 parameters simultaneously.

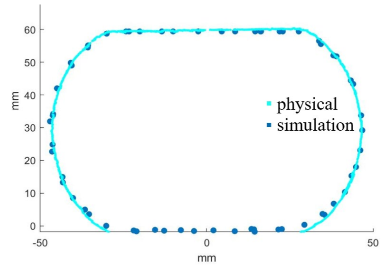

Fig. 6 The contours of the physical and simulated soft balls in PyBullet.

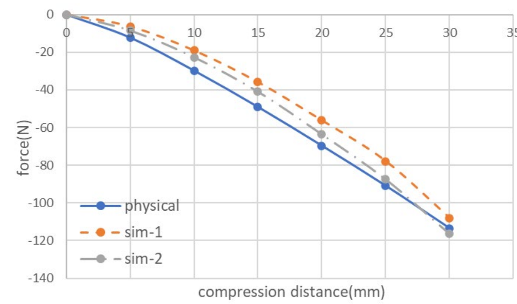

Fig. 7 The physical and simulated force's performances of the more elastic ball at steady states in different compression levels.
sim-1: (λ, μ) = (709.42, 611.32), sim-2: (λ, μ) = (815.43, 675.97)

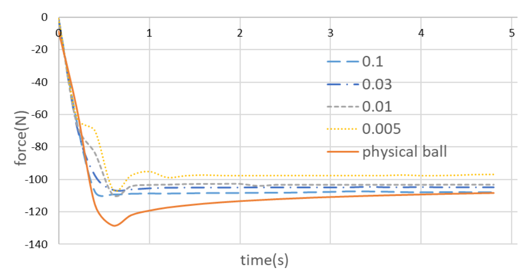

Fig. 8 The F-t diagram of the soft balls in compression. The orange line is the experimental data; the others are simulated data in different values of NeoHookeanDamping.

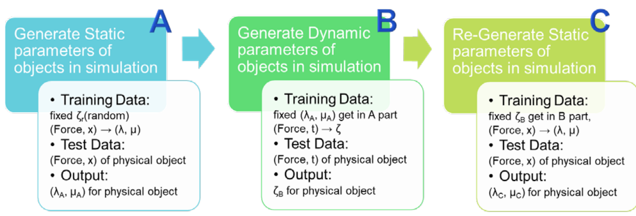

Fig. 9 The flow diagram of the parameters' prediction process.

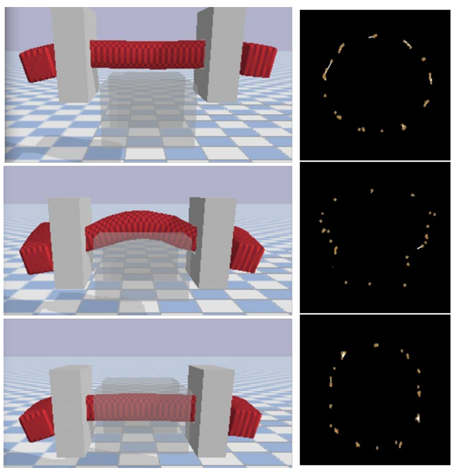

Fig. 10 The software inserting process (from top to bottom rows). The left column represents the simulation snapshots, and the left column plots a cross-section view of the wire in the middle section, where the deformation of the wire shape can be clearly seen.

IV. Application Scenario

By employing the method proposed in this study, a precise selection of parameters for the deformable object in the simulation leads to enhanced accuracy in simulating the deformable object, and the task environment of the simulation can be entirely established in the simulation engine. This section will take the situation where a soft wire is assembled into a slot as an example to demonstrate the functions related to deformable objects that Pybullet can support.

The parameters of the soft body in Pybullet, except neo-Hookean, are the same as those of the general rigid body. The mass and friction coefficient can be set. In addition to the current object's center of mass position, the state data of the soft body during motion can also be viewed in real-time, the contour point cloud data of the specified section as shown in Figure 10, and the force vector information of all contact points with other objects can be collected.

Based on the above functions, the changes of deformable objects when they move in the environment or interact with other objects can be fully observed in the simulation engine. It can also be extended to the reinforcement learning mentioned in section II for training on tasks related to deformable objects. In this way, the research field of manipulating deformable objects can be developed more efficiently and safely in a more complex and diverse direction.

V. Conclusion And Future Work

This work describes a methodology for investigating the parameters in the simulation of deformable objects and bridging the behavioral differences between simulations and empirical systems using a data-driven model. We used the best-predicted parameters of the model to demonstrate the simulation of a static deformable soft ball that exhibited a contour deviation of 1.67% compared to the real soft ball, with a forced error of 2.58% during steady-state conditions.

This framework can be extended to more actions involving higher complexity and multidirectional forces, as well as deformable objects with different materials and more intricate shapes, to validate the model's versatility and elasticity.

References

[1] Mahmood, A. R., Korenkevych, D., Komer, B. J., & Bergstra, J. (2018, October). Setting up a reinforcement learning task with a real-world robot. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 4635-4640). IEEE.

[2] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Openai gym. arXiv preprint arXiv:1606.01540.

[3] Yang, X., Ji, Z., Wu, J., & Lai, Y. K. (2021, September). An open-source multi-goal reinforcement learning environment for robotic manipulation with pybullet. In Annual Conference Towards Autonomous Robotic Systems (pp. 14-24). Cham: Springer International Publishing.

[4] Matas, J., James, S., & Davison, A. J. (2018, October). Sim-to-real reinforcement learning for deformable object manipulation. In Conference on Robot Learning (pp. 734-743). PMLR.

[5] Berenson, D. (2013, November). Manipulation of deformable objects without modeling and simulating deformation. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 4525-4532). IEEE.

[6] Laezza, R., Gieselmann, R., Pokorny, F. T., & Karayiannidis, Y. (2021, May). Reform: A robot learning sandbox for deformable linear object manipulation. In 2021 IEEE International Conference on Robotics and Automation (ICRA) (pp. 4717-4723). IEEE.

[7] Deng, Yuhong, et al. "Deep reinforcement learning based on local GNN for goal-conditioned deformable object rearranging." 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2022.

[8] Seita, D., Florence, P., Tompson, J., Coumans, E., Sindhwani, V., Goldberg, K., & Zeng, A. (2021, May). Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks. In 2021 IEEE International Conference on Robotics and Automation (ICRA) (pp. 4568-4575). IEEE.

[9] Lin, H., Guo, F., Wang, F., & Jia, Y. B. (2015). Picking up a soft 3D object by "feeling" the grip. The International Journal of Robotics Research, 34(11), 1361-1384.

[10] Hu, Z., Han, T., Sun, P., Pan, J., & Manocha, D. (2019). 3-D deformable object manipulation using deep neural networks. IEEE Robotics and Automation Letters, 4(4), 4255-4261.

[11] Coumans, E., & Bai, Y. (2016). Pybullet, a Python module for physics simulation for games, robotics, and machine learning.

[12] Weisstein, Eric. Eric Weisstein's. World of Science, A Wolfram Web Resource. https://scienceworld.wolfram.com/physics/LameConstants.html

[13] Pence, T. J., & Gou, K. (2015). On compressible versions of the incompressible neo-Hookean material. Mathematics and Mechanics of Solids, 20(2), 157–182.

[14] HU, Yixin, et al. Tetrahedral meshing in the wild. ACM Trans. Graph., 2018, 37.4: 60:1-60:14.