Robotics: Science and Systems 2023
Daegu, Republic of Korea, July 10-July 14, 2023

RoboNinja: Learning an Adaptive Cutting Policy for Multi-Material Objects

Zhenjia Xu[1], Zhou Xian[2], Xingyu Lin[3], Cheng Chi[1], Zhiao Huang[4], Chuang Gan[5†], Shuran Song[1†]
[1] Columbia University [2] CMU [3] UC Berkeley [4] UC San Diego [5] UMass Amherst & MIT-IBM AI Lab
https://roboninja.cs.columbia.edu/

***Abstract*—We introduce RoboNinja, a learning-based cutting system for multi-material objects (i.e., soft objects with rigid cores such as avocados or mangos). In contrast to prior works using open-loop cutting actions to cut through single-material objects (e.g., slicing a cucumber), RoboNinja aims to remove the soft part of an object while preserving the rigid core, thereby maximizing the yield. To achieve this, our system closes the perception-action loop by utilizing an interactive state estimator and an adaptive cutting policy. The system first employs sparse collision information to iteratively estimate the position and geometry of an object's core and then generates closed-loop cutting actions based on the estimated state and a tolerance value. The "adaptiveness" of the policy is achieved through the tolerance value, which modulates the policy's conservativeness when encountering collisions, maintaining an adaptive safety distance from the estimated core. Learning such cutting skills directly on a real-world robot is challenging. Yet, existing simulators are limited in simulating multi-material objects or computing the energy consumption during the cutting process. To address this issue, we develop a differentiable cutting simulator that supports multi-material coupling and allows for the generation of optimized trajectories as demonstrations for policy learning. Furthermore, by using a low-cost force sensor to capture collision feedback, we were able to successfully deploy the learned model in real-world scenarios, including objects with diverse core geometries and soft materials.**

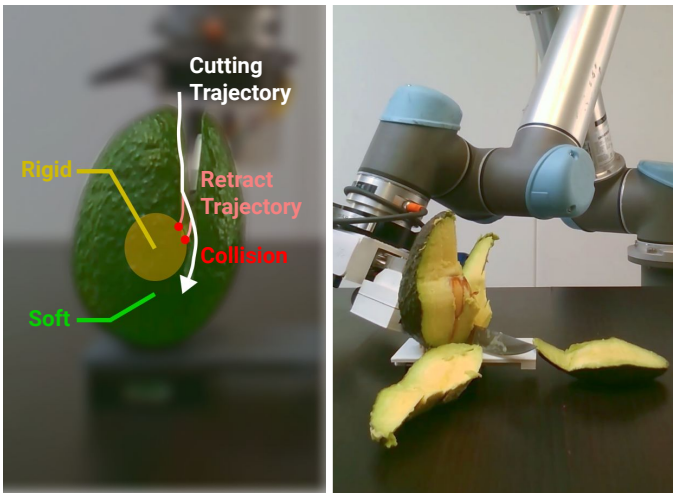

Fig. 1: **RoboNinja** is designed to cut multi-material objects with an interactive state estimator and adaptive cutting policy. Left: When the knife encounters a collision with the invisible core, the algorithm updates the core estimation and re-plans the cutting trajectory after a few retracting actions. Right: We deploy the learned model on a physical robot, allowing it to cut fruits in a way that maximizes the cut-off mass while minimizing collision occurrences.

I. Introduction

Imagine slicing a piece of avocado from its seed (Fig. 1) – we need to carefully slice through the soft outer flesh to locate the rigid seed and then follow the contours of the seed to maximize the volume of the slice. In some cases, we would need to switch the cutting trajectory when the knife collides with the seed. All of these maneuvers must be performed while adhering to the physical constraints of the knife and its interactions with both the soft and rigid parts of the avocado.

This simple yet intricate task illustrates the challenges of cutting multi-material objects, which is significantly more difficult than cutting through single-material objects that often could be accomplished with an open-loop cutting trajectory [13, 28, 46, 34, 60, 61]. In this paper, we are interested in enabling robots to effectively and efficiently perform this task. Using the above example, we could summarize the unique capabilities required for acquiring such a skill:

- **Multi-objective optimization under complex physical constraints.** To perform the task, the system needs to simultaneously optimize several objectives – maximizing the total yield (cut-off mass of the soft material), avoiding collisions with the rigid core, and minimizing energy consumption. Many of these objectives require comprehensive physical reasoning beyond simple geometry analysis.
- **Interactive state estimation for extreme partial observability.** In most cases, the rigid core is not observable on the surface. Hence, it requires the system to continuously estimate the location and geometry of the core through interaction, that is, through cutting and sensing the collision. This state estimator will continuously inform the cutting policy in a close-loop manner.
- **Adaptive policy for out-of-distribution scenarios.** While the state estimator could infer the core geometry based on contacts, there will always be instances where the shape of the core falls outside the training distribution. An inaccurate estimation could lead to repetitive collisions in the same location. In these scenarios, the cutting policy needs to "adaptively" update its cutting strategies to avoid getting stuck.

As the first step towards enabling this new robot capability,

† indicates equal advising.

we introduce **RoboNinja**, a learning-based cutting system that combines an interactive state estimator and an adaptive cutting policy. The interactive state estimator uses sparse contact information to iteratively estimate the position and shape of the core. The cutting policy, optimized to increase cut-off yield and reduce collision occurrences and energy consumption, produces cutting actions in a closed-loop manner, based on the estimated state and a tolerance value. The tolerance value is a function of past collision events and actively controls the policy conservativeness when encountering new collisions (e.g., keeping a distance from the estimated core location). This adaptivity is critical for handling out-of-distribution scenarios where the state estimation could be inaccurate.

Learning such cutting skills directly on a real-world robot system is challenging and potentially dangerous. However, existing simulators in the literature are limited in simulating multi-material objects, especially the coupling between rigid and soft bodies under forceful manipulations such as cutting. Therefore, we develop a new differentiable simulator for the proposed multi-material object cutting problem, allowing us to use gradient-based optimization for generating trajectories as demonstrations for policy learning.

Finally, when deploying the learned policy, we demonstrate that with the simple collision feedback captured by a low-cost (less than $10) force sensor, we can successfully transfer the model learned in the simulator directly to real-world scenarios, including out-of-distribution object geometries and materials, thanks to its adaptivity.

In summary, the primary contribution of this paper is RoboNinja – the first robotic system demonstrating the capability of multi-material object cutting. To build this system, we make the following technical advancements:

- The formulation of the multi-material cutting task and a differentiable simulator that could perform multi-objective trajectory optimization for collecting demonstrations.
- A learning-based cutting method with an *interactive* state estimator and an *adaptive* cutting policy.
- The deployment of our method on a real-world robotic system with low-cost sensory feedback.

Videos of experiments, the code for the simulator and the cutting system, as well as the CAD models for the benchmark objects are available at https://roboninja.cs.columbia.edu/.

II. Related Work

A. Differentiable Physics Simulation for Policy Learning

In recent years, a number of differentiable simulation environments have been proposed to accelerate policy learning. These include 1) simulators parameterized by neural networks [24, 23, 39], which haven't proved to be capable of accurate simulations involving complex interactions between multi-phase materials required in cutting scenarios, and 2) analytical simulators implemented in a differentiable way, leveraging either automatic differentiation tools [15] or analytical gradient computation rules [16]. The latter is used to provide gradient information to accelerate policy search in various robotic tasks, including locomotion [55], soft robot design [50], soft body manipulation [17, 54, 40], etc. To handle realistic sensory input outside of the simulation environment, prior methods distill the knowledge learned in simulators into visuomotor policies that take images [25] or point clouds [26] as input. In contrast, the state of multi-material objects in our task, such as the shape of the rigid core inside, is not directly observable. As such, our method learns an interactive state estimation network based on encountered collision events to address the challenge.

B. Simulation Environments for Cutting

There has been a significant amount of research conducted on simulating the cutting process of different materials. Theoretical analysis is commonly applied in metal cutting [32, 6] and brittle materials research [11, 33, 62]. Besides, various numerical methods are utilized to simulate the fracture of deformable objects. They could be further classified into mesh-based methods, such as the Finite Element Method (FEM) [13, 2, 22, 53, 38, 18], and mesh-free methods, including Position-based Dynamics (PBD) [36, 3] and Material Point Method (MPM) [14, 49, 51, 52]. The work most related to ours is "DiSECt" by Heiden et al. [13], where the FEM-based cutting simulator achieves differentiability through continuous contact formulation and damage modeling. In contrast to these prior works that only simulate the cutting process of a single material, our differentiable simulator stands out by accounting for both the external soft material and the internal rigid core, utilizing MPM for its ability to accurately and efficiently simulate the dynamics of elastoplastic objects and the coupling between different materials.

C. Interactive Perception

Interactive perception (IP) [5] leverages physical interaction to gather information about the environment. It is commonly used for scene reconstruction with occluded objects [58, 20, 43, 42] and kinematic object structure discovery [19, 10, 35, 30]. The integration of tactile signals becomes increasingly popular in this field, especially in material classification [8, 7], object recognition [27, 57, 56, 44, 45], and shape reconstruction [1, 4, 31, 29]. Recently, Xu et al. [56] recognize 3D objects with active tactile explorations. Wang et al. [29] present a curiosity-driven object reconstruction method using the modality of touch. These prior works only consider rigid or articulated objects. Our work advances the field of interactive perception by applying it to a new and challenging task: multi-material object cutting.

D. Robotic Cutting

Many robotic systems have been developed for cutting tasks in various domains such as meat [28], vegetables [46, 34, 60, 61, 41], and dough [25, 26]. Researchers also utilize multimodal haptic sensory data to enhance system robustness [59]. As an alternative to traditional cutting using knives, hot wire cutting tools [9, 47] are adopted in various sculpting and industrial applications. Previous works mostly deal with only single-material objects and study how to cut through them, where open-loop orthogonal cutting is typically sufficient. In

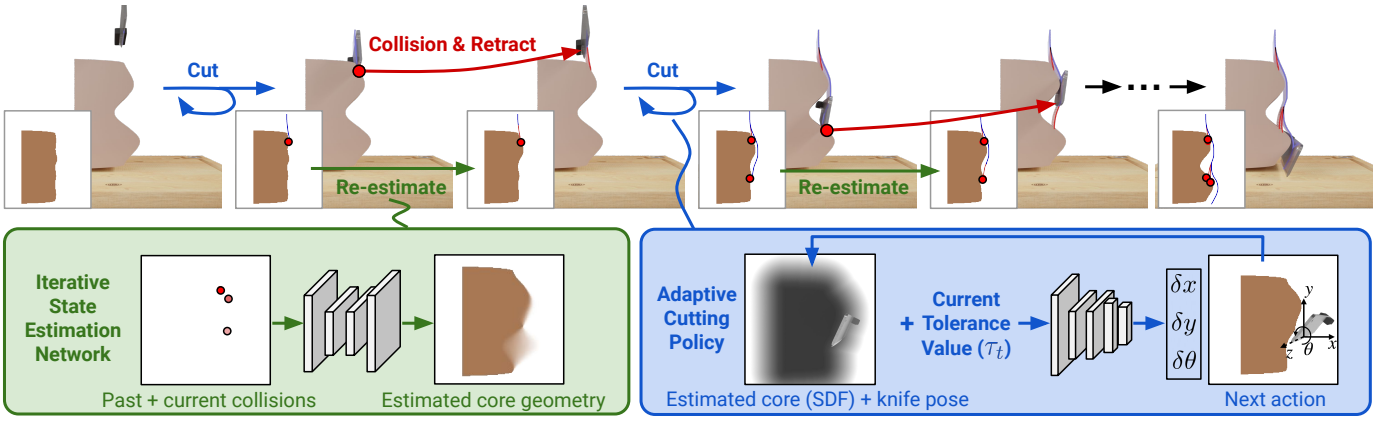

Fig. 2: **RoboNinja Overview.** The robot initially performs actions based on an initial estimation. In the event of a collision with the rigid core, the robot retracts a few steps (indicated in red). The robot then re-estimates the core's position and geometry (indicated in green) and generates the cutting action using the updated state estimation (indicated in blue).

this work, we study the problem of cutting multi-material objects with the goal of removing soft material from an invisible rigid core, which requires controlling under physical constraints such as avoiding collisions with the rigid core, and continuous state estimation from sensory feedback in a partially observable environment. Therefore, we propose to equip our system with an interactive state estimation network and an adaptive policy to address the task.

III. Method

In this work, we study a multi-material object cutting task, where the goal is to manipulate a tool to cut off soft material from a rigid core while maximizing the yield (the total amount of soft material being removed), as well as minimizing the number of collisions with the core and the total energy consumption.

Considering the complex objectives and the real-time nature of this task, we decide to employ the standard teacher-student framework: slow while accurate expert demonstrations act as the teacher (optimized with the differentiable simulator), and a lightweight learning-based policy as a student is trained to imitate the teacher's behavior and is deployed at inference time. Specifically, we first build a physics-based differentiable simulator supporting multi-material coupling to gather expert demonstrations via gradient-based trajectory optimization. Next, we train an interactive state estimation network to infer the position and the geometry of the core based on collected collision signals. Afterward, a cutting policy is trained to generate actions based on the core estimation and the knife state. This policy not only imitates the behavior of the expert demonstrations but also adaptively adjusts the conservativeness to retract from the core after the collision. An overview of the execution is illustrated in Fig. 2. The robot first starts the cutting process by executing actions based on an initial estimation from a learned prior. Upon collisions with the core, the state estimation is immediately updated. To avoid high energy consumption due to in-place rotation within fiber-rich flesh, the robot retracts a few steps and then replans the cutting trajectory using the updated state estimation. In this process, the system progressively updates the estimated state, leading to an accurate estimate of the position and geometry of the invisible core after multiple collision events, which in turn allows a physically plausible cutting trajectory to cut off most of the soft material.

In the following sections, we first present our expert demonstration process in §III-A, where we develop our differentiable cutting simulator and perform gradient-based trajectory optimization. We then detail the iterative state estimation in §III-B and the adaptive cutting policy in §III-C. Finally, we present a hardware setup that includes a low-cost force sensor for deploying our policy in real-world scenarios (§III-D).

A. Multi-objective Trajectory Optimization with a Differentiable Simulation Environment

Differentiable cutting simulation environment We build a cutting simulation environment to support the modeling of both soft and rigid materials, as well as the coupling between them. The soft material in our scene, e.g. flesh of fruit is represented using an elastoplastic continuum model simulated with MLS-MPM [14], treated by the von Mises yield criterion. In contrast to FEM [13], MPM-based methods naturally support arbitrary deformation and topology change. For rigid bodies in the scene, including the rigid core, the knife, and the support surface (a chopping board), we represent them as time-varying signed distance fields (SDFs), converted from imported external meshes. We model the contact between soft and rigid materials by computing surface normals of the SDFs and applying Coulomb friction [48]. Material separation occurring during the cutting process is handled by MLS-MPM, which inherently supports modeling sharp and clean split of material points in the soft material. The simulator is implemented with Taichi, a domain-specific language that supports auto differentiation. We implemented our computation process by building an explicit computation graph and keeping track of all the intermediate variables over time, which allows end-to-end gradient flow from the loss computation back to the action input.

Multi-objective trajectory optimization At each step, we

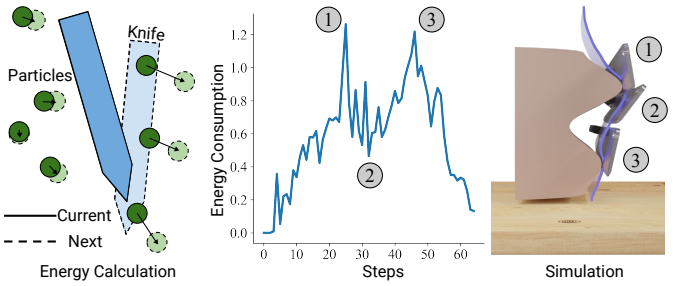

Fig. 3: **Energy Calculation.** [Left] Illustration of energy computation in simulation. Particle position and velocity are changed due to collision with the knife. The cumulative work from the knife to each collided particle is considered as the energy consumption of the agent. [Middle] A plot of energy consumption at each step. [Right] Visualization of knife poses at some representative steps. Note that large energy consumption incurs around the rapid rotations in the trajectory. (e.g., steps 1 and 3).

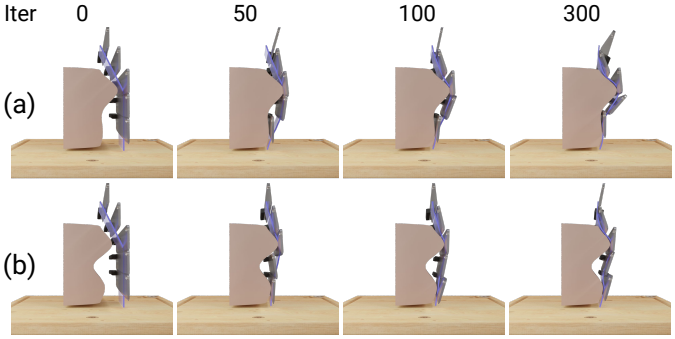

Fig. 4: **Optimization Process of Cutting Trajectory.** As illustrated in the first column, the cutting trajectory is initialized with a pre-designed collision-free path. After 300 optimization iterations, the knife is able to cut off most of the soft materials with optimized energy consumption.

consider a cutting action parameterized by the knife orientation θ along the z axis and a vertical displacement $\delta\vec{p} = (\delta x, \delta y)$ within the $x-y$ plane (see Fig. 2). To further ensure the smoothness of the cutting trajectory, we impose an additional constraint on the magnitude of the displacement: $\|\delta\vec{p}\| = 2mm$. Since our differentiable simulator allows gradient-based trajectory optimization, we collect cutting trajectories in the simulator assuming access to all ground truth information, including the mesh of the rigid core and the position and velocity of each particle representing the soft material. Trajectories are optimized using the following objectives:

- *Cut mass*: one primary goal of a cutting policy is to maximize the total amount of the soft material being removed by a cutting trajectory. This objective $\mathcal{L}_m$ is computed by accumulating the mass of all material points removed from the rigid core at the end of each episode.
- *Collision occurrences*: an ideal cutting trajectory should be able to avoid unnecessary collision events during its course. Our simulator represents both the core and the knife by time-varying SDFs. The collision between them is detected by sampling N uniform points on the knife surface and checking whether they penetrate the core. In order to make this optimization process differentiable, we model the discontinuous contact between the knife and the core in a soft manner, following [17, 54], and compute a differentiable collision loss for optimization: $\mathcal{L}_{col} = \sum_{i=1}^{N} \|\max(d_i + \hat{d}, 0)\|^k$, where d_i represents the penetration distance of the i-th sampled point, and $\hat{d}$ is an additional safety margin. In practice, we found $N = 5$, $k = 4$, and $\hat{d} = 2cm$ to be sufficient to produce good trajectories.
- *Energy consumption*: in order to produce a natural and smooth motion trajectory during a cutting process, our system also optimizes energy consumption. We estimate the energy loss $\mathcal{L}_e$ based on the work done by the knife during its motion at each step, which is computed by summing the product of the distance traveled and force experienced by each material point in contact with the knife: $\mathcal{L}_e = \Sigma_j \frac{m_j \Delta\vec{v}_j}{\Delta t} \cdot \Delta\vec{p}_j$, where $\Delta\vec{v}_j$ and $\Delta\vec{p}_j$ denote the change in velocity and position for each particle j, respectively. In Fig. 3, we illustrate the energy consumption of each step in an example cutting episode.

During trajectory optimization, we initialize using a pre-designed and translational collision-free trajectory that traverses down from above the object till touching the support surface, and optimize the trajectory loss by summing the above objectives: $\mathcal{L}_{total} = \mathcal{L}_m + \eta_{col}\mathcal{L}_{col} + \eta_e\mathcal{L}_e$, where η_{col} and η_e are coefficients for balancing different objectives. Examples of the optimization process are shown in Fig. 4.

B. Interactive State Estimation

The purpose of the state estimation module is to determine the location and shape of the initially invisible core using collision signals collected during the cutting process. As illustrated in Fig. 2, the input is a sparse collision map within the $x-y$ plane, where each filled pixel represents a collision point encountered during the trajectory traveled until the current time step. We employ an 11-layer U-Net architecture to estimate the 2D mask of the rigid core.

To facilitate offline training, we randomly select k points on the contour of the training cores to mimic collision signals that might be received during actual execution. Here k is an integer uniformly sampled from a uniform distribution with a range [0, 9]. The model is trained using the ground truth geometry as supervision and optimized with Binary Cross-Entropy loss. During the inference phase, a pre-determined threshold S_{thr} is used to convert the predicted probability map to a binary core mask.

C. Adaptive Cutting Policy

The goal of the cutting policy is to generate the cutting action at each step. The policy network takes the signed distance field of the estimated core mask, the current knife pose, and an additional tolerance value τ_t as input. It predicts the cutting action a_t, parameterized by the translational $\delta\vec{p} = (\delta x, \delta y)$ and rotational movement $\delta\theta$ of the knife. Tolerance value τ_t controls the level of the conservativeness of the action. A lower tolerance value results in the knife

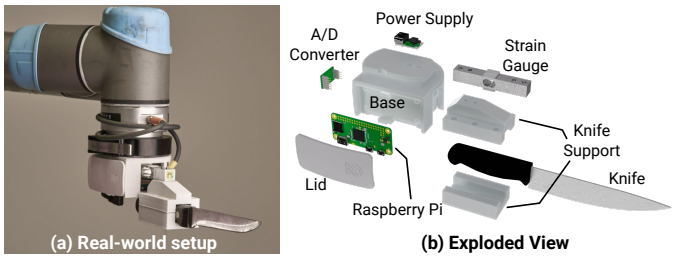

Fig. 5: **Hardware.** We design and construct a compact cutting tool equipped with a force sensor. The force is measured by a strain gauge as an analog electrical signal. The signal is then converted to a digital signal and transmitted to the robot controller through an A/D converter and a Raspberry Pi Zero, respectively.

getting closer to the estimated core, thus increasing the risk of collision. Conversely, a higher tolerance value leads to a more conservative action, keeping the knife farther from the core to prevent potential collisions. We initialize the tolerance at each step as 0. Upon collisions, the robot first retracts the knife for R_{dis} steps based on the history of actions, then executes actions with an increased tolerance value by τ^+ for R_{dis} step, and then linearly decay the tolerance τ afterwards to produce a smooth trajectory.

We use the demonstrations $\{D\}$ collected in the differentiable simulator to train our cutting policy. The tolerance value of the actions in the demonstration is assumed to be 0, and we use data augmentation to generate training data for tolerance values τ larger than 0 in the following way: For each demonstration, $D_i = [s_0, a_0, s_1, a_1, \cdots, s_N, a_N, s_{N+1}]$, where s_i and a_i represents the knife pose and action at each step, we generate another knife trajectory $\hat{S} = [\hat{s}_0, \hat{s}_1, \cdots, \hat{s}_N, \hat{s}_{N+1}]$ by moving the current knife trajectory away from the core in the direction of the x-axis by τ. To increase training robustness, a gaussian noise is added to the action sequence. Finally, the quadruple $(\tau, s_0^*, a_i, \hat{s}_{i+1})$, along with the core geometry is used for training, where s_0^* is calculated analytically using a_i and $\hat{s}_{i+1}$. The cutting policy is trained with Mean Squared Error (MSE) loss.

D. Real-world System Setup

We design and build a low-cost, force feedback system for deploying our method on a real-world UR5 platform. Figure 5 shows an image and the exploded view of our hardware system. A strain gauge load cell measures the shear force experienced by the connected knife as an analog signal. This signal is amplified and converted into digital readings by an HX711 A/D converter at 80 Hz. The digital measurement of the shear force is then processed by the Raspberry Pi Zero and transmitted to the robot controller via Wi-Fi. The entire system is powered by UR5's tool I/O power, and all these components are compactly assembled inside a 3D-printed container. The load cell and the AD converter are the core components of our hardware system to achieve real-time force feedback and are priced at $8 in total[1], significantly cheaper than a force-torque sensor. In practice, it is noteworthy that the cutting friction of different soft materials varies greatly; thus, we manually set the threshold for each material when converting the continuous force signal into a binary collision signal.

[1]Amazon link: https://www.amazon.com/gp/product/B08KRV8VYP

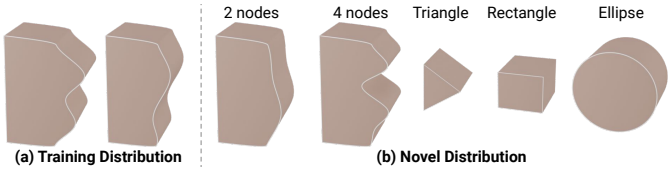

Fig. 6: **Generated cores for training and evaluation**

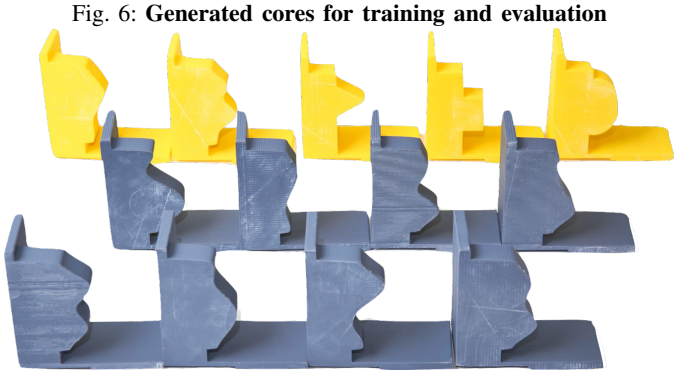

Fig. 7: **3D printed cores in real-world evaluation**. 8 in-distribution geometries (gray) and 5 out-of-distribution geometries (yellow).

IV. Evaluation

In this section, we first evaluate the cutting performance of our proposed method through comparison with various baselines and variants in simulated environments. We further validate our approach by conducting experiments on a real-world setup. Additionally, we conduct ablation studies to evaluate the contribution of each component of our system to its overall performance.

A. Policy Evaluation in Simulation

Dataset and Experimental Setup We generate 300 and 100 multi-material objects for training and evaluation, respectively. Each object comprises a rigid core surrounded by soft material. The soft material is simulated as elastoplastic material using the following parameters: $\lambda = 1388.89Pa$, $\mu = 2083.33Pa$, $\sigma = 200Pa$, $\rho = 10^3 kg/m^3$, where λ and μ are Lamé parameters, σ the yield stress, and ρ the density. The contour of the cores used in training is parameterized by a cubic spline with 3 equally spaced nodes, with 3 degrees of freedom in total. The horizontal coordinate for each node is sampled from a uniform distribution with range $[-0.035m, 0.035m]$. We subsequently concatenate a fixed back contour and extrude this 2D polygon into a 3D mesh. The 100 cores used in the evaluation are divided into two sets: 50 cores with the same distribution as the training cores and 50 out-of-distribution cores, which consist of 5 categories with 10 cores each: (1) 2 nodes, (2) 4 nodes, (3) triangle, (4) rectangle, (5) ellipse. Examples of the training and testing cores are shown in Fig. 6.

Implementation Details For trajectory optimization in the demonstration collection process described in Sec. III-A, we use $\eta_{col} = 2e4$ and $\eta_e = 0.15$. We generate one expert trajectory for each core in the training set, where optimizing

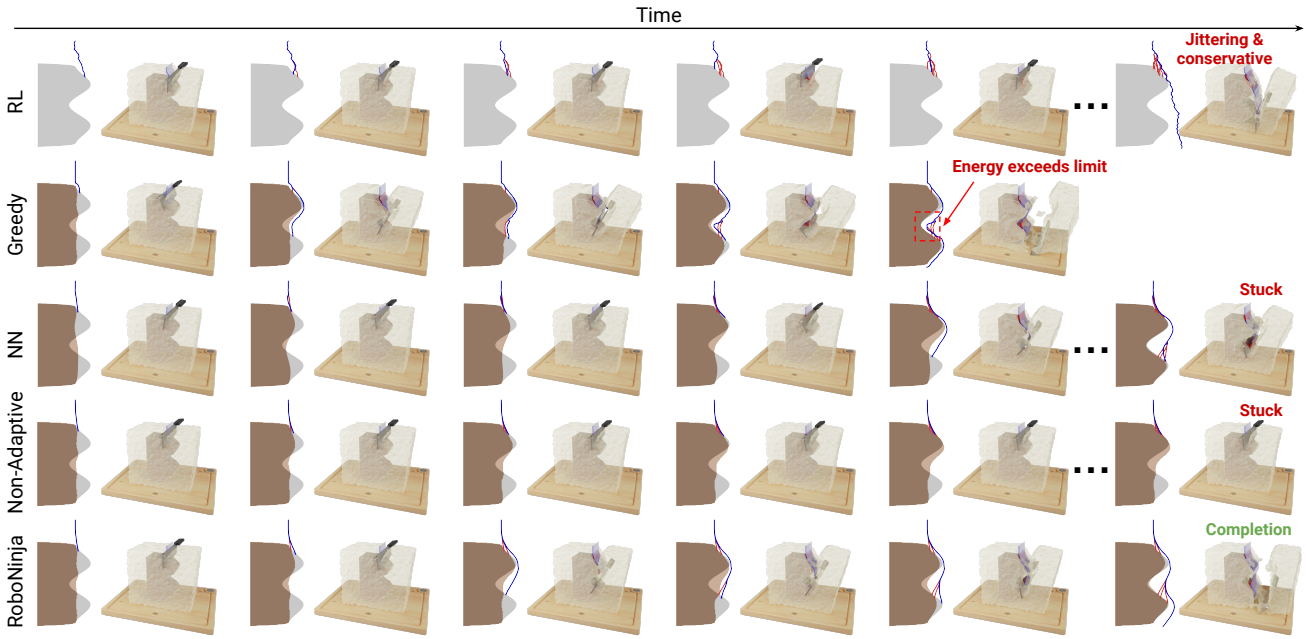

Fig. 8: **Evaluation on in-distribution geometries.** Each column shows an iteration (continuous execution until a collision). In the 2D view (left), the ground truth geometry is shown in gray, and the estimation is shown in brown. In both views, the forward trajectory of the knife is demonstrated in blue, and the retraction trajectories due to collision are visualized in red. The cutting trajectory of [RL] is very jittering and becomes too conservative after a few collisions. [Greedy] strictly follows the contour of the estimated geometry, leading to exceeding energy consumption during abrupt rotations. Both [NN] and [Non-Adaptive] can't complete the cutting task within 10 collisions. [RoboNinja] is able to iteratively update the estimate of the core after each collision and adaptively adjust the cutting trajectory with optimized energy consumption.

	In-distribution Geometries					Out-of-distribution Geometries				
	Completion↑	Cut Mass↑	Collision↓	Avg Eng↓	Max Eng↓	Completion↑	Cut Mass↑	Collision↓	Avg Eng↓	Max Eng↓
RL	0.600	0.540	0.030	0.380	1.33	0.540	0.534	0.038	0.326	1.270
Greedy	0.560	0.803	0.029	0.625	2.632	0.280	0.724	0.033	0.630	2.621
NN	0.400	0.585	0.045	**0.242**	1.106	0.300	0.472	0.048	0.233	1.072
Non-Adaptive	0.460	0.595	0.044	0.226	**0.857**	0.400	0.529	0.048	**0.224**	**0.945**
RoboNinja	**1.000**	**0.875**	**0.028**	0.357	0.862	**0.880**	**0.799**	0.033	0.342	0.988

TABLE I: **Cutting performance on in-distribution and novel geometries.**

each trajectory takes 300 gradient-based updates using the Adam optimizer [21] with a learning rate of $1e-2$. Examples of the optimization process are shown in Fig. 4. The state estimation network has an input and output size of 256×256. The classification threshold (S_{thr}) is set to 0.3. The training data is generated by randomly sampling 0 to 9 collision points. In the adaptive cutting policy, the number of retraction steps after a collision (R_{dis}) is set to 8, and the tolerance increment (τ^{+}) is set to 0.005. To achieve a smooth trajectory, the tolerance value is linearly decayed to 0 within 5 steps after increasing. Both networks are implemented in PyTorch [37] and trained using the Adam optimizer with a learning rate of $1e-4$ and a weight decay of $1e-6$. A comprehensive ablation study of some critical parameters is presented in Sec. IV-C.

Metrics. We use the following metrics to evaluate the cutting performance:

- Completion Rate. To measure the completion rate of each cutting task, we consider an execution as "completed" if the knife reaches the chopping board and as "failed" if either the number of collisions exceeds 10 or the energy loss value for any single step exceeds 3.0, as defined in Sec III-A. An episode is terminated as soon as it's considered "failed", and all subsequent metrics are evaluated considering only the action sequence executed before the termination.
- Cut Mass Ratio. This metric measures the ratio of the removed mass to the total mass of the soft material originally attached to the rigid core. In case of failed execution, the cut mass only considers the soft material on the right side of the cutting trajectory executed till termination.
- Collision Ratio. To account for variations in trajectory length, we normalize the number of collisions by the length of each trajectory.
- Avg / Max Energy. We also evaluate the energy consumption averaged over all steps, as well as the maximum energy consumption incurred at a single step during the whole trajectory.

Baselines. We compare our proposed system to the following alternative approaches:

- RL: A model-free reinforcement learning policy operating

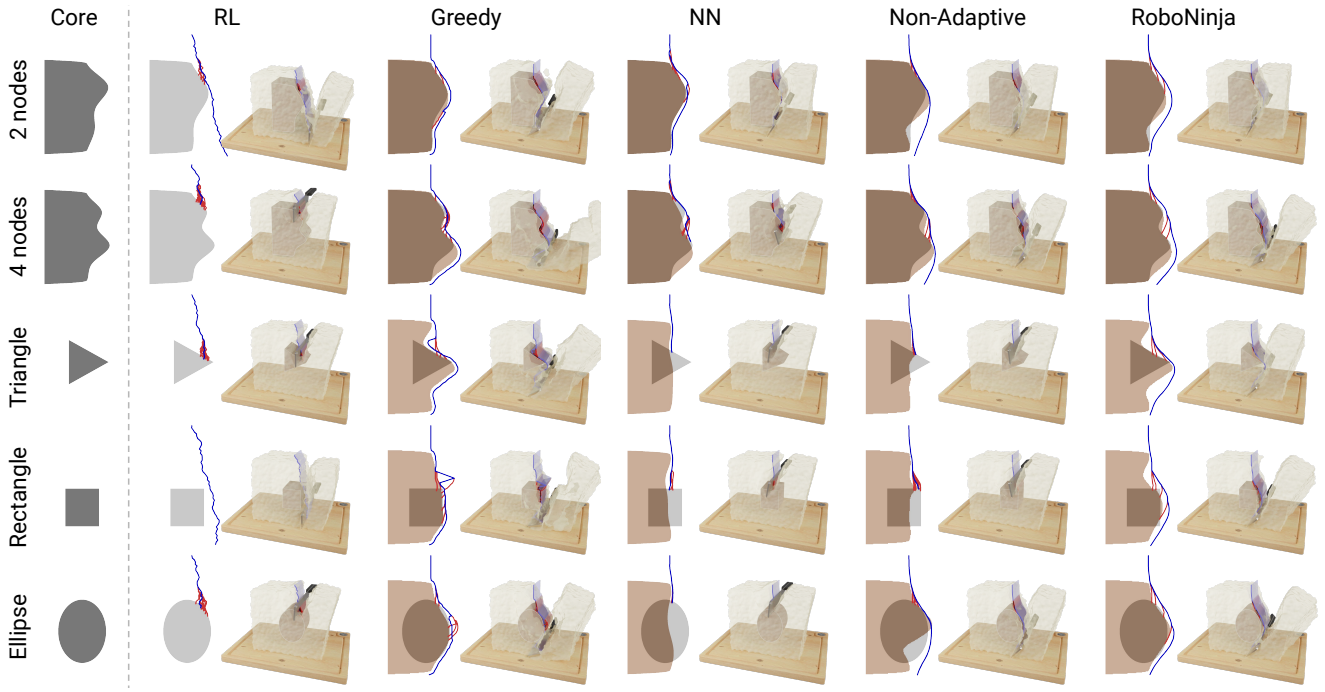

Fig. 9: **Evaluation on out-of-distribution geometries.** The cores sampled from each category are shown in the first column, followed by trajectories produced by each method. Both [NN] and [Non-Adaptive] may get stuck, especially when the geometries (e.g., Triangle and Rectangle) are significantly different from the training set. Both [Greedy] and [RoboNinja] have a strong generalization ability to handle various novel geometries. However, [RoboNinja] consumes much less energy than [Greedy], leading to smoother and more natural cutting trajectories.

within the same observation and action space, as well as using the same collision-retraction mechanism as our method. This policy is trained using Soft Actor-Critic (SAC) [12], using a dense reward function given by $\mathcal{R} = \mathcal{L}_m^t - \mathcal{L}_m^{t-1} - \eta(\mathcal{L}_e^t - \mathcal{L}_e^{t-1})$, where $\mathcal{L}_m^t$ and $\mathcal{L}_e^t$ are the cumulative cut mass and energy consumption till step t, respectively. Upon collision, the cumulative cut mass will decrease due to automatic retraction, resulting in a negative reward, which naturally functions as a collision penalty. The goal of the RL algorithm is to maximize the cumulative reward, which is equivalent to maximizing the total cut mass and meanwhile minimizing the total energy consumption in our setting. Through a grid search over the energy weight η, we found the best value to be $\eta = 0.05$. We train the policy with 3 random seeds and report the best performance.

- NN. This approach uses a nearest neighbor cutting policy in place of our adaptive cutting policy. At each step, it retrieves the most similar core from the training set based on the current state estimation result, and executes the closest action step selected from the associated demonstration trajectory based on the current knife pose.
- Greedy. This approach adopts a heuristic policy instead of our adaptive cutting policy. It determines the movement direction and knife rotation at each step in a greedy manner with the aim of maximizing the cut mass while avoiding collision with the estimated core. The primary difference between this method and ours is that it does not factor in energy consumption.
- Non-Adaptive. This is a non-adaptive variant of our system which uses a fixed tolerance value of 0.

B. Results and Analysis

Qualitative results are summarized in Fig. 8 and Fig. 9. Quantitative evaluations are reported in Tab. I.

Comparison to model-free RL. The cutting task is successfully completed by [RL] in over 50% of the cases. However, the resulting cut mass is significantly inferior compared to that of [RoboNinja]. As demonstrated in Fig. 8 and 9, the conservative cutting trajectories of [RL] maintain a significant distance from the bottom part of the core. In contrast, [RoboNinja] leverages explicit core estimation to follow the contour of the core, resulting in a significantly higher cut mass. Moreover, the cutting trajectories of [RL] display a noticeable level of jitter, deviating from ideal human-like behavior. Additionally, the knife could get stuck occasionally due to the lack of an explicit adaptive mechanism.

Comparison to Greedy. Compared to [Greedy], [RoboNinja] with a learning-based cutting policy achieves around +44% improvement in terms of completion rate. While [Greedy] is able to strictly follow the contour of the estimated geometry and maximize the cut mass, the policy does not consider energy consumption. This may result in physically implausible actions, such as abrupt knife rotations. In contrast, our cutting policy, which imitates trajectories optimized with the energy consumption objective, is able to sacrifice a small amount of cut mass ratio in exchange for much less energy

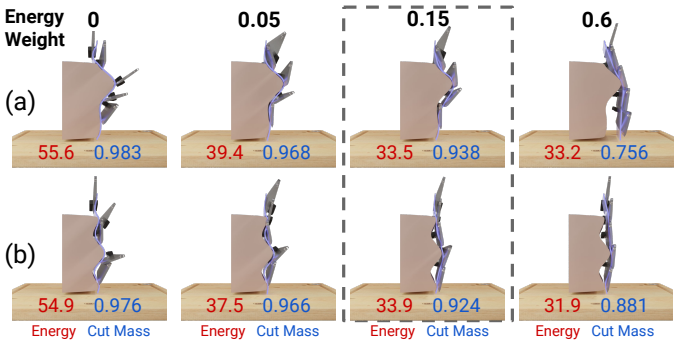

Fig. 10: **Trajectories optimized with different weights of energy loss.** Cumulative energy consumption and the cut mass ratio are shown in red and blue, respectively. Without any energy penalty, the trajectory strictly follows the contour of the core and cuts off most of the soft material. However, the knife has to rotate rapidly to avoid collision with the core, which results in large energy consumption. In contrast, a large energy penalty leads to a conservative policy where the rotation of the knife is less noticeable. An appropriate energy weight achieves a balance between energy consumption and cut mass. The one used in our final system [0.15] is able to cut off over 90% with similar energy consumption as [0.6].

Energy Weight (η_e)	0	0.05	0.15	0.6
Cut Mass Ratio ↑	0.977	0.955	0.923	0.854
Energy Consumption ↓	53.08	36.20	32.96	30.79

TABLE II: **Effects of different energy weights**

consumption. This is evident in the energy consumption values in Tab. II, where the average and maximum energy consumption of [RoboNinja] is 40% and 60% less than [Greedy].

Comparison to NN. [NN] directly leverages the action from the demonstration, which reduces the chance of exceeding the energy limit. However, the performance of [NN] relies heavily on the retrieved nearest neighbor trajectory, making it susceptible to any potential errors in the state estimation result. As shown in Fig. 8, in [NN]'s last step, a slight mismatch between the actual core and the retrieved one results in a collision. Afterward, if the state estimation doesn't have enough change after the collision, the policy will be stuck since the retrieved nearest neighbor remains the same. In contrast, [RoboNinja] is able to adapt the cutting policy to be more conservative when the state estimation is inaccurate and thereby avoiding being stuck due to collision.

Effect of the adaptive cutting policy. As shown in Fig. 8, [Non-Adaptive] behaves similarly to [NN] and suffers from getting stuck at the same location. Thanks to the adaptability of [RoboNinja], the policy successfully bypasses the peaks of the core with a higher tolerance, which improves the completion rate by over +54%.

Generalization to novel geometries. Despite being trained on a single type of core geometry, our policy is able to handle cores with novel geometries thanks to its adaptive cutting policy. Triangles and rectangles are particularly challenging, as they contain straight edges and sharp corners not present in the training data. The qualitative comparison in Fig. 9 shows that other baselines may fail due to exceeding energy consumption ([Greedy]) or getting stuck at one collision location ([RL], [NN], and [Non-Adaptive]). In contrast, [RoboNinja] can iteratively update state estimation after the collision and adjust the cutting policy to avoid getting stuck, achieving a balance between energy consumption and the cut mass.

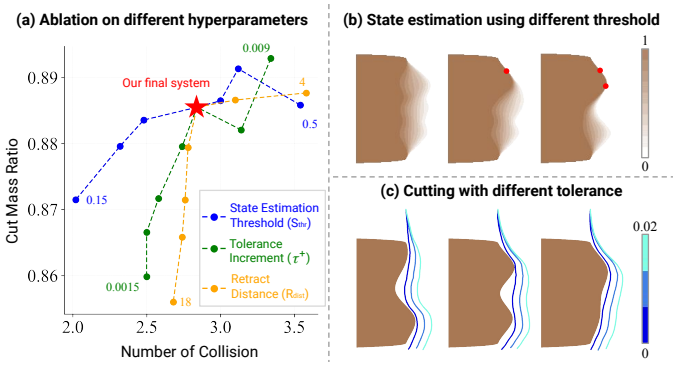

Fig. 11: **Ablations on algorithm parameters.** [Left] summarizes the effects of several critical hyperparameters in the algorithm. Detailed discussion about their effects and trade-off can be found in Sec. IV-C. [Right-Up] State estimation results with different thresholds. [Right-Bottom] Cutting trajectories with different tolerance increments.

	In-distribution Geometries			Out-of-distribution Geometries		
	COMP↑	Cut M↑	COLL↓	COMP↑	Cut M↑	COLL↓
Non-Adaptive	0.125	0.489	0.049	0.200	0.438	0.048
RoboNinja	**1.000**	**0.877**	**0.027**	**1.000**	**0.824**	**0.028**

TABLE III: **Cutting performance of real-world evaluation.** Here COMP, Cut M, and COLL represent Completion Rate, Cut Mass Ratio, and Collision Ratio respectively.

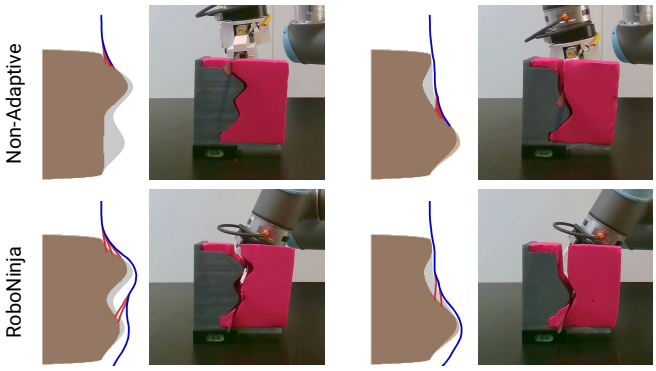

Fig. 12: **Real-world comparison**. [Non-Adaptive] has the drawback of getting stuck even when the state estimation is very close. In contrast, [RoboNinja] leverages an adaptive cutting policy that allows it to bypass the peak and successfully reach the bottom.

C. Ablation Studies

The following experiments study the effects of a few critical parameters and design choices. The experimental results are summarized in Fig. 10 and Fig. 11.

Energy weight (η_e) First, we want to validate the effect of energy penalty on trajectory optimizations. From the qualitative results in Fig. 10, we can observe that the trajectories optimized with no energy penalty strictly follow the contour of the core, but the knife rotation changes frequently to avoid collisions between the knife spine and the core. In contrast, a large energy weight derives an over-conservative cutting

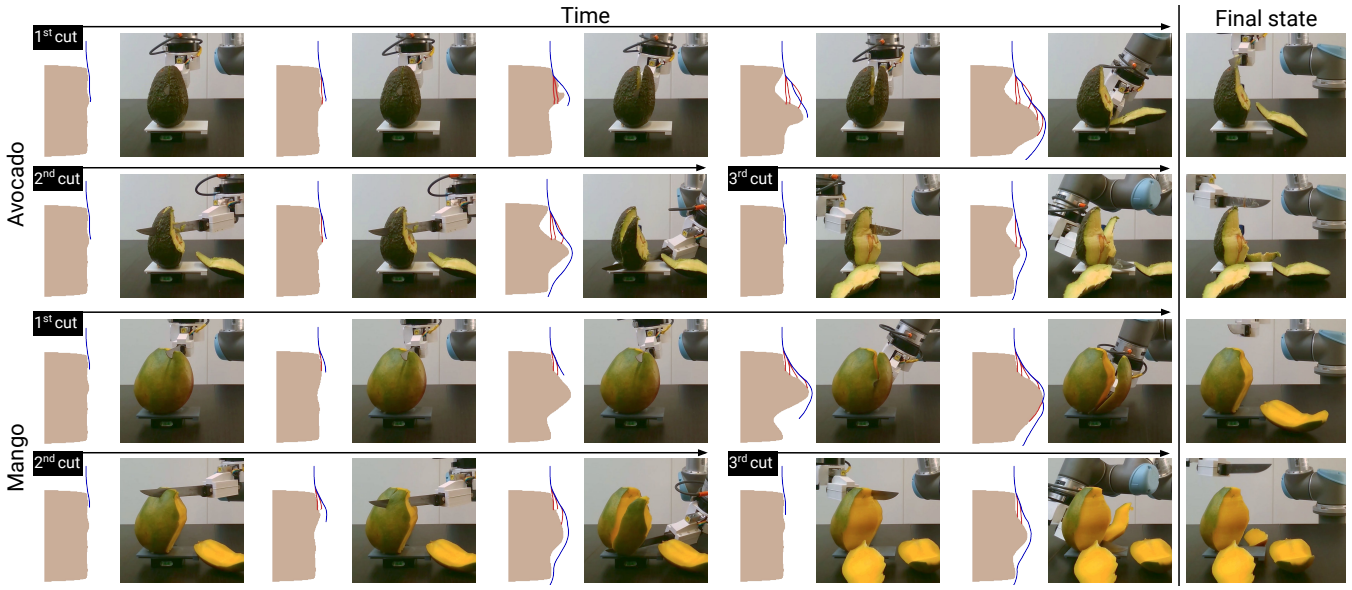

Fig. 13: **Evaluation on real fruits**. The left part illustrates the cutting execution on both an avocado and a mango, with an initial rotation angle of 0°, -45°, and 45°, respectively. The last column displays the final state after one cut and all three cuts.

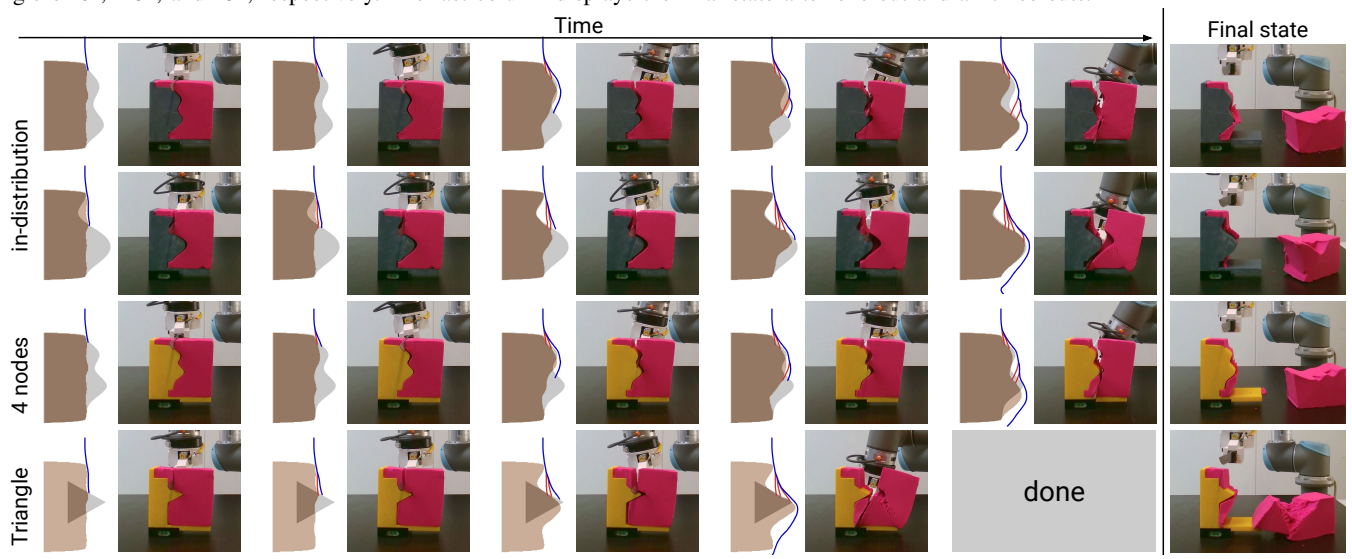

Fig. 14: **Realworld evaluation on 3d printed cores and kinetic sand.** Our simulation-trained policy demonstrates strong generalization capabilities, effectively handling both in-distribution and out-of-distribution cores in a real-world setting. With only a few collisions, it is able to accurately estimate the core geometry and cut off the majority of the sand with a smooth cutting trajectory.

behavior. An appropriate energy weight should achieve a good balance between energy consumption and cut mass. The one used in our final system (0.15) is able to cut off over 90% of the soft material while consuming a comparable amount of energy when compared to the one with a large energy weight.

State estimation threshold (S_{thr}) We evaluate our policy with different state estimation thresholds ranging from 0.15 to 0.5. Fig. 11 (b) indicates that the estimation results are consistent on the area to the left of the core boundary estimated from collision signals. However, the variation in the threshold value leads to a visible difference in the unexplored area. A small threshold suggests a larger estimated geometry, resulting in a more conservative policy with both fewer collisions and less cut mass (blue line in Fig. 11 (a)). We choose 0.3 in our system to achieve a balance between the number of collisions and the cut mass ratio.

Tolerance increment (τ^+) Fig. 11 (c) demonstrates the cutting trajectories on the same core with different tolerance values. An aggressive cutting policy with a small tolerance will increase the cut mass but also increase the chance of collisions. As to the tolerance adaptation strategy, the green line in Fig. 11 (a) shows that a large tolerance increment value τ_+ results in a conservative policy with fewer collisions and a smaller cut mass.

Retract distance (R_{dis}) Another important design choice is the retract distance, which determines the number of retraction steps after a collision. The yellow line in Fig. 11 (a) indicates different trends. From 18 to 8, the cut mass ratio increases significantly with a small increase in the number of collisions, but such benefits no longer exist afterward. Therefore, 8 is

selected as the retract distance in the final system.

D. Evaluation on a Real-world Setup

We directly evaluate the trained model on a real-world platform, where a UR5 robot is equipped with a knife and a force sensor described in Sec. III-D.

3D printed cores. In Fig. 7, we 3D print 8 in-distribution and 5 out-of-distribution geometries from the test set. As for the soft material, we use Kinect Sand as a proxy because of its stable physics property. Although a 1-DoF force sensor is sufficient for collision detection, it cannot reflect the energy consumption caused by abrupt rotations. Hence, in our evaluations, we only consider the first three metrics, which are *completion rate*, *cut mass ratio*, and *collision ratio*. For the sake of safety, we exclude [Greedy] from real-world evaluations and select [Non-Adaptive] for comparison. The termination criteria for the real-world evaluation are reaching the bottom (completed) or more than 10 collisions (failed). Following the same criteria as in simulation, if the execution is failed, the soft material to the right of the knife's trajectory is considered to be cut off. The quantitative results in Tab. III and qualitative comparison in Fig. 12 show that [Non-Adaptive] gets stuck in most cases. In contrast, [RoboNinja] is able to bypass the core after a few collisions and complete all test cases. The more detailed qualitative results in Fig. 14 demonstrate that [RoboNinja] is able to accurately estimate core geometry with sparse collision signals only, and cut the soft part off the cores with not only in-distribution but also out-of-distribution geometries in real-world scenarios.

Fruits. We then evaluate our model on real fruits, including avocados, mangos, and plums. To better resemble real-world scenarios, we execute the same policy multiple times with different initial rotation angles about the y-axis to cut off more soft material from different directions. Additionally, we augment our vertical cutting trajectory with an additional horizontal and repetitive back-and-forth slicing primitive to effectively cut through fiber-rich material, such as mango[2] and plum skin. Qualitative results on an avocado and a mango are shown in Fig. 13, demonstrating that our model trained with procedurally generated geometries is able to generalize to various fruit cores in the real world. Additionally, the ability to extend the policy allows it to cut off soft material from different directions, making it more practical in the real world.

Meat. We also evaluate our cutting policy on real bone-in meat, specifically, oxtail. In order to resemble real-world situations more realistically, we employ a bimanual setup, where one arm with a parallel-jaw gripper (WSG50) holds the bone and the other arm performs the cutting action. As raw meat is difficult to cut even with the bask-and-forth slicing primitive due to the high collagen content inside tendons, we choose to use cooked meat in this experiment. Qualitative results in Fig. 15 demonstrate RoboNinja's strong generalization ability in meat-cutting scenarios with novel bone geometry and soft material property.

[2]Note that since the force required to create the initial opening on the mango skin exceeds the force limit of our robot, we had to manually remove a small piece of skin at the top.

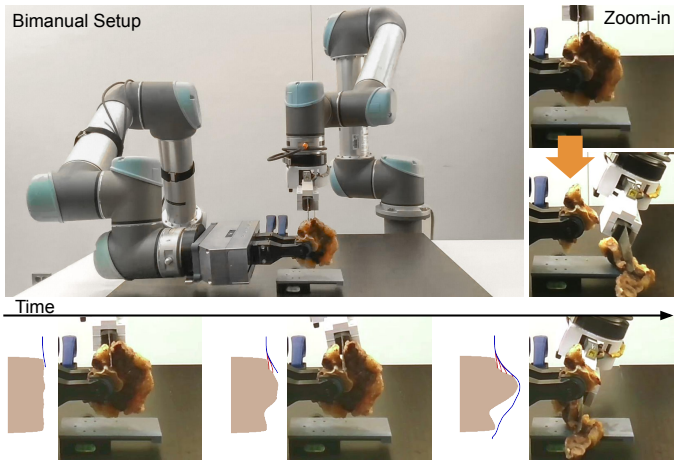

Fig. 15: **Evaluation on real meat.** We deploy our cutting policy on a bimanual setup, with one arm using a parallel-jaw gripper to hold the bone and the other arm using a knife to cut the meat. The cutting trajectory and core estimation are shown at the bottom.

E. Limitations and Potential Improvements

There are several limitations and potential improvements of our system: (a) The knife may exhibit visible deformations (i.e., bending) in real-world scenarios, causing deviations in behavior compared to simulation and disrupting the accuracy of state estimation due to misleading collision positions. This could be addressed by incorporating more accurate knife modeling in our simulator. (b) In this work, we consider the scenario where the object being cut is secured using an external fixture, and our primary objective is to optimize the trajectory of the cutting tool. In practice, cutting is a bimanual problem, which typically involves coordination between both arms, one to hold and reorient the object and the other to execute the cutting. In the meat-cutting scenario, we attempt to employ such a bimanual setup; however, this setup uses one arm to fix the meat bone and does not fully consider the coordination required between both arms. In addition, human counterparts use a dexterous hand with a soft exterior to firmly hold the object without damaging it, rather than a rigid gripper. Developing such hardware for a dual-arm robotic system, along with the coordinated control policies to perform cutting more efficiently and safely is a promising direction.

V. Conclusion

We introduce RoboNinja, a robotic system for cutting multi-material objects. The system utilizes demonstrations collected in our newly developed differentiable simulator to train an iterative state estimator and an adaptive cutting policy. It enables the robot to cut soft material off the rigid core while optimizing for both collision occurrences and energy consumption. We also present a low-cost real-world cutting system with real-time force feedback and collision detection, which is used to testify our proposed method on a real robotic setup. Our experiments show that our method is able to generalize well to novel core geometries and even real fruits. We hope our

experimental findings and the newly developed simulator could inspire future work on robot learning involving interactions with multi-material objects.

Acknowledgement

We would like to thank Huy Ha, Zeyi Liu, and Mandi Zhao for their helpful feedback and fruitful discussions. This work was supported in part by NSF Awards 2037101, 2132519, 2037101, and Toyota Research Institute. Dr. Gan was supported by the DARPA MCS program and gift funding from MERL, Cisco, and Amazon. We would like to thank Google for the UR5 robot hardware. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

References

[1] Peter K Allen and Paul Michelman. Acquisition and interpretation of 3-d sensor data from touch. 1990. 2

[2] P Areias and Timon Rabczuk. Steiner-point free edge cutting of tetrahedral meshes with applications in fracture. *Finite Elements in Analysis and Design*, 132:27–41, 2017. 2

[3] Iago Berndt, Rafael Torchelsen, and Anderson Maciel. Efficient surgical cutting with position-based dynamics. *IEEE computer graphics and applications*, 37(3):24–31, 2017. 2

[4] Alexander Bierbaum, Ilya Gubarev, and Rüdiger Dillmann. Robust shape recovery for sparse contact location and normal data from haptic exploration. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3200–3205. IEEE, 2008. 2

[5] Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav S Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017. 2

[6] THC Childs. Friction modelling in metal cutting. *Wear*, 260(3):310–318, 2006. 2

[7] Vivian Chu, Ian McMahon, Lorenzo Riano, Craig G McDonald, Qin He, Jorge Martinez Perez-Tejada, Michael Arrigo, Trevor Darrell, and Katherine J Kuchenbecker. Robotic learning of haptic adjectives through physical interaction. *Robotics and Autonomous Systems*, 63:279–292, 2015. 2

[8] Heather Culbertson, Juliette Unwin, and Katherine J Kuchenbecker. Modeling and rendering realistic textures from unconstrained tool-surface interactions. *IEEE transactions on haptics*, 7(3):381–393, 2014. 2

[9] Simon Duenser, Roi Poranne, Bernhard Thomaszewski, and Stelian Coros. Robocut: Hot-wire cutting with robot-controlled flexible rods. *ACM Transactions on Graphics (TOG)*, 39(4):98–1, 2020. 2

[10] Samir Yitzhak Gadre, Kiana Ehsani, and Shuran Song. Act the part: Learning interaction strategies for articulated object part discovery. *ICCV*, 2021. 2

[11] Alan Arnold Griffith. Vi. the phenomena of rupture and flow in solids. *Philosophical transactions of the royal society of london. Series A, containing papers of a mathematical or physical character*, 221(582-593):163–198, 1921. 2

[12] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018. 7

[13] Eric Heiden, Miles Macklin, Yashraj S Narang, Dieter Fox, Animesh Garg, and Fabio Ramos. DiSECt: A Differentiable Simulation Engine for Autonomous Robotic Cutting. In *Proceedings of Robotics: Science and Systems*, Virtual, July 2021. doi: 10.15607/RSS.2021.XVII.067. 1, 2, 3

[14] Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 2, 3

[15] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable programming for physical simulation. *arXiv preprint arXiv:1910.00935*, 2019. 2

[16] Yuanming Hu, Jiancheng Liu, Andrew Spielberg, Joshua B Tenenbaum, William T Freeman, Jiajun Wu, Daniela Rus, and Wojciech Matusik. Chainqueen: A real-time differentiable physical simulator for soft robotics. In *2019 International conference on robotics and automation (ICRA)*, pages 6265–6271. IEEE, 2019. 2

[17] Zhiao Huang, Yuanming Hu, Tao Du, Siyuan Zhou, Hao Su, Joshua B Tenenbaum, and Chuang Gan. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. *arXiv preprint arXiv:2104.03311*, 2021. 2, 4

[18] Lenka Jeřábková and Torsten Kuhlen. Stable cutting of deformable objects in virtual environments using xfem. *IEEE computer graphics and applications*, 29(2):61–71, 2009. 2

[19] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5616–5626, 2022. 2

[20] Jacqueline Kenney, Thomas Buckley, and Oliver Brock. Interactive segmentation for manipulation in unstructured environments. In *2009 IEEE International Conference on Robotics and Automation*, pages 1377–1382. IEEE, 2009. 2

[21] Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 6

[22] Dan Koschier, Sebastian Lipponer, and Jan Bender. Adaptive tetrahedral meshes for brittle fracture simula-

tion. In *Symposium on Computer Animation*, pages 57–66, 2014. 2

[23] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018. 2

[24] Yunzhu Li, Jiajun Wu, Jun-Yan Zhu, Joshua B Tenenbaum, Antonio Torralba, and Russ Tedrake. Propagation networks for model-based control under partial observation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1205–1211. IEEE, 2019. 2

[25] Xingyu Lin, Zhiao Huang, Yunzhu Li, Joshua B. Tenenbaum, David Held, and Chuang Gan. Diffskill: Skill abstraction from differentiable physics for deformable object manipulations with tools. *International Conference on Learning Representation (ICLR)*, 2022. 2

[26] Xingyu Lin, Carl Qi, Yunchu Zhang, Zhiao Huang, Katerina Fragkiadaki, Yunzhu Li, Chuang Gan, and David Held. Planning with spatial-temporal abstraction from point clouds for deformable object manipulation. In *6th Annual Conference on Robot Learning*, 2022. 2

[27] Huaping Liu, Yupei Wu, Fuchun Sun, and Di Guo. Recent progress on tactile object recognition. *International Journal of Advanced Robotic Systems*, 14(4): 1729881417717056, 2017. 2

[28] Philip Long, Amine Moughlbay, Wisama Khalil, and Philippe Martinet. Robotic meat cutting. In *Ict-pamm workshop*, 2013. 1, 2

[29] Yujie Lu, Jianren Wang, and Vikash Kumar. Curiosity driven self-supervised tactile exploration of unknown objects. *arXiv preprint arXiv:2204.00035*, 2022. 2

[30] Jun Lv, Qiaojun Yu, Lin Shao, Wenhai Liu, Wenqiang Xu, and Cewu Lu. Sagci-system: Towards sample-efficient, generalizable, compositional, and incremental robot learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 98–105. IEEE, 2022. 2

[31] Takamitsu Matsubara and Kotaro Shibata. Active tactile exploration with uncertainty and travel cost for fast shape estimation of unknown objects. *Robotics and Autonomous Systems*, 91:314–326, 2017. 2

[32] M Eugene Merchant. Mechanics of the metal cutting process. i. orthogonal cutting and a type 2 chip. *Journal of applied physics*, 16(5):267–275, 1945. 2

[33] O Miller, LB Freund, and A Needleman. Modeling and simulation of dynamic fragmentation in brittle materials. *International Journal of Fracture*, 96(2):101–125, 1999. 2

[34] Xiaoqian Mu, Yuechuan Xue, and Yan-Bin Jia. Robotic cutting: Mechanics and control of knife motion. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3066–3072. IEEE, 2019. 1, 2

[35] Neil Nie, Samir Yitzhak Gadre, Kiana Ehsani, and Shuran Song. Structure from action: Learning interactions for articulated object 3d structure discovery. *arxiv*, 2022. 2

[36] Junjun Pan, Junxuan Bai, Xin Zhao, Aimin Hao, and Hong Qin. Real-time haptic manipulation and cutting of hybrid soft tissue models by extended position-based dynamics. *Computer Animation and Virtual Worlds*, 26 (3-4):321–335, 2015. 2

[37] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 6

[38] Christoph J Paulus, Lionel Untereiner, Hadrien Courtecuisse, Stéphane Cotin, and David Cazier. Virtual cutting of deformable objects based on efficient topological operations. *The Visual Computer*, 31(6):831–841, 2015. 2

[39] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W Battaglia. Learning mesh-based simulation with graph networks. *arXiv preprint arXiv:2010.03409*, 2020. 2

[40] Carl Qi, Xingyu Lin, and David Held. Learning closed-loop dough manipulation using a differentiable reset module. *IEEE Robotics and Automation Letters*, pages 1–8, 2022. doi: 10.1109/LRA.2022.3191239. 2

[41] Amrita Sawhney, Steven Lee, Kevin Zhang, Manuela Veloso, and Oliver Kroemer. Playing with food: Learning food item representations through interactive exploration. In *International Symposium on Experimental Robotics*, pages 309–322. Springer, 2021. 2

[42] David Schiebener, Aleš Ude, Jun Morimoto, Tamim Asfour, and Rüdiger Dillmann. Segmentation and learning of unknown objects through physical interaction. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 500–506. IEEE, 2011. 2

[43] David Schiebener, Jun Morimoto, Tamim Asfour, and Aleš Ude. Integrating visual perception and manipulation for autonomous learning of object representations. *Adaptive Behavior*, 21(5):328–345, 2013. 2

[44] Alexander Schmitz, Yusuke Bansho, Kuniaki Noda, Hiroyasu Iwata, Tetsuya Ogata, and Shigeki Sugano. Tactile object recognition using deep learning and dropout. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 1044–1050. IEEE, 2014. 2

[45] Alexander Schneider, Jürgen Sturm, Cyrill Stachniss, Marco Reisert, Hans Burkhardt, and Wolfram Burgard. Object identification with tactile sensors using bag-of-features. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 243–248. IEEE, 2009. 2

[46] Mohit Sharma, Kevin Zhang, and Oliver Kroemer. Learning semantic embedding spaces for slicing vegetables. *arXiv preprint arXiv:1904.00303*, 2019. 1, 2

[47] Asbjørn Søndergaard, Jelle Feringa, Toke Nørbjerg, Kasper Steenstrup, David Brander, Jens Graversen, Steen Markvorsen, Andreas Bærentzen, Kiril Petkov, Jesper Hattel, et al. Robotic hot-blade cutting. In *Robotic*

fabrication in architecture, art and design 2016, pages 150–164. Springer, 2016. 2

[48] Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A material point method for snow simulation. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013. 3

[49] Stephanie Wang, Mengyuan Ding, Theodore F Gast, Leyi Zhu, Steven Gagniere, Chenfanfu Jiang, and Joseph M Teran. Simulation and visualization of ductile fracture with the material point method. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2(2): 1–20, 2019. 2

[50] Tsun-Hsuan Wang, Andrew Everett Spielberg, Pingchuan Ma, Zhou Xian, Hao Zhang, Joshua B. Tenenbaum, and Chuang Gan. Softzoo: A soft robot co-design benchmark for locomotion in diverse environments. In *International Conference on Learning Representations*, 2023. 2

[51] Joshuah Wolper, Yu Fang, Minchen Li, Jiecong Lu, Ming Gao, and Chenfanfu Jiang. Cd-mpm: continuum damage material point methods for dynamic fracture animation. *ACM Transactions on Graphics (TOG)*, 38 (4):1–15, 2019. 2

[52] Joshuah Wolper, Yunuo Chen, Minchen Li, Yu Fang, Ziyin Qu, Jiecong Lu, Meggie Cheng, and Chenfanfu Jiang. Anisompm: Animating anisotropic damage mechanics. *ACM Trans. Graph.*, 39(4), 2020. 2

[53] Jun Wu, Rüdiger Westermann, and Christian Dick. A survey of physically based simulation of cuts in deformable bodies. In *Computer Graphics Forum*, volume 34, pages 161–187. Wiley Online Library, 2015. 2

[54] Zhou Xian, Bo Zhu, Zhenjia Xu, Hsiao-Yu Tung, Antonio Torralba, Katerina Fragkiadaki, and Chuang Gan. Fluidlab: A differentiable environment for benchmarking complex fluid manipulation. In *International Conference on Learning Representations*, 2023. 2, 4

[55] Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, and Miles Macklin. Accelerated policy learning with parallel differentiable simulation. *arXiv preprint arXiv:2204.07137*, 2022. 2

[56] Jingxi Xu, Han Lin, Shuran Song, and Matei Ciocarlie. Tandem3d: Active tactile exploration for 3d object recognition. *arXiv preprint arXiv:2209.08772*, 2022. 2

[57] Jingxi Xu, Shuran Song, and Matei Ciocarlie. Tandem: Learning joint exploration and decision making with tactile sensors. *IEEE Robotics and Automation Letters*, 2022. 2

[58] Zhenjia Xu, Zhanpeng He, Jiajun Wu, and Shuran Song. Learning 3d dynamic scene representations for robot manipulation. In *Conference on Robot Learning (CoRL)*, 2020. 2

[59] Kevin Zhang, Mohit Sharma, Manuela Veloso, and Oliver Kroemer. Leveraging multimodal haptic sensory data for robust cutting. In *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, pages 409–416. IEEE, 2019. 2

[60] Debao Zhou, Mark R Claffee, Kok-Meng Lee, and Gary V McMurray. Cutting," by pressing and slicing", applied to robotic cutting bio-materials. i. modeling of stress distribution. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2896–2901. IEEE, 2006. 1, 2

[61] Debao Zhou, Mark R Claffee, Kok-Meng Lee, and Gary V McMurray. Cutting,'by pressing and slicing', applied to the robotic cut of bio-materials. ii. force during slicing and pressing cuts. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2256–2261. IEEE, 2006. 1, 2

[62] Fenghua Zhou, Jean-Francois Molinari, and Tadashi Shioya. A rate-dependent cohesive model for simulating dynamic crack propagation in brittle materials. *Engineering fracture mechanics*, 72(9):1383–1410, 2005. 2