MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Learning Non-prehensile Manipulation with Force and Vision Feedback Using Optimization-based Demonstrations

Shirai, Yuki; Ota, Kei; Jha, Devesh K.; Romeres, Diego

TR2026-011 January 07, 2026

Abstract

Non-prehensile manipulation is challenging due to complex contact interactions between objects, the environment, and robots. Model-based approaches can efficiently generate complex trajectories of robots and objects under contact constraints. However, they tend to be sensitive to model inaccuracies and require access to privileged information (e.g., object mass, size, pose), making them less suitable for novel objects. In contrast, learning-based approaches are typically more robust to modeling errors but require large amounts of data. In this paper, we bridge these two approaches to propose a framework for learning closed- loop non-prehensile manipulation. By leveraging computationally efficient Contact-Implicit Trajectory Optimization (CITO), we design demonstration-guided deep Reinforcement Learning (RL), leading to sample-efficient learning. We also present a sim- to-real transfer approach using a privileged training strategy, enabling the robot to perform non-prehensile manipulation using only proprioception, vision, and force sensing without access to privileged information. Our method is evaluated on several tasks, demonstrating that it can successfully perform zero-shot sim-to-real transfer.

IEEE Robotics and Automation Letters 2026

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Learning Non-prehensile Manipulation with Force and Vision Feedback Using Optimization-based Demonstrations

Yuki Shirai[1], Kei Ota[2], Devesh K. Jha[1], and Diego Romeres[1]

***Abstract*—Non-prehensile manipulation is challenging due to complex contact interactions between objects, the environment, and robots. Model-based approaches can efficiently generate complex trajectories of robots and objects under contact constraints. However, they tend to be sensitive to model inaccuracies and require access to privileged information (e.g., object mass, size, pose), making them less suitable for novel objects. In contrast, learning-based approaches are typically more robust to modeling errors but require large amounts of data. In this paper, we bridge these two approaches to propose a framework for learning closed-loop non-prehensile manipulation. By leveraging computationally efficient Contact-Implicit Trajectory Optimization (CITO), we design demonstration-guided deep Reinforcement Learning (RL), leading to sample-efficient learning. We also present a sim-to-real transfer approach using a privileged training strategy, enabling the robot to perform non-prehensile manipulation using only proprioception, vision, and force sensing without access to privileged information. Our method is evaluated on several tasks, demonstrating that it can successfully perform zero-shot sim-to-real transfer.**

***Index Terms*—Dexterous Manipulation, Learning from Demonstration, Deep Learning in Grasping and Manipulation, Multi-Contact Whole-Body Motion Planning and Control**

I. Introduction

NON-prehensile manipulation, such as pivoting, pushing, and sliding, plays an important role in enhancing the dexterity of robotic systems [1], [2]. These skills allow robots to interact with the environment more flexibly, enabling them to adapt to a wide range of tasks without requiring secure grasps. However, achieving such skills is challenging due to the inherently complex contact interactions (e.g., making-breaking contact, sliding-sticking contact). These interactions introduce non-smooth dynamics that are difficult to model and control as the number of contacts increases.

Model-based optimization methods, such as CITO and Model Predictive Control (MPC) [3], [4], [5], have demonstrated impressive performance, particularly in generating diverse trajectories at low computational cost. However, since these methods, in general, rely on simplified models of manipulation, they can be highly sensitive to uncertainties due to model inaccuracies. More critically, they often rely on offline system identification or online estimation of privileged information, such as object properties or contact states. This dependency limits the applicability of model-based controllers in real-world scenarios.

Learning-based methods, such as RL, have also shown impressive performance, especially in their robustness against various sources of uncertainty [6], [7], [8], [9]. These methods can operate without privileged information by directly learning policies from raw observations. However, they typically require a large number of training samples, resulting in long training times. This is especially problematic in non-prehensile manipulation, where the policy must reason object pose, contact locations, and contact forces from indirect and partial observations. As a result, RL may fail to discover viable solutions within a reasonable training time.

In this paper, we propose a framework that integrates the strengths of model-based planning with learning-based policy execution for non-prehensile manipulation. As illustrated in Fig. 2, we first employ CITO to collect a large number of task demonstrations. Second, a base policy is trained in a simulator using RL, leveraging the demonstrations (e.g., robot, object, & contact trajectories) generated by CITO. Third, we train a privileged information estimator to predict the privileged information required by the policy from observations. Finally, we evaluate the trained policy over various baselines in both simulation and hardware experiments, achieving zero-shot sim-to-real transfer. Our contributions are as follows.

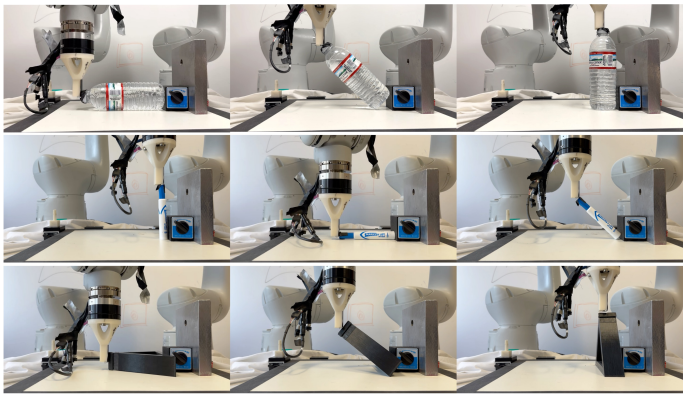
- A framework for learning contact-rich non-prehensile manipulation controllers and estimators by leveraging demonstrations generated by CITO.
- A sim-to-real transfer approach using a privileged information estimator to reconstruct required privileged information from temporal visual and force observations.

II. Related Work

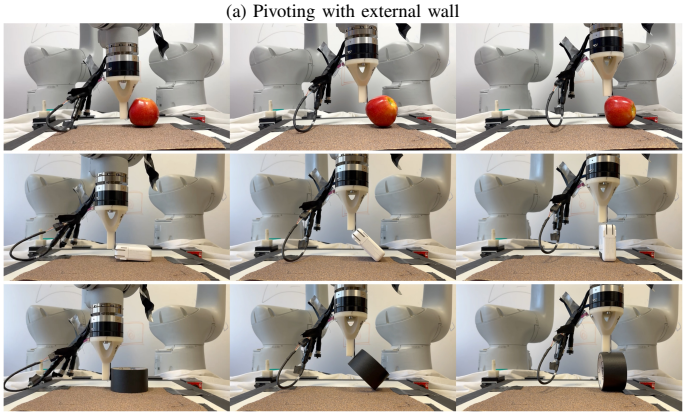
Model-Based Optimization for Contact-Rich Manipulation. Optimization-based methods have enabled various non-prehensile manipulation skills [5], [10], [11], [12]. These methods design manipulation skills computationally efficiently by leveraging techniques such as contact smoothing [13] and mixed-integer optimization [11], [14], [15]. However, these methods typically require privileged information (i.e., full-state feedback), which becomes increasingly impractical as task complexity grows. We relax such assumptions by adopting an RL approach, while still leveraging CITO to generate a large number of demonstrations. This strategy enables the agent to learn manipulation skills significantly more efficiently than standard RL methods.

[1]Y. Shirai, D. K. Jha, and D. Romeres are with Mitsubishi Electric Research Laboratories, USA yukishirai1926@gmail.com, devesh.dkj@gmail.com, romeres@merl.com

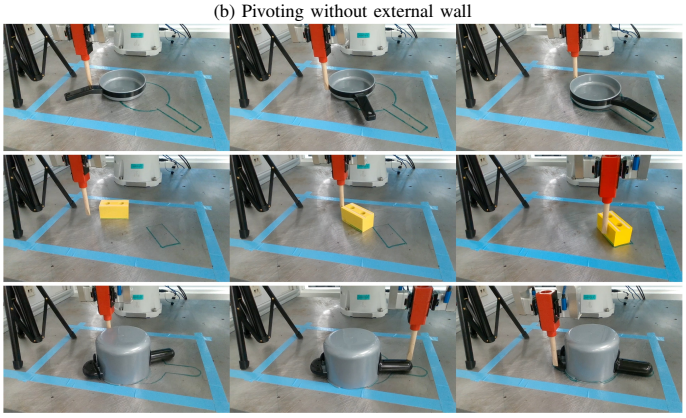
[2]K. Ota is with Mitsubishi Electric, Japan dev.ohtakei@gmail.com



(a) Pivoting with external wall



(b) Pivoting without external wall



(c) Pushing

Fig. 1: Snapshots of successful manipulation on diverse objects in real-world. Additional examples are provided in the accompanying video.

Learning Contact-Rich Manipulation. Learning-based methods have demonstrated remarkable success in robotic manipulation [16], [17], [18], [19]. However, all of these methods require a large number of training samples, resulting in long training times. To improve sample efficiency, demonstration-guided RL has been studied [20], [7], [21], where the demonstrations are used to guide exploration of RL agent to learn the policy and improve sample efficiency. However, these works [22], [23], [24] only consider kinematically feasible demonstrations. Incorporating contact force information into demonstrations could be critical to learn fine manipulation due to very thin margins of error imposed by contact constraints. Some works [25], [26] leverage human demonstrations to capture contact forces, but collecting such data at scale is challenging and often requires significant manual effort. In contrast, we use CITO to automatically generate robot, object, and contact force trajectories, providing richer supervision and greater scalability.

Bridging the sim-to-real gap is another key challenge. Privileged information used during training is often unavailable during real-world deployment. One line of work addresses this issue through student–teacher policy distillation, where a teacher policy is trained with privileged inputs and a student policy is subsequently trained via behavior cloning to imitate the teacher using only sensor observations [27], [28], [17]. Another strategy trains privileged-information estimators to recover latent physical variables (e.g., pose, friction, mass) from sensor observations, allowing a privileged-information policy to be deployed without modification [29], [30], [31], [32]. While conceptually related, these estimators are typically designed for settings where the latent variables are smooth and proprioception provides sufficient information. Also, they often rely on restrictive assumptions (e.g., fixed object size [31], [32]). In contrast, contact-rich non-prehensile manipulation requires estimating highly discontinuous latent states (e.g., sliding–sticking transitions, extrinsic contacts) and handling objects with variable geometries. By leveraging temporal histories of force measurements and segmentation images, our privileged information estimator avoids these limitations and generalizes to novel objects.

III. Method

In this section, we present our proposed framework, as shown in Fig. 2. The objective is to learn non-prehensile manipulation using only proprioceptive, visual, and force sensing. The proposed framework consists of three steps. In Step 1, task demonstrations are generated using CITO. In Step 2, a base policy which has access to the privileged information is trained using RL with the sampled demonstrations collected in Step 1. In Step 3, a privileged-information estimator is trained to estimate the privileged information, which serves as input to the base policy. The base policy with the predictions of the trained estimator is ultimately deployed on physical hardware for real-world validation.

While one could in principle train an RL policy end-to-end from raw visual observations, such approaches typically require substantially more samples and are unstable in real-robot settings [17]. Instead, we train the base policy leveraging privileged information for efficient training, and then the privileged information estimator learns to infer this information from raw sensory inputs (vision and touch).

In this work, we make the following assumptions: (1) both the objects and the robots are rigid, (2) manipulation occurs under quasi-static condition in SE(2), and (3) the robot end-effector pose, camera sensing, and robot contact force measurements are available throughout manipulation.

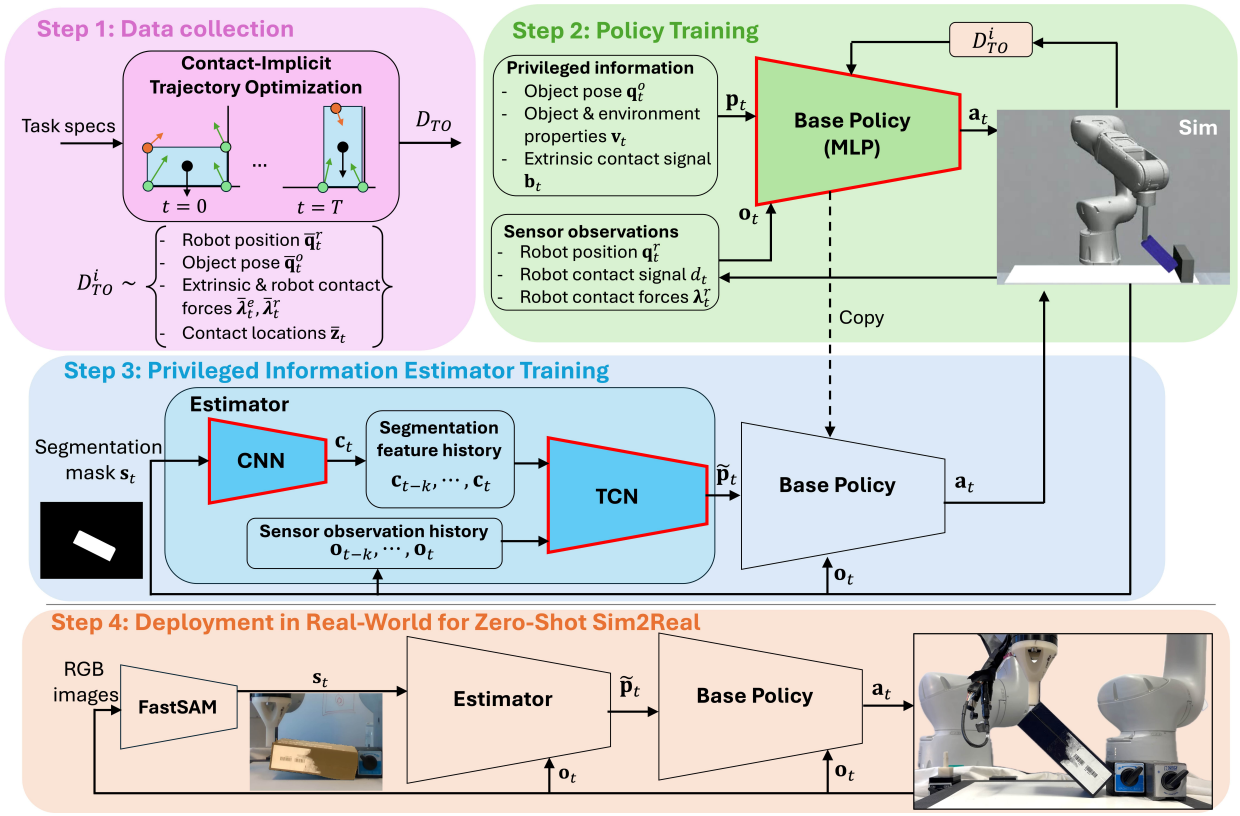


Fig. 2: Overview of our proposed framework. Trainable modules have red edges. **Step 1**: We collect data using CITO given a user-specified task. **Step 2**: The base policy is trained using RL with privileged information and sensor observations, leveraging the demonstrations D_{TO} collected in Step 1. Different RL variants are obtained by running this training pipeline with different rewards in (2). **Step 3**: The privileged information estimator is trained to estimate the privileged information. The estimator consists of a CNN and a TCN to process temporal sensor observations, including segmentation and force measurements. **Step 4**: During deployment in real-world, the learned estimator and policy run in zero-shot sim-to-real transfer on physical hardware.

A. Step 1: Collecting Demonstrations

We collect a large set of datasets using CITO in [33]. We consider the following CITO:

$$\begin{aligned} &\min_{\bar{\mathbf{q}}_t, \dot{\bar{\mathbf{q}}}_t, \bar{\mathbf{y}}_t} \sum_{t=0}^{T} \|\bar{\mathbf{q}}^o{}_t - \bar{\mathbf{q}}^{\text{ref}}_t\|^2_Q \\ &\text{s. t. }, f_{\text{dyn}}\left(\bar{\mathbf{q}}_t, \bar{\mathbf{q}}_{t+1}, \dot{\bar{\mathbf{q}}}_t, \bar{\mathbf{y}}_t\right) = \mathbf{0}, g\left(\bar{\mathbf{q}}_t, \dot{\bar{\mathbf{q}}}_t, \bar{\mathbf{y}}_t\right) = \mathbf{0}, \end{aligned} \tag{1}$$

where $\bar{\mathbf{q}}_t := [\bar{\mathbf{q}}^o_t, \bar{\mathbf{q}}^r_t]$ and $\bar{\mathbf{y}}_t := [\bar{\boldsymbol{\lambda}}^e_t, \bar{\boldsymbol{\lambda}}^r_t, \bar{\mathbf{z}}_t]$. $\bar{\mathbf{q}}^o_t \in \mathbb{R}^3$ represent an object pose in SE(2) and $\bar{\mathbf{q}}^r_t \in \mathbb{R}^2$ represent a robot end-effector position in SE(2), respectively. The end-effector orientation is kept fixed throughout the task. $\bar{\boldsymbol{\lambda}}^e_t \in \mathbb{R}^{2\times N_e}$ and $\bar{\boldsymbol{\lambda}}^r_t \in \mathbb{R}^2$ represent contact forces between an object and the environment, and between an object and the robots, respectively. N_e represents the potential extrinsic number of contacts between the object and the environment. We denote $\bar{\mathbf{z}}_t \in \mathbb{R}^{2\times N_e}$ as the extrinsic contact location between the object and the environment. $\bar{\mathbf{q}}^{\text{ref}}_t \in \mathbb{R}^3$ represents the linear interpolation between the start and goal object pose with T steps. Note that the cost penalizes only the object pose deviation from the reference trajectory, and no reference is imposed on the robot end-effector. We use the subscript t to represent the timestep t. We denote f_{dyn} as non-smooth dynamics of the non-prehensile manipulation, including nonsmooth contact switching, force and moment balance, and friction cone constraints. We denote g as non-dynamics related constraints, such as bounds of decision variables and collision-avoidance. We emphasize that the generation of trajectories that satisfy kinematic feasibility alone and not dynamic feasibility are simple to obtain by removing some of the f_{dyn} constraints, such as force and moment balance constraints. Thus, we denote kinematically feasible dynamics as f_{kin}. The problem (1) is solved using solvers such as Gurobi [34]. See [33] for more details. Solving (1) generates N demonstrations $D_{\text{TO}} := \{D^i_{\text{TO}}\}_{i=1}^N$, where $D^i_{\text{TO}} := \{\{\bar{\mathbf{q}}_t\}_{t=0}^T, \{\bar{\mathbf{y}}_t\}_{t=0}^T\}^i$. While previous works (e.g., [22], [23], [24]) only consider $\bar{\mathbf{q}}_t$ with f_{kin}, this work explicitly considers $\bar{\mathbf{q}}_t$ and $\bar{\mathbf{y}}_t$ with f_{dyn}. In particular, $\bar{\boldsymbol{\lambda}}^r_t$ guides for agents to learn robot motion direction, while $\bar{\boldsymbol{\lambda}}^e_t$ and $\bar{\mathbf{z}}_t$ offer insights into preferred extrinsic contacts.

B. Step 2: Learning Privileged Base-Policy

A base policy is trained to achieve the desired non-prehensile manipulation in a simulation where privileged information is accessible. We formulate the problem as a Markov Decision Process as follows.

States. States consist of the privileged and non-privileged information. The privileged information $\mathbf{p}_t$ includes the object pose $\mathbf{q}^o_t \in \mathbb{R}^3$, the object and environment properties $\mathbf{v}_t \in \mathbb{R}^{N_p}$, and the extrinsic contact signal $\mathbf{b}_t \in \mathbb{Z}^{N_e}$. The object pose $\mathbf{q}^o_t$ lies in the SE(2) and consists of two positional components and one orientation. $\mathbf{v}_t$ encodes physical properties, which are the mass and size of the object, and

the friction constants of both the object and the surrounding environment geometry. The extrinsic contact signal $\mathbf{b}_t$ is a binary vector where each element indicates whether a specific face of the object is in contact with a predefined environment surface (e.g., wall, table).

The non-privileged information $\mathbf{o}_t$ consists of the robot positions $\mathbf{q}_t^r \in \mathbb{R}^2$, the binary robot contact signal $d_t \in \mathbb{Z}^1$, and the 2D contact forces $\boldsymbol{\lambda}_t^r \in \mathbb{R}^2$ measured by force sensors mounted on the robots' wrists. All observations are approximately normalized to lie in the range $[-1, 1]$.

Actions. We consider linear translational actions in SE(2) for each robot, denoted as $\mathbf{a}_t^2$. Specifically, each action represents a relative position command for the robots' end-effector. These action commands are converted into joint torques using Operational Space Control (OSC) [35].

Rewards. Based on how demonstrations are used, we consider three distinct reward formulations. We denote three different RL polices using different demonstrations (i.e., using different reward formulation) as (1) *Vanilla RL*, which does not use any demonstrations, (2) *Kinematics-conditioned RL*, and (3) *Dynamics-conditioned RL*. These policies are obtained by 3 different rewards defined as:

$$\begin{aligned} \text{Vanilla RL:} \quad & r = r_p + r_s + r_a \\ \text{Kinematics-conditioned RL:} \quad & r = r_p + r_s + r_a + r_{\text{kin}} \\ \text{Dynamics-conditioned RL:} \quad & r = r_p + r_s + r_a + r_{\text{dyn}} \end{aligned} \tag{2}$$

First, the progress reward for the pivoting tasks is $r_p = \alpha_1 \left(\frac{\pi}{2} - \theta_e\right) + \alpha_2 \left(\theta_e^2\right)$, where $\theta_e = \arccos\left(\frac{1}{2}\left(Tr\left(R^{\text{G}} R\right) - 1\right)\right)$. $Tr(\cdot)$ denotes the matrix trace, and R and R^{G} are the goal and current rotation matrices, respectively. θ_e measures the angular deviation between the current and goal orientations, and $\frac{\pi}{2}$ is added as the offset. While the linear term in r_p is used in [36], [37], our experiments reveal that the inclusion of the quadratic term is necessary to achieve higher success rates under domain randomization (DR) [38] over the size of the objects, which was not discussed in [37]. The progress reward for the pushing task is $r_p = \alpha_1 \|\mathbf{q}_t^o - \mathbf{q}_{\text{goal}}^o\|_Q^2$, where $\mathbf{q}_{\text{goal}}^o$ is the desired goal state of the object. Second, the sparse success reward is defined as $r_s = \alpha_3 \mathbb{I}_G\left(\mathbf{q}_t^o\right)$, where $\mathbb{I}_G$ is the indicator function over the goal set $G := \left\{\mathbf{q}_t^o \in \mathbb{R}^3 \mid \|\mathbf{q}_t^o - \mathbf{q}_{\text{goal}}^o\| \leq \epsilon_s\right\}$, where ϵ_s is the user-specified positive constant. Third, the action smoothness reward is given by $r_a = \alpha_4 \|\mathbf{a}_{t-1} - \mathbf{a}_t\|^2$, for avoiding non-smooth actions.

Next, we define the reward based on demonstrations generated by CITO. For the kinematic reward r_{kin}, we use object and robot poses $\bar{\mathbf{q}}_t$ and extrinsic contact locations $\bar{\mathbf{z}}_t$ obtained by solving (1) with f_{kin}. Note that contact force demonstrations are not available in this setting, as f_{kin} does not have dynamics constraints. Thus, we compute r_{kin} as:

$$r_{\text{kin}} = \alpha_5 ||\mathbf{q}_t^r - \phi(\mathbf{q}_t^o)||^2 \tag{3}$$

where ϕ retrieves the closest reference robot configuration $\bar{\mathbf{q}}_t^r$ corresponding to the current object observation $\mathbf{q}_t^o$. Since both the object and environment parameters are sampled from a known dataset D_{TO} during simulation, the corresponding object reference trajectory $\bar{\mathbf{q}}_t^o$ is known. Using the current observation, we identify the closest object configuration within this trajectory, and consequently retrieve the closest robot configuration. This reward term encourages the robot to follow the kinematically feasible behaviors.

Similarly, we define the dynamics reward r_{dyn} by utilizing the demonstration $\bar{\mathbf{q}}_t$ and $\bar{\mathbf{y}}_t$ obtained by solving (1) with the dynamics model f_{dyn}:

$$r_{\text{dyn}} = \alpha_6 ||\mathbf{q}_t^r - \phi(\mathbf{q}_t^o)||^2 + \alpha_7 \arccos\left(\frac{\boldsymbol{\lambda}_t^r \cdot \psi(\mathbf{q}_t^o)}{||\boldsymbol{\lambda}_t^r|| ||\psi(\mathbf{q}_t^o)||}\right) + \alpha_8 \mathbf{b}_t \tag{4}$$

where ψ retrieves the closest reference robot contact forces $\bar{\boldsymbol{\lambda}}_t^r$ corresponding to $\mathbf{q}_t^o$, following the same logic as ϕ. This reward encourages the robot to follow the dynamically feasible behaviors. In particular, the arccosine term in r_{dyn} encourages the robot to perform a similar contact force direction as the demonstration shows. Importantly, we do not enforce matching the magnitude of the contact force, as we observe significant discrepancies between the dynamics model by f_{dyn} and those in simulators, leading to a potential sim2sim gap in contact force magnitudes. Hence, this work focuses on the direction of contact forces. The term $\mathbf{b}_t$ is used to count if the desired extrinsic contact states occur. The constants $\alpha_{i=1,2,3,8}$ are positive and the others are negative.

C. Step 3: Learning Privileged Information-Estimator

The goal of this step is to train the privileged information estimator only using sensor observations to predict the privileged information. Our estimator predicts the raw privileged information rather than learning a latent representation jointly with the policy. This choice preserves interpretability and allows the trained RL policy to be used without modification, enabling the estimator to serve as a drop-in replacement for privileged inputs during deployment.

We empirically observe that sensor observations alone are sufficient for the object whose geometry is in-distribution with the dataset. However, their reliability declines when there is uncertainty in object size, which is common when manipulating novel objects. To address this, we additionally incorporate vision inputs to improve the estimation of the privileged information. Directly using RGB images is avoided due to potential noise, and employing 3D point clouds is excluded due to their significant computational cost (see [17]). Instead, we leverage the object segmentation $\mathbf{s}_t$ derived from the RGB image, providing a compact but informative representation of the object.

Therefore, we define a privileged information encoder that takes the history of the sensor observations, $[\mathbf{o}_{t-T}, \cdots, \mathbf{o}_t]$, and the history of the segmentation features, $[\mathbf{s}_{t-T}, \cdots, \mathbf{s}_t]$. Since $\mathbf{s}_t$ is high-dimensional, we first apply a Convolutional Neural Network (CNN) to compress the segmentation into a lower-dimensional feature representation $\mathbf{c}_t$. Using the temporal histories of $\mathbf{o}_t$ and $\mathbf{c}_t$, we use a Temporal Convolutional Network (TCN) [39] to estimate the privileged information. We train CNN and TCN jointly via supervised learning using datasets collected by rolling out the base policy in the sim-

ulator under domain randomization. The supervised learning objective is to minimize the following loss function:

$$l = ||\mathbf{p}_t - \tilde{\mathbf{p}}_t||^2, \tag{5}$$

where $\mathbf{p}_t$ is the ground-truth privileged information and $\tilde{\mathbf{p}}_t$ is the estimated output from the privileged information encoder.

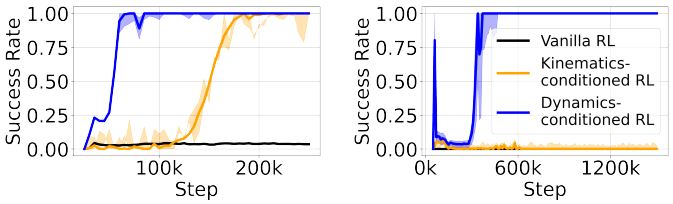
IV. Experiment Setup

We validate our framework across three distinct tasks (see Fig. 5): **Pivoting with Wall**: pivoting a box using an external wall, **Pivoting without Wall**: pivoting a box without relying on external support, **Pushing**: pushing an object on a table. For the second task, the table surface must provide very high friction. In simulation, increasing the friction coefficient alone was insufficient to replicate the real-world behavior. As a workaround, we add a thin virtual wall of height $1\,\mathrm{mm}$ to simulate the effect of high-friction contact (see Fig. 5). In the hardware experiments, we test on a variety of previously unseen objects (see Fig. 1 and the accompanying video) to assess generalization beyond the training set. We define a trial as successful if the final orientation error satisfies $|\theta_e| \leq 0.087$ rad (i.e., $5°$) for all tasks. We describe the setup as follows.

Demonstration Setup. We use the method in [33], randomizing object and environment parameters to generate diverse demonstrations. We randomize the mass of the object, the friction constant of the object and the environment, and the size of the object. For each task, to ensure sufficient coverage of contact-rich behaviors, we collect 5000 demonstrations using 30 Intel i9-13900K CPU cores. Each dataset is generated within minutes: about 4 minutes (pivoting with wall), 2.5 minutes (without wall), and 8.9 minutes (pushing).

Base Policy Setup. We train the base policy in MuJoCo simulator [40] using robosuite [41] as a wrapper. The agent is trained using Soft Actor Critic (SAC) [42]. For SAC, we use Multi-Layer Perceptron (MLP) for both actor and critic networks. The simulation runs at 500 Hz, while the policy operates at 10 Hz. For each episode, we set the maximum episode length to 300 steps. Overall, training converges within 4 hours on a single NVIDIA RTX 4090. During training, we apply domain randomization over the objects' mass and size, the friction constants of both the object and the environment (see Table V), and the controller gains used in OSC within robosuite. Furthermore, we introduce sensor noise to both privileged information and sensor observations to account for the estimation errors from the estimator.

Privileged Information Estimator Setup. We first rollout the trained base policy over 2000 episodes and collect a dataset containing ground-truth privileged information, sensor observations, and corresponding segmentation images (640×480 resolution) of the object using MuJoCo's rendering functionality. During data collection, we augment the segmentation images by introducing noise, such as randomly flipping, translating, and rotating segmentation masks, to improve robustness (see Table VI). We then train the estimator via supervised learning, minimizing the loss function (5) over multiple epochs. We use $T = 5$ step history of the observations for training corresponding to 0.5 second. Overall, training converges with 10 epochs (1 hour roughly), depending on the range of domain randomization.



(a) With external wall (b) Without external wall

Fig. 3: Learning curves for different RL training runs. Solid lines indicate average success rates, and shaded regions denote standard deviation across three different seeds. Every 10k step, the current policy is evaluated over 50 episodes, and the success rate is plotted. See Fig. 8 for the pushing result.

Hardware Setup. We use a 6 DoF MELFA robot equipped with a stiffness controller and a 6-axis force/torque sensor. This hardware enables users to get robot end-effector positions and the force measurements in the world frame. For object segmentation, we use FastSAM [43] to generate multiple instance segmentations from an RGB image captured by an Intel RealSense D435 camera.

Baselines. We consider the following baselines. (1) MPC following [12]: MPC solves (1) in a receding-horizon manner, running at the same frequency as the base policy. This setup mirrors the sensing and modeling assumptions of model-based controllers, making MPC a representative baseline. (2) Behavior Cloning (BC): A supervised policy trained on the same demonstration dataset as the base, mapping observations directly to actions using a mean-squared error loss. For both MPC and BC, we provide privileged information, including object mass, size, and friction (identified offline), as well as object pose estimated via AprilTags.

V. Results

In this section, we aim to address the following questions:

1) Do demonstrations generated by CITO facilitate more effective and efficient learning?
2) How does the base policy's performance vary with different demonstrations?
3) How robust is the base policy against baselines?
4) How accurately can the privileged information estimator predict the privileged information?
5) Can the trained policies be successfully transferred to real-world hardware experiments?

Do demonstrations generated by CITO facilitate learning? Across the three tasks, we compare RL performance using different types of demonstrations, corresponding to the different reward formulations in (2). RL with kinematic demonstrations is comparable to prior works such as [22], [23], which only consider kinematically feasible trajectories. Overall, RL with dynamics-based demonstrations achieves the fastest learning as shown in Fig. 3 and Fig. 8. In particular, in the pivoting without external wall task, neither vanilla RL nor kinematics-conditioned RL was able to learn the skill. We attribute this to the task's tighter feasible action space. In contrast, dynamics-conditioned RL successfully learns the skill, benefiting from enriched demonstration.

TABLE I: Number of successful attempts in real.

Mass	Kinematics-conditioned RL	Dynamics-conditioned RL
50 g	2 / 5	**5** / 5
110 g	**5** / 5	**5** / 5
300 g	0 / 5	**5** / 5

TABLE II: Number of successful attempts in sim.

Task	MPC	BC	Dynamics-conditioned RL
With wall	98 / 100	2 / 100	100 / 100
Without wall	81 / 100	0 / 100	100 / 100
Pushing	70 / 100	11 / 100	100 / 100

How does the base policy's performance vary with different demonstrations? For the pivoting-with-wall task, we deploy both kinematics- and dynamics-conditioned RL policies on a real system using a box of mass 110 g. During deployment, we vary the mass values used as privileged information. Table I shows the success rates over five trials. We observe that dynamics-conditioned RL consistently outperforms kinematics-conditioned RL. While both policies are trained with access to privileged information, the dynamics-conditioned policy benefits from demonstrations that include contact force references. This enables the policy to learn physically grounded interaction behaviors during training, leading to greater robustness against variations in dynamic properties. In contrast, the kinematics-conditioned policy is trained with demonstrations that satisfy only geometric feasibility, making it more sensitive to changes in object properties. These results highlight the importance of dynamics-aware demonstrations in contact-rich manipulation tasks.

How robust is the learned policy against baselines? We evaluate robustness at two levels: (i) performance under nominal randomized variations in simulation, and (ii) robustness to physical disturbances during real-world deployment. First, we compare the robustness of our dynamics-conditioned RL policy against MPC and BC in simulation, evaluating over 100 random seeds with variations in initial configuration and object properties. As shown in Table II, both MPC and our dynamics-conditioned RL policy achieve high success rates, whereas BC performs significantly worse. We attribute this to a sim-to-sim gap: the TO used in (1) employs a simplified contact model that is not compatible with MuJoCo, making BC policies brittle and unable to generalize. While BC followed by RL fine-tuning can be an alternative, poor BC initialization makes this equivalent to training from scratch.

Next, we compare the robustness of a dynamics-conditioned RL policy against an MPC on the real-world pivoting-with-wall task. The true object length is 0.16 m, and both controllers receive the same privileged input. To isolate the effect of model mismatch, we intentionally perturb only this privileged input

TABLE III: Number of successful attempts in real.

	MPC	Dynamics-conditioned RL
+5 mm	5 / 5	5 / 5
−5 mm	0 / 5	5 / 5
−10 mm	0 / 5	0 / 5

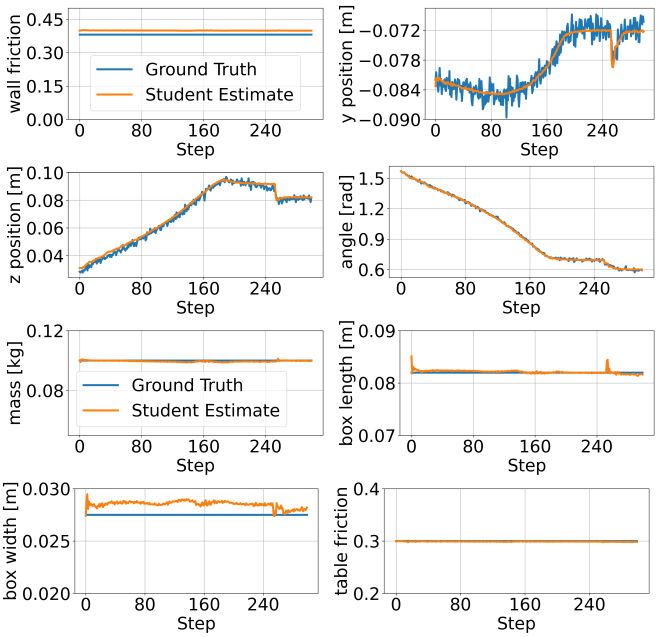


Fig. 4: Comparison of our privileged information estimator's predictions and the ground truth for the wall friction constant, y- and z-position of the object, orientation along x-axis, mass, length, and width of the box, and the table friction constant, for the pivoting with a wall.

during deployment. For example, a -5 mm offset means that the actual size of the box is shorter than what the controllers expect. As shown in Table III, both MPC and RL succeed when the actual object is longer than expected ($+5$ mm), as the contact with the wall is still maintained. However, when the actual object is shorter than expected (-5 mm), MPC fails completely, while RL remains successful. This suggests that the learned policy exhibits greater tolerance to moderate discrepancies in privileged information. At larger mismatches (-10 mm), even RL fails. These results highlight the importance of accurate privileged information and motivate us to develop reliable estimators.

Privileged Information Estimator Performance. We deploy the trained estimator and the policy in MuJoCo and collect both the ground-truth privileged information and the corresponding estimator's predictions. As shown in Fig. 4, our privileged information estimator can successfully predict the privileged information with reasonable accuracy.

Hardware Experiments. We deploy our policy and estimator on the real robot using zero-shot sim-to-real transfer. All objects used in hardware experiments are unseen during training. For all tasks, the policy successfully completed all 5 out of 5 trials, without access to privileged information as shown in Fig. 1.

To evaluate sim-to-real transfer, we deploy the learned dynamics-conditioned RL policy on both the simulation and hardware for the two pivoting tasks. The resulting object orientation trajectories over three trials are shown in Fig. 6. We observe a larger sim-to-real gap for the pivoting with external wall task than the pivoting without wall task. This is because for the pivoting with wall task, the object induces the sliding contact between the object and the wall, and between the

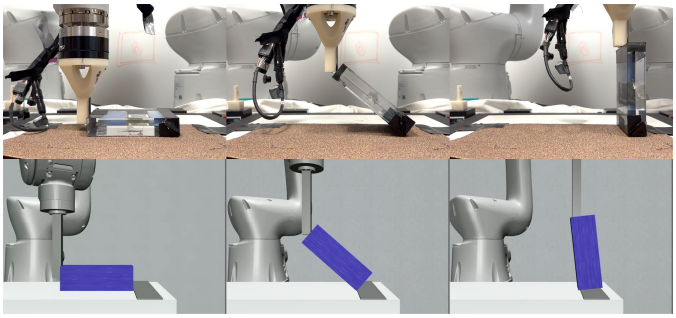
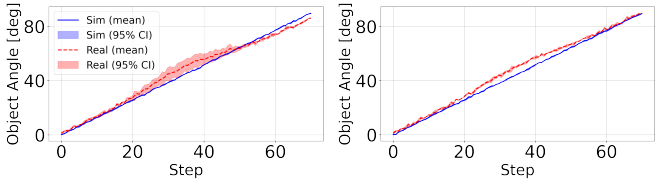


Fig. 5: Snapshots of pivoting without an external wall in sim and real-world.



(a) Pivoting with external wall (b) Pivoting without external wall

Fig. 6: Comparison of the object angle in simulation and the real-world during pivoting. We execute the same policy both in simulation and in hardware and collect the object orientation during manipulation over 3 trials. Due to sensor discrepancies and physical modeling differences, the resulting actions and motion can differ between simulation and hardware.

object and the table, which are challenging to model precisely in MuJoCo, leading to a larger sim-to-real gap. In contrast, the pivoting-without-wall task does not involve sliding contacts, resulting in better sim-to-real transfer.

VI. Conclusion

We present a framework for learning closed-loop controllers and estimators for contact-rich manipulation. We first leverage CITO to generate high-quality demonstrations. Then, we perform RL using these demonstrations for training a base policy, enabling sample-efficient learning. Furthermore, we train a privileged information estimator using only proprioception, vision, and force sensing, to predict such information that the policy uses. Our framework is evaluated over several tasks and achieves successful zero-shot sim-to-real transfer in real-world experiments.

Limitation. We assume that objects are rigid and manipulation occurs in SE(2). This limitation arises from the nature of CITO. The CITO we used in this paper [33] or other CITO methods [11] do not support such dynamics. As a result, the performance will decrease when handling deformable or when considering 3D manipulation. Another limitation is scalability to long-horizon manipulation. While our method can handle multiple contact transitions, extending it to significantly longer-horizon increases both the difficulty of generating demonstrations and the risk of error accumulation in the privileged-information estimator.

References

[1] M. T. Mason, "Toward robotic manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 1–28, 2018.

[2] A. Rodriguez, "The unstable queen: Uncertainty, mechanics, and tactile feedback," *Science Robotics*, vol. 6, no. 54, p. eabi4667, 2021.

[3] J.-P. Sleiman, J. Carius, R. Grandia, M. Wermelinger, and M. Hutter, "Contact-implicit trajectory optimization for dynamic object manipulation," in *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2019, pp. 6814–6821.

[4] S. Le Cleac'h, T. A. Howell, S. Yang, C.-Y. Lee, J. Zhang, A. Bishop, M. Schwager, and Z. Manchester, "Fast contact-implicit model predictive control," *IEEE Transactions on Robotics*, vol. 40, pp. 1617–1629, 2024.

[5] J. Moura, T. Stouraitis, and S. Vijayakumar, "Non-prehensile planar manipulation via trajectory optimization with complementarity constraints," in *2022 IEEE Int. Conf. Robot. Automat.* IEEE, 2022, pp. 970–976.

[6] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016.

[7] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

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

[9] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.

[10] A. Aydinoglu, A. Wei, W.-C. Huang, and M. Posa, "Consensus complementarity control for multi-contact mpc," *IEEE Trans. Robot.*, 2024.

[11] B. Aceituno-Cabezas and A. Rodriguez, "A global quasi-dynamic model for contact-trajectory optimization in manipulation," in *Robotics: Science and Systems Foundation*, 2020.

[12] Y. Shirai, D. K. Jha, and A. U. Raghunathan, "Robust pivoting manipulation using contact implicit bilevel optimization," *IEEE Transactions on Robotics*, vol. 40, pp. 3425–3444, 2024.

[13] Y. Shirai, T. Zhao, H. Suh, H. Zhu, X. Ni, J. Wang, M. Simchowitz, and T. Pang, "Is linear feedback on smoothed dynamics sufficient for stabilizing contact-rich plans?" *2025 International Conference on Robotics and Automation (ICRA)*, 2025.

[14] F. R. Hogan and A. Rodriguez, "Reactive planar non-prehensile manipulation with hybrid model predictive control," *The International Journal of Robotics Research*, vol. 39, no. 7, pp. 755–773, 2020.

[15] Y. Shirai, X. Lin, A. Schperberg, Y. Tanaka, H. Kato, V. Vichathorn, and D. Hong, "Simultaneous contact-rich grasping and locomotion via distributed optimization enabling free-climbing for multi-limbed robots," in *Proc. 2022 IEEE/RSJ Int. Conf. Intell. Rob. Syst.*, 2022, pp. 13 563–13 570.

[16] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE Int. Conf. Robot. Automat.* IEEE, 2017, pp. 3389–3396.

[17] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal, "Visual dexterity: In-hand reorientation of novel and complex object shapes," *Science Robotics*, vol. 8, no. 84, p. eadc9244, 2023.

[18] Z. Xu, R. Uppuluri, X. Zhang, C. Fitch, P. G. Crandall, W. Shou, D. Wang, and Y. She, "Unit: Data efficient tactile representation with generalization to unseen objects," *IEEE Robot. Autom. Lett.*, 2025.

[19] 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," *IEEE Robot. Autom. Lett.*, 2025.

[20] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv preprint arXiv:1707.08817*, 2017.

[21] L. Ankile, A. Simeonov, I. Shenfeld, M. Torne, and P. Agrawal, "From imitation to refinement–residual rl for precise assembly," *arXiv preprint arXiv:2407.16677*, 2024.

[22] K. Ota, D. Jha, T. Onishi, A. Kanezaki, Y. Yoshiyasu, Y. Sasaki, T. Mariyama, and D. Nikovski, "Deep reactive planning in dynamic environments," in *CoRL*. PMLR, 2021, pp. 1943–1957.

[23] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg, "Learning by watching: Physical imitation of manipulation skills from human videos," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 7827–7834.

[24] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE Int. Conf. Robot. Automat.* Ieee, 2018, pp. 5628–5635.

[25] Y. Hou, Z. Liu, C. Chi, E. Cousineau, N. Kuppuswamy, S. Feng, B. Burchfiel, and S. Song, "Adaptive compliance policy: Learning

approximate compliance for diffusion guided control," *arXiv preprint arXiv:2410.09309*, 2024.
[26] C. Chen, Z. Yu, H. Choi, M. Cutkosky, and J. Bohg, "Dexforce: Extracting force-informed actions from kinesthetic demonstrations for dexterous manipulation," *arXiv preprint arXiv:2501.10356*, 2025.
[27] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," in *Conference on robot learning*. PMLR, 2020, pp. 66–75.
[28] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science robotics*, vol. 7, no. 62, p. eabk2822, 2022.
[29] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," 2021.
[30] Z. Fu, X. Cheng, and D. Pathak, "Deep whole-body control: learning a unified policy for manipulation and locomotion," in *Conference on Robot Learning*. PMLR, 2023, pp. 138–149.
[31] Y. Fuchioka, C. C. Beltran-Hernandez, H. Nguyen, and M. Hamaya, "Robotic object insertion with a soft wrist through sim-to-real privileged training," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 9159–9166.
[32] J. D. A. Ferrandis, J. Moura, and S. Vijayakumar, "Learning visuo-tactile estimation and control for non-prehensile manipulation under occlusions," in *8th Annual Conference on Robot Learning*, 2024.
[33] Y. Shirai, A. Raghunathan, and D. K. Jha, "Hierarchical contact-rich trajectory optimization for multi-modal manipulation using tight convex relaxations," *2025 IEEE Int. Conf. Robot. Automat.*, 2025.
[34] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2024. [Online]. Available: https://www.gurobi.com
[35] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.
[36] X. Zhang, S. Jain, B. Huang, M. Tomizuka, and D. Romeres, "Learning generalizable pivoting skills," in *2023 IEEE Int. Conf. Robot. Automat.* IEEE, 2023, pp. 5865–5871.
[37] X. Zhang, C. Wang, L. Sun, Z. Wu, X. Zhu, and M. Tomizuka, "Efficient sim-to-real transfer of contact-rich manipulation skills with online admittance residual learning," in *Conference on Robot Learning*. PMLR, 2023, pp. 1621–1639.
[38] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 23–30.
[39] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.
[40] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.
[41] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, Y. Zhu, and K. Lin, "robosuite: A modular simulation framework and benchmark for robot learning," in *arXiv preprint arXiv:2009.12293*, 2020.
[42] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. Pmlr, 2018, pp. 1861–1870.
[43] X. Zhao, W. Ding, Y. An, Y. Du, T. Yu, M. Li, M. Tang, and J. Wang, "Fast segment anything," *arXiv preprint arXiv:2306.12156*, 2023.

APPENDIX

Training Base Policy Details. The training parameters are summarized in Table IV. The coordinate is illustrated in Fig. 7. In this work, we operate within the SE(2) group, restricting pivoting and pushing manipulation to the $y-z$ plane and $x-y$ plane, respectively. During the training of the base policy, we perform domain randomization and add sensor noises to robustify the policy, which is summarized in Table V.

Privileged Information Estimator Details. To train the privileged information estimator, we roll out the base policy in simulation and collect privileged information, sensor observations, and segmentation masks under the same randomization ranges used for base training. To better reflect real-world sensing, we augment segmentation masks as shown in Table VI.

TABLE IV: Hyperparameter setup for the base policy. Note that $\alpha_{i\in[1,\cdots,8]}$ are the coefficients of the reward terms in (2).

Parameter	Value
total # of steps	300k for pivoting with-wall task 1500k for without-wall task 2000k for pushing task
batch size	4096
max # of step for timeout	300
Networks	[128, 128] MLP
learning rate for policy	1e-4
learning rate for Q function	3e-4
discount factor	0.9
replay buffer size	1e6
# of episodes for evaluation	50
# of episodes for warmstart	50k
$[\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8]$	[1, 0.075, 10, -1, -50, -50, -0.005, 5]

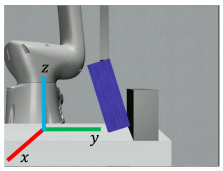


Fig. 7: Definition of world frame used in this work.

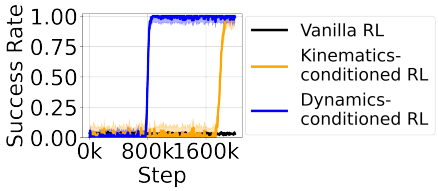


Fig. 8: Learning curves for different RL training runs for the pushing task. See Fig. 3 for other tasks.

TABLE V: Dynamics randomization and sensor noise. $\mathcal{N}(\mu, \sigma)$ denotes a Gaussian distribution with mean μ and standard deviation σ, and $\mathcal{U}(a, b)$ denotes a uniform distribution over the interval of $[a, b]$. A $+$ symbol indicates that the sampled noise is added to the original parameter value.

Parameter	Range
object mass	$\mathcal{U}(0.04, 0.4)$ kg
friction for table and wall	$\mathcal{U}(0.01, 0.4)$
friction for objects	$\mathcal{U}(0.2, 0.7)$
friction for robots	$\mathcal{U}(0.7, 1.7)$
object size scale	$\mathcal{U}(0.95, 1.05)$
proportional gain k_p in OSC	$\mathcal{U}(2000, 8000)$
derivative gain k_d in OSC	$2\sqrt{k_p}$
initial object position along y-axis	$+\mathcal{U}(-0.015, 0.015)$ m
initial robot position	$+\mathcal{U}(-0.015, 0.015)$ m
object position observation noise	$+\mathcal{N}(0, 0.015)$
robot position observation noise	$+\mathcal{N}(0, 0.00075)$
contact force observation noise	$+\mathcal{N}(0, 0.2)$

TABLE VI: Segmentation mask domain randomization parameters used during privileged information estimator data collection.

Noise Type	**Parameter**	**Value**
Erosion/Dilation	Probability	0.7
	Kernel size choices	$\{3, 5, 7\}$
	Erosion vs. dilation split	0.5
Random Holes	Number of holes	3
	Hole radius range	$[3, 9]$ pixels
	Hole probability	0.5
Full Mask Dropout	Probability	0.05
Flip Noise	Pixel flip probability	0.01
Edge Perturbation	Edge noise probability	0.75
Spatial Augmentation	Rotation range	$\pm 2.5°$
	Translation range	$\pm 7.5\%$
	Scaling range	$[0.95,\ 1.05]$