VT-Refine: Learning Bimanual Assembly with Visuo-Tactile Feedback via Simulation Fine-Tuning

Binghao Huang[1] **Jie Xu**[2] **Iretiayo Akinola**[2] **Wei Yang**[2] **Balakumar Sundaralingam**[2]
Rowland O'Flaherty[2] **Dieter Fox**[2] **Xiaolong Wang**[2,3] **Arsalan Mousavian**[2]
Yu-Wei Chao[2†] **Yunzhu Li**[1†]
[1]Columbia University [2]NVIDIA [3]University of California, San Diego
[†]Equal advising

Abstract: Humans excel at bimanual assembly tasks by adapting to rich tactile feedback—a capability that remains difficult to replicate in robots through behavioral cloning alone, due to the suboptimality and limited diversity of human demonstrations. In this work, we present VT-Refine, a visuo-tactile policy learning framework that combines real-world demonstrations, high-fidelity tactile simulation, and reinforcement learning to tackle precise, contact-rich bimanual assembly. We begin by training a diffusion policy on a small set of demonstrations using synchronized visual and tactile inputs. This policy is then transferred to a simulated digital twin equipped with simulated tactile sensors and further refined via large-scale reinforcement learning to enhance robustness and generalization. To enable accurate sim-to-real transfer, we leverage high-resolution piezoresistive tactile sensors that provide normal force signals and can be realistically modeled in parallel using GPU-accelerated simulation. Experimental results show that VT-Refine improves assembly performance in both simulation and the real world by increasing data diversity and enabling more effective policy fine-tuning. Our project page is available at https://binghao-huang.github.io/vt_refine/.

Keywords: Tactile Simulation, Bimanual Manipulation, RL Fine-Tuning

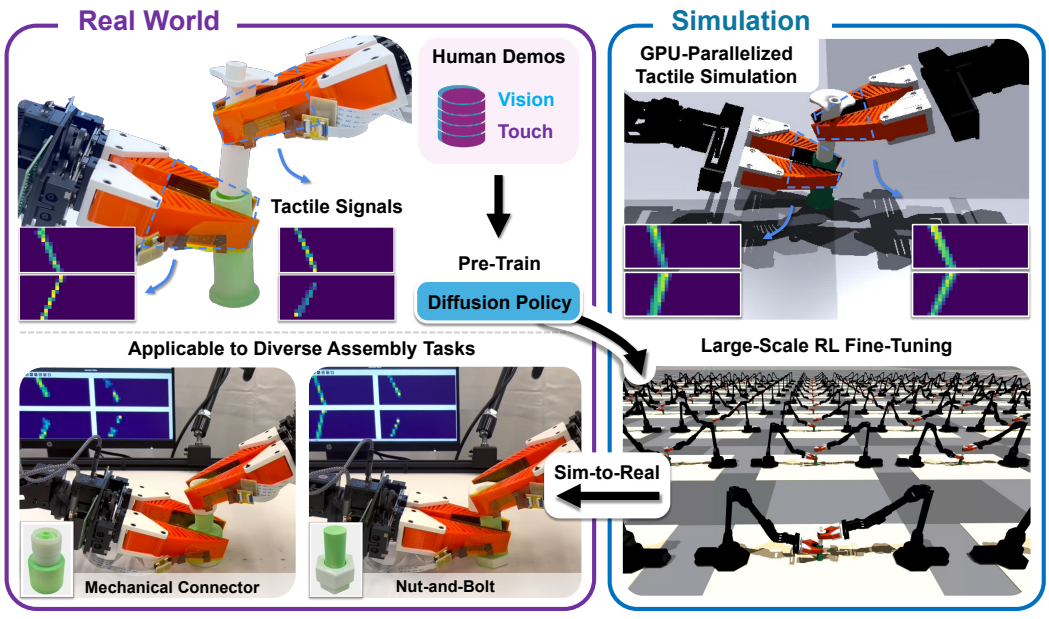


Figure 1: We propose **VT-Refine**, a novel visuo-tactile policy learning framework for precise, contact-rich bimanual assembly tasks. *Top Left:* We collect real-world demonstrations and pre-train a diffusion policy using visuo-tactile inputs. *Right to Bottom Left:* We leverage tactile simulation and large-scale reinforcement learning to fine-tune the policy, and subsequently transfer the fine-tuned policy back to the real world. The resulting policy demonstrates strong performance in both simulated and real environments.

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

1 Introduction

Solving precise assembly tasks with both hands requires sophisticated orchestration of vision and tactile sensing. Consider a bimanual plug-and-socket assembly: humans first rely on vision to locate the parts and coordinate the grasping and pickup of each part with both hands. Once the parts are held and positioned for insertion, tactile feedback becomes essential. This is because contact cues can be visually occluded during insertion (Fig. 1), and hence vision alone lacks the precision needed for fine-grained, contact-rich interactions.

Behavioral cloning with diffusion policies [1, 2] has recently shown promise in learning bimanual visuo-tactile policies from a limited number of human teleoperated demonstrations [3, 4]. However, scaling these methods for high-precision assembly tasks in real-world settings poses two major challenges. First, collecting real-world demonstrations is costly, and the demand for data only grows with increasing task precision and contact complexity, making large-scale collection prohibitively expensive. Second, current demonstration interfaces often lack tactile feedback, hindering the capture of how humans use touch for fine manipulation. Consequently, the collected demonstrations typically omit exploratory behaviors—such as iterative adjustments—that are critical for contact-rich tasks, resulting in suboptimal training data. Alternatively, simulation offers a promising path to scale visuo-tactile policy learning, but existing efforts primarily focus on visual modalities or tasks with limited reliance on touch [5–7]. While some recent work has explored simulation-based data collection for tactile inputs, these efforts are typically restricted to simpler tasks or setups (e.g., unimanual) [8–13], or have not yet addressed large-scale training or robust sim-to-real transfer for tactile-critical bimanual tasks [14].

To address these challenges, we introduce a novel real-to-sim-to-real framework designed for precise bimanual assembly. Our approach begins by collecting a small amount of real-world demonstrations (e.g., 30 episodes) to pre-train a bimanual visuo-tactile diffusion policy. The policy is subsequently fine-tuned using reinforcement learning (RL) on a digital twin of the scene within a parallelized simulation environment. Finally, the fine-tuned policy is transferred from simulation back to the real world. Our framework offers three key contributions: (1) We enhance visuo-tactile diffusion policies through RL-based fine-tuning in simulation, enabling policy improvement by exploring state-action regions near those seen in the initial human demonstrations. (2) We develop a GPU-parallelized tactile simulation module within a GPU-based physics simulator to accurately simulate piezoresistive tactile sensors that reliably capture normal force signals. This selection for tactile modality and simulation significantly narrows the sim-to-real gap and overcomes critical challenges in tactile modality transfer. (3) We adopt point-based representations for visual and tactile modalities, facilitating a seamless real-to-sim-to-real transfer. The unified representation preserves the spatial relationships between visual and tactile points, enhancing policy effectiveness. To the best of our knowledge, our work is the first to show successful RL with large-scale simulation and sim-to-real transfer for bimanual visuo-tactile policies.

We comprehensively evaluate our system on five challenging bimanual assembly tasks, demonstrating successful real-world execution and performance gains from simulation-based fine-tuning. A detailed analysis of each training phase shows that high-resolution tactile feedback significantly boosts policy effectiveness during both pre-training and fine-tuning. Additionally, our visuo-tactile point-based representation enables robust bidirectional transfer between real and simulated environments, playing a critical role in the success of our two-stage training framework across tasks and domains.

2 Related Work

Tactile Sensors and Simulation. Tactile information is critical in human daily life and plays an equally important role in enabling robots to interact with their environments [15]. Recognizing its importance, researchers have integrated vision and tactile sensing to enhance robotic manipulation [3, 4, 8, 13, 14, 16–25]. Most existing work focuses on optical-based tactile sensors, which can capture normal and shear forces, as well as fine-grained surface textures [10, 12, 26–30]. However, the high-resolution images produced by these sensors are difficult to simulate accurately and cause larger sim-real gap. Some approaches [31–33] attempt to sample marker positions from optical tactile

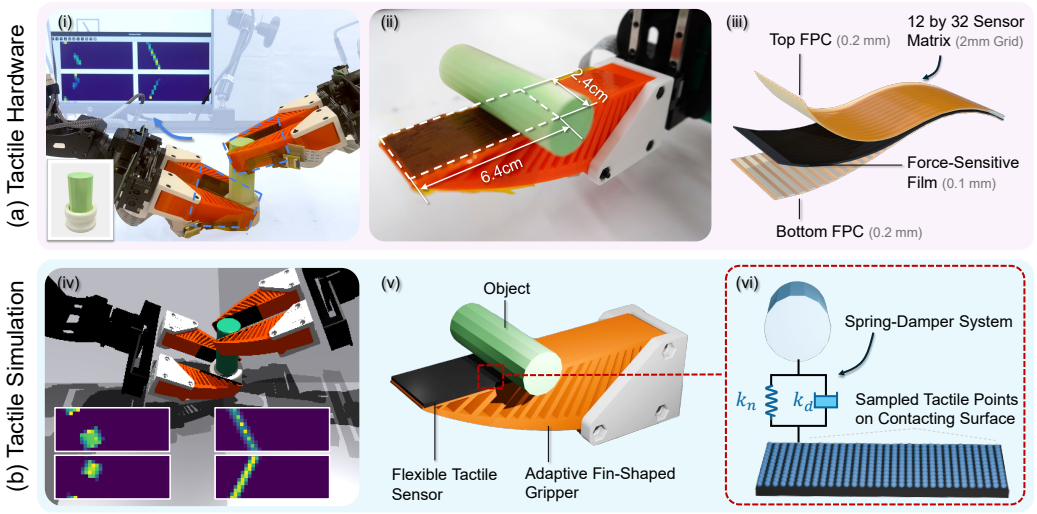


Figure 2: **Tactile Sensing in Real and Simulation.** (a) Our real-world hardware setup, including the design of the piezoresistive tactile sensor. Four tactile sensor pads (two per hand) are mounted on the soft gripper to capture contact forces. (b) Replication of the tactile sensing process in simulation. A spring-damper model is used to simulate the interaction between the tactile points and objects to generate realistic tactile signals.

images and infer normal and shear forces from marker deviations, but this indirect method further complicates sim-to-real transfer. In contrast, we select a tactile sensing modality that emphasizes structural contact patterns with normal force only rather than fine textures. Such signals are not only easier to simulate accurately but also more amenable to transfer between real and simulated environments, enabling scalable visuo-tactile data generation through simulation.

Bimanual Visuo-Tactile Manipulation. Bimanual robotic manipulation presents significant challenges across a range of applications [7, 17, 34–38], particularly for assembly tasks. Recently, there has been growing interest in learning-based methods, such as imitation learning [39–43], which leverage multimodal human demonstrations for fine-grained manipulation. However, achieving higher precision tasks vastly increases the amount of training data required in bimanual settings. To address this, simulation has been used to generate additional data and enhance policy robustness. Villasevil et al. [5] explored the use of reinforcement learning to fine-tune policies initialized by imitation learning. Nonetheless, most bimanual manipulation frameworks are still restricted to using the visual input alone [44–46], particularly for real-to-sim-to-real pipelines. This is largely because tactile signals, especially those from optical tactile sensors, are difficult to simulate and transfer [47], limiting their potential of being incorporated into simulation-based training. In contrast, our framework, along with the selection of a transfer-friendly tactile modality, enables effective real-to-sim-to-real learning with both vision and tactile inputs.

3 Visuo-Tactile System and Tactile Simulatiom

Tactile Sensor Hardware. Our sensor **FlexiTac** uses resistive sensing matrices, inspired by 3D-ViTac [3], to efficiently convert mechanical pressure into electrical signals. The choice of matrix-based flexible sensors is motivated by two key factors: (1) *Compatibility*: these sensors are versatile and can be mounted on a wide range of robotic end-effectors, including both rigid and compliant fingers. (2) *Sim-to-real transferability*: the sensing modality can be simulated with high fidelity in our environment, enabling consistent behavior across real-to-sim-to-real transfers.

As depicted in Fig. 2, each robotic finger is equipped with a tactile sensor pad composed of 12×32 sensing units, with a spatial resolution of $2mm$ (i.e., $2mm$ center-to-center distance between adjacent sensors). We use a triple-layer structure similar to [3], with a piezoresistive sensing layer sandwiched between two flexible printed circuit (FPC) layers (Fig. 2 (iii)). Utilizing FPC significantly enhances spatial consistency and increases the resolution of each sensor pad. Additionally, we developed a streamlined fabrication process capable of producing a single sensor in under 5 minutes, enabling

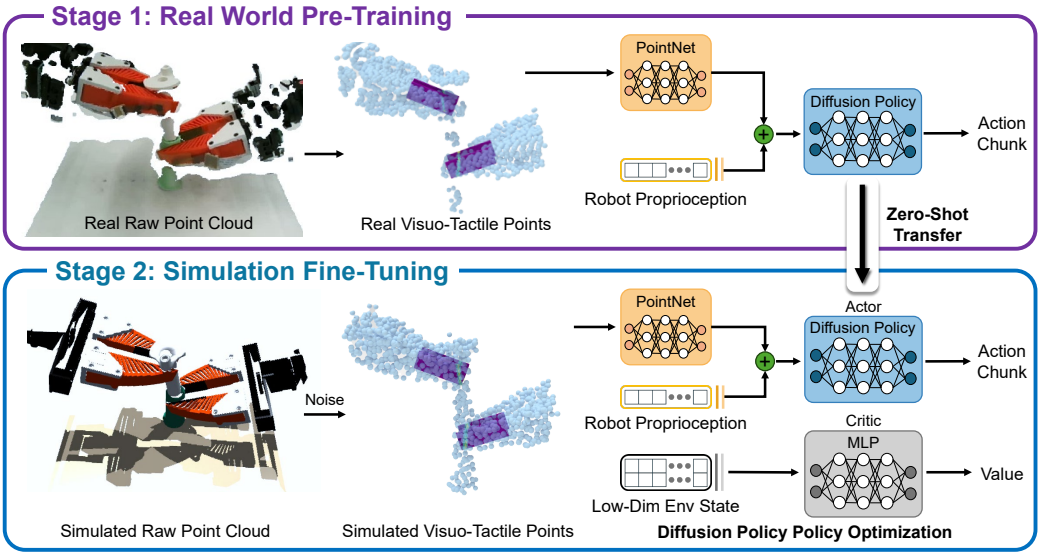


Figure 3: **Two-Stage Visuo-Tactile Policy Training.** *Stage 1:* We collect real-world human demonstrations with visual and tactile modalities and pre-train a diffusion policy. *Stage 2:* We simulate the same sensory modalities in simulation and fine-tune the pre-trained diffusion policy using policy-gradient-based RL.

cost-effective and scalable deployment. **We are committed to releasing comprehensive tutorials detailing the hardware design and fabrication process.**

Tactile Simulation. To simulate the tactile sensory input, we build on TacSL [12], a GPU-based tactile simulation library integrated with Isaac Gym [48]. We chose TacSL since the sensory signals acquired from our sensor pads are closely akin to TacSL's simulated tactile normal forces. To model the soft-contact interactions between our deformable tactile sensors (mounted on the soft grippers) and rigid contacting objects, we follow TacSL and employ a penetration-based tactile force model [49]. As shown in Fig. 2 (vi), the interaction between each *tactile point* (i.e., the 3D position of a sensing unit) and the rigid object is modeled using a Kelvin-Voigt model, consisting of a linear spring and a viscous damper connected in parallel. Each sampled tactile point independently computes the contact normal force f_n as: $f_n = -(k_n d + k_d \dot{d})\mathbf{n}$, where d and $\dot{d}$ represent the interpenetration depth and the relative velocity along the contact normal, respectively, and $\mathbf{n}$ denotes the outward contact normal vector. At each simulation timestep, the signed distance field (SDF) of the contacting object is queried to compute d, and the positions of tactile points are updated in real time via forward kinematics. The known resolution of our tactile sensor allows us to uniformly distribute contact points across the sensor surface. The shape and spatial resolution of the simulated sensor are fully customizable, ensuring consistency with their real-world counterparts. Further implementation details and force computation steps are provided in the Appendix.

Real-to-Sim-to-Real for Tactile. Alternative tactile sensors, such as vision-based ones like GelSight [16], rely heavily on internal illumination, making them difficult to simulate accurately and prone to sim-to-real gaps. In contrast, our approach uses normal force signals, which are easier to calibrate in simulation, as demonstrated in our experiments. Another challenge lies in simulating the deformable nature of flexible tactile sensors. While high-fidelity techniques like finite element methods (FEM) can model this softness, they are computationally expensive and impractical for large-scale reinforcement learning. By leveraging TacSL's GPU-accelerated simulation, we efficiently approximate the softness of flexible tactile sensors, enabling scalable training. As a result, our sensor design improves robustness and supports effective zero-shot sim-to-real transfer.

4 Visuo-Tactile Policy Policy Optimizatiom

Our goal is to learn a generalizable and robust control policy, denoted as $\pi : \mathcal{O} \rightarrow \mathcal{A}$ that maps multi-modal observation $o \in \mathcal{O}$ to robot actions $a \in \mathcal{A}$ with a few real-world demonstrations. As shown in Fig. 3, our method consists of two stages: (1) *real-world pre-training* and (2) *simulation fine-tuning*.

In the first stage, we pre-train a diffusion policy [1] using behavioral cloning on a small amount of human demonstrations. This pre-trained policy is expected to succeed on the task sporadically for a restricted range of object initial positions in both real-world and simulated environments. In the second stage, we initialize the policy (actor) model from the pre-trained weights and further optimize it with policy-gradient-based RL [50] in simulation. Finally, this fine-tuned policy is transferred back to the real world for evaluation.

Visuo-Tactile Representation. The choice of observation o is crucial for bridging the simulation and real world. We adopt a point cloud-based representation for its robust sim-to-real transferability [3, 10, 51]. Our observation contains three modalities: (1) *visual:* a colorless point cloud captured by an ego-centric camera, denoted as $P_t^{\text{visual}} \in \mathbb{R}^{N_{\text{vis}} \times 4}$, (2) *tactile:* a point cloud derived from the tactile sensors representing the 3D positions of the sensing units and their continuous sensory readings, denoted as $P_t^{\text{tactile}} \in \mathbb{R}^{N_{\text{tac}} \times 4}$. We set $N_{\text{tac}} = 384 \times N_{\text{finger}}$ for the tactile point cloud since each sensor pad consists of $12 \times 32 = 384$ tactile points, and (3) *proprioception:* joint positions from the two arms and two grippers.

As shown in Fig. 3, we merge the visual and tactile point clouds into a unified visuo-tactile representation: $o = P_t^{\text{tactile}} \cup P_t^{\text{visual}}$. The tactile sensor's position is computed via forward kinematics and transformed into the camera's 3D coordinate frame, preserving the spatial relationships between the two modalities. Following [2], the merged point cloud is processed by a PointNet encoder [52], and its output is concatenated with proprioceptive features encoded by a multilayer perceptron (MLP). The resulting feature vector is used as the conditioning input for the denoising diffusion network.

Stage 1: Real-World Pre-training. We begin by collecting a small real-world demonstration dataset (e.g., 30 episodes in our experiments) to pre-train a diffusion policy. At the beginning of each trial, the assembly parts are randomly placed within a designated region on the table. A human operator then teleoperates both robot arms to pick up the parts and complete the assembly task. During each demonstration, we record visual and tactile inputs, robot joint states, and action commands. To train the diffusion policy, we adopt a denoising diffusion probabilistic model (DDPM) and follow standard practice by predicting an action chunk [1] (Fig. 3, top). Given the limited number of demonstrations, the trained model may not succeed consistently, but is expected to occasionally complete the task, providing reward signals for reinforcement learning to further improve the policy during fine-tuning.

Stage 2: Simulation Fine-tuning. We fine-tune the pre-trained diffusion policy in an end-to-end manner using Diffusion Policy Policy Optimization (DPPO) [6] (Fig. 3, bottom). DPPO optimizes a diffusion policy using Proximal Policy Optimization (PPO) [50], by formalizing the denoising process as a Markov Decision Process (MDP), which allows the reward signal to propagate effectively through the denoising chain. For scalable training, we assume access to a digital twin of the scene equipped with simulated vision and tactile sensors. The pre-trained diffusion policy initializes the actor network, while the critic network is initialized randomly. We adopt an asymmetric actor-critic strategy [53], where the critic receives a low-dimensional representation of the robot and object state.
Reward Function. In line with the observations in DPPO [6], we find that pre-training on human demonstrations provides a strong prior that guides RL exploration, allowing us to avoid complex reward engineering. We therefore fine-tune using a sparse reward: the agent receives a reward of 1 when the parts are successfully assembled, and 0 otherwise [54].

5 Experimental Results

In this section, we address the following three questions through experiments: (1) How does our fine-tuned policy improve over the baseline diffusion policy? (2) How effectively does the proposed visuo-tactile representation transfer across domains (real-to-sim-to-real)? (3) How does policy performance scale with the number of human demonstrations?

5.1 Tactile Simulation Calibration

To align the simulated tactile response with that of the real sensor, we first characterize the sensor's force-reading curve using a *DMA 850 Dynamic Mechanical Analyzer*. We then fit a Kelvin–Voigt viscoelastic model by iteratively tuning the elastic modulus k_n (compliance stiffness) and viscosity

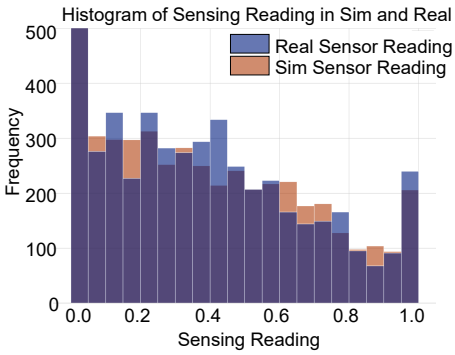
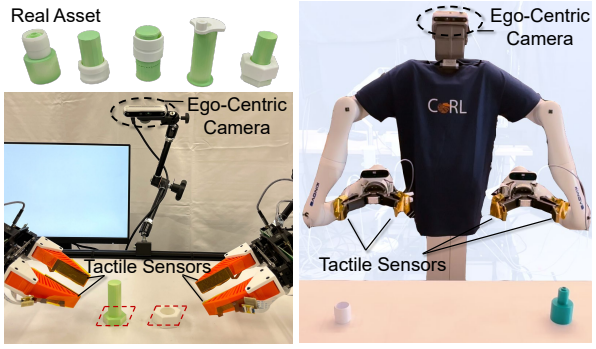


Figure 4: **Sensor Calibration Results.** The histogram shows consistent sensor reading distributions between the simulation and real world.



(a) Table-Top Bimanual Setup (b) Semi-Humanoid Bimanual Setup

Figure 5: **Real Robot Setups.** Both the socket and plug are randomly placed in a $3cm \times 3cm$ area. Both robot setups have four tactile sensing pads and an ego-centric camera.

coefficient k_d (damping) until the simulated curve closely matches the measured response. To validate the calibration, we grasp objects from multiple poses in the real world and record the corresponding tactile signals. We then replay the same trajectories in simulation to collect synthetic tactile data. A histogram comparison of the two datasets shows that the calibrated simulator accurately reproduces the distribution of real tactile signals (Fig. 4). We provide detailed calibration procedures in the supplementary materials.

5.2 Experiment Setup

We evaluate our multimodal real-to-sim-to-real system on challenging bimanual assembly tasks. Since our tactile sensors and simulation pipeline can be easily transferred across different robot platforms, we evaluate our method on two setups: (1) a tabletop bimanual robot arm setup, and (2) a semi-humanoid robot. For the **tabletop bimanual setup** (Fig.5 (a)), we adopt the teleoperation setup proposed in ALOHA 2 [55], using two 6-DoF WidowX robotic arms for manipulation, each equipped with a fin-shaped parallel soft gripper. A separate pair of identical arms is used for teleoperation. An Intel RealSense D455 camera is mounted on the table for egocentric visual sensing, and a tactile sensing pad is installed on each of the four soft fingers. For the **semi-humanoid setup** (Fig. 5 (b)), we use two 7-DoF Kinova Gen3 arms, each paired with a Robotiq 2F-140 gripper. The arms are mounted to a static torso structure. An Intel RealSense D455 camera is mounted at the head for visual sensing, and a tactile sensing pad is attached to each of the four gripper fingers. For teleoperation, we use the Meta Quest 2, with tracked controller poses mapped to target poses for the robot end effectors. Online trajectory generation is performed using the GPU-accelerated model predictive control framework provided by cuRobo [56].

Tasks are selected from the *AutoMate* dataset [57]. Each task involves a plug-socket pair: the robot must grasp both objects and complete an in-air insertion. Figure 5 shows the objects and robot configurations used in our experiments. For each task, we collect 30 demonstrations to pre-train a diffusion policy, which is then used to initialize the policy for fine-tuning (as described in Sec. 3). To reflect the variability introduced by bimanual insertion, we randomize the initial pose of each object within a $3cm$ range during both data collection and fine-tuning. The same visuo-tactile representation and encoder architecture are used throughout pre-training and fine-tuning.

5.3 Quantitative Analysis

Fine-tuning improves precise manipulation. Our RL fine-tuned policy significantly boosts performance on high-precision assembly tasks. *(i) Fine-tuning introduces necessary exploration.* Diffusion Policy performs well on lower-precision tasks, but behavior cloning alone lacks the small, repeated adjustments needed for tight-fit insertions. Encoding these subtle exploratory behaviors via demonstrations would require prohibitively large datasets. In contrast, RL fine-tuning introduces such behaviors efficiently by leveraging simulated rollouts. In real-world experiments (Tab. 1), our fine-tuned policy improves success rates by approximately 20% for the vision-only variant and 40% for

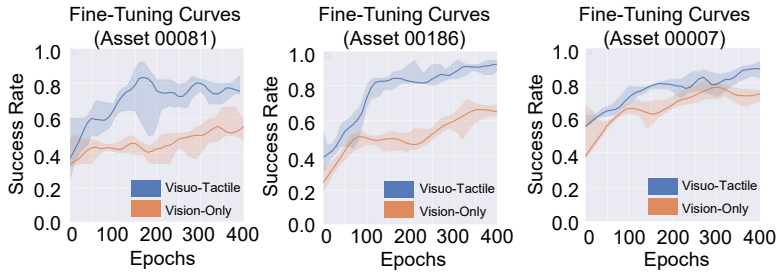


Figure 6: **Simulation Fine-Tuning of Pre-Trained Policies.** We compare the fine-tuning performance of visuo-tactile (blue) and vision-only (orange) policies. The visuo-tactile policy starts with not only a higher pre-trained performance but also continues to improve, achieving higher final performance after fine-tuning.

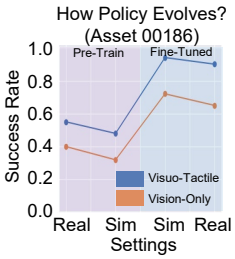


Figure 7: Performance of *Pre-Trained Policy* in sim and real, *Fine-Tuned Policy* in sim and real.

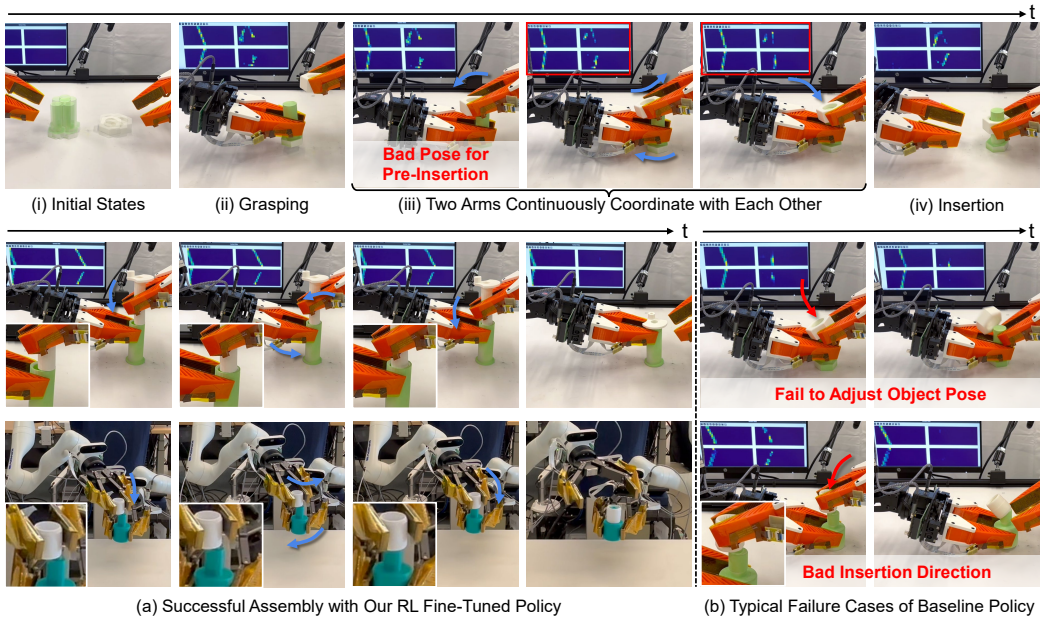


Figure 8: **Policy Rollout.** We evaluate our fine-tuned visuo-tactile policy on five plug-and-socket pairs with a clearance of roughly $\approx 2mm$. *Part(a)* shows the two arms keep re-orienting or moving the parts until they slide together smoothly, as indicated by the evolving tactile maps on the screen. *Part(b)* the robot either stops with a misaligned pose or pushes at a bad angle, leading to jams and incomplete insertions.

the visuo-tactile variant. Grasping often induces slight object slip, yielding an uncertain pre-insert pose that vision alone seldom detects due to occlusion. Fig. 8 (a) shows a representative success trajectory: following an imprecise pre-insertion pose, the two arms engage in rapid cycles of sensing, micro-adjusting, and re-sensing. These back-and-forth "wiggle-and-dock" maneuvers—commonly used by humans—emerged organically during RL fine-tuning, despite not being explicitly captured in demonstrations. This is because tactile feedback clearly signals when alignment improves, as indicated through the change of the contact forces. In contrast, policies without tactile input or sufficient exploration tend to stall or attempt insertion at poor angles, leading to failure or physical damage (Fig. 8 (b)).

(ii) Tactile feedback enhances policy fine-tuning. Visual input alone often fails to capture the fine contact cues needed to align parts. By incorporating tactile data, the visuo-tactile policy gains access to these subtle interactions, enabling it to start from a stronger baseline after pre-training and achieve higher precision after fine-tuning. As shown in our simulation experiments (Tab. 2), both the vision-only and visuo-tactile policies benefit from fine-tuning. However, the visuo-tactile policy not only begins at a higher performance level but also converges to greater precision. A common failure mode for the vision-only baseline is stalling with the plug hovering just above the socket, unable to close the final $2mm$ gap. In contrast, the visuo-tactile policy continues adjusting until successful insertion is achieved.

Table-Top Bimanual Setup

Settings	**Visual Policy** (Real)					**Visuo-Tactile Policy** (Real)				
	00081	00186	00007	00446	00581	00081	00186	00007	00446	00581
Pre-Train	0.35	0.40	0.40	0.20	0.35	0.55	0.55	0.65	0.40	0.35
RL Fine-Tuning	0.50	0.65	0.75	0.30	0.45	**0.85**	**0.90**	**0.95**	**0.80**	**0.75**

Semi-Humanoid Robot Setup

Settings	**Visual Policy** (Real)			**Visuo-Tactile Policy** (Real)		
	00081	00186	00007	00081	00186	00007
Pre-Train	0.15	0.25	0.35	0.30	0.30	0.35
RL Fine-Tuning	0.35	0.30	0.45	**0.60**	**0.65**	**0.65**

Table 1: **Real-World Experiments.** We compare the pre-trained diffusion policy with the policy after RL fine-tuning, as well as vision-only versus visuo-tactile representations. Five object assets are evaluated across two robot setups, with each column corresponding to an *AutoMate* [57] asset ID.

Table-Top Bimanual Setup

Settings	**Visual Policy** (Sim)					**Visuo-Tactile Policy** (Sim)				
	00081	00186	00007	00446	00581	00081	00186	00007	00446	00581
Pre-Train	0.28	0.32	0.42	0.12	0.18	0.45	0.48	0.54	0.34	0.31
Fine-Tune w/o Pretrain	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Fine-Tune w/ Pretrain	0.57	0.72	0.84	0.36	0.52	**0.82**	**0.94**	**0.98**	**0.76**	**0.78**

Table 2: **Simulation Results.** We compare three variants in simulation: the *Pre-Train Policy* (trained only on real-world demonstrations), *Fine-Tune w/o Pre-Train*, and *Fine-Tune w/ Pre-Train*. The results indicate that both pre-training and fine-tuning contribute significantly to final performance.

Num of Pretrain Data	**Visual Policy** (Sim)		**Visuo-Tactile Policy** (Sim)	
	Pretrain	RL Fine-Tune	Pretrain	RL Fine-Tune
10 demonstrations	0.08	0.21	0.02	0.34
30 demonstrations	0.40	0.65	0.48	0.94
50 demonstrations	0.37	0.67	0.57	0.92

Table 3: **Different Amount of Pre-Training Data.** We train *Pre-Train Policies* with different amounts of real-world data and transfer them to the simulation for fine-tuning. The results are only compared in simulation.

Representation transfer across domians (real-to-sim-to-real). Transferring a policy between the real robot and simulation inevitably introduces some performance loss due to domain mismatch. These discrepancies arise from differences in point cloud inputs, tactile readings, robot controller settings, and minor joint encoder errors (which affect the placement of tactile points, as they are computed from joint states). As shown in Fig. 7, even with our low-gap tactile modality, we observe a slight performance drop: transferring from real to simulation reduces success rates by approximately 5–10%, while sim-to-real transfer causes a smaller—and sometimes negligible—drop. However, since RL fine-tuning in simulation improves success rates by over 30%, this transfer loss is acceptable and does not outweigh the overall gain.

Ablation study: effect of pre-training data quantity. We trained three base policies using 10, 30, and 50 demonstrations, and applied the same RL fine-tuning procedure to each. As shown in Tab. 3, the policy trained with only 10 demonstrations performed poorly, achieving near-zero success. However, RL fine-tuning still improved its success rate to around 30%. The base policies trained on 30 and 50 demonstrations achieved reasonable performance and both fine-tuned to near-perfect success rates. Increasing the dataset from 30 to 50 demonstrations led to minimal improvement in the base policy. In both cases, the policy was already capable of completing the grasp phase; the main bottleneck was the fine, real-time adjustments required during the insertion phase. These micro-motions are difficult to capture with a modest increase in demonstration data, so adding more demonstrations brought limited additional benefit.

6 Conclusion

In this paper, we present a real-to-sim-to-real pipeline with multi-modal perception for precise bimanual manipulation. We introduce a tactile simulation capable of effectively modeling dense tactile sensing grids, achieving strong alignment between simulation and the real world. Finally, we demonstrate the effectiveness of RL finetuning, which substantially improves performance across diverse precise assembly tasks.

7 Limitations

7.1 Trade-offs with Vision-Based Tactile Sensors

We compare our **FlexiTac** sensor with vision-based tactile sensors along three dimensions:

(1) Resolution. Vision-based tactile sensors can provide high-resolution RGB tactile feedback at sub-millimeter ($<1mm$) scales, which benefits fine-grained manipulation. However, this high resolution introduces a significant sim-to-real gap. In contrast, our design, while lower in resolution ($2mm$ per unit), reduces simulation complexity and achieves more reliable sim-and-real alignment, while still supporting compact assembly tasks.

(2) Shear Force. A major advantage of vision-based tactile sensors is their ability to capture shear-force information. Although FlexiTac does not provide direct shear-force measurements, policies with temporal history can implicitly infer shear-related effects from contact dynamics.

(3) System Design. Vision-based tactile sensors are typically bulky (at least as large as the camera's focal length) and thus difficult to integrate into compliant grippers or small size fingertips. Moreover, customizing such sensors [58–60] requires significant engineering effort. In contrast, FlexiTac's thin, flexible force mat is lightweight, easy to install, and readily customizable, making it well-suited for compliant gripper integration.

7.2 Methodological Limitations

(1) Real-Sim Alignment. Like sim-to-real pipelines, our real-to-sim transfer pipeline requires manual calibration efforts. Visual domains, tactile distributions, and low-level control must all be aligned.

(2) Scope of the Method Applicability. The current pipeline is constrained by simulator capabilities: both Isaac Gym and our tactile simulation lack support for deformable objects. Longer-horizon tasks require significantly more training time, and the absence of shear force sensing limits applicability to more complex tasks. In future work, we aim to further enhance real-to-sim fidelity.

(3) Requirement for CAD models. Even though our sparse-reward formulation avoids excessive reward engineering, we still rely on object CAD models to train fine-tuned policies; in this work, we used 3D-printed replicas from an existing dataset. A CAD-free, plug-and-play pipeline would enable the extension of our approach to a much broader range of everyday objects.

Acknowledgement

This work is partially supported by the Toyota Research Institute (TRI), the Sony Group Corporation, Google, Dalus AI, Pickle Robot, and an Amazon Research Award. This article solely reflects the opinions and conclusions of its authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

References

[1] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. C. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *RSS*, 2023.

[2] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3D diffusion policy: Generalizable visuomotor policy learning via simple 3D representations. In *RSS*, 2024.

[3] B. Huang, Y. Wang, X. Yang, Y. Luo, and Y. Li. 3D-ViTac: Learning fine-grained manipulation with visuo-tactile sensing. In *CoRL*, 2024.

[4] T. Lin, Y. Zhang, Q. Li, H. Qi, B. Yi, S. Levine, and J. Malik. Learning visuotactile skills with two multifingered hands. In *ICRA*, 2025.

[5] M. T. Villasevil, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. In *RSS*, 2024.

[6] A. Z. Ren, J. Lidard, L. L. Ankile, A. Simeonov, P. Agrawal, A. Majumdar, B. Burchfiel, H. Dai, and M. Simchowitz. Diffusion policy policy optimization. In *ICLR*, 2025.

[7] L. Ankile, A. Simeonov, I. Shenfeld, M. Torne, and P. Agrawal. From imitation to refinement – residual RL for precise assembly. In *ICRA*, 2025.

[8] Z.-H. Yin, B. Huang, Y. Qin, Q. Chen, and X. Wang. Rotating without seeing: Towards in-hand dexterity through touch. In *RSS*, 2023.

[9] H. Qi, B. Yi, S. Suresh, M. Lambeta, Y. Ma, R. Calandra, and J. Malik. General in-hand object rotation with vision and touch. In *CoRL*, 2023.

[10] Y. Yuan, H. Che, Y. Qin, B. Huang, Z.-H. Yin, K.-W. Lee, Y. Wu, S.-C. Lim, and X. Wang. Robot synesthesia: In-hand manipulation with visuotactile sensing. In *ICRA*, 2024.

[11] J. Wang, Y. Yuan, H. Che, H. Qi, Y. Ma, J. Malik, and X. Wang. Lessons from learning to spin "pens". In *CoRL*, 2024.

[12] I. Akinola, J. Xu, J. Carius, D. Fox, and Y. Narang. TacSL: A library for visuotactile sensor simulation and learning. *T-RO*, 41:2645–2661, 2025.

[13] J. Yin, H. Qi, J. Malik, J. Pikul, M. Yim, and T. Hellebrekers. Learning in-hand translation using tactile skin with shear and normal force sensing. In *ICRA*, 2025.

[14] Y. Lin, A. Church, M. Yang, H. Li, J. Lloyd, D. Zhang, and N. F. Lepora. Bi-touch: Bimanual tactile manipulation with sim-to-real deep reinforcement learning. *RA-L*, 8(9):5472–5479, 2023.

[15] R. S. Johansson and G. Westling. Roles of glabrous skin receptors and sensorimotor memory in automatic control of precision grip when lifting rougher or more slippery objects. *Experimental Brain Research*, 56:550–564, 2004. URL https://api.semanticscholar.org/CorpusID:16631166.

[16] W. Yuan, S. Dong, and E. H. Adelson. GelSight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12), 2017.

[17] T. Lin, Z.-H. Yin, H. Qi, P. Abbeel, and J. Malik. Twisting lids off with two hands. In *CoRL*, 2024.

[18] S. Suresh, H. Qi, T. Wu, T. Fan, L. Pineda, M. Lambeta, J. Malik, M. Kalakrishnan, R. Calandra, M. Kaess, et al. Neural feels with neural fields: Visuo-tactile perception for in-hand manipulation. *arXiv preprint arXiv:2312.13469*, 2023.

[19] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg. Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8943–8950. IEEE, 2019.

[20] V. Dave, F. Lygerakis, and E. Rueckert. Multimodal visual-tactile representation learning through self-supervised contrastive pre-training. *arXiv preprint arXiv:2401.12024*, 2024.

[21] Z. Ding, G. Chen, Z. Wang, and L. Sun. Adaptive visual–tactile fusion recognition for robotic operation of multi-material system. *Frontiers in Neurorobotics*, 17, 2023. ISSN 1662-5218. doi:10.3389/fnbot.2023.1181383. URL https://www.frontiersin.org/articles/10.3389/fnbot.2023.1181383.

[22] H. Xue, J. Ren, W. Chen, G. Zhang, Y. Fang, G. Gu, H. Xu, and C. Lu. Reactive diffusion policy: Slow-fast visual-tactile policy learning for contact-rich manipulation. *arXiv preprint arXiv:2503.02881*, 2025.

[23] B. Ai, S. Tian, H. Shi, Y. Wang, C. Tan, Y. Li, and J. Wu. Robopack: Learning tactile-informed dynamics models for dense packing. *Robotics: Science and Systems (RSS)*, 2024. URL https://arxiv.org/abs/2407.01418.

[24] R. Bhirangi, V. Pattabiraman, E. Erciyes, Y. Cao, T. Hellebrekers, and L. Pinto. Anyskin: Plug-and-play skin sensing for robotic touch. *ICRA*, 2025.

[25] I. Guzey, Y. Dai, B. Evans, S. Chintala, and L. Pinto. See to touch: Learning tactile dexterity through visual incentives. *arXiv preprint arXiv:2309.12300*, 2023.

[26] E. Donlon, S. Dong, M. Liu, J. Li, E. Adelson, and A. Rodriguez. Gelslim: A high-resolution, compact, robust, and calibrated tactile-sensing finger. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1927–1934. IEEE, 2018.

[27] I. H. Taylor, S. Dong, and A. Rodriguez. Gelslim 3.0: High-resolution measurement of shape, force and slip in a compact tactile-sensing finger. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10781–10787. IEEE, 2022.

[28] D. Ma, E. Donlon, S. Dong, and A. Rodriguez. Dense tactile force estimation using gelslim and inverse fem. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5418–5424. IEEE, 2019.

[29] Z. Si, G. Zhang, Q. Ben, B. Romero, Z. Xian, C. Liu, and C. Gan. Difftactile: A physics-based differentiable tactile simulator for contact-rich robotic manipulation. *arXiv preprint arXiv:2403.08716*, 2024.

[30] A. Goncalves, N. Kuppuswamy, A. Beaulieu, A. Uttamchandani, K. M. Tsui, and A. Alspach. Punyo-1: Soft tactile-sensing upper-body robot for large object manipulation and physical human interaction. In *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*, pages 844–851, 2022. doi:10.1109/RoboSoft54090.2022.9762117.

[31] N. Sunil, S. Wang, Y. She, E. Adelson, and A. R. Garcia. Visuotactile affordances for cloth manipulation with local control. In *Conference on Robot Learning*, pages 1596–1606. PMLR, 2023.

[32] Y. She, S. Wang, S. Dong, N. Sunil, A. Rodriguez, and E. Adelson. Cable manipulation with a tactile-reactive gripper. *The International Journal of Robotics Research*, 40(12-14):1385–1401, 2021.

[33] S. Wang, Y. She, B. Romero, and E. H. Adelson. Gelsight wedge: Measuring high-resolution 3d contact geometry with a compact robot finger. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

[34] B. Huang, Y. Chen, T. Wang, Y. Qin, Y. Yang, N. Atanasov, and X. Wang. Dynamic handover: Throw and catch with bimanual hands. *arXiv preprint arXiv:2309.05655*, 2023.

[35] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. *arXiv preprint arXiv:2302.12422*, 2023.

[36] C. Wang, H. Shi, W. Wang, R. Zhang, L. Fei-Fei, and C. K. Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024.

[37] Y. Qin, W. Yang, B. Huang, K. Van Wyk, H. Su, X. Wang, Y.-W. Chao, and D. Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. In *Robotics: Science and Systems*, 2023.

[38] Y. Jiang, C. Wang, R. Zhang, J. Wu, and L. Fei-Fei. Transic: Sim-to-real policy transfer by learning from online correction. In *Conference on Robot Learning*, 2024.

[39] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[40] Y. Wang, G. Yin, B. Huang, T. Kelestemur, J. Wang, and Y. Li. Gendp: 3d semantic fields for category-level generalizable diffusion policy. In *8th Annual Conference on Robot Learning*, volume 2, 2024.

[41] L. Ankile, A. Simeonov, I. Shenfeld, and P. Agrawal. Juicer: Data-efficient imitation learning for robotic assembly. *arXiv*, 2024.

[42] K. Yu, Y. Han, Q. Wang, V. Saxena, D. Xu, and Y. Zhao. Mimictouch: Leveraging multi-modal human tactile demonstrations for contact-rich manipulation. *arXiv preprint arXiv:2310.16917*, 2023.

[43] X. Zhu, B. Huang, and Y. Li. Touch in the wild: Learning fine-grained manipulation with a portable visuo-tactile gripper. *arXiv preprint arXiv:2507.15062*, 2025.

[44] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang. Open-television: Teleoperation with immersive active visual feedback. *arXiv preprint arXiv:2407.01512*, 2024.

[45] C. Lu, X. Cheng, J. Li, S. Yang, M. Ji, C. Yuan, G. Yang, S. Yi, and X. Wang. Mobile-television: Predictive motion priors for humanoid whole-body control. *arXiv preprint arXiv:2412.07773*, 2024.

[46] R. Ding, Y. Qin, J. Zhu, C. Jia, S. Yang, R. Yang, X. Qi, and X. Wang. Bunny-visionpro: Real-time bimanual dexterous teleoperation for imitation learning. *arXiv preprint arXiv:2407.03162*, 2024.

[47] E. Su, C. Jia, Y. Qin, W. Zhou, A. Macaluso, B. Huang, and X. Wang. Sim2real manipulation on unknown objects with tactile-based reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9234–9241. IEEE, 2024.

[48] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac Gym: High performance GPU-based physics simulation for robot learning. In *NeurIPS Track on Datasets and Benchmarks*, 2021.

[49] J. Xu, S. Kim, T. Chen, A. R. Garcia, P. Agrawal, W. Matusik, and S. Sueda. Efficient tactile simulation with differentiability for robotic manipulation. In *CoRL*, 2022.

[50] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[51] Y. Qin, B. Huang, Z.-H. Yin, H. Su, and X. Wang. DexPoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation. In *CoRL*, 2022.

[52] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017.

[53] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel. Asymmetric actor critic for image-based robot learning. In *RSS*, 2018.

[54] M. Heo, Y. Lee, D. Lee, and J. J. Lim. FurnitureBench: Reproducible real-world benchmark for long-horizon complex manipulation. In *RSS*, 2023.

[55] J. Aldaco, T. Armstrong, R. Baruch, J. Bingham, S. Chan, K. Draper, D. Dwibedi, C. Finn, P. Florence, S. Goodrich, et al. Aloha 2: An enhanced low-cost hardware for bimanual teleoperation. *arXiv preprint arXiv:2405.02292*, 2024.

[56] B. Sundaralingam, S. K. S. Hari, A. Fishman, C. Garrett, K. V. Wyk, V. Blukis, A. Millane, H. Oleynikova, A. Handa, F. Ramos, N. Ratliff, and D. Fox. cuRobo: Parallelized collision-free minimum-jerk robot motion generation. *arXiv preprint arXiv:2310.17274*, 2023.

[57] B. Tang, I. Akinola, J. Xu, B. Wen, A. Handa, K. V. Wyk, D. Fox, G. S. Sukhatme, F. Ramos, and Y. Narang. AutoMate: Specialist and generalist assembly policies over diverse geometries. In *RSS*, 2024.

[58] J. Zhao, N. Kuppuswamy, S. Feng, B. Burchfiel, and E. Adelson. Polytouch: A robust multi-modal tactile sensor for contact-rich manipulation using tactile-diffusion policies. *arXiv preprint arXiv:2504.19341*, 2025.

[59] Y. Ma, J. A. Zhao, and E. Adelson. Gellink: A compact multi-phalanx finger with vision-based tactile sensing and proprioception. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1107–1113. IEEE, 2024.

[60] F. Liu, C. Li, Y. Qin, A. Shaw, J. Xu, P. Abbeel, and R. Chen. Vitamin: Learning contact-rich tasks through robot-free visuo-tactile manipulation interface. *arXiv preprint arXiv:2504.06156*, 2025.

[61] Learning the signatures of the human grasp using a scalable tactile glove. *Nature*, 569:698 – 702, 2019. URL https://api.semanticscholar.org/CorpusID:169033286.

[62] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2024.

[63] A. Z. Ren, J. Lidard, L. L. Ankile, A. Simeonov, P. Agrawal, A. Majumdar, B. Burchfiel, H. Dai, and M. Simchowitz. Diffusion policy policy optimization. In *arXiv preprint arXiv:2409.00588*, 2024.

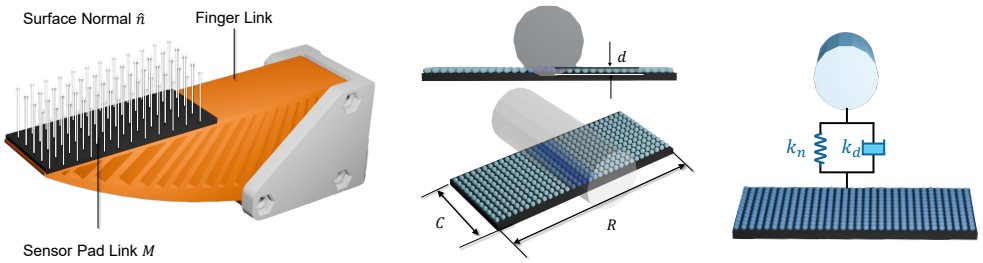
Supplementary Materials

Contents

A Details for Tactile Sensor Hardware

Our **FlexiTac** sensor builds on prior work [3, 61] with several key modifications. Instead of manually aligned conductive yarn, we employ *flexible PCBs*, which significantly improve durability and scalability, and we further adapt the design for seamless integration with soft-fin grippers. The sensor operates within a force range of approximately 0.2–10 N. Each sensor pad costs about $10, while the upgraded reading board costs roughly $30. Compared to the earlier version [3], the new reading board now supports up to 32×32 sensor units at 23 Hz, doubling the resolution from the previous 16×16 limit. The core innovations of this version lie in the *flexible PCB pad design* and the openly released circuit drawings and manufacturing guide, which are available on the FlexiTac website.

B Details for Tactile Simulation Implementation



(a) Tactile Finger $URDF$ (b) Sampling Tactile Points (c) Spring-Damper System

Figure 9: **Tactile-Simulation Pipeline.** *(a)* Finger model in the simulator with a dedicated elastomer sensing pad. *(b)* Uniform grid of taxels sampled on the pad surface and their contact interaction with an external object. *(c)* Penalty-based spring–damper model applied at every taxel to convert penetration into normal-force and depth signals.

B.1 Sampling Tactile Points

As shown in Fig. 9, each robot finger carries a sensor pad defined as a separate mesh in the URDF. Let M denote the triangular mesh of this pad and $\hat{\mathbf{n}}$ its outward surface normal (toward the "finger tip" link).

- M be the triangle mesh of the sensor pad link,
- $\hat{\mathbf{n}}$ the local surface normal that points towards the internal compliant layer (the "tip" link in the URDF).

Given a desired resolution $R \times C$ (the real robot uses 12×32), we generate taxel positions in three consecutive steps.

(i) Detect the flat face: the shortest bounding-box axis of M corresponds to pad thickness; the two remaining axes span the contact surface.

(ii) Create a planar lattice: A rectilinear grid is laid over this face, leaving a 1 mm margin to avoid edge artifacts; the grid spacing is equal to d_{taxel}.

(iii) Ray-cast for ground-truth locations: rays are shot from every lattice node along $-\hat{\mathbf{n}}$ until they intersect M. The resulting 3-D points populate `tactile_pos_local`$\in \mathbb{R}^{N\times 3}$ ($N = R \times C$) and are all assigned the fixed orientation $q_{\text{taxel}} = \text{Euler}(0, 0, -\pi)$ so that the $+y$ axis always points outward in world space.

This fully automatic procedure yields a dense, uniform taxel layout and requires no manual annotation.

B.2 Tactile Signal Computation

At every physics step we convert geometric contacts into a dense **two-channel** image that the policy ingests directly:

Chan.	Symbol	Quantity (tactile frame)
0	d	Penetration depth (m)
1	f_n	Normal force (N)

(i) Taxel pose in world coordinates. For every taxel i

$$\mathbf{x}_i^w = \mathbf{R}_e\, \mathbf{x}_i^l + \mathbf{p}_e, \qquad \dot{\mathbf{x}}_i^w = \boldsymbol{\omega}_e \times (\mathbf{R}_e \mathbf{x}_i^l) + \mathbf{v}_e,$$

where $(\mathbf{R}_e, \mathbf{p}_e, \boldsymbol{\omega}_e, \mathbf{v}_e)$ are the pose and twist of the sensor pad link returned by PhysX.

(ii) SDF query. A single GPU kernel returns the signed distance d_i, outward normal $\hat{\mathbf{n}}_i$, and normal relative velocity $\dot{d}_i = \hat{\mathbf{n}}_i \cdot \dot{\mathbf{x}}_i^w$ for every taxel.

(iii) Penalty contact model. The normal contact force is

$$f_{n,i} = -(k_n\, d_i + k_d\, \dot{d}_i),$$

with constants $k_n = 1.0$ and $k_d = 3 \times 10^{-3}$. Shear forces are not used in our cases.

(iv) Packing and normalisation. Negative depths ($d_i < 0$, meaning no contact) are clamped to zero. The pair $(d_i, f_{n,i})$ is linearly rescaled to $[0, 1]$ and reshaped into an $R \times C \times 4$ tensor that is streamed directly to the policy network.

The loop is fully vectorised over all environments (`N_envs`) and fully GPU-parallelized.

C Real-to-Sim-to-Real

C.1 Tactile Calibration

To match simulated tactile readings to the real hardware we adjust only the normal stiffness k_n and the damping term k_d of the penalty model. Calibration follows three steps: *(i)* a single taxel on the finger pad is chosen in both domains; *(ii)* a sequence of forces is applied to the real pad, yielding a ground-truth force–response curve; and *(iii)* the same normal loads are replayed in simulation while k_n and k_d are iteratively tuned until the mean-squared error between the two curves is minimised. Then apply this parameter to all sensor units.

Real pads exhibit a small, load-independent noise floor. We reproduce this behaviour by a two-stage normalisation

$$s^{\text{norm}} = \begin{cases} s / s_{\max}^{\text{fixed}}, & s < \tau, \\ s / s_{\max}^{\text{curr}}, & s \geq \tau, \end{cases}$$

where s is the raw taxel reading, τ is a noise threshold, $s_{\max}^{\text{fixed}}$ is a constant from the sensor data-sheet, and $s_{\max}^{\text{curr}}$ is the frame-wise maximum over the pad. The identical rule is applied to simulated readings so that both domains share the same dynamic range.

As illustrated in Fig. 4 of main text, the histograms of normalised signals overlap almost perfectly after calibration. The benefit is further demonstrated in Fig. 10, which shows fine-tuning performance with and without calibration using the *same* Pre-Trained policy (trained by real data). The calibrated run improves steady up, whereas the uncalibrated run initially degrades as the policy adapts to the shifted observation distribution. Although both eventually succeed in simulation, the uncalibrated policy exhibits a larger sim-to-real gap when deployed on the robot, underscoring the importance of distribution matching.

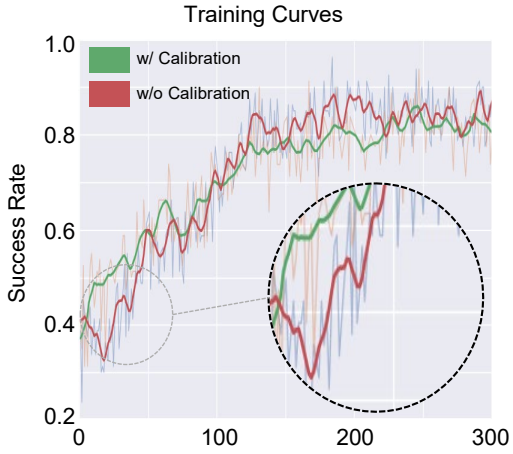


Figure 10: **Effect of Tactile Calibration on RL Fine-Tuning.** Training curves with and without sensor-simulation calibration. Without calibration the success rate initially falls as the policy re-adapts to the shifted tactile distribution, whereas the calibrated variant improves monotonically from the first iteration.

C.2 Vision for Sim–Real Alignment

To minimise the domain gap between simulated and real observations we normalise the depth stream from a single egocentric RealSense camera through three successive operations.

(i) Point-cloud generation: raw depth values are back-projected to camera space and transformed into the world frame whose origin coincides with the table centre.

(ii) Workspace cropping: the cloud is clipped to a manually specified axis-aligned bounding box that covers the robot's reachable workspace and excludes background clutter.

(iii) Uniform down-sampling: to accelerate both data loading and on-line RL roll-outs we subsample the cloud to a fixed budget of points using uniform lin-space indexing. Although farthest-point sampling (FPS) is common in diffusion-policy pipelines, uniform sampling is $\sim 10\times$ faster in our setting and was found to have no measurable impact on task performance.

(iv) Noise injection (simulation only): To emulate RealSense depth jitter, each simulated point $\mathbf{x}_w$ is perturbed by a multiplicative factor, $\mathbf{x}_w \leftarrow \mathbf{x}_w\big(1 + \mathcal{N}(0,\, 0.01\,\sigma)\big)$, where σ is a user-controlled noise level. We use noise level 3 for our pipeline. This step is omitted for real-camera data.

D DPPO Implementations and Parameters

D.1 Pre-Train Implementations and Parameters

Diffusion model. We employ the epsilon-prediction variant of *Diffusion Policy* [62] with $T = 100$ denoising steps and a rollout horizon of $H = 16$ actions. Conditioning information is injected during

the first $C=2$ steps for proprioception and during the first $C_{\text{img}}=2$ steps for point clouds, following the two-stream scheme in [62].

Backbone.

(i) Input structure. At each conditioning step t the policy receives merged visuo-tactile point cloud:

- *Visual cloud* $\mathcal{P}_t^v \in \mathbb{R}^{3\times N_v}$, whose fourth channel is initialised to zeros.
- *Tactile cloud* $\mathcal{P}_t^\tau \in \mathbb{R}^{4\times N_\tau}$ containing XYZ positions of taxels in the world frame and their normalised pressure readings as a fourth channel.

To allow the network to distinguish modalities we append a one-hot flag, yielding a 5-channel tensor

$$\mathbf{P}_t = \left[(\mathcal{P}_t^v;\, \mathbf{0}),\ (\mathcal{P}_t^\tau;\, \mathbf{1})\right] \in \mathbb{R}^{5\times(N_v+N_\tau)},$$

which is transposed to the $(N, 5)$ format expected by PointNet.

(ii) Per-step backbone. Each $\mathbf{P}_t$ is processed by *PointNetEncoderXYZ_Tactile*, an MLP with hidden sizes $\{64, 128, 256, 512\}$. Optionally, layer normalisation is inserted after every linear layer; we use the variant with `LayerNorm` and final projection to a 64-d feature. A global $\max_n$ pooling over points produces the per-step vector $\mathbf{f}_t \in \mathbb{R}^{64}$.

(iii) State feature. If proprioceptive state is available the 16-d joint states at each step is mapped through a two-layer MLP ($\{64, 64\}$) and concatenated with $\mathbf{f}$. The joint proprioception can be obtained both in sim and real.

D.2 Fine-Tune Implementations and Parameters

Actor network. The backbone is identical to pre-training (PointNet 64 $\rightarrow$ U-Net $\{512, 1024, 2048\}$, kernel 5, group norm 8), but only the last T_{ft} diffusion steps are optimised. A small K-L penalty ($\lambda = 2{\times}10^{-4}$) on the predicted noise keeps the decoder close to the pre-trained manifold.

Critic. A state-value network receives the proprioceptive history (concatenated over the $C{=}2$ conditioning steps, dimensionality 30×2) and passes it through an MLP (512–512–512, Mish activations, residual connections).

PPO hyper-parameters. We collect one segment of $n_{\text{steps}} = \lfloor 240/8 \rfloor = 30$ decisions from every environment before an update. Discount and GAE factors are $\gamma = 0.999$ and $\lambda = 0.95$. Ten PPO epochs are run per batch with a target K-L of 1.0. Learning rates are 10^{-5} (actor) and 10^{-3} (critic) with cosine decay to 10^{-6} and 10^{-3}, respectively. In the paper DPPO [63], they use learning rate 5×10^{-5}, but here we use 10^{-5} to ensure a more stable training as the task is even more fine-grained. Table 4 shows the important hyperparamter we truned for our fine-grained manipulation task.

Symbol	Value	Config Key
γ_{denoise}	0.99	`gamma_denoising`
λ_{KL}	2×10^{-4}	`clip_ploss_coef`
— base value	2×10^{-4}	`clip_ploss_coef_base`
— annealing rate	3	`clip_ploss_coef_rate`
σ_{rand}	3.0	`randn_clip_value`
$\hat{\sigma}_{\min}$	0.01	`min_sampling_denoising_std`
$\tilde{\sigma}_{\min}$	0.10	`min_logprob_denoising_std`

Table 4: DPPO–specific hyper-parameters used during fine-tuning.