2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

SliceIt! - A Dual Simulator Framework for Learning Robot Food Slicing

Cristian C. Beltran-Hernandez[1,*], Nicolas Erbetti[1,*], Masashi Hamaya[1]

***Abstract*— Cooking robots can enhance the home experience by reducing the burden of daily chores. However, these robots must perform their tasks dexterously and safely in shared human environments, especially when handling dangerous tools such as kitchen knives. This study focuses on enabling a robot to autonomously and safely learn food-cutting tasks. More specifically, our goal is to enable a collaborative robot or industrial robot arm to perform food-slicing tasks by adapting to varying material properties using compliance control. Our approach involves using Reinforcement Learning (RL) to train a robot to compliantly manipulate a knife, by reducing the contact forces exerted by the food items and by the cutting board. However, training the robot in the real world can be inefficient, and dangerous, and result in a lot of food waste. Therefore, we proposed SliceIt!, a framework for safely and efficiently learning robot food-slicing tasks in simulation. Following a real2sim2real approach, our framework consists of collecting a few real food slicing data, calibrating our dual simulation environment (a high-fidelity cutting simulator and a robotic simulator), learning compliant control policies on the calibrated simulation environment, and finally, deploying the policies on the real robot.**

I. INTRODUCTION

Cooking robots, which can safely work alongside humans, hold the potential to enhance home environments and ease daily chores. Tasks such as food slicing require the robot to skillfully and safely manipulate a knife. Our research focuses on enabling a robot to learn food-cutting tasks.

Cutting skills such as chopping and slicing, require manipulating the knife and carefully responding to the reaction forces that are exerted by the material and by the cutting board [1]. In particular, it is important for the robot to be able to adapt to the widely varying physical properties of each food product [2]. Learning-based methods are promising approaches to autonomously learning complex robotic behaviors, such as the one required for food slicing. However, such methods often need a large number of interactions to learn useful control policies, which for the food-slicing tasks could result in a lot of food waste. Furthermore, it could be dangerous to train the robot directly in the real world when the robot is handling hazardous tools such as a kitchen knife, as part of its learning paradigm involves the random exploration of actions. Thus, learning in a simulation environment stands out as a viable solution, where exploratory actions can be conducted safely.

Learning in simulation has its own challenges, namely the reality gap [3]. The difference between the simulation environment and the real world can render the learned control policy unusable in the real world. Recently, advanced simulators like DiSECt [4] have been introduced, offering a highly realistic representation of soft material cutting. These simulators can potentially bridge the reality gap. However, to facilitate more accessible evaluation in real-world scenarios, creating an interface between the specialized cutting simulation and real robots is essential. The existing simulators focus on the physical interactions between the knife and the object but do not consider whether the robot can feasibly realize the knife motion.

To address this challenge, we propose a dual simulation environment combining the cutting simulator (CutSim) with a robotics simulator (RoboSim). The CutSim provides more realistic interaction forces to the RoboSim. Meanwhile, the RoboSim provides the knife motion generated by the simulated robot to the CutSim. Additionally, the RoboSim is more practical to use since the motions generated can be more eas-

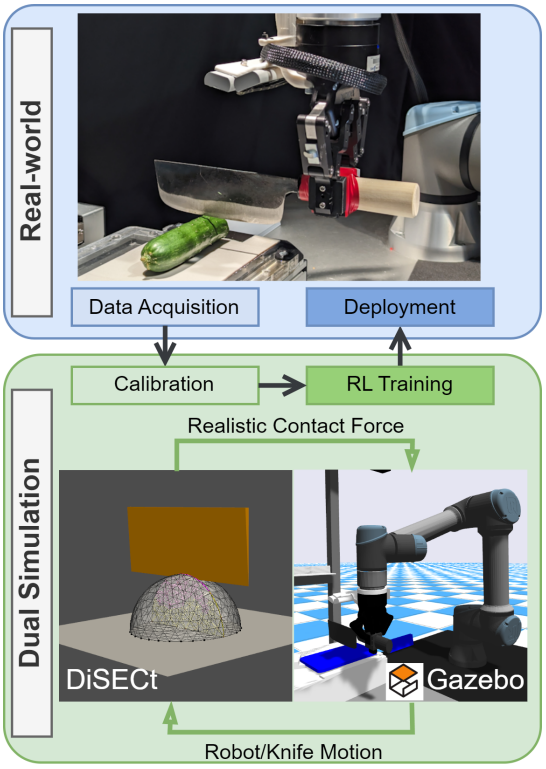

Fig. 1. Overview of proposed learning-based robot cutting framework, comprising four key stages: 1) Data collection on the real robot. 2) Calibration of the cutting simulator, DiSECt. 3) Learning a control policy within a dual simulation environment using Gazebo and DiSECt. 4) Deployment to the real robot.

* Equal contribution.

This work was supported by KAKENHI Grant Number 21H04910.

[1]OMRON SINIC X Corporation, Tokyo, Japan. Corresponding author cristian.beltran@sinicx.com

ily integrated into the real robot [5]. In this work, we present SliceIt!, a framework for safely learning robot cutting, that features a dual simulation environment. Our system follows a real-to-simulation-to-real (real2sim2real) [6] formulation and consists of (1) data collection from slicing real food items. (2) Calibration of the CutSim to simulate with high fidelity the collected slicing data. (3) Learning a control policy using our calibrated dual simulation environment. (4) Deployment of the policy on the real robotic system. An overview of our proposed method is depicted in Figure 1.

The main contribution of this work is SliceIt! - A framework for safely and autonomously learning robot food-slicing tasks in simulation. To develop this system, we introduce the following technical contributions:

- A dual simulation environment, that combines a high-fidelity cutting simulator and a robotic simulator, to enable better sim2real transfer of learning-based control policies. SliceIt! produced slicing motions with smaller contact forces than the baseline, where only RoboSim is used, thanks to the realistic simulation obtained from CutSim.
- A deep RL-based compliance control method for robotic cutting tasks, in particular for rigid robot manipulators. Our method adapts to unseen objects.

Our method was evaluated on a real robotic system and against a baseline that considers a simpler cutting simulation. Additionally, our project is open source[1] for the benefit of the research community.

This paper is organized as follows; related work is discussed in Section II, a detailed description of SliceIt! is provided in Section III, experiments and conclusion are provided in Section IV and V.

II. RELATED WORK

A. Slicing Robots

Prior works have formulated model-based control methods for robotic cutting based on analysis of stress and fracture forces of bio-materials, and blade sharpness and slicing angle [7], [8], as well as, considering cutting force models of the knife "pressing and slicing" [1]. Additionally, multimodal haptic sensory data has been used to enhance the cutting robotic system; in [9] a force/visual control approach is proposed while the use of tactile sensors was used to avoid slippage of the knife during cutting in [10]. To handle a wider diversity of food products, machine learning-based approaches have been proposed based on deep model predictive control [2], [11] and learning from demonstrations [12], [13], reinforcement learning [14]. However, these methods require several or many real-world interactions to collect data, which for food-slicing tasks, it may result in unnecessary food waste. Recent notable studies used sim-to-real transfer learning approaches. In RoboNinja [15], a multi-material cutting simulator is proposed to collect demonstrations to train a learning-based cutting method. DiSECt aimed at leveraging its differentiable simulation to optimize the motion of the knife. In their research, the optimization is focused on the parameters of a sinusoidal trajectory, the amplitude, downward velocity, and frequency of the slicing motion. In this study, we proposed a framework to handle more complex slicing motions by learning compliant control cutting motions using deep reinforcement learning in a real2sim2real formulation. Our method requires as few as one slice motion of each food item in the real world to obtain the force profile to optimize Cutsim. Besides, our trained policy adapts to unseen objects.

[1]Available at `https://github.com/omron-sinicx/sliceit`

B. Open-Source Software for Robot Manipulation

Recent studies offer open-source software for various robot manipulation, for example, grasping [16], learning [17], [18], [19], [20], assembly [21], cloth manipulation [22], and human-robot interaction [23]. We present an open-source software for robotic cutting that has not been explored in the existing studies. It provides an interface to integrate a dual simulation environment with real robots, making it practical to evaluate real-world scenarios. Given the hazards associated with cutting tasks, our approach offers the safety of training the robot in a simulated environment, and a collision-averse learning method based on [24], which learns adaptive compliant control policies to minimize high contact forces.

C. Multiple Simulator

Eaton et al [25] proposed using two simulators to compensate for the sim-to-real gap. One simulator was used first to train the policy, and then the policy was transferred to another simulator to evaluate its performance in a different environment. The successful policies in both simulators could be transferred to the real world. Unlike this study, which evaluated the robustness of the trained policy using different simulators, we propose a different use of the dual simulation to leverage the strengths of both simulators: the precise contact physics in CutSim (DiSECt) and RoboSim (Gazebo). In our study, we used DiSECt as our chosen cutting simulator. However, it's important to note that our method is not limited to this specific simulator. Our approach can be adapted to other simulators, like RoboNinja [15], to handle different complex cutting tasks.

III. METHODOLOGY

A. System Overview

We introduce a robot learning system designed for food-slicing tasks, employing a real2sim2real [6] approach. In Figure 1, we outline the core elements of our proposed method.

First, the "real2sim" phase involves data acquisition and calibration of our dual simulation environment. Our simulation environment consists of two concurrent simulators: a physics simulator tailored for cutting soft materials (CutSim), and a robotic simulator (RoboSim). We gather data by having the real-world robot slice food items. This data is then utilized to fine-tune the simulation parameters of the cutting simulator. Only a few data samples are required for the calibration.

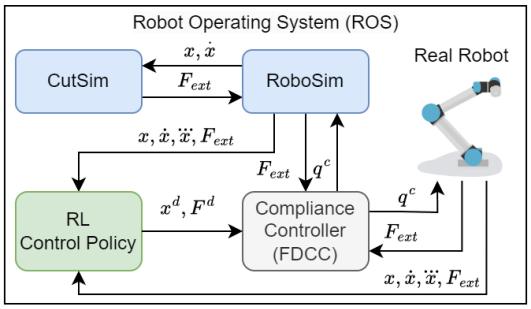

Fig. 2. ROS-powered proposed system for learning robotic cutting tasks using a dual simulation environment, reinforcement learning, and compliance control.

Second, the "sim2real" phase focuses on training a control policy using Reinforcement Learning (RL) within the simulation and deploying it in the real world. The combined cutting simulator and the robotic simulator are used for this purpose.

The following sections describe in detail the components of our method, the dual simulation environment, the calibration of our simulation environment, and the learning-based compliance control policy.

B. Dual Simulation Environment

1) CutSim: In this work, the DiSECt simulator was used. DiSECt is a differentiable physics simulator for cutting soft materials [4]. The simulator augments the finite element method (FEM) with a continuous contact model based on signed distance fields (SDF), as well as a continuous damage model that inserts springs on opposite sides of the cutting plane and allows them to weaken until zero stiffness to model crack formation. DiSECt was chosen because it allows us to realistically simulate food cutting by calibrating its simulation parameters. The differentiability of the simulator enables the calibration of the simulation parameters using gradient-based optimization methods [4].

Calibration: The calibration process involves simulating the robot's cutting actions, including motion and contact force, and adjusting the simulation parameters until the force profile matches the desired one. The simulator's differentiability allows us to fine-tune these parameters using gradient-based optimization methods [4]. However, gradient-based optimization can be computationally intensive, and inappropriate initial parameters sometimes cause learning instability. Therefore, in this study, we propose a two-step approach. Initially, a non-gradient-based optimization method is used to quickly and cost-effectively identify an initial set of simulation parameters. Subsequently, a gradient-based optimization method, specifically the Adam algorithm [26], is employed to further refine these simulation parameters. To optimize the initial simulation parameters, we utilize the Tree-structured Parzen Estimator algorithm [27] as implemented in Optuna [28].

2) RoboSim: We use the Gazebo simulator, which is an open-source 3D robotics simulator [5]. Our choice of the Gazebo simulator was motivated by its compatibility with the Robot Operating System (ROS) [29]. ROS is an open-source robotics middleware that facilitates the integration of different robotic components. In this work, ROS was used to integrate both simulators, the real robots, our proposed RL control policy, and its low-level compliance controller as depicted in Figure 2.

3) Simulators' Bridge: Our decision to employ two simulators is driven by the high computational cost of CutSim. To obtain better results from CutSim, the simulation needs to run at a much higher frequency. In practice, a time step of $dt = 1.0e^{-}5$ gave us the best results. Meanwhile, RoboSim can run at a lower frequency, thus requiring much less computational cost. A time step of $dt = 1.0e^{-}4$ was used for RoboSim. However, RoboSim cannot consider the interaction forces in the cutting plane, it can only consider the normal contact force against the surface of objects.

In our proposed framework, both simulators run in parallel with an interleaving simulation of a set time duration. In practice, we use the highest control frequency of the Universal Robots UR5e, which translates to a duration of $2.0e^{-3}$ seconds. The simulators exchange information through ROS messages. RoboSim provides the position x and velocity $\dot{x}$ of the knife with respect to the robot while CutSim provides the contact force F_{ext} for the robot compliance controller, as depicted in Figure 2.

C. Learning slicing with reinforcement learning

1) Markov Decision Process: In RL, agents learn a desired behavior through interaction. The slicing tasks can be formulated as an episodic Markov Decision Process (MDP) that has finite time steps with a limit of T steps per episode. MDP can be denoted as a tuple $(S, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A}$ is a transition probability function, $\mathcal{R}$ is a reward function, and $\gamma \in (0, 1)$ is a discount factor. At each time step t, the agent observes the current state $\mathbf{s}_t \in \mathcal{S}$ and takes an action space $\mathbf{a}_t \in A$ according to a policy π. The environment changes to the state $\mathbf{s}_{t+1}$ according to the transition function $p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$. Then, the agent receives a numerical reward $\mathbf{r}_t = R(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ for taking action $\mathbf{a}_t$ at the state $\mathbf{s}_t$ and landing to state $\mathbf{s}_{t+1}$. The goal is to find a policy π^* that maximizes the expected sum of discounted future rewards given by $R(t) = \sum_i^T \gamma \mathbf{r}_t$ [30].

We used the RL algorithm soft actor-critic (SAC). SAC [31] is an off-policy actor-critic deep RL algorithm based on maximal entropy. SAC aims to maximize the expected reward while also optimizing maximal entropy, H. The agent optimizes a maximal entropy objective, encouraging exploration according to a temperature parameter α.

$$\pi^* = arg\max_{\pi} E_{\pi}\left[\sum_{t=0}^{T} \gamma^t \Big(\mathcal{R}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) + \alpha H(\pi(\cdot|\mathbf{s}_t)\Big)\right]$$

The core idea of this method is to succeed at the task while acting as randomly as possible. SAC is an off-policy algorithm that uses a replay buffer to reuse information

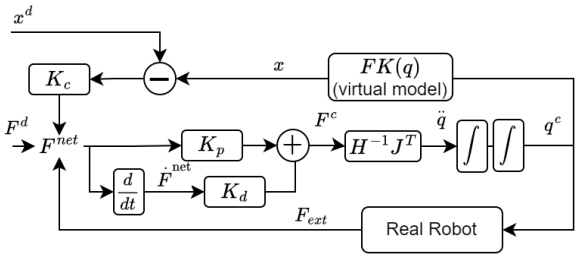

Fig. 3. Forward Dynamics Compliance Controller [33]

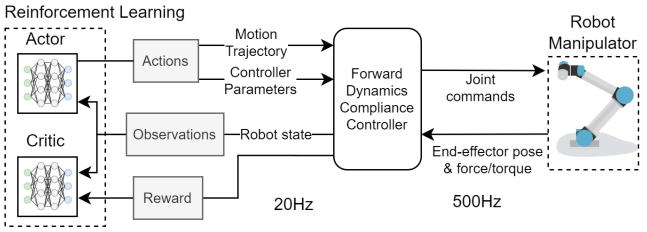

Fig. 4. Framework for learning food slicing using RL and compliance control. The RL agent controls provides the reference trajectory and control parameters for the compliance controller.

from recent rollouts for sample-efficient training. Thus, SAC can be enhanced with the prioritized experience replay [32] approach for further improvement.

2) Compliance control: For the contact-rich task of food slicing with a rigid robotic arm, the Forward Dynamics Compliance Control (FDCC) method [33] was used. FDCC combines three control principles: Impedance Control, Admittance Control, and Force Control into one new strategy to realize Cartesian compliance control. As a key component in FDCC, forward dynamics simulations of a virtual model are leveraged to directly map Cartesian inputs to joint control commands, leading to good stability in singularities. Our proposed system uses the open-source implementation [2] of FDCC for ROS.

The controller is described in Figure 3, where x represents the Cartesian position of the robot's end-effector and the joint positions. F_{ext} is the measured contact force. x^d and F^d are the desired pose and wrench of the robot's end-effector, and F^c is the commanded wrench. The virtual model's forward dynamics and forward kinematics are $H^{-1}J^T$ and $FK(q)$, respectively. q and $\ddot{q}$ correspond to the position and acceleration of the robot joints. The net force F^{net} encompasses the target forces, the forces measured by the sensor, and all the forces that result from virtual motion. The controller is parameterized by the stiffness K^c, and the PD gains K^p and K^d.

3) RL agent: Traditionally, fine-tuning a compliance controller for a given task is a time-consuming process that requires human expertise. To reduce these requirements, we use RL to learn the motion of the cutting action as well as the control parameters of the FDCC, inspired by previous work [34]. Compared to [34], we use FDCC because it provides better stability in singularities and requires fewer parameters to learn. As described in Figure 4, the actions of the RL agent provide the reference trajectory x^c and the control parameters $[K^c, K^p, K^d]$ to the FDCC at a low control frequency. Then the FDCC directly controls the robot with joint commands at the highest control frequency available. The feedback from the robot is the knife pose, computed using forward kinematics from the robot's joint positions, and the sensed force and momentum. The agent's observations consist of the knife's position relative to a target position, its velocity and jerkiness, as well as the previous action taken and the history of n force-torque readings from the sensor.

The SAC's actor-network architecture consists of a Temporal Convolutional Network (TCN) [35], that processes a history of n force/torque readings from the sensor, and a fully connected network that processes the remaining observations. Both networks output 64 features that are concatenated and processed together on an additional fully connected network, a simplified version of this structure is shown in Figure 4. The choice of network architecture is based on results from previous work [24].

4) Reward Function: The reward function is defined based on the height of the knife with respect to the cutting board x_{cut}, the contact force F_{ext}, and the jerkiness of the knife $\dddot{x}$. Additionally, we defined reward values for conditions that terminate the episode, completing the task (the knife reaches the desired target pose within some threshold), colliding with the cutting board (exceeding a maximum contact force), moving outside of the defined workspace, and taking too long to complete the task.

$$r = w_1 \tanh(|x_{cut}|) - w_2/(1+e^{||F_{ext}||}) - w_3||\dddot{x}|| + \mathfrak{P} \quad (1)$$

where $\mathfrak{P}$ is

$$\mathfrak{P} = \begin{cases} 100, & \text{Task completed} \\ -100, & \text{Collision} \\ -1, & \text{Otherwise} \end{cases}$$

IV. Experiments And Results

The following experiments were designed to answer these questions:

- How well does our system perform in the real world when it has only been trained in a simulated environment?
- When it comes to the robotic cutting task, does using a more realistic simulation environment lead to better performance compared to using a less detailed one?

For the latter, we evaluate the performance of our proposed system against a baseline. This baseline involves calibrating the simulation environment and training a policy exclusively using the Gazebo Simulator [5].

A. Experimental Setup

1) Robotic Platform: Our robotic platform is based on the dual-arm system initially introduced in [36]. It comprises two

[2]FZI Cartesian Controllers:
github.com/fzi-forschungszentrum-informatik/cartesian_controllers

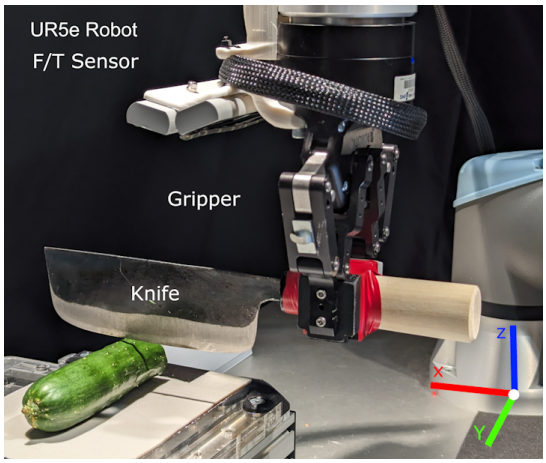

Fig. 5. Experimental setup with the real robot.

TABLE I

CALIBRATION PARAMETERS FOR THE DISECT SIMULATOR.

Parameter	Range
Cut Springs' stiffness	[100, 8000]
Cut Springs' softness	[10, 5000]
Contact stiffness	[200, 8000]
Contact damping	[0.1, 100]
Contact friction stiffness	[0.001, 8000]
Contact friction coefficient	[0.45, 1.0]

TABLE II

REAL-WORLD EXPERIMENTAL CONDITIONS

Food Item	# of Slices	Slice size (mm)
Cucumber	15	5
Tomato	5	5
Potato	10	3
Carrot	5	5

Universal Robot UR5e robotic arms, each equipped with a parallel gripper and a force-torque sensor positioned at the arm's end. In the context of this research, one of these robots serves a supporting role, such as holding the vegetable, while the second robot is responsible for executing the cutting task. The experimental setup is depicted in Figure 5. Notably, the robot engaged in the task is equipped with a Robotiq 85 parallel gripper featuring a custom finger adapter that enables the attachment and detachment of a kitchen knife.

2) Calibration: In this study, we gathered data from three distinct food items, each characterized by different material properties, to ensure diversity within our training dataset. Specifically, we selected a tomato, a cucumber, and a potato, representing items with low, medium, and high stiffness, respectively. The collected data consists of the knife's motion and the contact force applied during the cutting action. For simplicity, we maintained a constant knife speed throughout the experiments. The simulation parameters of the Disect simulator considered during the calibration process are detailed in Table I.

In the baseline case, denoted as *Gazebo only*, the cutting action was simulated using a compression spring. In the simulator, the compression spring is defined as a prismatic joint where the force constant is determined by specific simulation parameters, namely, the Error Reduction Parameter (ERP) and Constraint Force Mixing (CFM). These parameters were calibrated so that the force required to compress the spring to its maximum matches the maximum force observed in the real-world force profile, as depicted in Figure 6.

3) Cutting Task: In this study, the robot cutting task is defined as a single slice of the food item, which can then be executed multiple times as necessary. The goal of the robot is to maximize speed and minimize contact force and motion jerkiness. In particular, the task includes minimizing the force exerted on the cutting board with the knife. Four food items were used for validation: a cucumber, a tomato, a potato, and a carrot. As mentioned above, the first three were used for calibrating the simulation environments, while the carrot was used to evaluate the generalization capabilities of our method.

After calibration of both our method and the baseline using all of the food items, an RL agent was trained in each simulation environment for 80K time steps. The training included domain randomization by loading one of the calibrated food items into the CutSim as well as injecting a small uniformly sampled noise into the simulation parameters. A similar noise was injected into the baseline. The learning converges at around 60K time steps.

B. Real World Experiments

The evaluation of the learned control policies involved slicing each food item multiple times. To minimize food waste, only one unit of each food item was utilized in these experiments. The specific experimental conditions are presented in Table II.

Figure 7 and 8 illustrate the obtained results. In Figure 7, we compare the contact force observed during the slicing actions for each food item between our method and the baseline. These results include the contact force exerted not only on the food item but also on the cutting board. Notably, our method consistently outperforms the baseline by applying significantly lower force during the execution of slicing actions across all tasks. Remarkably, even in the case of the carrot, which was not part of the training dataset, our proposed method demonstrated superior performance when compared to the baseline.

Figure 8 centers on the experiments related to cucumber slicing. In this visualization, the grey region corresponds to the slicing of the vegetable, while the yellow region represents the contact force applied to the cutting board. These findings suggest that the policy acquired through our proposed method achieves superior performance by exhibiting a more adept response to the abrupt stiffness transition between the vegetable and the cutting board. Similar results are observed across all tasks. These results indicate that the policy learned with our proposed method performed better by more skillfully reacting to the sudden change of stiffness, between the vegetable and the cutting board. Figure 8 centers on the experiments related to cucumber

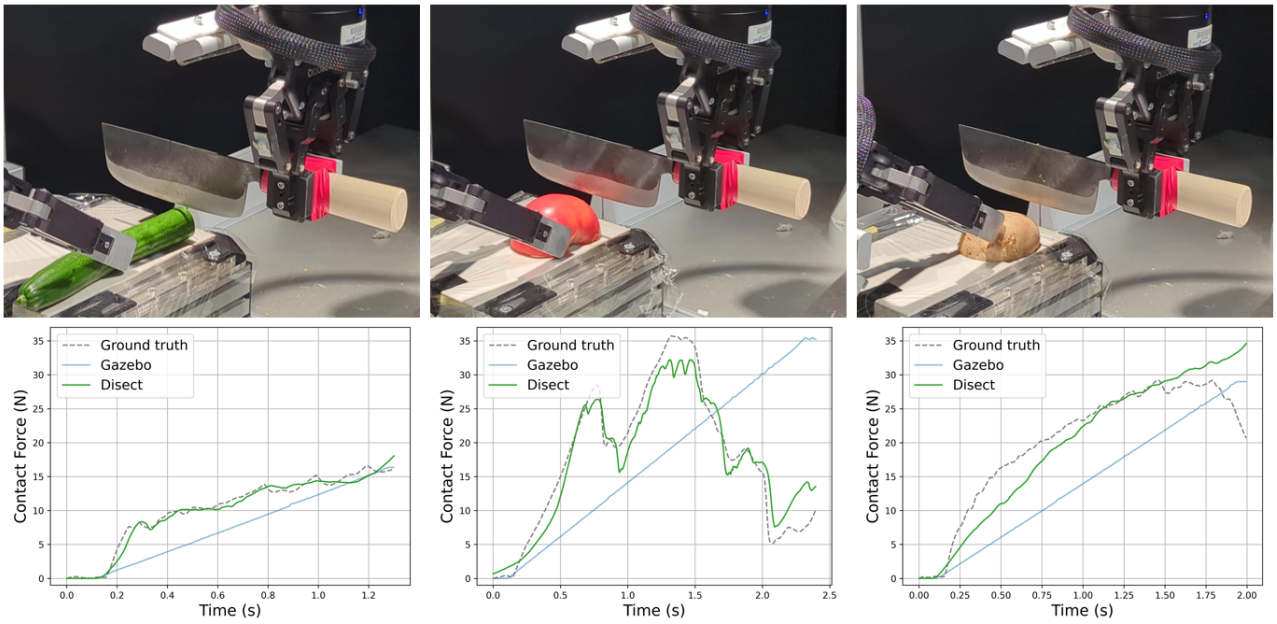

Fig. 6. Force profile of the simulated cutting after the calibration process for both simulators.

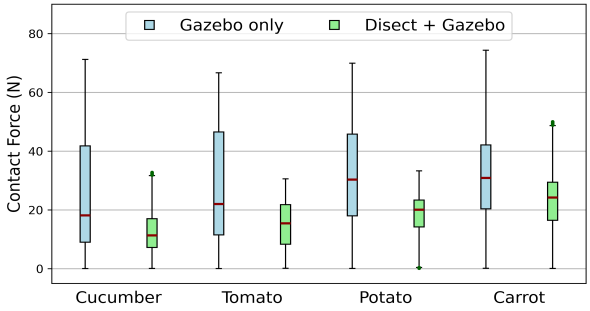

Fig. 7. Evaluation on the real robot: Slicing contact force for four vegetables, cucumber, tomato, potato, and carrot.

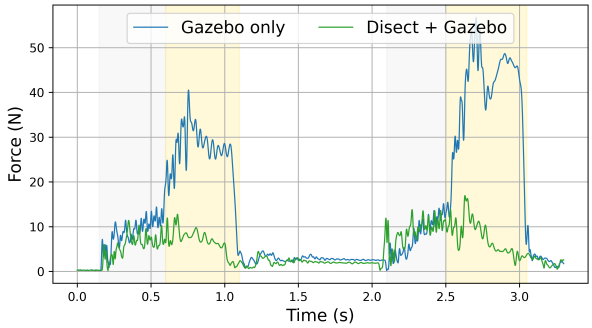

Fig. 8. Evaluation on the real robot: Normal contact force at the knife while slicing the cucumber twice. The gray region corresponds to slicing the vegetable, while the yellow region shows the contact with the cutting board.

slicing. In this visualization, the grey region corresponds to the slicing of the vegetable, while the yellow region represents the contact force applied to the cutting board. The results show clearly that our proposed method performs better than the baseline by more skillfully reacting to the change of stiffness between the food item and the cutting board. These results are consistent across all trials and tasks.

V. Conclusion

In this study, we introduced SliceIt! a learning-based robotic system for handling food-cutting tasks. Our system combines two key components: a dual simulation environment, and a control policy based on Reinforcement Learning. The aim is to enable a collaborative robot (cobot) or industrial robot arms to perform these tasks safely and accurately by adapting to varying conditions using compliance control. One of the advantages of using our proposed method is the reduction of food waste while learning the control policies, as only a few real-world samples are required. The experimental results support our hypothesis that using a highly realistic simulation environment is beneficial to learning safer control policies.

One limitation of our approach is the increased computation time when using a highly realistic cutting simulator compared to a simplified one. In our experiments, training the RL policy using our method took approximately 40 hours in total, whereas using Gazebo alone required only about 4 hours. This discrepancy arises from the more demanding computational requirements for each simulation time step in DiSECt. However, the additional computation time proves to be worthwhile, given the significant improvement in real robot performance achieved. A promising area for future research involves finding ways to reduce the extensive computational time.

REFERENCES

[1] X. Mu, Y. Xue, and Y.-B. Jia, "Robotic cutting: Mechanics and control of knife motion," in *IEEE International Conference on Robotics and Automation*, 2019, pp. 3066–3072.

[2] I. Lenz, R. A. Knepper, and A. Saxena, "Deepmpc: Learning deep latent features for model predictive control." in *Robotics: Science and Systems*, vol. 10, 2015, p. 25.

[3] N. Jakobi, P. Husbands, and I. Harvey, "Noise and the reality gap: The use of simulation in evolutionary robotics," in *Advances in Artificial Life: Third European Conference on Artificial Life*, 1995, pp. 704–720.

[4] E. Heiden, M. Macklin, Y. Narang, D. Fox, A. Garg, and F. Ramos, "Disect: a differentiable simulator for parameter inference and control in robotic cutting," *Autonomous Robots*, pp. 1–30, 2023.

[5] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2004, pp. 2149–2154.

[6] P. Chang and T. Padif, "Sim2real2sim: Bridging the gap between simulation and real-world in flexible object manipulation," in *IEEE International Conference on Robotic Computing*, 2020, pp. 56–62.

[7] D. Zhou, M. Claffee, K.-M. Lee, and G. Mcmurray, "Cutting, 'by pressing and slicing', applied to the robotic cut of bio-materials, part ii: Force during slicing and pressing cuts," in *IEEE International Conference on Robotics and Automation*, 2006, pp. 2256 – 2261.

[8] D. Zhou and G. McMurray, "Slicing cuts on food materials using robotic-controlled razor blade," *Modelling and Simulation in Engineering*, vol. 2011, pp. 36–36, 2011.

[9] P. Long, W. Khalil, and P. Martinet, "Robotic cutting of soft materials using force control & image moments," in *International Conference on Control Automation Robotics & Vision*, 2014, pp. 474–479.

[10] A. Yamaguchi and C. G. Atkeson, "Combining finger vision and optical tactile sensing: Reducing and handling errors while cutting vegetables," in *IEEE-RAS International Conference on Humanoid Robots*, 2016, pp. 1045–1051.

[11] I. Mitsioni, Y. Karayiannidis, J. A. Stork, and D. Kragic, "Data-driven model predictive control for the contact-rich task of food cutting," in *IEEE-RAS International Conference on Humanoid Robots*, 2019, pp. 244–250.

[12] C. Yang, C. Zeng, C. Fang, W. He, and Z. Li, "A dmps-based framework for robot learning and generalization of humanlike variable impedance skills," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 3, pp. 1193–1203, 2018.

[13] E. Anarossi, H. Tahara, N. Komeno, and T. Matsubara, "Deep segmented dmp networks for learning discontinuous motions," *arXiv preprint arXiv:2309.00320*, 2023.

[14] A. Padalkar, M. Nieuwenhuisen, S. Schneider, and D. Schulz, "Learning to close the gap: Combining task frame formalism and reinforcement learning for compliant vegetable cutting," in *International Conference on Informatics in Control, Automation and Robotics*, 2020, pp. 221–231.

[15] Z. Xu, Z. Xian, X. Lin, C. Chi, Z. Huang, C. Gan, and S. Song, "Roboninja: Learning an adaptive cutting policy for multi-material objects," *arXiv preprint arXiv:2302.11553*, 2023.

[16] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set," *IEEE Robotics & Automation Magazine*, vol. 22, no. 3, pp. 36–52, 2015.

[17] M. Wołczyk, M. Zając, R. Pascanu, Ł. Kuciński, and P. Miłoś, "Continual world: A robotic benchmark for continual reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28 496–28 510, 2021.

[18] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[19] B. Delhaisse, L. Rozo, and D. G. Caldwell, "Pyrobolearn: A python framework for robot learning practitioners," in *Conference on Robot Learning*, 2020, pp. 1348–1358.

[20] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu, "robosuite: A modular simulation framework and benchmark for robot learning," *arXiv preprint arXiv:2009.12293*, 2020.

[21] J. Collins, M. Robson, J. Yamada, M. Sridharan, K. Janik, and I. Posner, "RAMP: A benchmark for evaluating robotic assembly manipulation and planning," *arXiv preprint arXiv:2305.09644*, 2023.

[22] A. B. Clark, L. Cramphorn-Neal, M. Rachowiecki, and A. Gregg-Smith, "Household clothing set and benchmarks for characterising end-effector cloth manipulation," in *IEEE International Conference on Robotics and Automation*, 2023, pp. 9211–9217.

[23] C. Mower, T. Stouraitis, J. Moura, C. Rauch, L. Yan, N. Z. Behabadi, M. Gienger, T. Vercauteren, C. Bergeles, and S. Vijayakumar, "ROS-Pybullet interface: A framework for reliable contact simulation and human-robot interaction," in *Conference on Robot Learning*, 2023, pp. 1411–1423.

[24] C. C. Beltran-Hernandez, D. Petit, I. G. Ramirez-Alpizar, and K. Harada, "Variable compliance control for robotic peg-in-hole assembly: A deep-reinforcement-learning approach," *Applied Sciences*, vol. 10, no. 19, p. 6923, 2020.

[25] M. Eaton, "Bridging the reality gap a dual simulator approach to the evolution of whole-body motion for the nao humanoid robot," in *International Joint Conference on Computational Intelligence*, 2016, pp. 186–192.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[27] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in Neural Information Processing Systems*, vol. 24, 2011.

[28] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019, pp. 2623–2631.

[29] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng *et al.*, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, 2009, p. 5.

[30] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 2nd ed. MIT press, 2018.

[31] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*, 2018, pp. 1861–1870.

[32] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[33] S. Scherzinger, A. Roennau, and R. Dillmann, "Forward dynamics compliance control (FDCC): A new approach to cartesian compliance for robotic manipulators," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 4568–4575.

[34] C. C. Beltran-Hernandez, D. Petit, I. G. Ramirez-Alpizar, T. Nishi, S. Kikuchi, T. Matsubara, and K. Harada, "Learning force control for contact-rich manipulation tasks with rigid position-controlled robots," *IEEE Robotics and Automation Letters*, pp. 1–1, 2020.

[35] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv:1803.01271*, 2018.

[36] F. von Drigalski, C. C. Beltran-Hernandez, C. Nakashima, Z. Hu, S. Akizuki, T. Ueshiba, M. Hashimoto, K. Kasaura, Y. Domae, W. Wan *et al.*, "Team o2ac at the world robot summit 2020: towards jigless, high-precision assembly," *Advanced Robotics*, vol. 36, no. 22, pp. 1213–1227, 2022.