

# Learning Dexterous Manipulation Skills from Imperfect Simulations

Elvis Hsieh\*, Wen-Han Hsieh\*, Yen-Jen Wang\*, Toru Lin, Jitendra Malik, Koushil Sreenath<sup>†</sup>, Haozhi Qi<sup>†</sup>

UC Berkeley

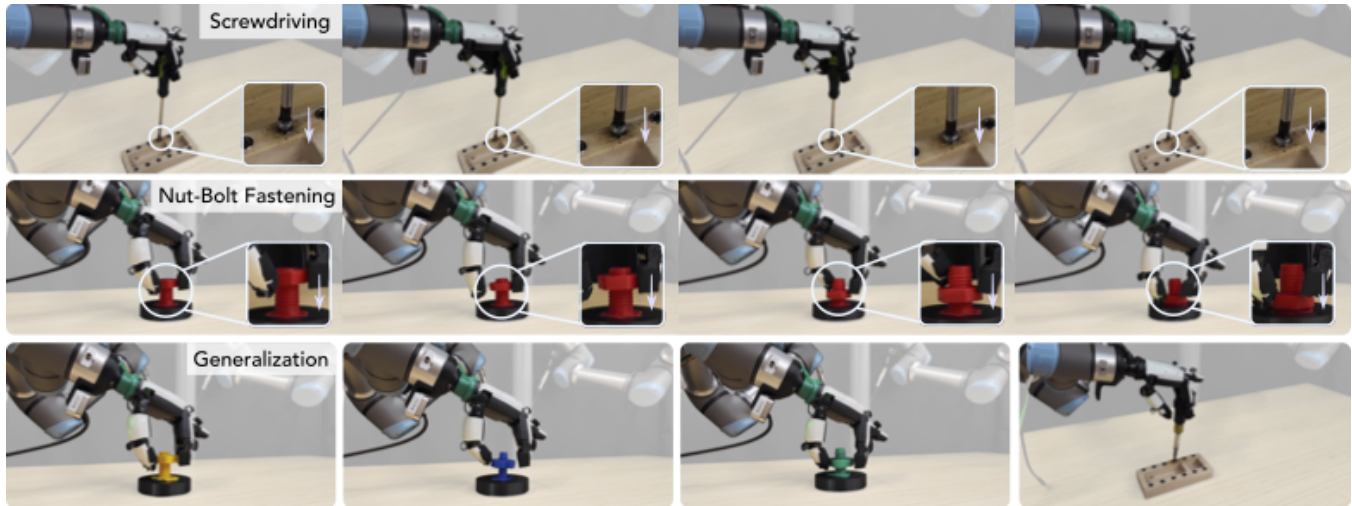


Fig. 1: We propose DexScrew, a sim-to-real framework for learning dexterous manipulation skills when the environment cannot be accurately simulated. In simulation, we use simplified objects to learn transferable rotational skills, which are then used to collect data and train tactile policies in the real world. We demonstrate the framework on contact-rich screwdriving (top row) and nut-bolt fastening (middle row). We also show generalization across different objects (bottom row). More videos and code are available on <https://dexscrew.github.io>.

**Abstract**—Reinforcement learning and sim-to-real transfer have made significant progress in dexterous manipulation. However, progress remains limited by the difficulty of simulating complex contact dynamics and multisensory signals, especially tactile feedback. In this work, we propose DexScrew, a sim-to-real framework that addresses these limitations and demonstrates its effectiveness on nut-bolt fastening and screwdriving with multi-fingered hands. The framework has three stages. First, we train reinforcement learning policies in simulation using simplified object models that lead to the emergence of correct finger gaits. We then use the learned policy as a skill primitive within a teleoperation system to collect real-world demonstrations that contain tactile and proprioceptive information. Finally, we train a behavior cloning policy that incorporates tactile sensing and show that it generalizes to nuts and screwdrivers with diverse geometries. Experiments across both tasks show high task progress ratios compared to direct sim-to-real transfer and robust performance even on unseen object shapes and under external perturbations.

## I. INTRODUCTION

Reinforcement learning (RL) paired with sim-to-real transfer has recently delivered a number of promising results in dexterous manipulation [1]–[5]. Policies trained in massively parallel simulation [6] with domain randomization [7] have demonstrated strong robustness and generalization capabilities in the real world.

However, in practice, sim-to-real transfer faces two major limitations. First, due to the complexity of physics simulation, only a limited range of tasks can be accurately modeled. Prior work either relies on specialized techniques for high-fidelity simulation [8], [9] or seeks generalization through domain randomization [1], [10]–[12]. However, as the task becomes more dynamic, the sim-to-real gap grows [13], and simulation alone becomes insufficient. Second, existing sensing modalities have an intrinsic sim-to-real gap. While vision can be partially mitigated through domain randomization [14], tactile sensing remains difficult to approximate reliably. Although some work aims to improve tactile simulation [15], [16] or trains policies using alternative proxy representations [17]–[19], these approaches cannot leverage the full power of tactile sensing. These limitations remain widely viewed as major constraints on the complexity of tasks that can be achieved.

On the other hand, teleoperation and imitation learning [20], [21] remove the need for simulation entirely. In this setting, policies can learn directly from real-world interactions and sensorimotor signals, which avoids the challenges introduced by sim-to-real transfer. However, teleoperating dexterous hands is challenging because of the intrinsic morphology differences between human and robot hands [22]–[24]. As a result, it is difficult to collect datasets that are large and diverse enough to achieve the desired behavior and generalization.

\* Equal contribution (listed in alphabetical order).

<sup>†</sup> Equal advising.

Motivated by these observations, we introduce DexScrew, a framework that combines the strengths of both approaches to expand the capability of sim-to-real reinforcement learning under imperfect simulation. The key idea is that *the motion primitives underlying contact-rich dexterous manipulation do not need to be learned from a perfect physics model*. A simplified simulator is sufficient to induce the core rotational behaviors required for these tasks. Once this motion is learned, the resulting policy can be used as a skill primitive to collect real-world demonstrations, from which a new policy can be learned. In this way, sensing modalities and physical interactions that are difficult to simulate can be obtained directly from real-world data, while the fine-grained motions that are hard to teleoperate are provided by the simulation-trained policy. We demonstrate this idea through two tasks, nut-bolt fastening and screwdriving with a multifingered hand. Both tasks are traditionally viewed as requiring complex contact dynamics understanding and tactile sensing. We instead show that effective policies can be learned without relying on high-fidelity simulation.

More specifically, our framework consists of three stages. First, we train reinforcement learning (RL) policies in simulation using a simplified physics model. Instead of modeling the thread structure of the nut and screw, we approximate their interaction with a revolute joint that connects two simple geometric shapes, which allows the policy to efficiently learn rotational behavior. Second, we use this learned skill as a primitive within a teleoperation system to collect real-world demonstrations. The operator controls the arm motion and triggers the finger rotation skill rather than issuing low-level joint commands, which enables efficient collection of tactile data during teleoperated execution. Finally, using the resulting multisensory dataset, we train a behavior cloning policy that coordinates arm and finger motions while leveraging tactile feedback.

We evaluate our framework on two tasks: nut-bolt fastening and screwdriving. Policies trained with simplified dynamics can generate reasonable rotational behavior but cannot complete the tasks. By learning from real-world multisensory demonstrations, our method overcomes these limitations and achieves stable and reliable performance under challenging contact conditions. These results show that complex contact-rich manipulation skills can be bootstrapped from simplified simulators and that real-world tactile feedback is essential. Our framework provides a scalable path toward dexterous manipulation and supports broader deployment of general-purpose robot hands in unstructured environments.

## II. RELATED WORK

Dexterous manipulation has been a long-standing challenge in robotics [25], [26]. Early work focused on classical model-based control and analytic grasp planning [27]–[31]. Recent years have seen rapid progress in learning-based approaches, which can be grouped into two primary directions: sim-to-real learning paired with reinforcement learning [1], [4], [5] and imitation learning from teleoperation [23], [24], [32] or human data [33], [34].

Both directions, however, face notable limitations. Sim-to-real methods benefit from large-scale simulated data and can generalize across diverse objects, yet they remain limited by inaccuracies in modeling complex contact dynamics and sensing, a challenge that becomes more significant as task complexity increases [13], [35]. Imitation learning benefits from multisensory real-world data, yet collecting high-quality dexterous demonstrations is considerably more difficult than collecting data for simpler end-effectors. Our work seeks to combine the strengths of both approaches. We use large-scale simulation to learn motion primitives while leveraging real-world data to close the dynamics and sensing gaps. Moreover, our skill-based framework enables efficient collection of dexterous real-world data by using the simulation-trained policy itself as a reusable skill primitive.

In the context of nut fastening and screwdriving, there has been recent work combining sim-to-real transfer with teleoperation. For example, Liu et al. [36] build a residual model from real-world interactions to compensate for the sim-to-real gap and achieve robust in-hand manipulation. Yin et al. [37] use simulation-trained policies as stability controllers to enable complex manipulation skills. Both approaches can be integrated with teleoperated arm control to complete these tasks. However, they do not produce autonomous policies that incorporate tactile sensing. Another line of work is Kumar et al. [38], who demonstrate screwdriver turning by combining learning with trajectory optimization. Noseworthy et al. [39] present an autonomous sim-to-real policy but only show results with a parallel-jaw gripper and do not demonstrate regrasping.

Another way to address the sim-to-real gap is to refine the policies in the real world. For example, Transic [40] shows that sim-to-real policies can adapt to complex real-world dynamics with only a few human interventions as demonstrations, although the demonstrations are primarily performed with simple end-effectors. In contrast, we apply this idea to dexterous hands. Maddukuri et al. [41] show that co-training with both simulation and real-world data can reduce the gap and improve manipulation performance. RLPD [42] and Sparsh-X [43] share a similar philosophy of augmenting modalities through human-guided data collection; however, they are not motivated by the possibility of learning skills from imperfect simulations, as their policies are designed to be directly transferable.

## III. DEXTERITY FROM IMPERFECT SIMULATION

An overview of our method is shown in Figure 2. It consists of three stages. First, we train a reinforcement learning (RL) policy in simulation using a simplified object model (Section III-A). The resulting policy learns the desired finger motions but does not experience real-world dynamics and lacks tactile feedback. To address this, we collect real-world trajectories using the learned policy as a skill primitive for teleoperation (Section III-B). Finally, using this dataset, we train a new multisensory policy using behavior cloning (Section III-C).

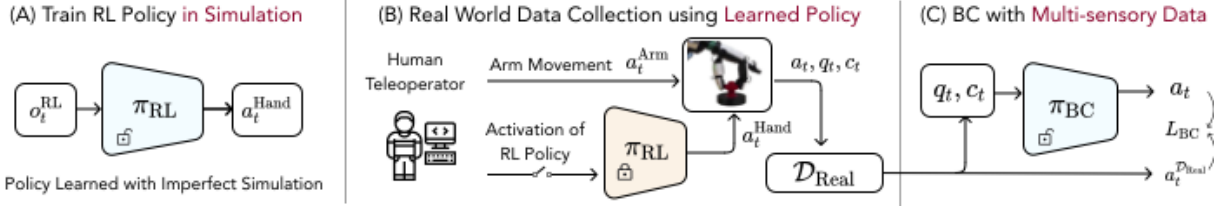


Fig. 2: **An overview of our approach.** We first train a reinforcement learning policy in simulation using a simplified object model, which serves as a motion prior for nut-bolt fastening and screwdriving. We then collect real-world trajectories by using the learned policy as a skill primitive during teleoperation. Finally, we train a behavior cloning policy on the collected data to obtain coordinated behavior between the arm and the fingers.

### A. Training a Reinforcement Learning Policy in Simulation

**Simplified Object Modeling.** Our goal is to design a simulation environment that enables fast training and encourages the emergence of desired finger gaits for rotation. To achieve this, we construct a simplified simulated object (Figure 3) that captures the essence of rotational motion. The object consists of a fixed cylindrical base with a nut or handle attached via a revolute joint. This setup allows the policy to learn rotational motion efficiently without relying on expensive contact-rich simulations. A similar idea was explored in [44] to model bottle caps using a heuristic friction design. In contrast, we further simplify the model, since we can leverage real-world data to compensate for the resulting dynamics mismatch.

Specifically, for the nut-bolt task, we use a thick triangular shape as the training object (Figure 3A). The extra thickness is used to prevent the policy from learning suboptimal strategies that apply a large force from the bottom. The learned policy also discovers a high-clearance gait that transfers well to diverse real-world nuts such as hexagonal and cube-shaped nuts. For the screwdriver task, where the primary difficulty arises from slippage around the handle, we use spherical primitives to keep the learned behavior conservative (Figure 3B). This observation, that different training shapes lead to different rotational gaits, is also consistent with the findings in [5]. Note that these objects do not need to be visually aligned with real world objects, as they are only used to learn the coarse motions used for real world data collection, as we discussed in Section III-B.

**Training Pipeline.** Following [13], [17], we first train an oracle policy and then distill it into a sensorimotor policy. The oracle policy  $f$  is trained with access to an embedding of privileged information [45]  $z_t$ . The sensorimotor policy operates without privileged sensing and instead conditions on a predicted embedding  $\hat{z}_t = \phi(h_t)$  inferred from proprioceptive history  $h_t$  by a prediction module  $\phi$ .

**Privileged Information.** The oracle policy has access to ground-truth environment and object properties, including object attributes (e.g., position, scale, mass, center of mass, friction coefficients), hand pose and finger configurations, and low-level controller parameters. The full set of privileged inputs is documented in the appendix.

**Actions.** At each step, the policy outputs a relative target

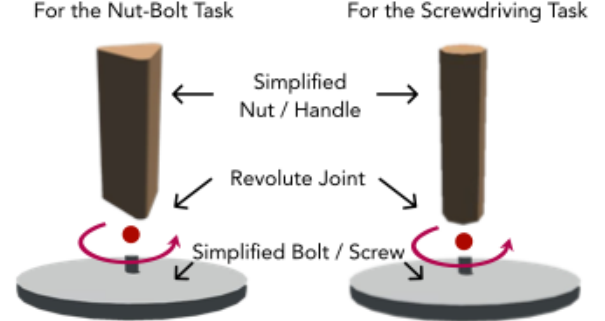


Fig. 3: **Simplified Object Models.** Each nut or handle is modeled as a rigid body attached to a fixed base through a revolute joint. This abstraction ignores thread-level mechanics while retaining the essential rotational dynamics needed for learning.

position. The position command is computed as  $a_t^{\text{Hand}} = \eta f(o^{\text{RL}}) + a_{t-1}^{\text{Hand}}$ , where  $\eta$  is the action scale. This command is sent to the robot and converted into torque via a low-level PD controller. Here,  $o^{\text{RL}}$  contains the robot’s proprioceptive state, including joint positions and previous target positions from a sliding window of recent 3 timesteps.

**Reward.** The goal of the policy in simulation is to rotate the simplified object around the revolute joint. The reward consists of a task reward, energy penalties, and stability penalties (time index  $t$  omitted for simplicity). Each component includes several terms defined in the appendix:

$$r_t = \lambda_{\text{task}} r_t^{\text{task}} + \lambda_{\text{energy}} r_t^{\text{energy}} + \lambda_{\text{stability}} r_t^{\text{stability}}.$$

**Oracle Policy Training.** We train the oracle policy using proximal policy optimization (PPO) [46] with the reward described above. The robot state and privileged information are each encoded with separate MLPs. These embeddings are concatenated and passed through an MLP to produce the final action and value predictions. We train the policy for  $1.5 \times 10^9$  environment steps.

**Sensorimotor Policy Training.** The sensorimotor policy receives proprioceptive states and a latent code  $\hat{z}_t = \phi(h_t)$  inferred from a 30-timestep history. We train the policy using DAgger [47]: at each step, the sensorimotor policy acts in the environment, while the oracle policy provides target actions and ground-truth privileged embeddings. The training objective is  $\mathcal{L} = \|a_t^{\text{Hand}} - \hat{a}_t^{\text{Hand}}\|_2^2 + \|z_t - \hat{z}_t\|_2^2$ , where  $\hat{a}_t^{\text{Hand}}$

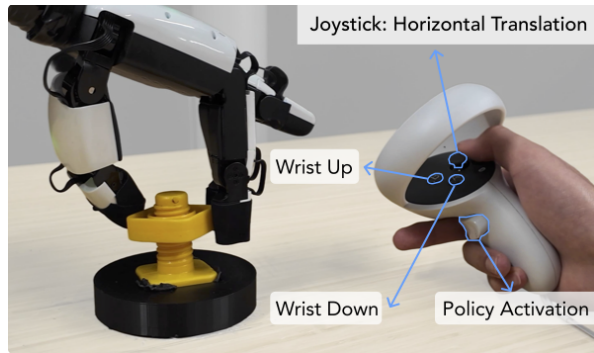


Fig. 4: **Teleoperation Interface.** The human operator controls the wrist position using the VR controller buttons and adjusts yaw and pitch through the joystick. This setup allows the operator to guide the arm motion while relying on the learned finger-rotation skill during data collection.

denotes the actions produced by the sensorimotor policy. The embedding predictor  $\phi$  is optimized using Adam [48] until convergence.

**Randomization.** We apply domain randomization during training to improve the robustness of the RL policy [7]. Following [13], we randomize the nut/handle mass, center of mass, friction coefficient, size, and PD gains, and we also add observation noise. Detailed parameters are provided in the appendix.

**Termination Conditions.** To prevent the policy from getting stuck in unrecoverable states, we terminate an episode when any of the following conditions are met: (1) the distance between the thumb or index finger and the nut/handle exceeds a reset threshold (7 cm for nuts, 10 cm for handles); (2) the nut or handle remains stagnant over a sliding time window (3.5 s for nuts, 3 s for handles); or (3) near-zero contact forces are detected for the same duration. These conditions accelerate training by penalizing failure modes such as drifting away from the object or failing to maintain contact.

### B. Real World Data Collection with Learned Policy

The policy trained in simulation with the simplified object model learns the desired rotational behavior; however, it inevitably misses key physical dynamics such as thread interactions. These effects are difficult to model but are crucial for reliable real-world fastening. It also lacks tactile information, which is hard to simulate but crucial for fine-grained wrist adjustments.

To bridge this gap, we introduce a skill-based *assisted teleoperation* for real-world data collection. The core idea is to reuse the simulation-trained policy as a skill primitive for finger motion control. Instead of commanding every joint individually, the human operator controls only the wrist movement and decides when to activate the skill primitive (Figure 4). Wrist position is specified using the Quest VR<sup>1</sup> controller’s joystick. This approach is inspired by [32], but we use a much finer-grained skill for data collection.

<sup>1</sup><https://www.meta.com/quest/products/quest-2>

This framework offers several advantages. First, it delegates complex finger motions to a robust simulation-trained policy that generalizes across different objects, eliminating the need for humans to learn finger coordination under morphological differences. Second, using a joystick for arm control enables precise and intuitive wrist positioning. While arm motion is relatively straightforward to simulate, we intentionally exclude it from the initial simulation training stage, as learning the specific downward progression required for nut-bolt fastening would necessitate simulating thread-level mechanics. Finally, collecting data in the physical environment provides the multisensory observations, particularly tactile signals, that are essential for reliable task completion but remain difficult to approximate reliably in simulation.

Concretely, at each timestep we record two actions: (1) the action generated by the RL policy  $\pi_{\text{RL}}$ , which defaults to the current hand joint positions when the policy is not activated, and (2) the arm action generated by human teleoperation. Formally, we define  $\mathbf{a}_t = [\mathbf{a}_t^{\text{Hand}}, \mathbf{a}_t^{\text{Arm}}]$ , where  $\mathbf{a}_t^{\text{Hand}} \in \mathbb{R}^{12}$  denotes the hand target joint positions, and  $\mathbf{a}_t^{\text{Arm}} \in \mathbb{R}^6$  denotes the arm joint positions. We also record the multisensory observation  $(\mathbf{q}_t, \mathbf{c}_t)$ , where  $\mathbf{q}_t = [\mathbf{q}_t^{\text{Hand}}, \mathbf{q}_t^{\text{Arm}}]$  contains all joint positions, and  $\mathbf{c}_t$  represents the raw tactile signals from all five fingers.

**Tactile Signal.** In this work, we use the XHand’s built-in tactile sensors to capture contact information. Each fingertip is equipped with a pressure-based tactile array comprising 120 sensing elements, each measuring three-axis forces with a minimum detectable force of 0.05 N. At each timestep, we record the tactile signal as  $\mathbf{c}_t \in \mathbb{R}^{5 \times 120 \times 3}$ , which includes the signals from all five fingers across three axes.

### C. Behavior Cloning with Multisensory Data

With the dataset  $\mathcal{D}_{\text{Real}}$  collected using the skill-based *assisted teleoperation*, we can train a behavior cloning (BC) policy  $\pi_{\text{BC}}$  using the paired multisensory observations and expert actions. Vision is not used in our work.

**Neural Network Architecture.** We use a feedforward network as the policy. The past  $K$  timesteps of observations  $(\mathbf{q}_{t-K+1:t}, \mathbf{c}_{t-K+1:t})$  are concatenated into a single feature vector. Tactile signals are first flattened and passed through an MLP. The fused feature vector is then processed by an hourglass encoder [49], which outputs the action predictions. We also apply an action chunking strategy [20], [21], where the policy predicts a sequence of future actions  $\hat{\mathbf{a}}_{t:t+H}$  rather than a single-step action. We use  $K = 5$  and  $H = 16$  unless otherwise noted.

**Training.** The policy is trained with supervised learning using the loss  $\mathcal{L}_{\text{BC}} = \sum_{t=1}^T \sum_{h=0}^H \|\hat{\mathbf{a}}_{t+h} - \mathbf{a}_{t+h}\|_2^2$ , where  $\hat{\mathbf{a}}_{t:t+H}$  is the action chunk predicted by  $\pi_{\text{BC}}$ , and  $\mathbf{a}_{t:t+H}$  is the corresponding expert sequence. This objective encourages consistent predictions over the full horizon. We train the policy using Adam [48] for 200 epochs and normalize observations following [50].

## IV. EXPERIMENTS

In this section, we first introduce the experiment setup (Section IV-A). We then evaluate the performance of our

TABLE I: Real-world fastening performance on square, triangular, hexagonal, and cross-shaped nuts. We report progress ratio and rotation time (mean  $\pm$  standard deviation over 10 trials) for different observation modalities. Tactile sensing and temporal history both improve performance, and their combination yields the highest accuracy and fastest execution. \* indicates that only one completely successful trial was recorded, so no standard deviation is reported.

Tactile	Hist.	Square Nuts		Triangular Nuts		Hexagonal Nuts		Cross-Shaped Nuts	
		Prog. Ratio (%) $\uparrow$	Time (s) $\downarrow$	Prog. Ratio (%) $\uparrow$	Time (s) $\downarrow$	Prog. Ratio (%) $\uparrow$	Time (s) $\downarrow$	Prog. Ratio (%) $\uparrow$	Time (s) $\downarrow$
		63.75 $\pm$ 33.05	148.76 $\pm$ 30.57	30.00 $\pm$ 39.62	229.39*	75.00 $\pm$ 29.46	102.36 $\pm$ 25.00	63.75 $\pm$ 33.05	101.07 $\pm$ 0.51
$\checkmark$		62.50 $\pm$ 33.85	202.21 $\pm$ 59.69	66.25 $\pm$ 27.67	134.19 $\pm$ 46.72	75.00 $\pm$ 16.67	205.02 $\pm$ 82.44	82.50 $\pm$ 10.54	127.17 $\pm$ 47.98
	$\checkmark$	87.50 $\pm$ 20.41	129.88 $\pm$ 48.06	80.00 $\pm$ 32.91	<b>111.47</b> $\pm$ 55.10	85.00 $\pm$ 32.17	<b>68.97</b> $\pm$ 13.99	<b>100.00</b> $\pm$ 00.00	91.49 $\pm$ 44.89
$\checkmark$	$\checkmark$	<b>97.50</b> $\pm$ 7.91	<b>124.20</b> $\pm$ 33.22	<b>96.25</b> $\pm$ 8.44	117.79 $\pm$ 52.13	<b>95.00</b> $\pm$ 10.54	75.07 $\pm$ 17.41	98.75 $\pm$ 3.95	<b>84.21</b> $\pm$ 57.19

policies on two challenging tasks, nut-bolt fastening (Section IV-B) and screwdriving (Section IV-C). We conclude with qualitative experiments that provide additional analysis and design ablations in simulation.

### A. Experiment Setup

**Hardware Setup.** Our system consists of a UR5e robot arm (6 DoF) and a 12-DoF XHand. The XHand has five fingers: the thumb and index finger each have 3 DoF, and the remaining fingers have 2 DoF. Only the thumb and index provide abduction/adduction.

**Simulation.** We train our policies in IsaacGym [6] using 8,192 parallel environments. Each environment contains a simulated XHand and the simplified object model described in Section III-A. The simulation runs at 200 Hz, with control applied at 20 Hz. Each episode lasts up to 800 simulation steps (40 s).

**Object Set.** For the nut-bolt task, we train on triangular nuts. For the screwdriver task, we approximate handles by octagon- and dodecagon-shaped nuts. This multi-geometry training in simulation helps the policy generalize to diverse real-world shapes.

**Metrics.** In simulation, we report the episode reward and episode length during training. In real-world evaluation, we measure the *progress ratio*, defined as the number of successful rotations divided by the total number of rotations required for full fastening. We also report the *time* for trials that fully complete the fastening process (progress ratio = 100), defined as the time needed to fully fasten the nut or fully tighten the screw. Note that some baseline methods do not achieve a single successful fastening or screwing attempt across ten trials. In such cases, we cannot report the standard deviation or the completion-time metric.

### B. Nut-Bolt Experiments

We first evaluate the system on the nut-bolt task. This task requires the fingers to establish correct contact patterns, sense progress through tactile feedback, and adjust the arm position accordingly. We choose this task because nut-bolt interactions are difficult to simulate efficiently, and completing the task relies heavily on tactile sensing, making it a strong testbed for our method.

We note that direct sim-to-real transfer can rotate the nut, but it cannot drive the nut downward because the arm does

not move. Since thread interactions are not simulated, the nut remains at the same height even after completing full revolutions in simulation.

**Setting.** In simulation, we use the thick triangular nut shown in Figure 3 (left). Training with this shape produces high-clearance gaits that transfer well and can rotate both square and triangular nuts in the real world.

During skill-based assisted teleoperation, we collect 50 trajectories each for the square and triangular nuts. Each trajectory lasts about 80 seconds. We then train a behavior cloning policy using the combined dataset. We evaluate performance on four types of nuts, which include square and triangular nuts as well as two unseen shapes, namely hexagonal nuts and cross-shaped nuts. Our main results are shown in Table I.

**Observation History.** We first study the effect of providing a short temporal history in the observation. Adding history significantly improves progress ratio and reduces execution time across all modalities and nut geometries. Temporal cues help the policy track fine-grained rotational progress and distinguish local geometric features. The benefit is especially clear for non-tactile policies, where history stabilizes performance and narrows much of the gap to tactile-based policies on easier geometries. When combined with tactile sensing, history yields the strongest overall performance, achieving the highest accuracy and fastest completion times across all nut types, including unseen shapes.

**Tactile Information.** Across all nut geometries, adding tactile input generally improves progress ratio. The effect is most pronounced on challenging shapes such as triangular and cross-shaped nuts, where progress ratios rise from roughly 30 to 65 percent for triangular nuts and from about 60 to 80 percent for cross-shaped nuts. This underscores the importance of tactile feedback for maintaining stable contact and detecting effective rotation. We also observe that certain non-tactile settings achieve high performance, even on unseen shapes, particularly on well-constrained geometries such as hexagonal and cross-shaped nuts. In these cases, the geometry provides strong passive guidance, and the finger gait learned in simulation is often sufficient to maintain contact without relying on tactile cues.

**Failure Modes.** We observe two main failure modes. First, the policy without observation history struggles to infer

TABLE II: **Real-world screwdriving performance.** We report progress ratio and rotation time (mean  $\pm$  standard deviation over 10 trials) for direct sim-to-real, expert replay, and our behavior cloning (BC) ablations. Tactile sensing and temporal history each improve performance, and their combination achieves the highest progress ratio and fastest execution. Here \* indicates that the policy never fully completed the task, so no rotation time is reported.

		Screwdriver Task		
Method	Tactile	Hist.	Prog. Ratio (%) $\uparrow$	Time (s) $\downarrow$
Direct Sim2Real			41.60 $\pm$ 26.21	N.A.*
Expert Replay			50.80 $\pm$ 19.27	N.A.*
Ours			69.20 $\pm$ 35.25	266.33 $\pm$ 91.60
	$\checkmark$		67.63 $\pm$ 35.65	264.06 $\pm$ 76.57
		$\checkmark$	87.50 $\pm$ 18.61	195.15 $\pm$ 44.83
	$\checkmark$	$\checkmark$	<b>95.00<math>\pm</math>13.24</b>	<b>187.87<math>\pm</math>24.87</b>

object shape from single-step proprioception. As a result, it fails to generalize across nut geometries, since different nuts require different rotational gaits. The policy cannot adjust its gait because the limited sensing information prevents it from identifying the correct motion pattern. Second, non-tactile policies frequently drift into unstable contact states and lose alignment with the bolt. Once misalignment occurs, the nut cannot sustain continuous rotation. In contrast, tactile policies can recover by adjusting wrist orientation or applying corrective downward force to re-establish contact.

### C. Screwdriving Experiments

We also demonstrate that our method extends to the more challenging screwdriving task. Compared to nut-bolt fastening, screwdriving is inherently less stable: the shaft is not kinematically constrained along the screw axis, and even small tilts or misalignments can lead to slipping or loss of contact. As a result, the task requires more fine-grained control to maintain continuous rotation. The complex interaction between the screwdriver and the screw is also difficult to simulate accurately, which is why we include this task in our study.

**Setting.** In simulation, we use a mix of octagonal and dodecagonal handles as the training objects, as shown in Figure 3 (right). This curated set encourages the learned rotational gaits to remain conservative in clearance and maintain stability. In the real world, we collect 72 trajectories, each lasting between 120 and 180 seconds.

**Sim-to-Real Policy.** Unlike the nut-bolt experiments, the screwdriving task can still make progress even when the wrist does not move, since downward motion is not required. We first evaluate a direct sim-to-real policy that uses only proprioception. The results are shown in Table II. The direct sim-to-real policy achieves a 41.60% progress ratio, indicating that it can produce meaningful behavior, which is a prerequisite for collecting expert trajectories. However, because it inevitably makes mistakes, it never completes the task even once, and therefore we cannot report statistics

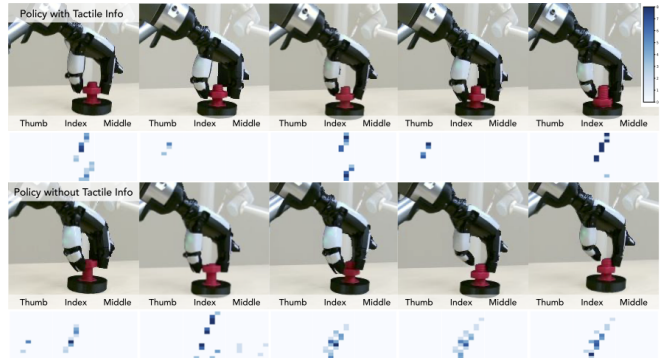


Fig. 5: **Top:** The policy with tactile information maintains a consistent alternating pattern of thumb and index finger contact, which supports stable engagement as the nut is rotated downward. **Bottom:** The policy without tactile information does not maintain a clear contact pattern. This leads to unsuccessful engagement and prevents proper downward wrist motion. The resulting pattern reflects the index finger pressing against the bolt after losing stable contact.

for completion time. In contrast, during data collection, the human operator can adjust the wrist position to recover from such mistakes.

As a result, when we replay the expert data from the data we collected, it can achieve higher success rate to be 50.80%. However, because it cannot adapt changes during deployment. It also fails at completely finishing the task. Next, we show the results of our policies.

**Main Results.** Our behavioral cloning policies show clear improvements over the direct sim-to-real and expert-replay baselines. Adding tactile sensing or temporal history individually improves progress ratio. Combining both modalities gives the strongest performance.

The baseline behavior cloning model already achieves a 69.20% progress ratio, which substantially outperforms expert replay. This phenomenon, where a behavior cloning policy surpasses the policy that generated the data, is consistent with filtered behavior cloning [51]. In this approach, only successful trials are used for training. A similar effect has also been reported in [13].

Using history alone produces a comparable improvement of 67.63%. This indicates that temporal information helps the policy track rotational progress and recover from partial failures. When history is not used, tactile information provides only limited benefit. However, once history is included, the two modalities become complementary. The progress ratio increases to 95.00%, and the average rotation time decreases substantially. These results show that tactile feedback and temporal history work together to produce stable, consistent, and efficient screwdriving behaviors.

**Failure Modes.** We observe that open-loop baselines frequently fail due to gradual handle slipping and accumulated orientation drift. Without feedback, small misalignments grow over time. Among behavioral cloning methods, the policy trained without observation history struggles to stabi-

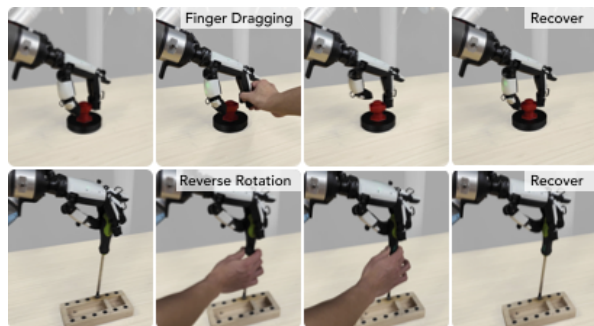


Fig. 6: **Top row:** The policy recovers back to the nut-bolt fastening motion when the fingers are dragged by an external force. **Bottom row:** The policy recovers back to the screwdriving motion when the screwdriver is rotated counterclockwise during the clockwise rotation by the policy.

lize the screwdriver since it lacks the temporal information needed to infer handle pose and orientation. Non-tactile policies fail to detect subtle torque imbalances and often lose stable contact under slight perturbations. With both tactile feedback and temporal history, BC with tactile and history compensates for these effects by adjusting wrist orientation and applying appropriate forces.

#### D. Qualitative Experiments

**Out-of-distribution Robustness.** We study the robustness of the learned policy under external perturbations that are not encountered during training. These disturbances include (1) dragging the fingers away from the object or rotating the screwdriver in the opposite direction of the intended direction. In Figure 6, we show that despite these perturbations, the policy consistently recovers to stable fastening behavior. Specifically, it recovers from reverse rotation of the fastening object by re-establishing contact and restoring the correct rotational direction. When finger contact is disrupted or temporarily blocked, the policy repositions the fingers and wrist to regain stable engagement.

**Tactile Visualization.** Tactile signals provide structured spatiotemporal patterns that the policy uses to infer contact phases. As shown in Fig. 5, stable activation signatures emerge when the nut or screwdriver handle is correctly engaged. The policy learns to preserve these patterns by adjusting wrist orientation and contact force, effectively using tactile feedback as a local reference for alignment. When these patterns deviate, corrective actions such as re-engagement or downward pressing are triggered.

#### E. Simulation Ablations

In simulation, we verify the design choices used to train the screwdriving policies. We evaluate our two-stage training procedure against two alternatives: training without any privileged information, and using an asymmetric actor-critic [52] architecture where the critic has access to full tactile information while the actor does not. The actor in this setting is directly deployable in the real world. The results are shown in Figure 7.

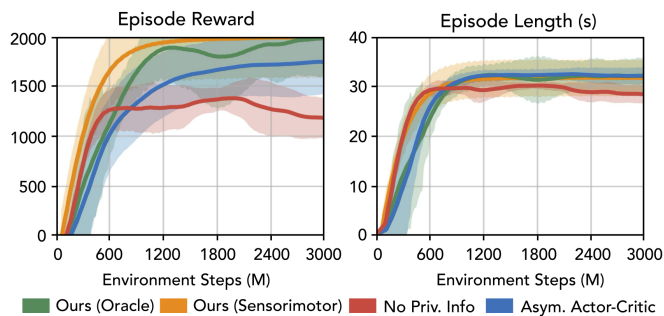


Fig. 7: Simulation ablations of screwdriving policy training. We compare our privileged-information oracle policy, its sensorimotor policy, an asymmetric actor-critic variant, and a policy trained without privileged information. Providing privileged information during training leads to significantly higher reward and more stable episode lengths. Each curve shows the mean and standard deviation over 5 seeds.

We compare episode reward and episode length across these training strategies. When both the actor and critic have access to privileged information, our method (Ours, Oracle) achieves the highest performance. Removing privileged information from the actor, as in the asymmetric actor-critic variant, leads to a noticeable drop in performance. Removing privileged information entirely results in a further decline. These results show that privileged information plays an important role during policy learning. We also observe that a sensorimotor policy can approach similar performance when proprioceptive history is provided as input. We also experimented with training the asymmetric and non-privileged models for longer horizons but have not observed further improvement. This suggests that the performance gap is not due to insufficient training but instead reflects the importance of privileged information during learning.

#### V. CONCLUSION AND FUTURE WORK

We present a framework for learning dexterous manipulation skills for contact-rich tasks using imperfect simulation. The approach first learns transferable rotational skills through reinforcement learning with simplified object modeling. It then uses these skills for skill-based teleoperation to collect real-world trajectories, and finally incorporates tactile feedback and learns a sensorimotor policy through behavior cloning. Experiments on nut-bolt fastening and screwdriver usage show that simulation alone cannot capture the complex dynamics required for reliable task execution. However, when behavior cloning is combined with tactile sensing and temporal history, the resulting policies become robust and reliable across diverse and unseen object geometries. This staged pipeline provides a practical and scalable solution for contact-rich manipulation and highlights the value of tactile sensing and skill-based teleoperation as effective bridges between simulation and real-world deployment.

**Limitations.** Although our skill-based teleoperation reduces the burden on the human operator, it remains a constraint for scalable data acquisition. Fully autonomous data collection or learning skill-level guidance from human videos would

further improve efficiency. In our current tasks, the nut is already installed and the screwdriver is already inserted. Extending the approach to fully long-horizon assembly will require vision sensing and possibly high-accuracy force-torque sensing. A broader evaluation across more contact-rich manipulation tasks is also needed to assess generality.

#### ACKNOWLEDGMENT

This work is supported in part by the program “Design of Robustly Implementable Autonomous and Intelligent Machines (TIAMAT)”, Defense Advanced Research Projects Agency award number HR00112490425. We thank Mengda Xu for his valuable feedback.

#### REFERENCES

- [1] OpenAI *et al.*, “Learning dexterous in-hand manipulation,” *IJRR*, 2019.
- [2] —, “Solving rubik’s cube with a robot hand,” *arXiv:1910.07113*, 2019.
- [3] A. Handa *et al.*, “Dextreme: Transfer of agile in-hand manipulation from simulation to reality,” in *ICRA*, 2023.
- [4] T. Chen *et al.*, “Visual dexterity: In-hand reorientation of novel and complex object shapes,” *Science Robotics*, 2023.
- [5] H. Qi *et al.*, “In-hand object rotation via rapid motor adaptation,” in *CoRL*, 2022.
- [6] V. Makovychuk *et al.*, “Isaac gym: High performance gpu-based physics simulation for robot learning,” in *NeurIPS Datasets and Benchmarks*, 2021.
- [7] X. B. Peng *et al.*, “Sim-to-real transfer of robotic control with dynamics randomization,” in *ICRA*, 2018.
- [8] Y. Narang *et al.*, “Factory: Fast contact for robotic assembly,” in *RSS*, 2022.
- [9] E. Heiden *et al.*, “Disect: A differentiable simulation engine for autonomous robotic cutting,” in *RSS*, 2021.
- [10] J. Lee *et al.*, “Learning quadrupedal locomotion over challenging terrain,” *Science robotics*, 2020.
- [11] A. Kumar *et al.*, “Rma: Rapid motor adaptation for legged robots,” in *RSS*, 2021.
- [12] A. Loquercio *et al.*, “Learning high-speed flight in the wild,” *Science Robotics*, 2021.
- [13] J. Wang *et al.*, “Lessons from learning to spin ‘pens’,” in *CoRL*, 2024.
- [14] R. Singh *et al.*, “Dextrah-rgb: Visuomotor policies to grasp anything with dexterous hands,” *arXiv:2412.01791*, 2024.
- [15] S. Wang *et al.*, “Tacto: A fast, flexible, and open-source simulator for high-resolution vision-based tactile sensors,” *RA-L*, 2022.
- [16] I. Akinola *et al.*, “Tacs: A library for visuotactile sensor simulation and learning,” *T-RO*, 2025.
- [17] H. Qi *et al.*, “General in-hand object rotation with vision and touch,” in *CoRL*, 2023.
- [18] M. Yang *et al.*, “Anyrotate: Gravity-invariant in-hand object rotation with sim-to-real touch,” in *CoRL*, 2024.
- [19] 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.
- [20] 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,” *IJRR*, 2024.
- [21] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, “Learning fine-grained bimanual manipulation with low-cost hardware,” in *RSS*, 2023.
- [22] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto, “Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation,” in *ICRA*, 2023.
- [23] S. P. Arunachalam, I. Güzey, S. Chintala, and L. Pinto, “Holo-dex: Teaching dexterity with immersive mixed reality,” in *ICRA*, 2023.
- [24] 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 *RSS*, 2023.
- [25] A. Bicchi, “Hands for dexterous manipulation and robust grasping: A difficult road toward simplicity,” *IEEE Transactions on robotics and automation*, 2002.
- [26] A. M. Okamura, N. Smaby, and M. R. Cutkosky, “An overview of dexterous manipulation,” in *ICRA*, 2000.
- [27] D. Rus, “In-hand dexterous manipulation of piecewise-smooth 3-d objects,” *IJRR*, 1999.
- [28] I. Mordatch, Z. Popović, and E. Todorov, “Contact-invariant optimization for hand manipulation,” in *Eurographics*, 2012.
- [29] R. Fearing, “Implementing a force strategy for object re-orientation,” in *ICRA*, 1986.
- [30] A. S. Morgan, K. Hang, B. Wen, K. Bekris, and A. M. Dollar, “Complex in-hand manipulation via compliance-enabled finger gaiting and multi-modal planning,” *RA-L*, 2022.
- [31] S. Patidar, A. Sieler, and O. Brock, “In-hand cube reconfiguration: Simplified,” in *IROS*, 2023.
- [32] 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.
- [33] 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,” in *RSS*, 2024.
- [34] M. Xu, H. Zhang, Y. Hou, Z. Xu, L. Fan, M. Veloso, and S. Song, “Dexumi: Using human hand as the universal manipulation interface for dexterous manipulation,” in *CoRL*, 2025.
- [35] B. Huang, Y. Chen, T. Wang, Y. Qin, Y. Yang, N. Atanasov, and X. Wang, “Dynamic handover: Throw and catch with bimanual hands,” in *CoRL*, 2023.
- [36] X. Liu, H. Wang, and L. Yi, “Dexndm: Closing the reality gap for dexterous in-hand rotation via joint-wise neural dynamics model,” *arXiv:2510.08556*, 2025.
- [37] Z.-H. Yin, C. Wang, L. Pineda, F. Hogan, K. Bodduluri, A. Sharma, P. Lancaster, I. Prasad, M. Kalakrishnan, J. Malik, M. Lambeta, T. Wu, P. Abbeel, and M. Mukadam, “Dexteritygen: Foundation controller for unprecedented dexterity,” *arXiv:2502.04307*, 2025.
- [38] A. Kumar, T. Power, F. Yang, S. A. Marinovic, S. Iba, R. S. Zarrin, and D. Berenson, “Diffusion-informed probabilistic contact search for multi-finger manipulation,” in *ICRA*, 2025.
- [39] M. Noseworthy, B. Tang, B. Wen, A. Handa, C. Kessens, N. Roy, D. Fox, F. Ramos, Y. Narang, and I. Akinola, “Forge: Force-guided exploration for robust contact-rich manipulation under uncertainty,” *RA-L*, 2025.
- [40] Y. Jiang, C. Wang, R. Zhang, J. Wu, and L. Fei-Fei, “Transic: Sim-to-real policy transfer by learning from online correction,” in *CoRL*, 2024.
- [41] A. Maddukuri, Z. Jiang, L. Y. Chen, S. Nasiriany, Y. Xie, Y. Fang, W. Huang, Z. Wang, Z. Xu, N. Chernyadev, S. Reed, K. Goldberg, A. Mandelkar, L. Fan, and Y. Zhu, “Sim-and-real co-training: A simple recipe for vision-based robotic manipulation,” *RSS*, 2025.
- [42] H. Zhang, Z. Li, X. Zeng, L. Smith, K. Stachowicz, D. Shah, L. Yue, Z. Song, W. Xia, S. Levine, *et al.*, “Traversability-aware legged navigation by learning from real-world visual data,” *T-RO*, 2025.
- [43] C. Higuera, A. Sharma, T. Fan, C. K. Bodduluri, B. Boots, M. Kaess, M. Lambeta, T. Wu, Z. Liu, F. R. Hogan, and M. Mukadam, “Tactile beyond pixels: Multisensory touch representations for robot manipulation,” in *CoRL*, 2025.
- [44] T. Lin, Z.-H. Yin, H. Qi, P. Abbeel, and J. Malik, “Twisting lids off with two hands,” in *CoRL*, 2024.
- [45] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, “Learning by cheating,” in *CoRL*, 2020.
- [46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv:1707.06347*, 2017.
- [47] S. Ross, G. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *AISTATS*, 2011.
- [48] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *ICLR*, 2015.
- [49] A. Newell, K. Yang, and J. Deng, “Stacked hourglass networks for human pose estimation,” in *ECCV*, 2016.
- [50] J. Barreiros, A. Beaulieu, A. Bhat, R. Cory, E. Cousineau, H. Dai, C.-H. Fang, K. Hashimoto, M. Z. Irshad, M. Itkina, *et al.*, “A careful examination of large behavior models for multitask dexterous manipulation,” *arXiv:2507.05331*, 2025.
- [51] A. Kumar, J. Hong, A. Singh, and S. Levine, “When should we prefer offline reinforcement learning over behavioral cloning?” in *ICLR*, 2022.
- [52] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, “Asymmetric actor critic for image-based robot learning,” *arXiv:1710.06542*, 2017.