DexNinja: Learning Robust Dexterous Cutting Policy with a Real-to-Sim-to-Real Data Engine

Haozhe Lou[1*], Runyi Yang[2*], Wanqi Zhong[1], Caiyun Liu[3], Yicheng Liu[1], Wenhao Liu[4], Wenlong Ma[1]
Jiawei Xia[1], Xu Zheng[5], Danda Pani Paudel[2], Luc Van Gool[2], Hang Zhao[1], Yiming Li[1]

[1]Tsinghua University [2]INSAIT, Sofia University "St. Kliment Ohridski" [3]Peking University [4]University of Southern California [5]HKUST (GZ)

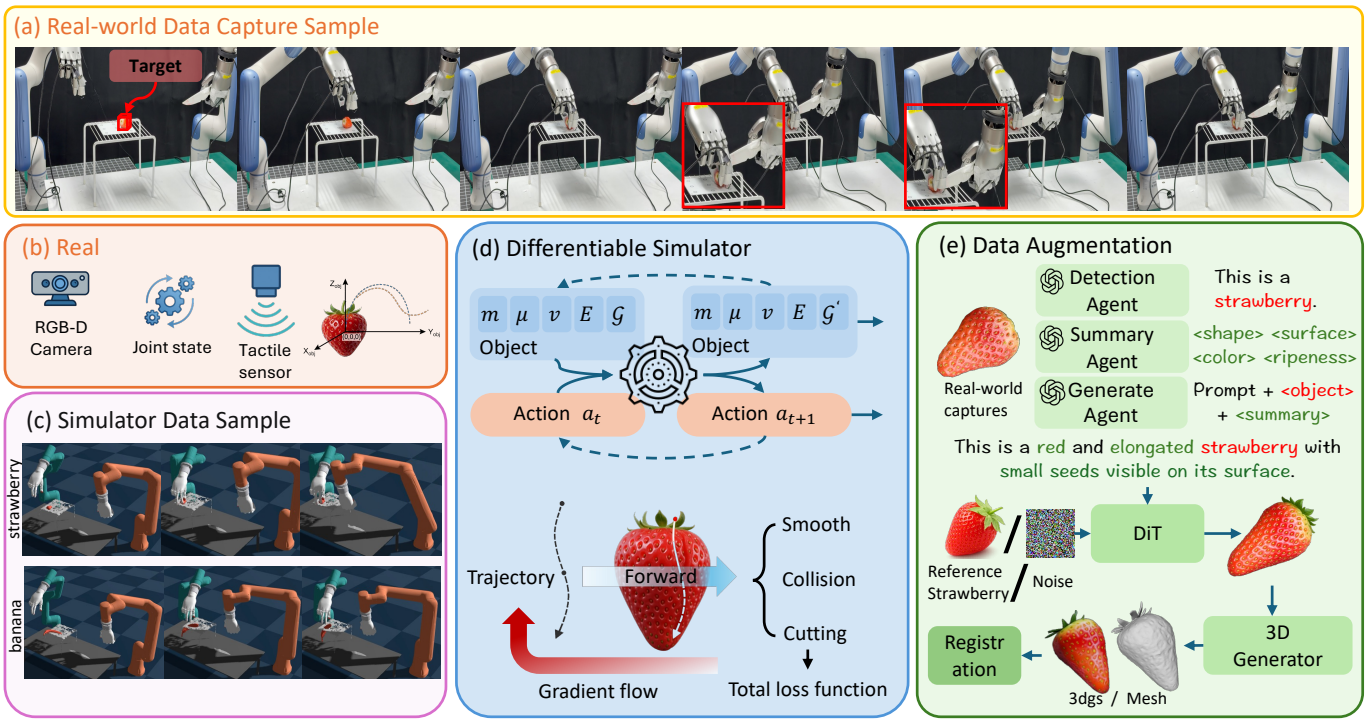

Fig. 1: **Real2Sim data densification pipeline.** (a) Real-world demonstration sample via teleoperation. (b) Real-world multi-modal sensing, and register object into a unified coordinate frame. (c) Simulation Data example (d) Differentiable simulator empowers the learning within simulator. The trajectory is optimized by minimizing cutting policy, collision penalty and smoothness to minimize the energy consumption. (e) Detection and summary agents extract structured object knowledge, which guides generative agents to produce diverse, but in-domain instances to get controlable 3D generation (3DGS or meshes).

***Abstract*—Cutting a piece of food looks effortless in human hands: we nudge the blade, adjust the angle, modulate pressure as the skin yields and cuts. For a robot, however, every cut is a high-contact interaction with a deformable, elasto-plastic object whose behavior changes with pose, contact mode, and cutting stage. Collecting real-world data is also inherently expensive and wasteful: each trial typically contaminates or damages the food, so obtaining scalable real world data for dexterous cutting are expensive and difficult. Simulation is an appealing alternative, but a large sim-to-real gap exists in both physics and perception. This gap is especially severe for topology-changing interactions like cutting, where small errors in contact, friction, or material and elastic transformations response quickly compound. We propose DexNinja, a differentiable real2sim2real framework that turns a handful of real demonstrations into a dense and realistic training distribution. DexNinja (i) reconstructs object instances from real trajectories, (ii) randomizes physically meaningful parameters under category-level constraints, and (iii) augmented dexterous manipulation episodes using a custom differentiable simulator that couples robot dynamics, tactile contact, and deformable cutting. We evaluate DexNinja on a dexterous food-slicing task and show that (1) the augmented data consistently improves sim-to-real transfer, and (2) the resulting policies generalize to out-of-distribution objects with varying shapes and sizes.**

I. Introduction

Robotic manipulation has recently made strong progress on everyday household behaviors such as table cleaning, dish washing, and trash pickup [29, 24, 23]. Yet these advances largely center on rigid objects and quasi-static contacts, while cooking remains a frontier where the robot must purposefully change an object's state. Food manipulation is particularly challenging because many ingredients are elasto-plastic: they deform, yield, and recover in ways that depend on contact history, friction, and internal material properties. Classical approaches for food interaction often rely on handcrafted rules or analytic heuristics [73, 57, 16, 45], which struggle to generalize across object instances and to express the long-horizon,

* indicates equal contribution.

contact-rich motions required for cutting. Prior systems have also explored cutting with specialized tool and hardware designs [85], but such solutions can limit cross-tool and cross-task reuse. In parallel, dexterous hands have emerged as a powerful end-effector for manipulation [60, 72], enabling richer stabilization strategies and finer force modulation, where these capabilities are central to cooking.

Seemingly simple actions like slicing an apple demand rapid, continuous decisions about tool placement, object stabilization, and force control. Much of this competence comes from a tight perception-action loop that combines vision with touch: tactile cues reveal micro-slips, sticking events, and the onset and propagation of a cut. Fruit cutting stresses this closed-loop physical reasoning in an unforgiving regime: deformable elasto-plastic objects, intermittent contacts, and topology changes as the object splits [25]. These properties make fruit cutting a representative benchmark for cooking-style manipulation, and a natural testbed for learning-based methods that must operate under irreversible state changes.

Learning from real demonstrations is the most direct path to acquiring such behaviors, but for cutting it is unusually expensive and wasteful. Each trial damages the fruit; after a small number of executions the object becomes unsuitable for repeated collection due to contamination and accumulated deformation. Consequently, real datasets are sparse and biased: they cover few object instances, and they concentrate around limited stages of the cutting process, leaving large portions of the state space underexplored.

Simulation is an attractive alternative, but food deformation amplifies the sim-to-real (sim2real) gap. Success depends on accurate contact forces, frictional conditions, and deformable material behavior, while perception must remain stable as geometry and topology evolve [20]. Small mismatches in physics parameters or sensing can derail the policy [26, 31, 7].

In this paper, we propose a differentiable **real-to-sim-to-real** (real2sim2real) strategy [40] to bridge this gap for cooking-style, topology-changing manipulation. Our key idea is to use *category-level structure* as a scaffold for diversity: given a category label (e.g., "apple-like fruits"), we reconstruct object instances from sparse real demonstrations, then generate a family of physically plausible variations that preserve category identity while meaningfully changing geometry, mass, friction, and appearance. By grounding this augmentation in real data and enforcing physically meaningful constraints, we densify training distributions without drifting into unrealistic simulation artifacts. Crucially, differentiability couples reconstruction, parameter estimation, and simulation in a single loop, allowing gradients to guide policy learning.

This design targets two outcomes that are difficult to obtain with either sparse real data or pure simulation alone:

1) **Sim2real through realistic densification:** starting from limited real demonstrations, we generate dense training data that remains close to real-world behavior, improving sim2real transfer across multiple learning baselines.
2) **Generalization within a category:** by training on physically plausible category-level variations, we aim to learn policies that remain effective on unseen object with substantial shifts in geometry, mass, and appearance.

Our contributions are:

1) **A differentiable real2sim2real data augmentation framework** that reconstructs object instances from sparse real demonstrations and perturbs geometry, mass, friction, and appearance under category-level constraints to produce dense yet realistic training distributions.
2) **A learning framework for multi-modal, topology-changing manipulation** that leverages physically guided gradients for visual–tactile policy learning on deformable objects, moving beyond rigid-only assumptions in sim2real transfer.

II. Related Works

A. Data generation and differentiable simulation

Scalable data generation is increasingly studied for robot manipulation [13, 96, 88]. Trajectory-centric methods such as MimicGen generate large-scale simulation trajectories [43], while DemoGen extends this capability to real-world demonstrations [86]. R2RGen samples over object pose distributions [84], and Giga-0 further expands variation to appearance and motion [68, 90]. Pointworld directly synthesizes 3D point flows as motion representations [22]. In contrast, we focus on *physical densification* for topology-changing tasks: rather than only generating trajectories or poses, we densify *mass*, *geometry*, and *friction* (optionally appearance) in a differentiable loop to improve system identification and policy robustness. Since contact forces are difficult to infer from vision alone, we further leverage tactile signals to constrain contact estimation [1].

Gaussian Splatting has become practical for robotics perception [28, 32, 41, 33, 30] and simulation [99]. Robo-GS presents a Real2Sim pipeline for identifying object physical parameters [39], SplatSim shows that photo-realistic simulation improves policy robustness [55], and GSWorld unifies Gaussian splatting with URDF for simulation datasets [27]. These works motivate appearance randomization and controllable rendering to reduce sim2real gaps [89]; in our framework, appearance augmentation complements physics densification and can be integrated into the same pipeline [70, 44, 69, 12, 82, 50]. More broadly, system identification recovers latent physical properties by matching predicted and observed dynamics, including approaches that infer parameters from vision [65] or videos [4], and physics-integrated Gaussian representations that connect rendering primitives with simulated states [81]. Related works also use object-centric tracking and reward signals to link visual observations with action outcomes, reducing supervision for manipulation [15, 30].

For topology-changing manipulation specifically, we build on physics-based simulators and differentiable cutting models such as Taccel [34] and DiSECt [17], which enable gradient-based simulation of fracture and topology change. RoboNinja studies cutting under randomized object properties [85, 36]. Classic rule-based cutting plans tool paths and interaction

forces along predefined trajectories [46], while vision-based cutting incorporates visual feedback for alignment and deformation control [16] and force-aware controllers adapt motions using force sensing [78]. We treat mesh geometry as a initial condition for MPM based cutting dynamics, enabling structured densification around real objects [79, 100, 98, 9]. Unlike pure randomization or heuristic control, we enforce topology consistency and anchor the distribution to real2sim reconstructed meshes or Gaussian splatting.

B. Dexterous manipulation and Teleoperation

Dexterous manipulation spans universal grasping, task-specific skills, learning from video, and reward learning. Universal grasping enables zero-shot, contact-aware control [93, 35], while task-specific methods learn skills such as in-hand rotation and compliant grasping [52, 8, 53, 2]. Learning from human videos reduces robot data needs but is often open-loop [59]. Representation- and theory-driven works seek unified interaction abstractions for cross-task and cross-embodiment generalization [74, 56, 49, 60, 6, 83], while reward-based methods learn transferable visual rewards [42].

Tactile sensing is key to robust dexterous manipulation, especially for visual–tactile cutting where spatio-temporal force–contact cues matter [18]. We therefore leverage differentiable tactile simulation to improve learning and generalization [21, 71, 63]. To obtain physically executable real2sim data, we retarget teleoperation demonstrations into the robot base frame under kinematic constraints. This enables efficient collection of contact-rich trajectories with improved operator perception, while UMI further standardize and scale cross-embodiment datasets that support training generalist manipulation policies [58, 11, 97, 38].

III. Problem Formulation

We consider the problem of *dexterous food cutting with tool use*, where a robot equipped with a dexterous hand and a knife interacts with deformable food items (e.g., strawberries) whose physical properties vary across instances.

A. Observation space

At each discrete time step t, the robot receives a multi-modal observation

$$\mathbf{o}_t = \left\{\mathbf{o}_t^{\text{vis}},\ \mathbf{o}_t^{\text{tac}},\ \mathbf{o}_t^{\text{prop}}\right\} \in \mathcal{O}, \tag{1}$$

where: (i) $\mathbf{o}_t^{\text{vis}}$ denotes the visual observation from RGB-D sensors, represented as a point cloud of the scene including the target instance fruit, tools and background, (ii) $\mathbf{o}_t^{\text{tac}}$ contains tactile measurements from fingertip sensor arrays, and (iii) $\mathbf{o}_t^{\text{prop}}$ denotes robot proprioceptive signals such as joint positions, joint velocities, and hand configuration states.

a) Target Instance: We represent each instance instantiated with physically interpretable parameters

$$\vartheta_t = \big(m, \mu_f, E, \nu, \mathcal{G}(t)\big), \tag{2}$$

where m is mass, μ_f is an effective Coulomb friction coefficient, $\mathcal{G}$ is the geometry (mesh/particles) regarding time, and (E, ν) are Young's modulus and Poisson's ratio for isotropic linear elasticity. For reference, uniaxial stress–strain satisfies $\sigma = E\,\varepsilon$ in the small-strain regime, and in 3D isotropic elasticity we convert (E, ν) to Lamé parameters

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \qquad G = \frac{E}{2(1+\nu)}, \tag{3}$$

where λ controls volumetric response and G (shear modulus) controls shear resistance [81].

b) Knife and Scene background: The knife is treated as a known rigid tool with structured model together with scene background directly represented as a colored point primitives

$$P = \{p_k\}_{k=1}^{N_P}, \qquad p_k = (x_k, c_k), \quad x_k \in \mathbb{R}^3,\ c_k \in \mathbb{R}^d, \tag{4}$$

(e.g., $d = 3$ for RGB point cloud, $d = 11$ for Gaussian Splatting), obtained from the RGB-D stream. Thus, we can represent the visual observation as eq. (5)

$$\mathbf{o}_t^{\text{vis}} = (\vartheta_t, P) \in \mathcal{O}^{vis}. \tag{5}$$

c) Tactile observation: The tactile observation $\mathbf{o}_t^{\text{tac}}$ encodes distributed contact measurements from fingertip tactile sensor arrays, as seen example in fig. 3.

$$\mathbf{o}_t^{\text{tac}} = \{\mathbf{U}_t^{(f)}\}_{f=1}^{N_f}, \qquad \mathbf{U}_t^{(f)} \in \mathbb{R}^{H\times W\times C}, \tag{6}$$

where N_f is the number of tactile-equipped fingers, and each $\mathbf{U}_t^{(f)}$ is a spatial taxel array of resolution $H \times W$. Each taxel stores a C-dimensional local contact feature vector. For a taxel at grid location (i, j), $\mathbf{u}_{ij} = \big(p_{ij},\ \boldsymbol{\gamma}_{ij},\ \delta_{ij}\big)$, where p_{ij} denotes normal pressure, $\boldsymbol{\gamma}_{ij} \in \mathbb{R}^2$ represents tangential shear stress, and δ_{ij} is the local surface deformation.

d) Robot Proprioceptive observation: The proprioceptive observation $\mathbf{o}_t^{\text{prop}}$ describes the internal state of the robot kinematics. We represent it as

$$\mathbf{o}_t^{\text{prop}} = \big(\mathbf{q}_t,\ \dot{\mathbf{q}}_t,\ \mathbf{h}_t\big), \tag{7}$$

where $\mathbf{q}_t \in \mathbb{R}^n$ and $\dot{\mathbf{q}}_t \in \mathbb{R}^n$ denote the joint positions and joint velocities of the robot arm, respectively, and $\mathbf{h}_t$ represents hand configuration states, including finger joint angles and grasp mode indicators.

B. Policy and control

a) Actions: Let $\mathbf{o}_t \in \mathcal{O}$ be the deployable observation at time t from (1). We decompose the robot action as

$$a_t = \big(a_t^{\text{kin}},\, a_t^{\text{tac}}\big), \qquad a_t^{\text{kin}} \in \mathcal{A}^{\text{kin}},\ \ a_t^{\text{tac}} \in \mathcal{A}^{\text{tac}}, \tag{8}$$

where a_t^{kin} controls large-scale kinematic motions (left arm, right arm, and left hand), while a_t^{tac} modulates contact-sensitive right-hand behavior (e.g., grip force, finger impedance, or local compliance).

Conditioned on the static visual geometry, we model the kinematic component as a deterministic mapping

$$a_t^{\text{kin}} = f\big(\mathbf{o}^{\text{vis}}\big), \tag{9}$$

implemented in practice via object-centric motion specification and deterministic planning that maintains the desired

blade–fruit geometry. The tactile component is modeled as a observation-conditioned feedback policy

$$a_t^{\text{tac}} = \pi\big(\mathbf{o}_{0:t}^{\text{tac}}, \mathbf{o}_{0:t}^{\text{prop}}\big)\,, \tag{10}$$

which is closed-loop to tactile and proprioceptive observation.

b) Demonstration data: We assume access to a dataset of each episode of expert demonstrations

$$\mathcal{D} = \Big\{\big(\mathbf{o}_t,\, a_t\big)\Big\}, \tag{11}$$

where t indexes time steps within each episode. Each tuple consists of the deployable observation $\mathbf{o}_t \in \mathcal{O}$, the executed action $a_t \in \mathcal{A}$.

c) Objective functions: We learn a hybrid control architecture with two parameterized policies: a kinematic policy $f_\phi : \mathcal{O}^{vis} \to \mathcal{A}^{\text{kin}}$ and a tactile policy $\pi_\theta : \mathcal{O}^{\text{tac}}, \mathcal{O}^{\text{prop}} \to \mathcal{A}^{\text{tac}}$. Given demonstration data $\mathcal{D}$, we optimize

$$\begin{aligned} \hat{a}_t^{\text{kin}} &= f_\phi\big(\mathbf{o}_t^{\text{vis}}\big)\,, \\ \hat{a}_t^{\text{tac}} &= \pi_\theta\big(\mathbf{o}_{0:t}^{\text{tac}}, \mathbf{o}_{0:t}^{\text{prop}}\big)\,, \\ \mathcal{L}(\hat{a}_t, a_t) &= \mathcal{L}_{\text{kin}}\big(\hat{a}_t^{\text{kin}}, a_t^{\text{kin}}\big) + \mathcal{L}_{\text{tac}}\big(\hat{a}_t^{\text{tac}}, a_t^{\text{tac}}\big)\,. \end{aligned} \tag{12}$$

IV. Real-Sim-Real pipeline

We propose a real-sim-real learning pipeline [85, 5, 39, 51, 67] for dexterous elasto-plastic object cutting (Fig. 5). Starting from a single teleoperated demonstration, we use a differentiable cutting simulator to expand the data into a large set of physically consistent training episodes by varying object instances and feasible robot configurations [91, 92, 3, 54]. The policy is trained on these augmented trajectories using aligned multimodal observations, while deployment relies only on real sensing (vision, tactile, proprioception). The framework is object-category-centric: motions are represented in the food frame, enabling transfer across variations in object geometry, material properties, grasp configuration, and robot kinematics within the same food category. Tactile sensing is modeled consistently between the real system and simulator, allowing direct use of simulated contact signals during training.

Our pipeline consists of three stages:

- **Stage I (Sec IV-A1):** Collect a single expert cutting episode on a real food item (e.g., strawberry) via teleoperation and extract an object digital asset through real-to-simulation reconstruction [39].
- **Stage II (Sec IV-A2):** Construct the task in simulation and augment the data distribution by sampling instance physical parameters and feasible trajectory perturbations.
- **Stage III (Sec IV-A3):** Learn a policy from the augmented simulation episodes and deploy it on the real robot using new sensory observations.

A. Pipeline

We denote coordinate frames by $\mathcal{F}_{(\cdot)}$ and rigid transforms by ${}^{A}\mathbf{T}_B \in SE(3)$, which maps a point expressed in frame $\mathcal{F}_B$ into frame $\mathcal{F}_A$. We use frames: world $\mathcal{F}_W$, food $\mathcal{F}_S$, knife $\mathcal{F}_K$, and end-effector $\mathcal{F}_E$. A time-parameterized knife trajectory is $\{{}^{W}\mathbf{T}_K(t)\}_{t=0}^{T}$, and the ending time is t_f.

1) Stage I: Teleoperation for a Single Expert Episode: We teleoperate the robot to perform one successful cutting trial and record multi-modal trajectories. Each time step includes the observation $\mathbf{o}_t = \{\mathbf{o}_t^{\text{vis}}, \mathbf{o}_t^{\text{tac}}, \mathbf{o}_t^{\text{proprio}}\}$ and the executed action (joint angles). We additionally estimate the food pose ${}^{W}\mathbf{T}_S$ and its geometry [39, 75]. Then extracting the cutting trajectory in the food/object frame:

$${}^{S}\mathbf{T}_K(t) = \big({}^{W}\mathbf{T}_S\big)^{-1}\,{}^{W}\mathbf{T}_K(t), \qquad {}^{S}\mathbf{T}_{K,f} = {}^{S}\mathbf{T}_K(t_f). \tag{13}$$

The sequence $\{{}^{S}\mathbf{T}_K(t)\}$ captures the demonstrated cutting skill as a relative knife-object motion, which is transferable across reconstructed food instances.

2) Stage II: Real-to-Sim Reconstruction and Data Densification: A single real episode provides a narrow slice of the state-action distribution. The goal of Stage II is to turn this single slice into a broad and physically grounded training set. We reconstruct the task assets (object, knife, support surface) in metric scale and refine relative poses so that the object-centric template in eq. (13) is consistent with the reconstructed geometry. When the knife is rigidly grasped, we estimate the constant transform from end-effector to knife via teleoperation logs:

$${}^{E}\mathbf{T}_K(t) = \big({}^{W}\mathbf{T}_E(t)\big)^{-1}\,{}^{W}\mathbf{T}_K(t), \tag{14}$$

This estimate provides the fixed grasp geometry needed to convert object-centric knife motions into executable end-effector trajectories. We then initialize the reconstructed assets in our simulator [85, 5, 14] and enable deformable, topology-changing dynamics for the food. This is essential: cutting changes contact geometry and can induce separation, so a rigid-body proxy is insufficient for producing reliable feedback during slicing. We sample strawberry instances by varying shape and physical parameters θ (e.g., mass, compliance, friction) and generate perturbed object-centric knife trajectories $\{{}^{S}\mathbf{T}_K^{(i)}(t)\}$ around the demonstrated template (start offsets, approach angles, small pose noise). For each sampled trajectory, we map the knife motion back to an end-effector motion using the estimated grasp geometry:

$${}^{W}\mathbf{T}_E^{(i)}(t) = {}^{W}\mathbf{T}_S\;{}^{S}\mathbf{T}_K^{(i)}(t)\;\big({}^{E}\mathbf{T}_K\big)^{-1}. \tag{15}$$

We then solve inverse kinematic [66] to obtain feasible joint-space trajectories $\mathbf{q}^{(i)}(t)$ and execute them in simulation to generate augmented dataset $\mathcal{D} = \big\{\big(o_t^{(i)}, a_t^{(i)}\big\}$ as in eq. (11).

a) Trajectory generation loss.: We optimize a knife trajectory $\{(\mathbf{x}_i, \theta_i)\}_{i=0}^{T}$ by differentiating through the unrolled simulator. Knife–ground interaction is evaluated with the ground signed distance field (SDF) $d(\mathbf{p})$. At each step we sample K points across the blade width [85, 36, 45, 46]

$$\mathbf{p}_{i,k} = \mathbf{x}_i + \alpha_k\, w_{\text{knife}}\, \mathbf{dir}(\theta_i), \quad \alpha_k = \frac{k}{K-1} \tag{16}$$

$$\mathbf{dir}(\theta_i) = \big[-\sin\theta_i \quad \cos\theta_i \quad 0\big]^{\top} \tag{17}$$

and use a smooth hinge $\operatorname{srelu}(x) = \frac{1}{2}\big(x + \sqrt{x^2+\varepsilon}\big)$.

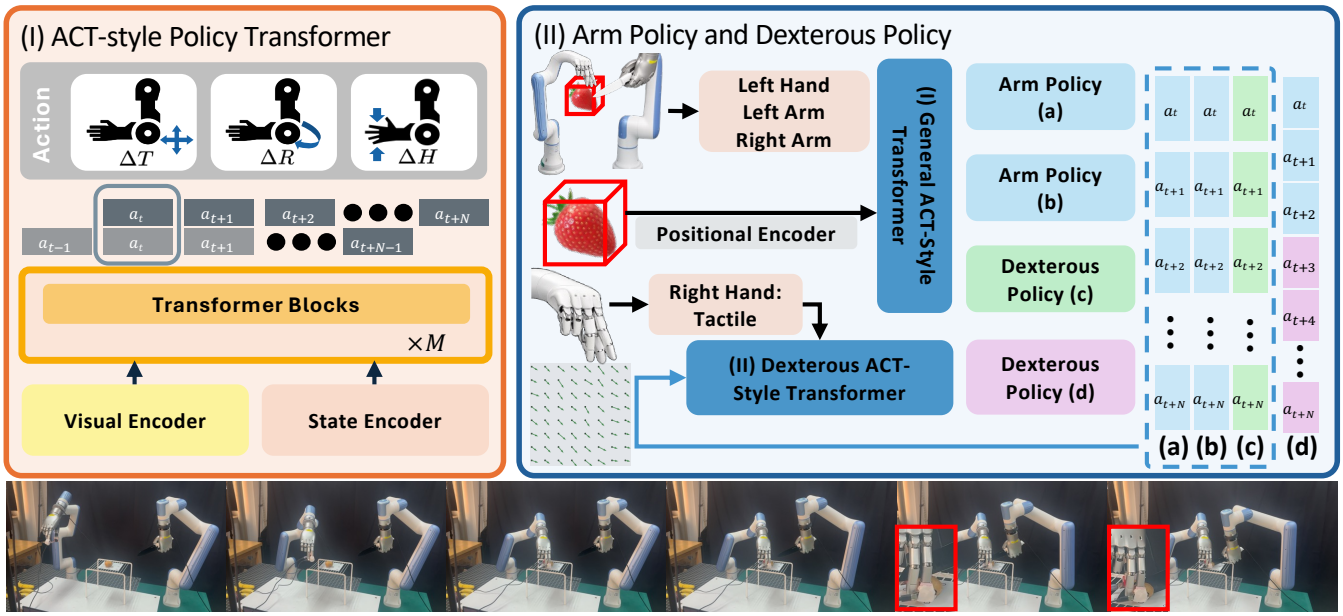

Fig. 2: **Pipeline** (I) A general ACT encodes aligned multimodal observations and predicts short action chunks. (II) Learning to control arm and dexterous hand separately via adaptive action chunking. We give an real-world inference result at bottom.

We define the nonnegative knife–ground gap

$$g_i = \frac{1}{K}\sum_{k=0}^{K-1} \mathrm{srelu}\big(d(\mathbf{p}_{i,k})\big). \tag{18}$$

The cutting objective encourages approaching the surface and making stepwise progress:

$$\mathcal{L}_{\text{cut}} = \frac{1}{T+1}\sum_{i=0}^{T} w_i\, g_i^2 + \frac{\lambda_{\text{prog}}}{T}\sum_{i=1}^{T}\Big[\mathrm{srelu}\big(g_i-(1-\rho)g_{i-1}\big)\Big]^2, \tag{19}$$

$$w_i = \left(\frac{i+1}{T+1}\right)^2. \tag{20}$$

To prevent interpenetration (with margin m), we penalize negative SDF values:

$$\mathcal{L}_{\text{col}} = \frac{1}{(T+1)K}\sum_{i=0}^{T}\sum_{k=0}^{K-1}\Big[\mathrm{srelu}\big(-(d(\mathbf{p}_{i,k})+m)\big)\Big]^4. \tag{21}$$

We add smoothness/consistency and effort regularizers:

$$\mathcal{L}_{\text{rot}} = \frac{1}{T}\sum_{i=0}^{T-1}(\theta_{i+1}-\theta_i)^2, \quad \mathcal{L}_{\text{move}} = \frac{1}{T+1}\sum_{i=0}^{T}(\theta_i^k-\theta_i^v)^2 \tag{22}$$

and minimize the weighted sum

$$\mathcal{L} = w_{\text{cut}}\mathcal{L}_{\text{cut}} + w_{\text{col}}\mathcal{L}_{\text{col}} + w_{\text{rot}}\mathcal{L}_{\text{rot}} + w_{\text{move}}\mathcal{L}_{\text{move}} \tag{23}$$

3) Stage III: Policy Learning and Real-World Deployment: With the augmented simulation episodes, we train a closed-loop policy that only observes the initial point cloud of the strawberry. We also train a second-stage control policy for the right hand for dexterous hand–fruit interaction, as shown in 5.Finally, we deploy the learned policy on the real robot. When failures are observed, we expand the simulated distribution in a targeted way (e.g., larger perturbations in approach angle, end-effector pose, fruit compliance, or friction) and repeat Stage II IV-A2, prioritizing coverage of configurations that are difficult to obtain from teleoperation alone. The details of policy learning are provided in Sec. VI.

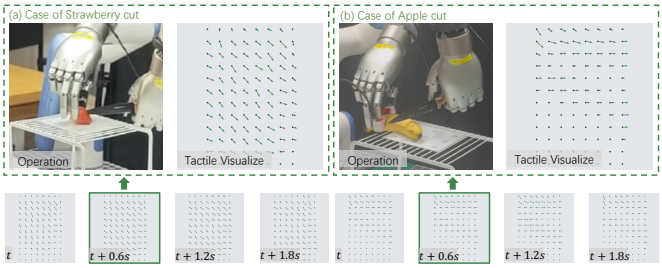

Fig. 3: Tactile perception visualiztion of the food examples.

B. Tactile simulation while cutting

The right-hand data generation has two components. (1) Pre-Grasp: we reuse expert grasp priors [94] to generate a kinematically feasible hand pose that encloses the strawberry. (2) Tactile adaptation: the hand control policy can handle deformation and contact during cutting.

We implement tactile sensing in our differentiable MPM-based simulator [85]. Concretely, for each fingertip (or tactile pad), we build a SDF in the sensor's local frame. At each simulation step, we detect particles that enter the near-surface band of the SDF. Using their positions and velocities, we estimate contact events and compute contact impulses/forces by combining (i) penetration depth and (ii)relative tangential velocity between particles and the sensor surface. We then ag-

gregate these per-particle contributions into a compact tactile observation by spatially binning forces over the sensor surface.

To generate tactile adaptation data, we simulate the full grasp-and-cut sequence starting from the initial grasp. When tactile signals indicate incipient slip , we apply small corrective finger motions to maintain stable contact while allowing compliant deformation. We record these state–action pairs as supervision for the second-stage right-hand policy. To bridge sim-to-real, we calibrate a lightweight mapping from simulated forces to sensor readings (scale/offset and noise models), improving robustness to contact variations.

C. Scene level and hardware registration

We register the canonical object–knife trajectory into the real-world hardware setting. Specifically, we estimate the fruit pose in the world frame, compute the rigid transform from canonical to world coordinates, and apply it to the nominal knife trajectory. The resulting world-frame end-effector targets are converted into robot joint trajectories using inverse kinematics and time-parameterization, while enforcing joint limits, velocity and acceleration limits, and end-effector orientation constraints. For bi-manual execution, we synchronize both arms through a shared timeline so that the stabilizing grasp precedes cutting and maintains contact throughout the interaction.To ensure safety and feasibility, we perform collision-checking and trajectory filtering using fast motion planning libraries [5, 66]. We reject trajectories that violate self-collision constraints, collide with the environment (table, branches, etc.), or approach kinematic singularities.Finally, we bridge residual sim2real discrepancies with targeted randomization at fruit pose noise. This complements the object–knife canonical planning, and improves robustness when transferring the learned policy to different robot embodiments .

V. Continuation on a Shape Manifold for Data Augmentation

We model each food category as a continuous family of shapes induced by a learned geometric prior. Starting from a nominal real instance for which a cutting motion succeeds, we densify training data by (i) sampling nearby instances on the category shape family and (ii) transferring the nominal motion to these nearby instances via local continuation (small corrective adjustments that preserve feasibility).

a) Data augmentation pipeline: Fig. 1 summarizes the full pipeline. Given simulated observations or real-captured references, we first infer the object category using a detector. A summarization module then extracts attribute-level descriptors (e.g., coarse shape, surface characteristics) and encodes them into a structured semantic representation designed to remain consistent across domains. Conditioned on the predicted category and these attributes, a generative module synthesizes diverse yet physically plausible instances that remain within the target domain distribution. This knowledge-guided generation enables controlled variation in geometry and appearance while preserving task-relevant physical properties, yielding large-scale, domain-consistent data densification. The resulting in-domain textual descriptions are used to generate controllable images, which are then lifted to 3D assets (including 3D Gaussian Splatting (3DGS) representations and meshes). Additional implementation details are provided in the supplementary material. This pipeline induces a learned shape prior and yields a nominal reconstructed instance, which we formalize next.

b) Shape generation and local manifold structure: Let $\mathcal{Z} \subset \mathbb{R}^{d_z}$ denote the latent space and let

$$\mathcal{G} : \mathcal{Z} \to \mathcal{M}, \qquad z \mapsto \mathcal{G}(z), \tag{24}$$

be an implicit geometry generator, where $\mathcal{M}$ is the family of physically plausible food geometries represented by the implicit model. We use the term *shape manifold* in the local sense that, under an appropriate geometry metric $d_{\mathcal{M}}(\cdot,\cdot)$, the generator is locally regular: small perturbations in z induce small changes in geometry.[1] Given a reconstructed real instance, we obtain a nominal latent code z_0 and geometry $\mathcal{G}(z_0)$. We densify the training distribution by sampling z within a neighborhood of z_0, thereby producing a continuum of related instances (Fig. 4).

c) Local regularity and analytical reliability: Our continuation argument relies on local smoothness of the deterministic mapping from deployable visual observations to actions, $f(\mathbf{o}^{vis})$ (cf. (9)). Assuming f is differentiable in a neighborhood of a successful anchor observation $o^\star$, a first-order expansion gives

$$f(o) \;=\; f(o^\star) \;+\; J_f(o^\star)(o - o^\star) \;+\; r(o), \tag{25}$$

where $J_f(o^\star)$ is the Jacobian and the remainder satisfies $\|r(o)\| = o(\|o - o^\star\|)$ as $o \to o^\star$. Equivalently, by the mean-value theorem, there exists a neighborhood $\mathcal{N}(o^\star)$ and a constant L_f such that $\|f(o) - f(o^\star)\| \le L_f \|o - o^\star\|$ for all $o \in \mathcal{N}(o^\star)$. This formalizes the key intuition: small observation/instance perturbations admit proportionally small action corrections, which is precisely the regime in which continuation is numerically stable and analytically justified.

d) Neighborhood of demonstrations: Let $\mathcal{D}_0 = \{(o_i, a_i)\}_{i=1}^{N_0}$ denote deployable observation–action pairs collected from a teleoperated episode. We evaluate candidate actions in simulation using a binary feasibility predicate

$$S(o, a) \in 0, 1, \tag{26}$$

which indicates whether the action succeeds under both local (object-level) and global (scene-level) constraints, i.e., it passes the cutting and collision tests.

We capture this neighborhood by fitting a local corrective policy Γ_{θ_i} around each anchor:

$$\theta_i^\star \;=\; \arg\max_{\theta} \; \mathbb{E}_{o\sim\mu_i}\Big[S\big(o, \Gamma_\theta(o)\big) \;-\; \beta\, \|\Gamma_\theta(o) - a_i\|^2\Big], \tag{27}$$

where μ_i is a distribution over observations induced by (i) small state perturbations, (ii) physics randomization, and (iii) local shape variation $z \sim \nu_i$ through (24). The quadratic term

[1] Concretely, it suffices that $\mathcal{G}$ is locally Lipschitz: for z in a neighborhood of z_0, $d_{\mathcal{M}}(\mathcal{G}(z), \mathcal{G}(z_0)) \le L_{\mathcal{G}} \|z - z_0\|$.

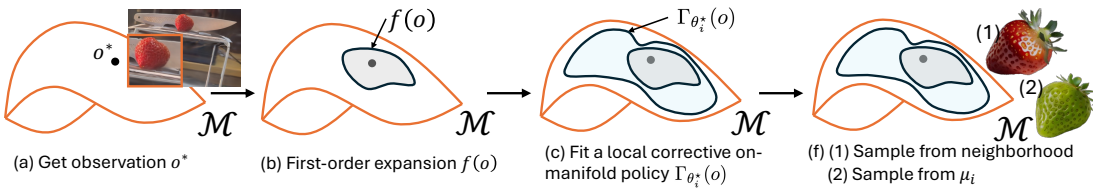

Fig. 4: On-manifold approximation of real-data neighborhood for data augmentation pipeline.

anchors the correction to the demonstrated action a_i, while allowing the policy to compensate for mismatches introduced by deformation and partial separation during cutting. In this sense, $\Gamma_{\theta_i^\star}$ acts as a *continuation operator*: it tracks a successful behavior as instance parameters vary locally along the learned shape family, while remaining close to the nominal action to ensure numerical stability.

e) Augmented data generation: We generate additional samples by rolling out the local corrective policies in simulation and retaining only successful outcomes:

$$\mathcal{D}_{\text{aug}} = \mathcal{D}_0 \cup \bigcup_{i=1}^{N_0} \Big\{ (o,a) \;:\; o \sim \mu_i,\; a = \Gamma_{\theta_i^\star}(o),\; S(o,a) = 1 \Big\}. \tag{28}$$

Compared to naive perturbation, continuation concentrates samples on regions that remain physically feasible under geometry variation and cutting-induced topology changes.

A single teleoperated episode captures only a narrow slice of the skill (one instance, one grasp, one approach). We therefore use simulation as a data densification engine (Fig. 4) to expand this slice into a broad set of consistent rollouts over diverse instances sampled from the category prior, while restricting observations to deployable sensing modalities.

VI. Learning Model

We train an ACT-style model [95] on the augmented dataset. The policy consumes multimodal observations and predicts action chunks executed open-loop for a brief horizon, then replans at chunk boundaries. Cutting is a long-horizon, contact-rich manipulation task that requires (i) object awareness under occlusion, (ii) precise end-effector/knife placement, and (iii) contact regulation via tactile feedback. We adopt three design choices for stable training and deployment.

A. Model insight

At the start of each trial, we treat the left hand, left arm, and right arm states as fixed context. The right-hand policy conditions on the left hand and both arms, and additionally uses tactile observations to stabilize the hold and regulate interaction during cutting. Observations include the segmented object point cloud from a calibrated fixed-view RGBD camera, along with proprioceptive and tactile signals. We (i) register the object point cloud over time to reduce viewpoint drift, and (ii) temporally synchronize tactile signals with the visual/proprioceptive streams so contact events are aligned. This simplifies learning for contact-rich behaviors.

Using a single action head for both arm motion and hand/knife interaction is unstable, so we predict separately:

Training regime	Food	Progress ↑	Success ↑	Integrity ↑
Sim-only	Strawberry	10/25	6/25	10/25
	Apple	5/25	4/25	5/25
	Banana	16/25	12/25	14/25
	Potato	15/25	**12/25**	11/25
	Avg.	46/100	34/100	40/100
Real-only	Strawberry	12/25	8/25	10/25
	Apple	6/25	4/25	4/25
	Banana	17/25	12/25	14/25
	Potato	9/25	7/25	9/25
	Avg.	44/100	31/100	37/100
Sim+Real (Ours)	Strawberry	**19/25**	**12/25**	**14/25**
	Apple	**14/25**	**8/25**	**7/25**
	Banana	**24/25**	**18/25**	**23/25**
	Potato	**19/25**	**12/25**	**16/25**
	Avg.	**76/100**	**50/100**	**60/100**

TABLE I: Sim-to-real gap and the effect of mixing simulation and real demonstrations (25 trials per food).

- **Arm:** global geometric motion (approach and pose alignment relative to the object).
- **Hand/knife:** local contact interaction (grasp stability and cutting dynamics).

We also process tactile signals as *effector-specific token groups* (tied to the hand/knife they come from), which improves robustness to brief, localized tactile transients [76].

Let o_t denote the aligned observation at time t (point cloud, proprioception, tactile). The policy predicts arm and hand action chunks over a stage-dependent horizon H_t:

$$\pi_\theta\big(a^{\text{arm}}_{t:t+H_t},\, a^{\text{hand}}_{t:t+H_t} \mid o_t\big)\,. \tag{29}$$

B. Adaptive action chunking for Sim2Real deployment

We use variable action chunking as a post-training Sim2Real strategy to improve robustness to execution-rate mismatch and contact timing uncertainty. Following [64, 77], we execute long chunks for coarse free-space motion and short chunks with frequent re-planning near contact.

From demonstrations, we annotate two levels of temporal anchors: (i) key checkpoints that mark stage transitions (e.g., grasp and contact initiated), and (ii) subkey steps within a stage that correlate with perceptual change. These anchors define a discrete stage token $z_t \in \mathcal{Z}$ and a nominal per-stage chunk horizon $\bar{H}(z_t)$, with larger $\bar{H}(z)$ for long-horizen stages and smaller $\bar{H}(z)$ for contact-sensitive stages. We sample the

chunk length around the nominal horizon:

$$H_t \sim p(H \mid z_t) = \lambda \,\mathrm{Unif}\big(H_{\min}(z_t), \bar{H}(z_t)\big) + (1-\lambda)\,\mathrm{Unif}\big(\bar{H}(z_t), H_{\max}(z_t)\big). \tag{30}$$

The first component performs interpolation, which helps when real contact occurs earlier than in simulation. The second performs extrapolation (longer-than-nominal chunks, more open-loop execution), improving tolerance to slower dynamics and perception latency. We set $H_{\min}(z)$ smaller for contact-heavy stages and allow larger $H_{\max}(z)$ for free-space motion.During post-training, we supervise the expert action chunk of length H_t for the current stage z_t. At inference, we execute H_t steps open-loop:

$$\mathbf{a}_{t:t+H_t-1} \leftarrow \pi_\theta(\mathbf{o}_t, z_t), \qquad \text{replan from } \mathbf{o}_{t+H_t}. \tag{31}$$

In the real system, the policy conditions on the segmented strawberry point cloud, proprioceptive and tactile signals. We use three stages with nominal horizons: (i) **approach & grasp** ($\bar{H}$=64), (ii) **knife positioning** ($\bar{H}$=128), and (iii) **cutting with tactile feedback** ($\bar{H}$=16).

VII. Experimental Evaluation

This section evaluates (i) whether the learned cutting policy transfers to real-world food setups under real sensing and contact dynamics, and (ii) whether simulation-based data densification improves generalization to novel instances. Standard manipulation benchmarks (e.g., [47, 37, 87]) do not capture key characteristics of food cutting, including deformability, contact-rich tool use, and long-horizon execution with intermittent sticking and slip. We therefore conduct experiments on a real food-halving task and deploy the policy on-robot under real contact conditions.

We structure the evaluation around two questions:

1) **Sim-to-real transfer:** Can the policy execute meaningful cutting behaviors in real-world setups?
2) **Generalization:** Does simulation-based data densification improve performance, particularly on unseen instances within the same food category?

A. Task setup and evaluation protocol

We train and evaluate 4 food types: **strawberry, apple, banana, potato**. In the training, for each instance, we adopt 5 demonstrations for real-world and 50 sequences from simulation. In supplementary, we show the results of training-set scaling-up. For each type, we select 25 instances per category for evaluation. Natural variation in geometry and physical properties is treated as uncontrolled but bounded variability representative of real deployment. We consider a food-halving task following [78], where a robot must cut a food item into two halves without causing crumbling. A trial is successful only if the knife fully traverses the object and the halves can be cleanly separated. To assess robustness to shifts in initial conditions, we randomize object placement across trials. Specifically, the food item is placed on the table with a random planar offset relative to the robot base frame within a predefined reachable workspace region. We do not pre-assume or pre-calibrate strawberry shape.

a) Metrics: We report 3 metrics, each measured as the number of trials satisfying the criterion out of 25:

- **Progress (↑):** whether the policy can approach the food with the knife and reach a valid pre-cut state, i.e., the knife successfully reaches the target region and establishes an initial cutting contact with the surface.
- **Success (↑):** whether the knife *cuts through* the food, i.e., the cut fully traverses the item, yielding two separable halves and ending with a safe termination/retreat.
- **Integrity (↑):** whether the food remains in acceptable condition after the attempt.

B. Sim-to-real gap and the effect of mixing sim and real data

To quantify the sim2real gap and evaluate the benefit of densification (Q 1–2), we train policies under 3 data regimes:

1) **Sim-only:** simulated cutting trajectories only.
2) **Real-only:** teleoperated real demonstrations only.
3) **Sim+Real (Augmented):** real demonstrations combined with augmented simulation data.

All policies are evaluated on the same held-out real-world setups, which controls for evaluation conditions and isolates the effect of training data. This protocol also directly tests whether densification improves within-category generalization to unseen instances. In total, we conduct 300 real-world experiments across different data regimes and different instances.

As shown in Tab. I, policies trained on either sim-only or real-only data generalize poorly to unseen real-world setups. Aggregated over 100 real trials (4 foods × 25 instances), Sim-only achieves 46% progress, 34% success, and 40% integrity, while Real-only achieves 44% progress, 31% success, and 37% integrity. In contrast, Sim+Real (Augmented) achieves the strongest performance, reaching 76% progress, 50% success, and 60% integrity. Overall, co-training with densified simulation improves progress, success, and integrity compared to the two single-source baselines, indicating that our augmentation pipeline mitigates the sim-to-real gap and improves within-category generalization. Additional experimental results are provided in the supplementary material.

VIII. Conclusion

We presented DexNinja, a differentiable real-sim-real data augmentation framework for dexterous food cutting. From a small set of teleoperated demonstrations, DexNinja reconstructs and registers object assets with multi-modal sensing, estimates physically interpretable parameters, and expands the training distribution via physics-aware randomization and knowledge-guided instance generation. A custom differentiable simulator couples robot dynamics, tactile contact, and deformable cutting for scalable rollouts and gradient-based fitting, while policies use an ACT-style transformer with separated arm/hand control and adaptive action chunking for robust execution. Experiments on varies instance indicate that sim-only or real-only training is insufficient: combining real demonstrations with DexNinja's augmented pipeline improves robustness to unseen shapes and sizes.

References

[1] Christopher Agia, Rohan Sinha, Jingyun Yang, Rika Antonova, Marco Pavone, Haruki Nishimura, Masha Itkina, and Jeannette Bohg. Cupid: Curating data your robot loves with influence functions, 2025. URL https://arxiv.org/abs/2506.19121.

[2] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[3] Joao Pedro Araujo, Yanjie Ze, Pei Xu, Jiajun Wu, and C. Karen Liu. Retargeting matters: General motion retargeting for humanoid motion tracking. *arXiv preprint arXiv:2510.02252*, 2025.

[4] Martin Asenov, Michael Burke, Daniel Angelov, Todor Davchev, Kartic Subr, and Subramanian Ramamoorthy. Vid2param: Modelling of dynamics parameters from video, 2020. URL https://arxiv.org/abs/1907.06422.

[5] Genesis Authors. Genesis: A universal and generative physics engine for robotics and beyond, December 2024. URL https://github.com/Genesis-Embodied-AI/Genesis.

[6] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. *CVPR*, 2023.

[7] Peter Yichen Chen, Chao Liu, Pingchuan Ma, John Eastman, Daniela Rus, Dylan Randle, Yuri Ivanov, and Wojciech Matusik. Learning object properties using robot proprioception via differentiable robot-object interaction. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5997–6004, 2025. doi: 10.1109/ICRA55743.2025.11127955.

[8] Sirui Chen, Jeannette Bohg, and C. Karen Liu. Springgrasp: Synthesizing compliant, dexterous grasps under shape uncertainty, 2024. URL https://arxiv.org/abs/2404.13532.

[9] Siwei Chen, Yiqing Xu, Cunjun Yu, Linfeng Li, Xiao Ma, Zhongwen Xu, and David Hsu. Daxbench: Benchmarking deformable object manipulation with differentiable physics. *arXiv preprint arXiv:2210.13066*, 2022.

[10] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[11] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[12] Filipe de Avila Belbute-Peres, Kevin Smith, Kelsey Allen, Josh Tenenbaum, and J. Zico Kolter. End-to-end differentiable physics for learning and control. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[13] Hongwei Fan, Hang Dai, Jiyao Zhang, Jinzhou Li, Qiyang Yan, Yujie Zhao, Mingju Gao, Jinghang Wu, Hao Tang, and Hao Dong. Twinaligner: Visual-dynamic alignment empowers physics-aware real2sim2real for robotic manipulation, 2025. URL https://arxiv.org/abs/2512.19390.

[14] Haoran Geng, Feishi Wang, Songlin Wei, Yuyang Li, Bangjun Wang, Boshi An, Charlie Tianyue Cheng, Haozhe Lou, Peihao Li, Yen-Jen Wang, et al. Roboverse: Towards a unified platform, dataset and benchmark for scalable and generalizable robot learning. *arXiv preprint arXiv:2504.18904*, 2025.

[15] Irmak Guzey, Yinlong Dai, Georgy Savva, Raunaq Bhirangi, and Lerrel Pinto. Bridging the human to robot dexterity gap through object-oriented rewards. *arXiv preprint arXiv:2410.23289*, 2024.

[16] Lijun Han, Hesheng Wang, Zhe Liu, Weidong Chen, and Xiufeng Zhang. Vision-based cutting control of deformable objects with surface tracking. *IEEE/ASME Transactions on Mechatronics*, 26(4):2016–2026, 2020.

[17] 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.

[18] Liang Heng, Haoran Geng, Kaifeng Zhang, Pieter Abbeel, and Jitendra Malik. Vitacformer: Learning cross-modal representation for visuo-tactile dexterous manipulation, 2025. URL https://arxiv.org/abs/2506.15953.

[19] Yixin Hu, Teseo Schneider, Bolun Wang, Denis Zorin, and Daniele Panozzo. Fast tetrahedral meshing in the wild. *ACM Trans. Graph.*, 39(4), July 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392385. URL https://doi.org/10.1145/3386569.3392385.

[20] 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 Trans. Graph.*, 37(4), July 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201293. URL https://doi.org/10.1145/3197517.3201293.

[21] Jialei Huang, Yang Ye, Yuanqing Gong, Xuezhou Zhu, Yang Gao, and Kaifeng Zhang. Spatially anchored tactile awareness for robust dexterous manipulation, 2025. URL https://arxiv.org/abs/2510.14647.

[22] Wenlong Huang, Yu-Wei Chao, Arsalan Mousavian, Ming-Yu Liu, Dieter Fox, Kaichun Mo, and Li Fei-Fei. Pointworld: Scaling 3d world models for in-the-wild robotic manipulation. *arXiv preprint arXiv:2601.03782*,

2026.

[23] Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $pi_{0.6}$: a vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

[24] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $pi_{0.5}$:a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[25] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *Acm siggraph 2016 courses*, pages 1–52. 2016.

[26] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses*, SIGGRAPH '16, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342896. doi: 10.1145/2897826.2927348. URL https://doi.org/10.1145/2897826.2927348.

[27] Guangqi Jiang, Haoran Chang, Ri-Zhao Qiu, Yutong Liang, Mazeyu Ji, Jiyue Zhu, Zhao Dong, Xueyan Zou, and Xiaolong Wang. Gsworld: Closed-loop photo-realistic simulation suite for robotic manipulation. *arXiv preprint arXiv:2510.20813*, 2025.

[28] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

[29] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[30] Sizhe Lester Li, Annan Zhang, Boyuan Chen, Hanna Matusik, Chao Liu, Daniela Rus, and Vincent Sitzmann. Controlling diverse robots by inferring jacobian fields with deep networks. *Nature*, Jun 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09170-0. URL https://doi.org/10.1038/s41586-025-09170-0.

[31] Wenxuan Li, Hang Zhao, Zhiyuan Yu, Yu Du, Qin Zou, Ruizhen Hu, and Kai Xu. Pin-wm: Learning physics-informed world models for non-prehensile manipulation, 2025. URL https://arxiv.org/abs/2504.16693.

[32] Yue Li, Qi Ma, Runyi Yang, Huapeng Li, Mengjiao Ma, Bin Ren, Nikola Popovic, Nicu Sebe, Ender Konukoglu, Theo Gevers, et al. Scenesplat: Gaussian splatting-based scene understanding with vision-language pretraining. *arXiv preprint arXiv:2503.18052*, 2025.

[33] Yue Li, Qi Ma, Runyi Yang, Mengjiao Ma, Bin Ren, Nikola Popovic, Nicu Sebe, Theo Gevers, Luc Van Gool, Danda Pani Paudel, et al. Chorus: Multi-teacher pretraining for holistic 3d gaussian scene encoding. *arXiv preprint arXiv:2512.17817*, 2025.

[34] Yuyang Li, Wenxin Du, Chang Yu, Puhao Li, Zihang Zhao, Tengyu Liu, Chenfanfu Jiang, Yixin Zhu, and Siyuan Huang. Taccel: Scaling up vision-based tactile robotics via high-performance gpu simulation. *arXiv preprint arXiv:2504.12908*, 2025.

[35] Toru Lin, Yu Zhang, Qiyang Li, Haozhi Qi, Brent Yi, Sergey Levine, and Jitendra Malik. Learning visuotactile skills with two multifingered hands, 2024. URL https://arxiv.org/abs/2404.16823.

[36] 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, 2022. URL https://arxiv.org/abs/2203.17275.

[37] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36: 44776–44791, 2023.

[38] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation, 2025. URL https://arxiv.org/abs/2410.07864.

[39] Haozhe Lou, Yurong Liu, Yike Pan, Yiran Geng, Jianteng Chen, Wenlong Ma, Chenglong Li, Lin Wang, Hengzhen Feng, Lu Shi, Liyi Luo, and Yongliang Shi. Robo-gs: A physics consistent spatial-temporal model for robotic arm with hybrid representation, 2024. URL https://arxiv.org/abs/2408.14873.

[40] Haozhe Lou, Mingtong Zhang, Haoran Geng, Hanyang Zhou, Sicheng He, Zhiyuan Gao, Siheng Zhao, Jiageng Mao, Pieter Abbeel, Jitendra Malik, Daniel Seita, and Yue Wang. Dream: Differentiable real-to-sim-to-real engine for learning robotic manipulation. In *3rd RSS Workshop on Dexterous Manipulation: Learning and Control with Diverse Data*, June 2025. URL https://openreview.net/forum?id=S0FmCZ6by5. OpenReview submission (Dex-RSS-25), published June 25, 2025.

[41] Mengjiao Ma, Qi Ma, Yue Li, Jiahuan Cheng, Runyi Yang, Bin Ren, Nikola Popovic, Mingqiang Wei, Nicu Sebe, Luc Van Gool, et al. Scenesplat++: A large dataset and comprehensive benchmark for language gaussian splatting. In *NeurIPS*, 2025.

[42] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[43] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. *arXiv preprint arXiv:2310.17596*, 2023.

[44] Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Ritvik

Singh, Yunrong Guo, Hammad Mazhar, Ajay Mandlekar, Buck Babich, Gavriel State, Marco Hutter, and Animesh Garg. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, 2023. doi: 10.1109/LRA.2023.3270034.

[45] 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.

[46] Xiaoqian Mu, Yuechuan Xue, and Yan-Bin Jia. Dexterous robotic cutting based on fracture mechanics and force control. *IEEE Transactions on Automation Science and Engineering*, 21(4):5198–5215, 2024. doi: 10.1109/TASE.2023.3309784.

[47] Yao Mu, Tianxing Chen, Shijia Peng, Zanxin Chen, Zeyu Gao, Yude Zou, Lunkai Lin, Zhiqiang Xie, and Ping Luo. Robotwin: Dual-arm robot benchmark with generative digital twins (early version). In *European Conference on Computer Vision*, pages 264–273. Springer, 2024.

[48] Masaki Murooka, Takahiro Hoshi, Kensuke Fukumitsu, Shimpei Masuda, Marwan Hamze, Tomoya Sasaki, Mitsuharu Morisawa, and Eiichi Yoshida. Tact: Humanoid whole-body contact manipulation through deep imitation learning with tactile modality. *IEEE Robotics and Automation Letters*, 2025.

[49] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[50] Austin Patel, Andrew Wang, Ilija Radosavovic, and Jitendra Malik. Learning to imitate object interactions from internet videos. *arXiv:2211.13225*, 2022.

[51] Nicholas Pfaff, Evelyn Fu, Jeremy Binagia, Phillip Isola, and Russ Tedrake. Scalable real2sim: Physics-aware asset generation via robotic pick-and-place setups, 2025. URL https://arxiv.org/abs/2503.00370.

[52] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-Hand Object Rotation via Rapid Motor Adaptation. In *Conference on Robot Learning (CoRL)*, 2022.

[53] Haozhi Qi, Brent Yi, Sudharshan Suresh, Mike Lambeta, Yi Ma, Roberto Calandra, and Jitendra Malik. General in-hand object rotation with vision and touch. In *Conference on Robot Learning*, pages 2549–2564. PMLR, 2023.

[54] Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dieter Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. In *Robotics: Science and Systems*, 2023.

[55] M Nomaan Qureshi, Sparsh Garg, Francisco Yandun, David Held, George Kantor, and Abhisesh Silwal. Splatsim: Zero-shot sim2real transfer of rgb manipulation policies using gaussian splatting. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6502–6509. IEEE, 2025.

[56] Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. In *Conference on Robot Learning*, pages 416–426. PMLR, 2023.

[57] D Rico, Ana Belen Martin-Diana, JM Barat, and C Barry-Ryan. Extending and measuring the quality of fresh-cut fruit and vegetables: a review. *Trends in Food Science & Technology*, 18(7):373–386, 2007.

[58] Bipasha Sen, Michelle Wang, Nandini Thakur, Aditya Agarwal, and Pulkit Agrawal. Learning to look around: Enhancing teleoperation and learning with a human-like actuated neck, 2024. URL https://arxiv.org/abs/2411.00704.

[59] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. Videodex: Learning dexterity from internet videos, 2022. URL https://arxiv.org/abs/2212.04498.

[60] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. Videodex: Learning dexterity from internet videos. In *Conference on Robot Learning*, pages 654–665. PMLR, 2023.

[61] Haochen Shi, Huazhe Xu, Zhiao Huang, Yunzhu Li, and Jiajun Wu. Robocraft: Learning to see, simulate, and shape elasto-plastic objects with graph networks. *arXiv preprint arXiv:2205.02909*, 2022.

[62] Haochen Shi, Huazhe Xu, Samuel Clarke, Yunzhu Li, and Jiajun Wu. Robocook: Long-horizon elasto-plastic object manipulation with diverse tools. *arXiv preprint arXiv:2306.14447*, 2023.

[63] Zilin Si, Gu Zhang, Qingwei Ben, Branden Romero, Zhou Xian, Chao Liu, and Chuang Gan. Difftactile: A physics-based differentiable tactile simulator for contact-rich robotic manipulation, 2024. URL https://arxiv.org/abs/2403.08716.

[64] Junhyuk So, Chiwoong Lee, Shinyoung Lee, Jungseul Ok, and Eunhyeok Park. Improving generative behavior cloning via self-guidance and adaptive chunking, 2025. URL https://arxiv.org/abs/2510.12392.

[65] Trevor Standley, Ozan Sener, Dawn Chen, and Silvio Savarese. image2mass: Estimating the mass of an object from its image. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 324–333. PMLR, 13–15 Nov 2017. URL https://proceedings.mlr.press/v78/standley17a.html.

[66] Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, et al. curobo: Parallelized collision-free minimum-jerk robot motion generation. *arXiv preprint arXiv:2310.17274*, 2023.

[67] Priya Sundaresan, Rika Antonova, and Jeannette Bohgl. Diffcloud: Real-to-sim from point clouds with

differentiable simulation and rendering of deformable objects. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10828–10835. IEEE, 2022.

[68] GigaWorld Team, Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Haoyun Li, Jiagang Zhu, Kerui Li, Mengyuan Xu, Qiuping Deng, Siting Wang, Wenkang Qin, Xinze Chen, Xiaofeng Wang, Yankai Wang, Yu Cao, Yifan Chang, Yuan Xu, Yun Ye, Yang Wang, Yukun Zhou, Zhengyuan Zhang, Zhehao Dong, and Zheng Zhu. Gigaworld-0: World models as data engine to empower embodied ai, 2025. URL https://arxiv.org/abs/2511.19861.

[69] Dylan Turpin, Liquan Wang, Eric Heiden, Yun-Chun Chen, Miles Macklin, Stavros Tsogkas, Sven Dickinson, and Animesh Garg. Grasp'd: Differentiable contact-rich grasp synthesis for multi-fingered hands, 2022. URL https://arxiv.org/abs/2208.12250.

[70] Luobin Wang, Runlin Guo, Quan Vuong, Yuzhe Qin, Hao Su, and Henrik Christensen. A real2sim2real method for robust object grasping with neural surface reconstruction. In *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*, pages 1–8, 2023. doi: 10.1109/CASE56687.2023.10260554.

[71] Luobin Wang, Runlin Guo, Quan Vuong, Yuzhe Qin, Hao Su, and Henrik Christensen. A real2sim2real method for robust object grasping with neural surface reconstruction, 2023. URL https://arxiv.org/abs/2210.02685.

[72] Ruicheng Wang, Jialiang Zhang, Jiayi Chen, Yinzhen Xu, Puhao Li, Tengyu Liu, and He Wang. Dexgraspnet: A large-scale robotic dexterous grasp dataset for general objects based on simulation. *arXiv preprint arXiv:2210.02697*, 2022.

[73] Yi Wang, Yan Yang, Hongmei Zhao, Bo Liu, Jitong Ma, Yu He, Yitan Zhang, and Hongbin Xu. Effects of cutting parameters on cutting of citrus fruit stems. *Biosystems Engineering*, 193:1–11, 2020.

[74] Zhenyu Wei, Zhixuan Xu, Jingxiang Guo, Yiwen Hou, Chongkai Gao, Zhehao Cai, Jiayu Luo, and Lin Shao. $\mathcal{D}(\mathcal{R},\mathcal{O})$ grasp: A unified representation of robot and object interaction for cross-embodiment dexterous grasping, 2025. URL https://arxiv.org/abs/2410.01702.

[75] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. FoundationPose: Unified 6d pose estimation and tracking of novel objects. In *CVPR*, 2024.

[76] Ruoshi Wen, Guangzeng Chen, Zhongren Cui, Min Du, Yang Gou, Zhigang Han, Liqun Huang, Mingyu Lei, Yunfei Li, Zhuohang Li, Wenlei Liu, Yuxiao Liu, Xiao Ma, Hao Niu, Yutao Ouyang, Zeyu Ren, Haixin Shi, Wei Xu, Haoxiang Zhang, Jiajun Zhang, Xiao Zhang, Liwei Zheng, Weiheng Zhong, Yifei Zhou, Zhengming Zhu, and Hang Li. Gr-dexter technical report, 2026. URL https://arxiv.org/abs/2512.24210.

[77] Yueyang Weng, Xiaopeng Zhang, Yongjin Mu, Yingcong Zhu, Yanjie Li, and Qi Liu. Temporal action selection for action chunking, 2025. URL https://arxiv.org/abs/2511.04421.

[78] Lasitha Wijayarathne, Ziyi Zhou, Ye Zhao, and Frank L Hammond. Real-time deformable-contact-aware model predictive control for force-modulated manipulation. *IEEE Transactions on Robotics*, 39(5):3549–3566, 2023.

[79] 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.

[80] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. *arXiv preprint arXiv:2412.01506*, 2024.

[81] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. *arXiv preprint arXiv:2311.12198*, 2023.

[82] 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.

[83] Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. *arXiv preprint arXiv:2407.15208*, 2024.

[84] Xiuwei Xu, Angyuan Ma, Hankun Li, Bingyao Yu, Zheng Zhu, Jie Zhou, and Jiwen Lu. R2rgen: Real-to-real 3d data generation for spatially generalized manipulation. *arXiv preprint arXiv:2510.08547*, 2025.

[85] Zhenjia Xu, Zhou Xian, Xingyu Lin, Cheng Chi, Zhiao Huang, Chuang Gan, and Shuran Song. Roboninja: Learning an adaptive cutting policy for multi-material objects. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[86] Zhengrong Xue, Shuying Deng, Zhenyang Chen, Yixuan Wang, Zhecheng Yuan, and Huazhe Xu. Demogen: Synthetic demonstration generation for data-efficient visuomotor policy learning. *arXiv preprint arXiv:2502.16932*, 2025.

[87] Adina Yakefu, Bin Xie, Chongyang Xu, Enwen Zhang, Erjin Zhou, Fan Jia, Haitao Yang, Haoqiang Fan, Haowei Zhang, Hongyang Peng, et al. Robochallenge: Large-scale real-robot evaluation of embodied policies. *arXiv preprint arXiv:2510.17950*, 2025.

[88] Sizhe Yang, Wenye Yu, Jia Zeng, Jun Lv, Kerui Ren, Cewu Lu, Dahua Lin, and Jiangmiao Pang. Novel demonstration generation with gaussian splatting enables robust one-shot manipulation, 2025. URL https://arxiv.org/abs/2504.13175.

[89] Chongjie Ye, Yinyu Nie, Jiahao Chang, Yuantao Chen,

Yihao Zhi, and Xiaoguang Han. Gaustudio: A modular framework for 3d gaussian splatting and beyond, 2024. URL https://arxiv.org/abs/2403.19632.

[90] Justin Yu, Letian Fu, Huang Huang, Karim El-Refai, Rares Andrei Ambrus, Richard Cheng, Muhammad Zubair Irshad, and Ken Goldberg. Real2render2real: Scaling robot data without dynamics simulation or robot hardware. *arXiv preprint arXiv:2505.09601*, 2025.

[91] Yanjie Ze, Zixuan Chen, João Pedro Araújo, Zi ang Cao, Xue Bin Peng, Jiajun Wu, and C. Karen Liu. Twist: Teleoperated whole-body imitation system. *arXiv preprint arXiv:2505.02833*, 2025.

[92] Yanjie Ze, Siheng Zhao, Weizhuo Wang, Angjoo Kanazawa, Rocky Duan, Pieter Abbeel, Guanya Shi, and Jiajun Wuand C. Karen Liu. Twist2: Scalable, portable, and holistic humanoid data collection system. *arXiv preprint arXiv:2511.02832*, 2025.

[93] Hui Zhang, Zijian Wu, Linyi Huang, Sammy Christen, and Jie Song. Robustdexgrasp: Robust dexterous grasping of general objects, 2025. URL https://arxiv.org/abs/2504.05287.

[94] Jialiang Zhang, Haoran Liu, Danshi Li, Xinqiang Yu, Haoran Geng, Yufei Ding, Jiayi Chen, and He Wang. Dexgraspnet 2.0: Learning generative dexterous grasping in large-scale synthetic cluttered scenes, 2024. URL https://arxiv.org/abs/2410.23004.

[95] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware, 2023. URL https://arxiv.org/abs/2304.13705.

[96] Yujie Zhao, Hongwei Fan, Di Chen, Shengcong Chen, Liliang Chen, Xiaoqi Li, Guanghui Ren, and Hao Dong. Real2edit2real: Generating robotic demonstrations via a 3d control interface, 2025. URL https://arxiv.org/abs/2512.19402.

[97] Zhaxizhuom Zhaxizhuoma, Kehui Liu, Chuyue Guan, Zhongjie Jia, Ziniu Wu, Xin Liu, Tianyu Wang, Shuai Liang, Pengan CHEN, Pingrui Zhang, Haoming Song, Delin Qu, Dong Wang, Zhigang Wang, Nieqing Cao, Yan Ding, Bin Zhao, and Xuelong Li. Fastumi: A scalable and hardware-independent universal manipulation interface with dataset. In Joseph Lim, Shuran Song, and Hae-Won Park, editors, *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 3069–3093. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/zhaxizhuoma25a.html.

[98] Dongzhe Zheng, Siqiong Yao, Wenqiang Xu, and Cewu Lu. Differentiable cloth parameter identification and state estimation in manipulation. *IEEE Robotics and Automation Letters*, 9(3):2519–2526, 2024.

[99] Yuhang Zheng, Xiangyu Chen, Yupeng Zheng, Songen Gu, Runyi Yang, Bu Jin, Pengfei Li, Chengliang Zhong, Zengmao Wang, Lina Liu, et al. Gaussiangrasper: 3d language gaussian splatting for open-vocabulary robotic grasping. *arXiv preprint arXiv:2403.09637*, 2024.

[100] Yifan Zhu, Tianyi Xiang, Aaron M Dollar, and Zherong Pan. One-shot real-to-sim via end-to-end differentiable simulation and rendering. *IEEE Robotics and Automation Letters*, 2025.

TABLE II: Baseline comparison between RoboNinja and our method on four objects. We report successes over 10 trials per object; **Avg.** aggregates over all 40 trials.

Method	Strawberry	Potato	Banana	Apple	Avg.
RoboNinja (baseline)	1/10	2/10	2/10	7/10	12/40
Ours	6/10	5/10	9/10	3/10	23/40

IX. Additional Experiments

We present additional ablations to isolate the contributions of: (i) our full DexNinja pipeline relative to a standard Reinforcement Learning baseline (RoboNinja) [85], (ii) the scale of simulation-based densification, and (iii) robustness to variability in the initial object pose. (iv) system-level design choices, including action-head factorization and action chunking.

Except for the components explicitly ablated in Sec. IX-D, we keep the model architecture and optimization hyperparameters fixed. Therefore, differences in the remaining experiments are attributable primarily to the training data source/scale or to test-time perturbations.

- **Baseline comparison:** How does DexNinja compare to RoboNinja under identical real-world evaluation?
- **Data scaling:** Does adding more simulated densification episodes improve real-world success?
- **Pose robustness:** Does performance degrade under randomized object placement?
- **System-level ablation:** How do action-head factorization and adaptive action chunking affect performance?

A. Baseline comparison against RoboNinja

Tab. II compares DexNinja to RoboNinja under the same real-world protocol (40 trials total; 10 per object). DexNinja achieves $23/40$ successes (57.5%) versus $12/40$ (30.0%) for RoboNinja, a $+27.5$ percentage-point improvement and a $1.92\times$ relative gain in success rate.

At the object level, we observe improvements on Strawberry ($+5/10$), Potato ($+3/10$), and Banana ($+7/10$), suggesting that gains are not isolated to a single instance type but extend across multiple deformable foods. Performance decreases on Apple ($-4/10$), indicating a remaining category-specific gap. We view this as a targeted augmentation opportunity: Apple may require stronger coverage of category-specific geometry, surface conditions, and contact/cutting regimes than the current densification distribution provides.

B. Computation and cost

Our simulation-based densification is GPU-accelerated and memory intensive. Generating one simulated episode (deformable cutting + tactile synthesis) requires approximately **20 GB GPU memory** and about **10 minutes** on a single GPU in our current implementation. The total data-generation cost scales approximately linearly with the number of simulated episodes: for N episodes, the nominal cost is $\approx 10N$ minutes (equivalently, $N/6$ GPU-hours), excluding parallelization across multiple GPUs.

TABLE III: Scaling law of simulation augmentation for real-world cutting. We train with a fixed real dataset and add N simulated episodes; values are real-world **success** (25 trials per food), and Avg. aggregates 100 trials.

+Sim episodes N	**Strawberry**	**Banana**	**Potato**	**Apple**	**Avg.**
10	9/25	14/25	8/25	5/25	36/100
50	12/25	18/25	12/25	8/25	50/100
100	14/25	20/25	13/25	9/25	56/100
300	**16/25**	**22/25**	**15/25**	**10/25**	**63/100**

Because all ablations in this section hold the model architecture, training hyperparameters, and evaluation protocol constant, observed performance differences can be attributed primarily to data scale or test-time conditions rather than training confounders.

C. Scaling simulation data

We evaluate how simulation scale affects real-world success by varying the number of simulated densification episodes used during training, while keeping (i) the real teleoperated demonstrations, (ii) the architecture, and (iii) optimization hyperparameters fixed. Each policy is evaluated with the same real-world protocol described above.

As shown in Tab. III, increasing the simulation budget generally improves real-world performance, consistent with the hypothesis that densification expands coverage of contact-rich regimes that are hard to capture from limited teleoperation. Returns are not strictly linear, due to a combination of (i) saturation once key interaction modes are sufficiently covered and (ii) evaluation variance from a finite number of real trials. Given the near-linear episode generation cost (Sec. IX-B), these results motivate prioritizing *diversity and coverage* in the simulated distribution in addition to simply scaling episode count.

D. System-level ablation study

We further evaluate two system-level design choices: the action-head design and the action-chunking strategy. The action-head ablation compares a universal action head, which predicts arm and dexterous-hand actions using a shared output head, against separate heads for the kinematic and dexterous action components.The chunking ablation compares a fixed chunk size of 64 against our adaptive chunking strategy.

Tab. IV summarizes the real-world success rates across objects. The universal head struggles to bridge the distribution gap between global arm motion and fine dexterous-hand actions, while separate heads improve performance across all four objects. We also observe that a fixed chunk size of 64 is insufficient for manipulation phases with different temporal scales, such as grasping, repositioning, and cutting. Overall, the full system, which combines separate action heads with adaptive action chunking, achieves the best aggregate performance.

TABLE IV: System-level ablation study on head design and action chunking across objects. Metric: task success rate (successes / 10 trials).

Object	Universal Head	Separate Heads	Fixed Action Chunk	Full System
Apple	2/10	4/10	1/10	5/10
Banana	4/10	7/10	6/10	7/10
Strawberry	2/10	5/10	2/10	7/10
Potato	3/10	5/10	4/10	5/10
Average	11/40	21/40	13/40	24/40

E. RoboNinja baseline implementation details

We implement RoboNinja as faithfully as possible based on the publicly available description, matching task setup and evaluation. Our system requires the dexterous hand to grasp and manipulate the knife. In the original ROBONINJA setting, the knife is rigidly attached to the hand (e.g., taped), which removes grasp slippage and changes tool–hand interaction dynamics. To remain faithful to the baseline assumption, we rigidly attach the knife when evaluating ROBONINJA.

We could not determine the exact force sensor model and calibration used in the original ROBONINJA submission (the released repository version has an misleading link). We therefore approximate force feedback using a scalar contact/force estimate derived from our tactile readings. This yields a comparable contact-aware signal, but may not match the original sensor noise/dynamics; baseline results should be interpreted with this approximation in mind.

F. Robustness under randomized object placement

Real deployments cannot assume fixed object placement. Following prior work on generalizable visuomotor policies [10, 95, 38], we evaluate robustness to object placement variation by randomizing the food position and orientation on the cutting board. For each trial, we estimate the object 6D pose using a calibrated RGB-D camera and FoundationPose [75], then align observations/actions to the canonical object frame used during training. Tab. V summarizes results and indicates that the policy remains robust under these pose variations.

These experiments highlight three conclusions: (1) Under identical real-world evaluation, DexNinja outperforms the standard RL baseline RoboNinja in aggregate success, supporting improved sim2real robustness for contact-rich cutting; (2) Increasing densification data generally improves performance, but with diminishing returns relative to compute cost; and (3) canonical-frame alignment with pose estimation supports practical robustness to deployment-time placement variability, while residual failure modes (e.g., Apple) motivate targeted augmentation and improved category-specific contact modeling.

X. KNOWLEDGE-GUIDED DATA AUGMENTATION

We implement the knowledge-guided augmentation pipeline as a staged, loop-closure curation process that converts real-world captures into: (i) clean single-object reference images with transparent backgrounds, and (ii) validated 3D assets (3DGS or meshes) that pass multi-view consistency checks.

a) Inputs: For each raw capture, we start from an RGB (or RGB-D) image I that may contain clutter, occlusion, and background context. A detector predicts the food category $y \in \mathcal{Y}$. A summary agent extracts attribute-level descriptors

$$\mathbf{s} = \mathrm{Summarize}(I, y),$$

including coarse shape (e.g., elongated vs. round), surface (e.g., smooth vs. bumpy), color (e.g., pale green to deep red), and ripeness cues (e.g., glossiness, bruises). We store $\mathbf{s}$ as a structured record to enforce cross-domain consistency (simulation and real) and to condition subsequent generation.

b) Template-constrained description synthesis: Raw captures often have partial views or quality issues (motion blur, occlusions). Instead of directly using I as a canonical reference, a generation agent produces a detailed text description

$$p = \mathrm{Describe}(I, y, \mathbf{s}),$$

formatted in a constrained template to improve controllability, e.g., *"This is a `<adj> <object>` with `<features>`"*, where `<features>` enumerates extracted attributes in a fixed order (shape/surface/color/ripeness).

c) Image cleaning via inpainting and transparency: Given $(I, p, \mathbf{s})$, we inpaint a single-object canonical view:

$$\tilde{I} = \mathrm{InpaintToCleanSingleObject}(I, p, \mathbf{s}),$$

enforcing: (i) one centered object, (ii) no hands/tools/background clutter, and (iii) transparent background (alpha matte). The goal is to remove occluders and background while preserving instance attributes encoded by $\mathbf{s}$ and p.

d) Automatic quality control (QC) for 2D: We filter generated images using a vision-language judge. Given $(\tilde{I}, y, \mathbf{s}, p)$, the judge returns a decision $q_{\mathrm{2D}} \in \{0, 1\}$ and optional rejection reasons (e.g., *multiple objects*, *background not transparent*, *attribute mismatch*, *artifacts*). Rejected samples may be re-inpainted with stricter constraints derived from rejection reasons.

e) 3D asset generation and multi-view validation: For each retained clean image $\tilde{I}$, we generate a 3D representation using an image-to-3D backbone (e.g., Trellis [80]), producing either a 3D Gaussian Splatting asset $\mathcal{G}$ or a mesh $\mathcal{M}$:

$$A \in \{\mathcal{G}, \mathcal{M}\} = \mathrm{LiftTo3D}(\tilde{I}).$$

We render K views $\{\hat{I}_k\}_{k=1}^{K}$ from predefined camera poses and perform 3D QC:

$$q_{\mathrm{3D}} = \mathrm{JudgeMultiView}(\{\hat{I}_k\}_{k=1}^{K}, y, \mathbf{s}),$$

checking (i) identity preservation across views, (ii) geometric plausibility (no missing lobes / melted structure), and (iii) appearance stability (no view-dependent texture tearing). Assets passing $q_{\mathrm{3D}} = 1$ are added to the category pool for simulation rollouts and local continuation around real anchors.

TABLE V: Robustness to random food placement in the real world. For **Random**, we place the object at different positions/orientations on the board and estimate its 6D pose with FoundationPose [75]. We report the number of trials achieving **Progress** (knife reaches the pre-cut contact region), **Success** (object is cut/halved), and **Integrity** (food remains intact aside from the intended cut)

Food	Placement	Progress ↑	Success ↑	Integrity ↑
Strawberry	Canonical	19/25	12/25	14/25
	Random (FoundationPose)	*13/25*	*8/25*	*11/25*
Banana	Canonical	24/25	18/25	23/25
	Random (FoundationPose)	*20/25*	*13/25*	*17/25*
Potato	Canonical	19/25	12/25	16/25
	Random (FoundationPose)	*14/25*	*8/25*	*12/25*
Apple	Canonical	14/25	8/25	7/25
	Random (FoundationPose)	*14/25*	*6/25*	*5/25*

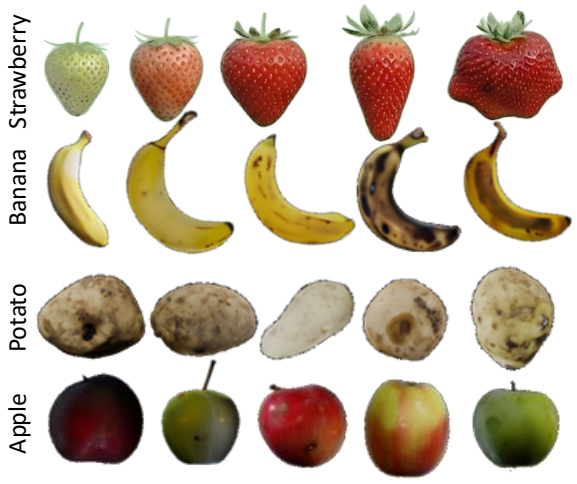

Fig. 5: **Generated instance samples** from the knowledge-guided augmentation pipeline.

Algorithm 1: Knowledge-guided instance generation with 2D/3D quality control

Input: Raw real-world captures $\{I_n\}_{n=1}^{N}$
Output: Curated assets $\mathcal{A} = \{(\tilde{I}, A, y, \mathbf{s}, p)\}$
$\mathcal{A} \leftarrow \emptyset$
for $n = 1$ **to** N **do** `// Process each capture`
  $y \leftarrow \text{Detect}(I_n)$
  $\mathbf{s} \leftarrow \text{Summarize}(I_n, y)$
  $p \leftarrow \text{Describe}(I_n, y, \mathbf{s})$
  $\tilde{I} \leftarrow \text{InpaintToCleanSingleObject}(I_n, p, \mathbf{s})$
  $(q_{2D}, r_{2D}) \leftarrow \text{Judge2D}(\tilde{I}, y, \mathbf{s}, p)$
  if $q_{2D} = 0$ **then**
    $\tilde{I} \leftarrow \text{ReInpaint}(\tilde{I}, I_n, p, r_{2D})$
    $(q_{2D}, r_{2D}) \leftarrow \text{Judge2D}(\tilde{I}, y, \mathbf{s}, p)$
  if $q_{2D} = 0$ **then**
    continue
  $A \leftarrow \text{LiftTo3D}(\tilde{I})$ `// A is 3DGS or mesh`
  $\{\hat{I}_k\}_{k=1}^{K} \leftarrow \text{RenderMultiView}(A)$
  $(q_{3D}, r_{3D}) \leftarrow \text{JudgeMultiView}(\{\hat{I}_k\}, y, \mathbf{s})$
  if $q_{3D} = 0$ **then**
    continue
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{(\tilde{I}, A, y, \mathbf{s}, p)\}$
return $\mathcal{A}$

f) Outputs: The pipeline returns curated paired assets:

$$\{(\tilde{I}_j, A_j, y_j, \mathbf{s}_j, p_j)\}_{j=1}^{N_{\text{keep}}},$$

where each element is a transparent single-object image, a validated 3DGS/mesh, and its structured attributes. These assets approximate a controllable neighborhood around real anchors for densification (Sec. V of the main paper).

XI. Simulation Details

We treat each physical food item as a unique deformable instance with distinct geometry and contact response. Real demonstrations are collected via teleoperation and serve both as imitation targets and as anchors for real2sim2real transfer.

To expand coverage beyond limited real demonstrations, we densify interaction data using simulation. Our pipeline follows prior work on tool interaction and deformable simulation (e.g., [62, 61]) and includes:

- **Object reconstruction:** mesh reconstruction from visual observations (e.g., Robo-GS [39, 19]).
- **Deformable cutting:** particle-based cutting simulation (e.g., [5, 85, 17]).
- **Tactile synthesis:** tactile modeling for aligned contact supervision (e.g., [34, 48]).

We model cutting as the composition of three *coupled* interactions: (i) **robot–fruit** contact that stabilizes the object and constrains motion on the board [85], (ii) **robot–tool** interaction that determines how the robot kinematically and dynamically excites the tool (e.g., wrist compliance, grasp stability, and force transmission) [18, 94], and (iii) **tool–fruit**

interaction that governs penetration, frictional sliding, and fracture-like separation within deformable material [57, 73]. Although these interactions are strongly interdependent in execution, this decomposition is useful because the *tool* is the physical mediator: the robot influences the fruit primarily through the tool, and the fruit reacts back to the robot through forces and moments applied on the tool.

Visual–tactile alignment. To improve sim-to-real transfer, we synthesize tactile observations in a frame-consistent manner. Concretely, we anchor tactile signals to the *active end-effector/tool frame* (rather than the world frame), using visual pose estimates and calibrated extrinsics to align contact location and force direction with the current tool pose [21]. This frame-consistent representation reduces spurious variation caused by object placement and camera viewpoint, making tactile features more transferable across scenes and embodiments.

Local-to-scene registration (implementation). A practical challenge is that deformable fruit modeling is not natively supported in the Genesis backend used for scene-level simulation, while prior cutting simulators such as RoboNinja and DiSect do not provide a full-stack articulated robot model with kinematics/dynamics at the scene level [85, 17]. We therefore introduce a *local-to-scene registration* bridge that uses the knife as the shared interface between two simulators. We run a local, tool-centric cutting simulation to resolve **tool–fruit** contact (including tactile signals), then *project* the interaction onto the knife as external contact constraints. Specifically, we aggregate contact into (a) contact points expressed in the knife frame, and (b) the corresponding force/moment (wrench) applied on the knife (optionally summarized at the knife center of mass). We then register the resulting knife pose/trajectory to the scene-level simulator (Genesis), where the articulated robot executes the same knife motion under robot kinematic constraints, while receiving the mapped external wrench as disturbance/feedback. This design is slightly slower due to per-step data transfer, but it is modular, general across objects/tools, and avoids re-implementing deformable cutting inside a scene-level robot engine.

In practice, the minimal set of quantities passed across the bridge includes: *knife pose/trajectory* $\{T_{\text{knife}}(t)\}$, *contact points* $\{p_i(t)\}$, *center of mass* $c(t)$ (or a fixed tool COM), and *external forces/moments* $\{f_i(t), \tau_i(t)\}$ (or an equivalent net wrench).

A. Canonical object representation and scale fixing

Real2Sim reconstruction yields a mesh in an arbitrary scale. Before simulation, we estimate a similarity transform (scale s, canonical rotation R, translation t) to place the object into a metric canonical frame $\mathcal{F}_S$.

We define $\mathcal{F}_S$ via PCA on mesh vertices, with the first principal axis aligned to the long axis of the object. We then set metric scale by matching a target physical length L_{target} (measured in-scene or assigned by a size bucket) to the reconstructed extent $\ell_{\max}$ along that axis:

$$s = \frac{L_{\text{target}}}{\ell_{\max}}. \tag{32}$$

Consistent scale is critical: otherwise, identical knife motions imply inconsistent penetration depth and contact forces across instances.

XII. Implementation Details

A. Hardware and sensing

All real-world experiments use a bimanual platform with:

- **Robot:** bimanual dexterous hands (Linkerhand L10) with DR3 and DR5 robotic arms.
- **Tactile sensing:** pressure-based tactile sensors on both hands for contact feedback during holding and cutting.
- **Vision:** fixed RGB-D camera (Intel RealSense D435) as the primary exteroceptive input.

Expert demonstrations are obtained using data-collection gloves. Unless otherwise noted, we treat expert trajectories as noise-free supervision after standard timestamp alignment and calibration.

B. Object-centric trajectory extraction

Let ${}^{W}\mathbf{T}_S$ denote the food pose and ${}^{W}\mathbf{T}_K$ the knife pose. From teleoperation, we extract the knife trajectory in the object frame:

$${}^{S}\mathbf{T}_K(t) = \left({}^{W}\mathbf{T}_S\right)^{-1} {}^{W}\mathbf{T}_K(t). \tag{33}$$

This object-centric trajectory serves as a template: it encodes the relative blade–object geometry that should be preserved across instances.

When the knife is rigidly grasped, we estimate the approximately constant tool transform ${}^{E}\mathbf{T}_K$ from logs:

$${}^{E}\mathbf{T}_K(t) = \left({}^{W}\mathbf{T}_E(t)\right)^{-1} {}^{W}\mathbf{T}_K(t), \tag{34}$$

and aggregate it robustly over time (e.g., averaging in $\mathfrak{se}(3)$ with outlier rejection).

C. Controller and safety filters

Cutting couples arm motion, knife pose, and grasp regulation. We make three practical choices: We separate control into arm and dexterous-hand groups: the arm tracks end-effector pose targets, while the hand controller maintains a stable grasp and regulates contact forces. Human cutting relies on fast tactile reflexes. Our tactile pipeline is slower; aggressive high-rate reaction can destabilize the hand. We therefore downsample tactile features and run dexterous hand control at a lower rate (e.g., 10 Hz rather than 30 Hz), using tactile primarily to clamp grip force and detect incipient slip or over-compression. The Inverse Kinematic problem is more constrained than standard reaching: the knife is long and collision-prone near the table and object. We use Genesis to solve Inverse Kinematic for batched knife-pose targets and discard/repair trajectories violating collision, joint-limit, or singularity constraints before entering the cutting simulator.

XIII. Training Details

a) Why a single shared arm–hand policy is unstable.: Although both subsystems are controlled in joint space, their motion statistics and effective actuation scales differ sharply. Finger joints typically traverse a large fraction of their physical range within an episode (e.g., sweeping from open to closed), whereas arm joints often perform comparatively small corrections for approach, alignment, and compliance. If we normalize all joints with one global scale, optimization becomes ill-conditioned: either (i) gradients and exploration are dominated by the high-amplitude hand joints, or (ii) the action scale is reduced to stabilize the arm and the hand becomes underactuated. This mismatch is amplified by contact dynamics: once the object is touched, small pose bias, friction differences, and compliance errors can dominate the outcome.

To stabilize optimization and align learning with task structure, we use: (i) **group-wise action factorization** with separate normalization/scaling for arms vs. hands, and (ii) a **two-stage curriculum** that separates free-space approach from contact-rich refinement. Stage 1 reliably reaches a task-relevant pre-contact configuration; Stage 2 uses tactile feedback to regulate contact and complete the manipulation/cutting.

A. Group-wise action parameterization and normalization

We factor the policy output into arm and hand groups,

$$\mathbf{a}_t = \big[\mathbf{a}_t^{\text{arm}},\ \mathbf{a}_t^{\text{hand}}\big], \qquad \mathbf{a}_t^{(\cdot)} \in [-1,1]^{d_{(\cdot)}}, \tag{35}$$

and map to joint increments using group-specific step sizes (or per-joint scales),

$$\Delta\mathbf{q}_t^{\text{arm}} = \mathbf{s}^{\text{arm}} \odot \mathbf{a}_t^{\text{arm}}, \qquad \Delta\mathbf{q}_t^{\text{hand}} = \mathbf{s}^{\text{hand}} \odot \mathbf{a}_t^{\text{hand}}, \tag{36}$$

where $\mathbf{s}^{\text{arm}}$ is intentionally smaller than $\mathbf{s}^{\text{hand}}$. In practice, we additionally apply standard safety clipping and joint-limit handling:

$$\mathbf{q}_{t+1} = \text{clip}\big(\mathbf{q}_t + [\Delta\mathbf{q}_t^{\text{arm}}, \Delta\mathbf{q}_t^{\text{hand}}],\ \mathbf{q}_{\min}, \mathbf{q}_{\max}\big), \tag{37}$$

and enforce rate limits to avoid impulsive commands near contact. This simple factorization removes the need for brittle global normalization and yields significantly more stable gradients during training.

B. Stage 1: approach to pre-grasp / pre-cut

Stage 1 drives both end-effectors to a reliable pre-contact configuration (pre-grasp or pre-cut) where contact can be initiated predictably. The objective is not to solve contact, but to (i) reduce geometric error, (ii) avoid collisions, and (iii) bring the system into a small neighborhood where tactile control can take over.

Stage 1 primarily uses low-bandwidth global signals that remain reliable in free space: robot proprioception (joint positions/velocities), end-effector kinematics, and coarse geometric cues such as estimated object pose and target approach direction. We optionally inject mild observation noise (pose perturbations, depth/registration noise) to reduce over-reliance on any single sensing channel.

We train Stage 1 with imitation learning from teleoperation trajectories and/or shaped objectives that mirror the approach structure:

- **Pose/waypoint tracking:** end-effector pose error to the pre-contact target.
- **Collision avoidance:** penalties for self-collision and undesired contacts with the environment.
- **Smoothness:** regularization on action magnitude and temporal variation (e.g., $\|\Delta\mathbf{q}_t\|$ and $\|\Delta\mathbf{q}_t - \Delta\mathbf{q}_{t-1}\|$).

To improve robustness, we use a curriculum that gradually increases the distribution shift between the initial state and the pre-contact target.

Stage 1 outputs both arm and hand increments using the group-wise scaling above. Hands are guided to a task-appropriate pre-shape (e.g., pre-grasp aperture or pre-cut finger posture), while arms execute the primary positioning and alignment.

C. Stage 2: tactile contact refinement

Once contact begins, residual errors that were negligible in free space (small pose bias, friction variation, compliance mismatch) can cause slip, unstable grasps, or failed cuts. Stage 2 therefore focuses on local contact regulation using tactile feedback, treating vision as optional due to frequent occlusion and viewpoint changes during interaction.

Stage 2 augments proprioception with tactile measurements. Depending on the sensor, we use either raw taxel values or compact tactile features (e.g., contact area, center-of-pressure, normal force proxies), along with temporal cues (finite differences or short histories) to detect slip/shear. We also include lightweight contact indicators to improve robustness under partial contact.

Stage 2 outputs incremental finger joint targets using the same group-wise scaling. Arms are executed in a compliant mode near the contact configuration, optionally allowing low-rate wrist micro-adjustments to maintain alignment while the hand regulates contact.

We train Stage 2 from contact-rich segments of demonstrations and/or reinforcement-style objectives that explicitly reward stable interaction:

- **Stable contact:** encourage sustained contact without excessive force or chatter.
- **Slip/shear suppression:** penalize rapid tactile changes correlated with tangential motion.
- **Task completion:** grasp stability and cutting progress/success, plus smoothness regularization.

In simulation, we randomize contact-relevant parameters (friction, mass/inertia, compliance proxies, and minor geometry perturbations) so the tactile policy does not overfit to a single contact regime.

D. Why tactile is important?

Tactile sensing is needed not simply to grasp more tightly, but to keep the system in the narrow regime between slip and over-compression during cutting. In the rebuttal we add a guava comparison showing that harder pressing without tactile

regulation damages the object, while tactile feedback better preserves integrity, as seen in fig. 6.

E. Sim2Real deployment

At deployment, we keep the same observation normalization and group-wise action scaling as used in training to avoid distribution shift. We enforce standard safety constraints: joint/rate limits, workspace bounds, and self-collision checks, and apply mild filtering near contact to suppress high-frequency jitter.

Since sensing is asynchronous, we align streams by timestamps and use the most recent synchronized sample .

We use the same stage gating as training: Stage 1 runs until a geometric proximity criterion is met and/or tactile onset is detected; Stage 2 takes over once contact is established. If contact is lost unexpectedly, the controller falls back to Stage 1 to re-acquire the pre-contact configuration before re-entering tactile refinement.

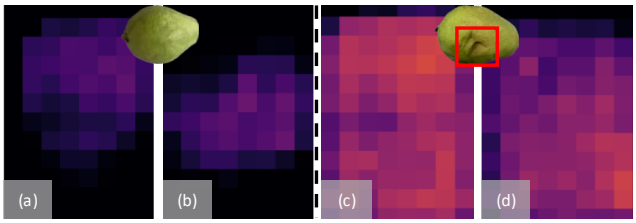

Fig. 6: Tactile responses under nominal and high-force contact, and the revised embodiment. (a,b) Thumb and index tactile maps for an undeformed guava. (c,d) Thumb and index tactile maps for a compressed guava.

F. New Embodiment Experiments.

To further demonstrate the robustness of the proposed real2sim2real pipeline, we also evaluated it under a new embodiment setting, shown in **??**(e). In this setting, we replace the original hardware configuration with Franka arms and Linkerbot L20 dexterous hands, while keeping the overall pipeline unchanged. The system remains stable under this embodiment shift, supporting the robustness of the design beyond a single platform.

XIV. Qualitative Results

We include representative real-world executions in Fig. 8. Background colors denote object type: orange (banana), red (strawberry), and yellow (potato).

a) Knife initialization and handover.: Knife pre-grasping is initialized using DexGraspNet 2.0 [94]. Concretely, we use the generative grasp prior to propose stable handle grasps and an associated end-effector pre-grasp pose, then execute a short approach to bring the knife into a consistent, repeatable starting configuration before closed-loop control begins. This initialization serves two purposes: (i) it reduces the burden on the downstream policy by standardizing the initial knife pose (position, yaw/pitch/roll, and approach height), and (ii) it improves safety and repeatability by ensuring the knife starts outside the contact region with sufficient clearance from the board and the object. After the pre-grasp is reached, our policy takes over to refine alignment, initiate contact, and complete the slice.

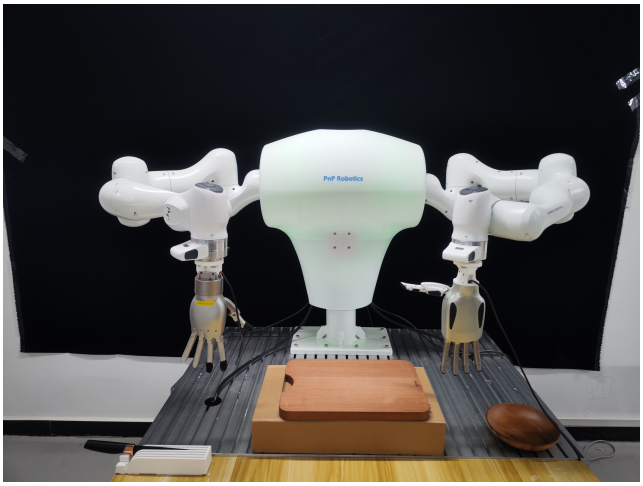

Fig. 7: New embodiment setting.

b) Phase structure in real rollouts.: During approach, the knife translates to the estimated pre-cut region while keeping a conservative clearance and a stable orientation. In contact initiation, the policy performs small, corrective motions (typically millimeter-scale translations and a slight pitch adjustment) to settle the blade against the surface without inducing large lateral forces that would slip or topple the object. Finally, during slicing, the knife follows a smooth, monotonic progression through the object; we often observe minor orientation compensation to maintain an effective cutting angle as the contact state changes (e.g., as the blade transitions from skin contact to deeper penetration).

Qualitatively, different food categories induce distinct contact regimes that are reflected in the knife trajectory and pose evolution. For **banana** (soft, high compliance), successful trials tend to use a shallower entry and shorter, more continuous slicing strokes to avoid excessive deformation that would cause the target region to drift. For **strawberry** (delicate surface and easily damaged interior), the rollouts emphasize gentle contact initiation and reduced tangential loading; the knife trajectory typically prioritizes stable alignment before committing to depth, which helps preserve *integrity* (i.e., limiting damage to the intended cut). For **potato** (stiff, higher cutting resistance), we observe slower progression with more pronounced maintenance of blade orientation, consistent with the need to sustain cutting pressure while avoiding sudden jumps that could lead to chatter or board collision.

c) Common qualitative failure modes.: When failures occur, they are visually attributable to a small set of recurring issues: (i) *initial misalignment* (the blade contacts off-center, leading to a biased cut or early slip), (ii) *premature contact* during approach (insufficient clearance causes the knife to bump the object and disturb its pose), and (iii) *late-stage jamming or excessive friction* (the knife stalls or deviates as

Fig. 8: **Qualitative real-world deployment results.** Representative executions across food categories. Background colors denote the object type: orange (banana), pink (strawberry), red (apple), and yellow (potato).

the contact patch grows).

XV. Notation and Conventions

We summarize the most frequently used symbols in Tab. VI. We use homogeneous transforms ${}^{A}\mathbf{T}_{B} \in SE(3)$ to denote the pose of frame $\mathcal{F}_B$ expressed in frame $\mathcal{F}_A$ (mapping points from $\mathcal{F}_B$ to $\mathcal{F}_A$). Time indices use discrete steps $t \in \{0, \dots, T\}$ unless otherwise stated.

TABLE VI: **Notation summary.**

Symbol	Meaning
t, T, t_f	Discrete time index, episode horizon, final time of an episode.
$\mathcal{F}_W, \mathcal{F}_S, \mathcal{F}_K, \mathcal{F}_E$	World, food/object, knife/tool, end-effector frames.
${}^A\mathbf{T}_B \in SE(3)$	Rigid transform from frame $\mathcal{F}_B$ to $\mathcal{F}_A$.
$o_t = (o_t^{vis}, o_t^{tac}, o_t^{prop})$	Multimodal observation: vision, tactile, proprioception.
$a_t = (a_t^{kin}, a_t^{tac})$	Action split into kinematic (arm/pose) and tactile/contact (hand) components.
$\vartheta = (m, \mu_f, E, \nu, G(t))$	Instance parameters: mass, friction, elasticity (Young's modulus E, Poisson ratio ν), and geometry $G(t)$.
D, D_0, D_{aug}	Demonstrations, anchor demo set, augmented (densified) dataset.
f_ϕ	Kinematic (geometry-driven) policy/module (object-centric motion specification / planning).
π_θ	Tactile feedback policy (closed-loop contact regulation).
z_t	Stage token (e.g., approach / knife positioning / cutting).
H_t	Action-chunk horizon executed open-loop before replanning.
$S(o, a) \in \{0, 1\}$	Feasibility predicate in simulation (success under cutting + collision constraints).
I, $\tilde{I}$	Raw captured image and cleaned single-object canonical image (transparent BG).
$y \in \mathcal{Y}$	Predicted object category label.
$\mathbf{s}$	Structured attribute summary (shape/surface/color/ripeness cues).
p	Template-based text prompt used for controllable generation.
$q_{\text{2D}}, q_{\text{3D}}$	2D and 3D quality-control decisions (accept/reject).
$\mathcal{G}$, $\mathcal{M}$	3D Gaussian Splatting asset and mesh asset.