2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Zero-Shot Transfer of a Tactile-based Continuous Force Control Policy from Simulation to Robot

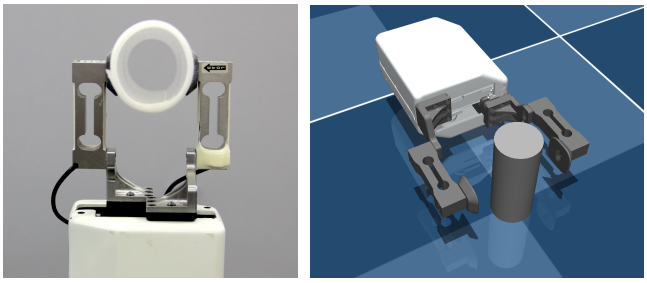
Luca Lach[1,2], Robert Haschke[2], Davide Tateo[3], Jan Peters[3,4], Helge Ritter[2], Júlia Borràs[1], Carme Torras[1]

***Abstract*— The advent of tactile sensors in robotics has sparked many ideas on how robots can leverage direct contact measurements of their environment interactions to improve manipulation tasks. An important line of research in this regard is grasp force control, which aims to manipulate objects safely by limiting the amount of force exerted on the object. While prior works have either hand-modeled their force controllers, employed model-based approaches, or not shown sim-to-real transfer, we propose a model-free deep reinforcement learning approach trained in simulation and then transferred to the robot without further fine-tuning. We, therefore, present a simulation environment that produces realistic normal forces, which we use to train continuous force control policies. A detailed evaluation shows that the learned policy performs similarly or better than a hand-crafted baseline. Ablation studies prove that the proposed inductive bias and domain randomization facilitate sim-to-real transfer. Code, models, and supplementary videos are available on `https://sites.google.com/view/rl-force-ctrl`**

I. INTRODUCTION

For humans and robots, tactile information is crucial in manipulation tasks involving environment contacts, visual occlusions, or both. As a consequence, efforts of the robotics community to include this essential sensor modality have heavily increased over the past years [1]–[3]. Prominent examples of tasks where tactile-based methods have shown successful include force control [4], [5], object pushing [6]–[8], and opening doors [9], [10].

The application domain of tactile sensors is as diverse as the sensing principles within the field. Complex sensors are commonly used in more challenging, high-level tasks such as surface following or edge prediction, which often involve deep learning methods [7], [11]–[14] or dexterous manipulation tasks [15], [16]. Low-level tasks like force control are commonly modeled by hand [4], [5], [17], [18]. Those approaches that employ machine learning either rely on classical methods, do not investigate sim-to-real transfer, or both [9], [19]–[22]. In contrast, this paper presents a deep reinforcement learning (DRL) approach for the low-level task of grasp force control for parallel-jaw grippers with two degrees of freedom (DoFs). Similar to [5], our controller has two distinct control objectives to ensure safe object grasping and holding: I) reaching and maintaining a given goal force, and II) minimizing object movements while closing and holding the object. Using normal forces measured at the fingertips, we train a continuous control policy in simulation and transfer it to the real robot. We employ an inductive bias [23], [24] and domain randomization [25], [26] to facilitate zero-shot sim-to-real transfer. In an extensive real-world evaluation, we compare our approach with the hand-modeled baseline from [5] and conduct an ablation study over the two proposed methods. Both the inductive bias and the domain randomization are vital for policies to generalize across real-world objects of various shapes, sizes, and softness and to achieve zero-shot sim-to-real transfer.



(a) Exemplary real-life grasp (b) Simulated scenario

Fig. 1: Safely holding an easily deformable object in real-life (a) vs. the simulation environment the policies are trained on (b).

In the following, we propose a simulation environment based on MuJoCo [27], where we tuned contact model parameters to match a few real-world samples. Then, we detail our learning process based on deep reinforcement learning, where we apply domain randomization and introduce a learning curriculum and an inductive bias to learn policies for subsequent zero-shot sim-to-real transfer. Lastly, we compare our policy to a hand-modeled force controller [5] and perform an ablation study on some model choices. Our main contributions are: i) a training procedure based on reinforcement learning that generalizes zero-shot to the real robot, ii) a novel simulation environment for 2-DoF grippers with realistic fingertip forces, and iii) open-sourcing the code for the environment, all methods, and their evaluation as well as CAD models of the sensorized gripper. To the best of our knowledge, this is the first paper proposing a tactile-based continuous grasp force controller learned with DRL, which was transferred to the real robot without further refinement.

II. RELATED WORK

Grasp Force Control In their review of human grasping, Johansson and Flanagan [28] highlight the importance of tactile sensations and divide the human grasping sequence

This work was supported by the European Union's Horizon 2020 Marie Curie Actions under grant no. 813713 NeuTouch and the Horizon Europe research and innovation program under grant no. 101070600 SoftEnable.

[1] Institut de Robòtica i Informàtica Industrial, CSIC-UPC. Correspondence: `llach@iri.upc.edu`
[2] Neuroinformatics Group, Technical Faculty, Bielefeld University
[3] Computer Science Department, Technical University Darmstadt
[4] German Research Center for AI (DFKI), Research Department: Systems AI for Robot Learning, Hessian.AI, Centre for Cognitive Science.

into distinct phases, where tactile events often mark phase transitions. Other experimental works have supported their findings [29], and many contributions from robotics have adopted this human-inspired approach [4], [5], [30], [31]. Romano et al. [4] closely follow the phase distinctions from [28] when modeling their force controller for the PR2 robot but assume the fingers to be equidistant to the object while grasping. Hsiao et al. [30] also proposed a force controller for the PR2 that was integrated into its grasping pipeline [32] but without the assumption of equidistant finger-object placement. These approaches hand-model controllers for grippers with a single degree of freedom, while our work relies on a learning approach that works for 2-DoF grippers.

Other works have proposed classical force controllers for end-effectors with two or more degrees of freedom. The authors in [31] designed a custom sensor that measures forces and proximity to ensure safe object grasps for their 2-DoF gripper. The studies [33] and [34] use two fingers of a multi-fingered hand to perform force control, where the former detects slippage and regulates the grasp force accordingly. The latter optimizes the fingertip positions based on force-closure constraints. Another body of works has presented force controllers for more complex, multi-fingered hands [17], [18], [35], [36]. Unlike our work, these studies focus on more complex end-effectors while not employing DRL methods.

Learning grasp force control has also become popular in recent years. In [20], the authors learn a grasping policy that controls forces on rigid objects in simulation, while others focus on increasing grasp success using tactile feedback [37]. Others have learned to control grasping forces for more complex tasks like door opening [10], high-precision assembly tasks [22], or surface tracking [11]. The work of [9] uses classical RL to learn force control policies, [19] combines deep learning with Gaussian Mixture Models, and [21] uses RL and admittance control to control forces in unknown environments. Although these works learn force control behaviors, none investigates the potential of learning in simulation only and transferring to the real robot afterward.

Sim-to-real transfer Sim-to-real transfer is widely used in robotics to avoid the time-consuming and labor-intensive task of real-world data collection [25], [26]. For policies without tactile information, a commonly used approach is domain randomization [25], [26] of the visual input [38], [39]. In the domain of optical tactile sensors, [6], [8] presented Tactile Gym, a simulation environment containing the TacTip [40], DIGIT [41], and DigiTac [42] sensors, and propose a domain adaptation approach using a generative adversarial network trained to mimic real-world tactile feedback in simulation. Later studies [43], [44] have reported successful sim-to-real transfer using Tactile Gym on various tasks, while others proposed simulators for the GelSlim [45], [46] and GelSight [47]–[49] sensors. In contrast to these studies, we focus on low-level force control tasks that solely require force measurements as inputs. Peng et al. [7] chose to learn and transfer an object-pushing policy for the Fetch robot. Their work relies heavily on domain randomization, and the authors argue that the robustness of the transferred policies was greatly increased. Ding et al. [12] have proposed a Unity-based simulation of the TacTip sensor for edge prediction and have achieved sim-to-real transfer through domain randomization. In a later work, Ding et al. [13] use MuJoCo to simulate a self-made tactile sensor array to open a cabin door. They also employ domain randomization for transferring the policy, but they binarized the sensor readings due to the low sensitivity of the built-in MuJoCo touch sensor. Although the authors in [39] also mention a potential lack of realism in simulated continuous force measurements, we find that continuous force control policies can indeed be learned in MuJoCo and then be successfully transferred to the real world without fine-tuning.

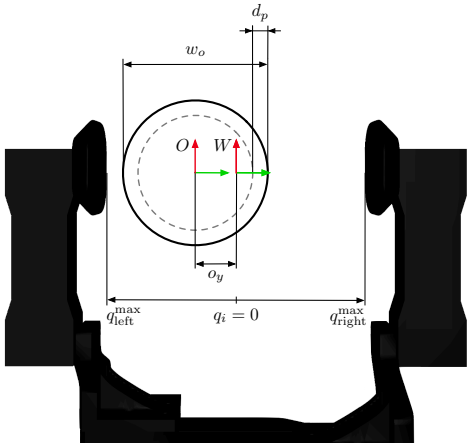


Fig. 2: Schematic overview of the grasping scenario with the world frame W and the object frame O. w_o is the object width, d_p its maximum deformation, and o_y its position on the world's y-axis.

Approaches based on Finite Element Methods (FEM) are capable of generating accurate simulation data of complex tactile sensors. In [50], [51], the authors use FEM to estimate deformations of a BioTac [52] sensor and synthesize simulated data by learning a latent space representation. Other papers have applied FEM to their custom-built soft tactile sensors [53]–[55]. Due to their high computational cost, FEM is typically not well-suited for data-driven approaches like DRL, which we use, unless some simplifying assumptions can be made [56].

III. FORCE CONTROL SIMULATION

To train force control policies, we first modeled TIAGo's 2-DoF parallel jaw gripper in MuJoCo with one tactile sensor per finger and an object of variable softness to grasp. The simulated tactile sensors were designed to mimic load cell sensors, as they can easily be integrated on TIAGo, as done in [5]. Similarly, the control frequency was set to 25 Hz to match that of a real TIAGo.

A. Grasping Scenario

A schematic overview of the grasping scenario is shown in Fig. 2, which details all parameters needed to define it. The gripper is depicted in its fully open state ($q_i = q_i^{\max} = 0.045$), with an object located between the fingers somewhere on

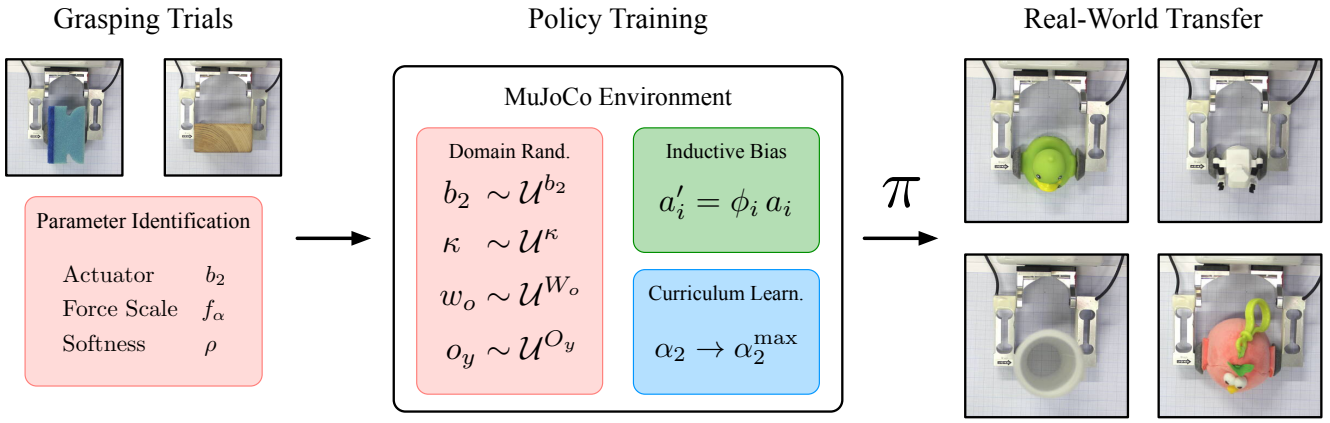


Fig. 3: Method overview. **Step I**: From a few real-world grasping trials with a position controller, MuJoCo parameter values were identified that mimic the real-world actuator behavior and force readings. **Step II**: For policy training, sampling ranges for the domain randomization were set based on the results from the prior step, an inductive bias that scales policy actions, and a learning curriculum were introduced. **Step III**: Zero-shot transfer of the policies to real-world grasps on objects of various shapes, sizes, and softness.

the grasping axis. W and O refer to the world and object frames, where W is considered fixed w.r.t. the gripper base and centered between the fingertips. If the object offset o_y is non-zero, the object center is displaced w.r.t. W, causing one fingertip to touch it earlier. The object width is defined by w_o, and d_p denotes the maximum penetration depth (or object deformation). Softer objects can be deformed more heavily and thus have larger values of d_p. The object's softness (see Sec. III-C) is chosen randomly in each episode, resulting in a variation of d_p. As the controller should learn to maintain any initial object position during grasping, we sample o_y during episode initialization, thus exposing the learner to different object-gripper offsets. w_o is also varied, such that the policy does not implicitly assume all objects to be of equal width. The following three constraints are imposed on the sampling of these parameters:

$$|o_y| + r_o < q^{\max} \tag{1}$$
$$r_o - d_p > |o_y| \tag{2}$$
$$d_p < r_o \tag{3}$$

where $r_o = \frac{1}{2} w_o$ is the object radius. Equation 1 ensures that the object doesn't initially collide with a finger and equation 2 that the fingers can fully close up to the maximum penetration without dislocating the object core. Finally, equation 3 states that the object deformation is limited by the object radius.

In order to facilitate sim-to-real transfer, we identified suitable ranges for various simulation parameters to match the behavior observed in a few simple grasping experiments performed in simulation and real-world. The detailed procedure will be discussed in the following subsections.

B. Actuator Behavior

First, we identify the actuator parameters within MuJoCo to match the actual robot behavior. TIAGo uses position controllers, and to keep control signals identical, we opt to use them in the simulation as well. Our policies output desired position *deltas*, which are subsequently integrated into absolute positions and forwarded to the controller. The joints are thus controlled by

$$q_i^{\text{des}} = q_i + u_i$$

at each simulation step t, where $u_i = \Delta q_i^{\text{des}}$ is the control signal generated by a policy. Using position deltas instead of absolute positions has several advantages for the learning process. First, the action space is symmetric and centered around zero with known bounds, making it easy to normalize while satisfying assumptions some RL algorithms make about the action space. Second, by constraining the control signals via $|u_i| \leq \Delta q^{\max}$, we can easily limit the maximum joint velocity, thus preventing the policy from executing erratic and potentially dangerous movements. We set $\Delta q^{\max} = 0.003$ in all experiments, representing the maximal velocity observed in real-world experiments.

Next, we tuned the MuJoCo actuators to match the behavior of the real TIAGo. First, `gainprm` and `ctrlrange` were set to $(100, 0, 0)$ and $[0.0, 0.045]$ to mimic the real joint actuation range $[0, 0.045]$ measured in meters (see Fig. 2). The damping parameter of `biasprm`, b_2, was tuned to match the finger's closing velocity and acceleration. To find a range of realistic values for b_2, we executed power grasps with the real robot and in simulation and then manually tuned b_2 until the joint trajectories matched. We found $b_2 = -9$ to mirror the actual behavior best and sample $b_2 \in [-13, -6]$ for domain randomization and robust actuator behavior.

C. Force & Softness Modeling

To yield similar contact force readings and object deformations in simulation and real-world, we compared the final forces $f(T)$ and penetration depths $d_p(T)$ achieved when closing the gripper with a constant action Δq^{des} on a soft and a rigid object (Sponge and Wood from Fig. 5). The corresponding real-world results, shown in Fig. 4, exhibit a linear relationship. Then, the simulation parameters were identified that determine the force magnitude and penetration depth. For the former, a factor f_α is introduced that scales the contact forces generated by MuJoCo and for the latter,

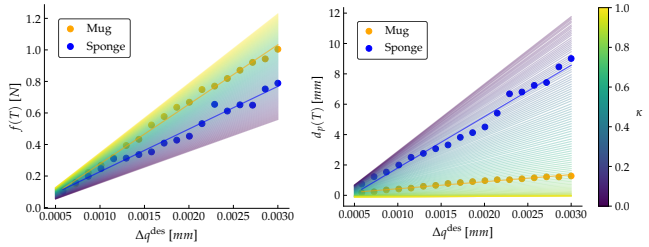


Fig. 4: Final gripper force and penetration depth when closing the real gripper with constant actions Δq^{des} on a soft and a rigid object.

the stiffness parameter k from MuJoCo's *solref* parameter tuple was identified. Both parameters were tuned using real-world experimental results. As both parameters are strongly correlated, we link them via an extra softness parameter κ, which is sampled once per episode between $[0, 1]$ and then used to determine f_α and k. In this fashion, we prevent unrealistic configurations, e.g., a soft object ($\kappa = 0$) with a high f_α or small k.

f_α was determined to be in the range $[0.5, 5]$, corresponding to the required scaling factor to match the slopes of the regression lines observed in real-world and simulation for the soft and rigid object respectively. κ was then used as an interpolation factor on this interval. Due to the non-linear dependence of d_p on k with $d_p \approx \frac{1}{k}$, linear interpolation on some interval is not viable for calculating k given κ. Instead, κ is transformed non-linearly using the derivative of their dependence ($\frac{1}{k^2}$), arriving at:

$$k(\kappa) = \frac{1}{\max(1.1\,\kappa^2, 0.001)}$$

where the scaling factor before κ determines the change of the slope of the regression lines in Fig. 4. During training, κ is sampled with $\kappa \sim \mathcal{U}(0, 1)$ to randomize the object's stiffness.

IV. LEARNING METHODS

As force control is a sequential decision-making problem, we can model it as a Markov Decision Process (MDP), allowing us to solve it using RL algorithms. We follow the classical formulation of an MDP, which is defined as a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, r, \gamma, \iota \rangle$, with the state space $\mathcal{S}$, the action space $\mathcal{A}$, the transition kernel $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^+$, the reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, the discount factor γ, and the initial state distribution $\iota : \mathcal{S} \to \mathbb{R}^+$. The objective of Reinforcement learning algorithms is to find the optimal policy π^* maximizing the expected discounted reward $\mathcal{J}(\pi) = \mathbb{E}_{\tau \sim P, \pi, \iota}\left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right]$, where τ is a trajectory composed of a sequence of states s_t and actions a_t generated by executing the policy π in the MDP. Our simulation environment defines the transition kernel P.

A. Observation Space

The agent receives the following observation at each time step t:

$$o(t) = \big(q_i(t), f_i(t), \Delta f_i(t), a_i(t-1), h_i(t)\big)$$

where subscript i indicates that the observation is given for joint i, $\Delta f_i = f^{\text{goal}} - f_i$ refers to the difference of the current force to the goal force, $a_i(t-1)$ to the previous action taken by the agent (with $a_i(0) = [0, 0]$) and h_i to the `had_contact` flag, which is defined as:

$$c_i(t) = \begin{cases} 1 & \text{if } f_i(t) > f_\theta \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$h_i(t) = c_i(t) \vee h_i(t-1) \tag{5}$$

where $c_i(t)$ indicates whether a finger is considered to be in contact with the object at time t by comparing the current force to a noise threshold f_θ, and $h_i(0) = 0$. h_i stays 1 once c_i gets 1, even if the finger loses contact again later. This flag provides important contextual information w/o the need for a recurrent policy (which may be more difficult to train) or a long history of observations. We add Gaussian noise to the joint position and fingertip force with $\sigma_q = 0.000027$ and $\sigma_f = 0.013$ and stack the observation $k = 3$ times for the policy to have access to a short history of position and force deltas so that it can estimate the object stiffness.

B. Action Space

The action space is simply a two-vector with one desired position delta per finger $(a_{\text{left}}, a_{\text{right}})^T$, where $a_i = \Delta q_i^{\text{des}}$ refers to an individual action for joint i. Each a_i is first clipped to lie within $[-1, 1]$ and then multiplied with $\Delta q^{\max}$, effectively denormalizing a_i. Additionally, we define a contact-state dependent inductive bias:

$$\phi_i = \begin{cases} \max(0.9, 1 - \frac{|\Delta f_i|}{f^{\text{goal}}}) & \text{if } h_i = h_j = 1 \\ 0.1 & \text{if } h_i = 1 \wedge h_i \neq h_j \\ 1 & \text{else} \end{cases}$$

that is multiplied with the individual actions at each time step, yielding $a_i' = \phi_i\, a_i$, which is then passed to the MuJoCo actuators. It is inspired by the human grasping phases [28] commonly used in other controllers [4], [5], and serves two purposes: The first line dampens the policy commands for in-contact states far from the force goal as a safety measure. The second line ensures that the in-contact finger is slowed down when the other finger is not yet in contact, thus ensuring that the object is not pushed away.

C. Reward Function

Our reward function mainly reflects the two controller objectives and adds a third term for smooth control. We propose the following individual reward terms:

$$r^{\text{force}} = 1 - \tanh\left(\sum_i |\Delta f_i|\right) \tag{6}$$

$$r^{\text{obj}} = \begin{cases} -1 & \text{if } \dot{o}_y > \dot{o}_y^{\max} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$r^{\text{act}} = -\sum_i |a_i(t-1) - a_i(t)| \tag{8}$$

The term r^{force} reflects the first control objective, namely, to reach and maintain the target force. $\tanh$ limits the summed force deltas to 1, and subtracting it from 1 yields

Fig. 5: Evaluation objects: Plush Toy, Rubber Mat, Sponge, Duck, Spray, TIAGo, Pringles, Banana, Wood, and Mug. Ordered by increasing stiffness from left to right.

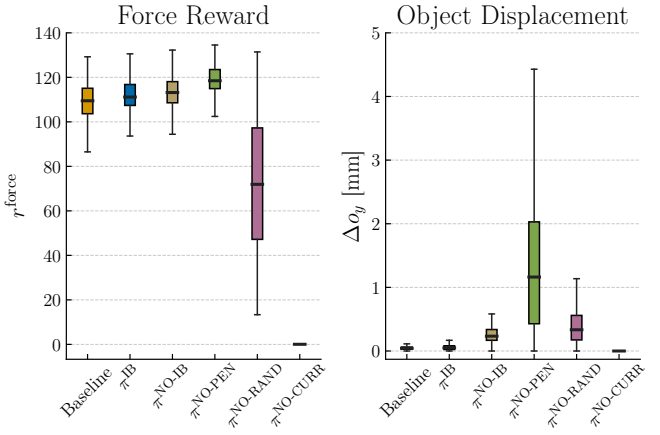


Fig. 6: Policy evaluation and ablation studies with 200 simulation trials per policy. Force reward and object displacement are shown separately to identify the impact of specific controller components on both metrics.

the highest reward if the goal forces are reached. The second objective, maintaining the object pose, is achieved via r^{obj} by penalizing large object motion with a negative reward of -1. In equation 7, $\dot{o}_y$ refers to the current object velocity and $\dot{o}_y^{\text{max}} = 5 \cdot 10^{-5}$ to the accepted velocity threshold, which is slightly larger than the observed noise level of $\dot{o}_y$ during finger contact. The third term, r^{act}, penalizes high changes in policy actions at consecutive time steps to encourage a smooth control behavior. Other, more involved procedures have been proposed before [57], but this simple penalty has shown to be sufficient for our purposes. Finally, the total reward per time step is defined as

$$r = \alpha_1 r^{\text{force}} + \alpha_2 r^{\text{obj}} + \alpha_3 r^{\text{act}} \tag{9}$$

D. Curriculum Learning

Our reward function, as given in equation 9, poses a challenging problem for learning: During early exploration, it is very likely that a random agent will push the object with one finger before it learns to control the grasping force precisely. As a result, the policy will converge to a local minimum avoiding contact with the object since it will receive a large negative reward from r^{obj} before being rewarded by r^{force}. We, therefore, employ a learning curriculum, a common approach to gradually increase the task complexity throughout training by annealing specific environment parameters [58], [59].

TABLE I: Annealed parameters with their initial and final values.

Parameter	Initial	Final
α_2	0	1.0
$\dot{o}_y^{\text{max}}$	2×10^{-4}	5×10^{-5}
w_o	$[0.020, 0.025]$	$[0.015, 0.035]$
o_y	$[0.0, 0.0]$	$[-0.040, 0.040]$

The parameters controlling the likelihood of these high negative rewards are α_2, the weight of r^{obj}, the range of possible object displacements o_y, the range of object radii w_o, and the object velocity threshold $\dot{o}_y^{\text{max}}$. Initial and final values are defined for the scalars α_2 and $\dot{o}_y^{\text{max}}$. Since o_y and w_o are sampled from parameter ranges, the limits of their respective sampling ranges are annealed. The annealing phase ends at timestep $t = s_{\text{end}}$, which is a hyperparameter.

E. Domain Randomization

In Sec. III, we described the intervals for different simulation parameters, such as the actuator bias or object softness, and several parameters defining specific task instances, such as the object width and offset o_y. At the beginning of each training episode, these parameters are sampled uniformly within their respective interval, e.g., $b_2 \sim \mathcal{U}^{b_2}(-13, -6)$. As described in the previous subsection, the interval borders for some parameters are annealed during training, as they impact episode difficulty. Fig. 3 provides an overview of our proposed method and its components that were described in the last two sections.

V. EXPERIMENTAL EVALUATION

For the experiments, we use Proximal Policy Optimization (PPO) [60] to train the grasping policies. We first evaluate our proposed method in simulation and then apply the policies to the real robot and compare them regarding force maintenance and object movements.

A. Simulation experiments

We train for a total of 4M steps with an episode length of 150 steps and anneal the parameters mentioned above until $s_{\text{end}} = 1.5$M. At the beginning of each episode, all randomization parameters are sampled anew. The policy network consists of two fully connected layers with 50 neurons each and ReLu activations. The output layer has two neurons, one for each finger's action output. We found a learning rate of $6 \cdot 10^{-4}$ to perform best, and the remaining parameters are set to their default values from stable-baselines3 [61]. We store network weights each time it surpasses the prior best reward averaged over 30 episodes.

We compare our policy with inductive bias π^{IB} to a Python implementation of the hand-crafted baseline controller presented in [5]. Additionally, we introduce four ablations on specific controller components to assess their impact on the task objectives, namely force and object pose maintenance. $\pi^{\text{NO-IB}}$ was trained without the inductive bias, $\pi^{\text{NO-PEN}}$ without object velocity penalty ($\alpha_2 = 0$), $\pi^{\text{NO-RAND}}$ with neither inductive bias nor domain randomization, and $\pi^{\text{NO-CURR}}$

TABLE II: Real-world experimental results for ten household testing objects. Our method π^{IB} performs the strongest of all policies in terms of reward and minimizes object movements as efficiently as the baseline controller. The other policies consistently perform worse.

Policy	Metric	Plush Toy	Rubber Mat	Sponge	Duck	Spray	TIAGo	Pringles	Banana	Wood	Mug	Average
Baseline	r^{force}	74 ± 12	92 ± 15	96 ± 14	90 ± 21	$\mathbf{114 \pm 5}$	84 ± 14	$\mathbf{118 \pm 7}$	83 ± 9	$\mathbf{108 \pm 7}$	115 ± 10	97 ± 12
	Obj.Mov.	1.0 ± 0.7	1.6 ± 0.8	$\mathbf{3.1 \pm 2.1}$	2.0 ± 0.9	1.9 ± 0.6	$\mathbf{1.6 \pm 0.8}$	$\mathbf{0.9 \pm 0.4}$	1.6 ± 0.7	1.0 ± 0.9	-	$\mathbf{1.6 \pm 0.9}$
π^{IB}	r^{force}	$\mathbf{89 \pm 13}$	$\mathbf{96 \pm 15}$	$\mathbf{103 \pm 7}$	$\mathbf{101 \pm 12}$	109 ± 17	$\mathbf{102 \pm 12}$	115 ± 16	$\mathbf{97 \pm 13}$	105 ± 15	$\mathbf{120 \pm 10}$	$\mathbf{104 \pm 13}$
	Obj.Mov.	$\mathbf{0.9 \pm 0.7}$	$\mathbf{1.5 \pm 0.8}$	3.2 ± 1.4	$\mathbf{1.9 \pm 0.7}$	$\mathbf{1.8 \pm 0.8}$	1.8 ± 0.9	1.2 ± 0.6	$\mathbf{1.3 \pm 0.7}$	$\mathbf{0.7 \pm 0.7}$	-	$\mathbf{1.6 \pm 0.9}$
$\pi^{\text{NO-IB}}$	r^{force}	86 ± 12	77 ± 10	74 ± 26	94 ± 15	88 ± 33	91 ± 20	98 ± 34	82 ± 9	101 ± 16	106 ± 17	88 ± 20
	Obj.Mov.	2.0 ± 1.1	2.8 ± 1.4	4.0 ± 0.9	3.1 ± 1.0	2.4 ± 0.7	2.95 ± 1.0	1.9 ± 0.6	2.3 ± 0.8	1.7 ± 0.4	-	2.6 ± 0.9
$\pi^{\text{NO-RAND}}$	r^{force}	75 ± 13	82 ± 12	97 ± 15	63 ± 22	57 ± 37	48 ± 12	53 ± 39	56 ± 18	49 ± 33	40 ± 41	64 ± 25
	Obj.Mov.	1.9 ± 1.2	2.6 ± 1.6	3.6 ± 1.3	2.5 ± 0.7	2.6 ± 0.9	3.3 ± 1.5	1.7 ± 0.7	2.8 ± 0.8	1.8 ± 0.9	-	2.5 ± 1.0

without curriculum learning (all values in table I were set to their final values already at the start of the training). For each policy, 2,000 simulation trials were executed where all environment parameters and f^{goal} were randomly sampled for each trial, summing up to 12,000 trials in total. Fig. 6 shows box plots for the force reward r^{force} and the object displacement for all policies.

The results show that our proposed approach π^{IB} achieves slightly better performance than the baseline regarding force reward and is nearly as good at preventing object movements during grasping. Without the inductive bias, the force control performance is unchanged, but object movements increase by approximately half a millimeter, as evident by the results of $\pi^{\text{NO-IB}}$. When comparing the object movements of both policies without the inductive bias ($\pi^{\text{NO-IB}}$ and $\pi^{\text{NO-RAND}}$) with that of $\pi^{\text{NO-PEN}}$, it is apparent that the object movement penalty alone minimizes object movements substantially, although not as effectively as the bias. $\pi^{\text{NO-PEN}}$ achieved the highest force reward among all six policies, as it does not halt finger movements upon object contact and thus collects force rewards already while pushing the object. The poor force control performance of $\pi^{\text{NO-RAND}}$ suggests that the policy overfitted the single environment configuration and is not able to generalize to differently parameterized environment instances. Finally, $\pi^{\text{NO-CURR}}$ achieved no force rewards at all and did not cause any object movements. An analysis of the behavior of $\pi^{\text{NO-CURR}}$ during the trials revealed that it simply learned not to move the fingers at all, always returning $a_i = 0$. Overall, the results indicate that our proposed method yields policies that slightly outperform a hand-crafted force controller and underline the efficacy of the individual components.

B. Real-World Evaluation

To assess whether our method is transferable to the real world without fine-tuning, we evaluate several policies on TIAGo using ten test objects of varying stiffness. We chose not to include $\pi^{\text{NO-CURR}}$ as it did not learn any rewarding behavior and neither $\pi^{\text{NO-PEN}}$ since it was clearly unable to minimize object movements. On the other hand, $\pi^{\text{NO-RAND}}$ is included in the real-world evaluation to test whether it performs well on a specific object type that is similar to the environment configuration it was trained on. We perform 20 grasping trials per object and method, yielding $20 \times 10 \times 4 = 800$ real-world trials in total. f^{goal} is sampled randomly, where the real-world experiment results from Sec. III-C

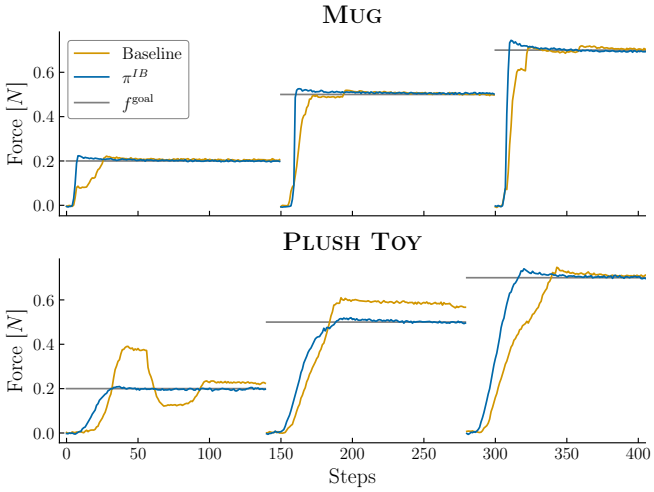


Fig. 7: Comparison of force trajectories between baseline and π^{IB} on two objects of different stiffness for $f^{\text{goal}} \in \{0.2, 0.5, 0.7\}$. While both policies perform similarly well on Mug, π^{IB} controls the target force much more reliably on the Plush Toy than the baseline.

were used to determine the upper and lower bounds of the sampling interval. In each trial, the object is offset to one finger; in half of the trials, it is placed closer to the left finger, and in the other half, closer to the right. Then, the policy is commanded to perform a grasp, and after 6 seconds (150 steps at 25 Hz), the gripper is automatically opened again. The process repeats after the reward is computed and the traveled distance is measured. Note that the force reward is not comparable between objects since it depends on the object's softness and width because they determine the amount of time a force reward can be achieved. Wider objects come in contact with the fingers earlier, and force rewards are generated in more time steps than for narrower objects. The softer an object is, the slower the force builds up, leading to a reduced reward. Instead of measuring and integrating the object velocity to calculate the total object displacement for each trial, we measured it by placing the objects on millimeter paper, annotating the start and end positions, and calculating the difference. No displacements for the Mug are reported as it is almost as wide as the gripper opening and would, therefore, not move during a grasp regardless of the policy. As a consequence of using millimeter paper, the reported real-world object displacement measurements are less precise than the ones in the simulation, meaning they can not be compared directly.

Table II shows the results of the real-world evaluation.

π^{IB} shows the strongest overall performance with an average reward of 104, while the baseline achieved an average reward of 97. To further analyze the performance difference between the two policies, we performed additional grasping trials with both policies on two objects with three different target forces f^{goal} and compared the force trajectories (see Fig. 7). Mug and Plush Toy were chosen for the comparison, as they are on opposite sides of the softness spectrum. The Plush Toy had the largest performance difference between the policies. On the rather stiff Mug, the performance of both policies is similar. While π^{IB} tends to slightly overshoot f^{goal} before converging, the baseline takes longer to reach the target force. For the Plush Toy, π^{IB} reliably reaches and maintains the target force, while the baseline fails in doing so for the two lower goal forces. The trajectory comparison indicates that our method generalizes better across objects of different softness than the hand-crafted baseline. Since the PID-based baseline controller overcomes steady-state errors by means of an integral term, it can be slow in adapting to them, thus having difficulties in accurately reaching f^{goal}. This behavior is especially apparent for small target forces on very soft objects like the Plush Toy. On the other hand, π^{IB} learned to react to discrepancies between current and desired grasping force immediately and more accurately. The performance of the two policies is comparable for several objects, with the baseline performing better on three objects by a small margin.

Furthermore, our experiments indicate that the inductive bias facilitates sim-to-real performance, as evidenced by the lower average force reward and higher object movements of $\pi^{\text{NO-IB}}$. When comparing $\pi^{\text{NO-IB}}$ to π^{IB}, the difference in performance was larger on the real robot than in simulation. This suggests that the inductive bias helps to adapt to real-world effects that could not be modeled in simulation, such as sensor hysteresis. The (mostly) poor performance of $\pi^{\text{NO-RAND}}$ shows that domain randomization is vital for both generalization over different objects and sim-to-real transfer. $\pi^{\text{NO-RAND}}$ was able to perform well on relatively soft objects, however, likely because they behave similarly to the simulation object configuration it was trained on with $\kappa = 0.5$, and our default choice for b_2 closely mirroring the robot behavior. Note that all policies exhibit slightly worse performance in terms of object movements for the Sponge than for other objects. This is because the Sponge is so light that the sensors sometimes fail to detect first contact, a phenomenon also reported in [5].

We also conducted a qualitative assessment of the grasp stability of π^{IB}, which is shown in the accompanying video. To that end, we executed ten grasps per object, lifted it, and executed a pre-defined perturbation trajectory with the arm. In the second part of the experiment, the arm was put into gravity compensation mode and randomly shaken by a human operator. These trials were conducted to assess whether our method is able to maintain a stable grasp even under external disturbances induced by the movement along the trajectories. None of the objects were dropped during any of the trials, indicating that our method is able to maintain a secure grip even if the shear forces of the object change.

The evaluation shows that domain randomization is crucial for successful zero-shot policy transfer and that domain knowledge in the form of an inductive bias further facilitates the transfer. Without domain randomization, policies will overfit to their narrow training distribution and fail to generalize. Our proposed simulation environment has been shown to generate realistic forces, making the transfer possible for continuous control policies. Videos of all policies can be found in the supplementary material.

VI. CONCLUSION

In this work, we presented a DRL method to train grasp force controllers for 2-DoF grippers in simulation and transfer them to the real robot without fine-tuning. We proposed a novel simulation environment that generates realistic grasp forces, which we used to train our policies. To strengthen the transfer performance, we proposed to use an inductive bias and domain randomization. An extensive real-world evaluation has shown that our method can successfully grasp objects of various sizes, shapes, and softness while minimizing object movements during the grasp. Our results show that continuous force control policies can be learned end-to-end in simulation and outperform hand-crafted controllers on real robots. An exciting direction for future work is integrating grasp force control in more complex tasks for which DRL methods are used.

REFERENCES

[1] C. Bartolozzi, L. Natale, F. Nori, and G. Metta, "Robots with a sense of touch," *Nature Materials*, no. 9, 2016.

[2] R. S. Dahiya and M. Valle, *Robotic Tactile Sensing*. Springer Netherlands, 2013.

[3] R. Dahiya, G. Metta, M. Valle, and G. Sandini, "Tactile Sensing—From Humans to Humanoids," *IEEE T-RO*, 2010.

[4] J. M. Romano, K. Hsiao, G. Niemeyer, S. Chitta, and K. J. Kuchenbecker, "Human-inspired robotic grasp control with tactile sensing," *IEEE T-RO*, no. 6, 2011.

[5] L. Lach, S. Lemaignan, F. Ferro, H. Ritter, and R. Haschke, "Bio-Inspired Grasping Controller for Sensorized 2-DoF Grippers," in *2022 IROS*. IEEE, 2022.

[6] A. Church and J. Lloyd, "Tactile Sim-to-Real Policy Transfer via Real-to-Sim Image Translation," *CoRL 2021*, 2021.

[7] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-Real Transfer of Robotic Control with Dynamics Randomization," in *2018 ICRA*, 2018.

[8] Y. Lin, J. Lloyd, A. Church, and N. F. Lepora, "Tactile Gym 2.0: Sim-to-real Deep Reinforcement Learning for Comparing Low-cost High-Resolution Robot Touch," *arxiv:2207.10763*, 2022.

[9] M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal, "Learning Force Control Policies for Compliant Manipulation," in *2011 IROS*, 2011.

[10] G. Kang, H. Seong, D. Lee, and D. H. Shim, "A Versatile Door Opening System with Mobile Manipulator through Adaptive Position-Force Control and Reinforcement Learning," *arxiv:2307.04422*, 2023.

[11] T. Zhang, M. Xiao, Y.-b. Zou, J.-d. Xiao, and S.-y. Chen, "Robotic Curved Surface Tracking with a Neural Network for Angle Identification and Constant Force Control based on Reinforcement Learning," *InJ. Precis. Eng. Manuf.*, 2020.

[12] Z. Ding, N. F. Lepora, and E. Johns, "Sim-to-Real Transfer for Optical Tactile Sensing," *arxiv:2004.00136*, 2020.

[13] Z. Ding, Y.-Y. Tsai, W. W. Lee, and B. Huang, "Sim-to-Real Transfer for Robotic Manipulation with Tactile Sensory," *arxiv:2103.00410*, 2021.

[14] L. Lach, N. Funk, R. Haschke, S. Lemaignan, H. J. Ritter, J. Peters, and G. Chalvatzaki, "Placing by Touching: An empirical study on the importance of tactile sensing for precise object placing," in *IROS23*, 2023.

[15] X. Mao, Y. Xu, R. Wen, M. Kasaei, W. Yu, E. Psomopoulou, N. F. Lepora, and Z. Li, "Learning Fine Pinch-Grasp Skills using Tactile Sensing from Real Demonstration Data," *arxiv:2307.04619*, 2023.

[16] A. Melnik, L. Lach, M. Plappert, T. Korthals, R. Haschke, and H. Ritter, "Using Tactile Sensing to Improve the Sample Efficiency and Performance of Deep Deterministic Policy Gradients for Simulated In-Hand Manipulation Tasks," *Front. Robot. AI*, 2021.

[17] K. Tahara, S. Arimoto, and M. Yoshida, "Dynamic object manipulation using a virtual frame by a triple soft-fingered robotic hand," in *ICRA*, 2010.

[18] Q. Li, R. Haschke, H. Ritter, and B. Bolder, "Towards unknown objects manipulation," *10th IFAC Symposium on Robot Control*, 2012.

[19] Z. Deng, Y. Jonetzko, L. Zhang, and J. Zhang, "Grasping Force Control of Multi-Fingered Robotic Hands through Tactile Sensing for Object Stabilization," *Sensors*, 2020.

[20] H. Merzic, M. Bogdanovic, D. Kappler, L. Righetti, and J. Bohg, "Leveraging Contact Forces for Learning to Grasp," *arxiv:1809.07004*, 2018.

[21] A. Perrusquía, W. Yu, and A. Soria, "Position/force control of robot manipulators using reinforcement learning," *Industrial Robot: the international journal of robotics research and application*, 2019.

[22] J. Luo, E. Solowjow, C. Wen, J. A. Ojea, A. M. Agogino, A. Tamar, and P. Abbeel, "Reinforcement Learning on Variable Impedance Controller for High-Precision Robotic Assembly," in *2019 ICRA*, 2019.

[23] V. Zambaldi, D. Raposo, A. Santoro, V. Bapst, Y. Li, I. Babuschkin, K. Tuyls, D. Reichert, T. Lillicrap, E. Lockhart, M. Shanahan, V. Langston, R. Pascanu, M. Botvinick, O. Vinyals, and P. Battaglia, "Deep Reinforcement Learning with Relational Inductive Biases," *ICLR*, 2019.

[24] M. Hessel, H. van Hasselt, J. Modayil, and D. Silver, "On Inductive Biases in Deep Reinforcement Learning," *arxiv:1907.02908*, 2019.

[25] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: A Survey," *2020 SSCI*, 2020.

[26] H. Ju, R. Juan, R. Gomez, K. Nakamura, and G. Li, "Transferring policy of deep reinforcement learning from simulation to reality for robotics," *Nat Mach Intell*, 2022.

[27] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *2012 IROS*. IEEE, 2012.

[28] R. S. Johansson and J. R. Flanagan, "Coding and use of tactile signals from the fingertips in object manipulation tasks," *Nature Reviews Neuroscience*, no. 5, 2009.

[29] S. Sundaram, P. Kellnhofer, Y. Li, J. Y. Zhu, A. Torralba, and W. Matusik, "Learning the signatures of the human grasp using a scalable tactile glove," *Nature*, 5 2019.

[30] K. Hsiao, S. Chitta, M. Ciocarlie, and E. G. Jones, "Contact-reactive grasping of objects with partial shape information," in *IROS*, 2010.

[31] R. Patel, R. Cox, and N. Correll, "Integrated proximity, contact and force sensing using elastomer-embedded commodity proximity sensors," *Autonomous Robots*, 2018.

[32] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Şucan, "Towards reliable grasping and manipulation in household environments," in *Springer Tracts in Advanced Robotics*. Springer Verlag, 2014.

[33] Z. Su, K. Hausman, Y. Chebotar, A. Molchanov, G. E. Loeb, G. S. Sukhatme, and S. Schaal, "Force estimation and slip detection/classification for grip control using a biomimetic tactile sensor," in *2015 Humanoids*, 2015.

[34] A. Montaño and R. Suárez, "Manipulation of unknown objects to improve the grasp quality using tactile information," *Sensors (Switzerland)*, 2018.

[35] K. Hang, M. Li, J. A. Stork, Y. Bekiroglu, F. T. Pokorny, A. Billard, and D. Kragic, "Hierarchical fingertip space: A unified framework for grasp planning and in-hand grasp adaptation," *IEEE T-RO*, 8 2016.

[36] Z. Deng, Y. Jonetzko, L. Zhang, and J. Zhang, "Grasping force control of multi-fingered robotic hands through tactile sensing for object stabilization," *Sensors (Switzerland)*, 2 2020.

[37] B. Wu, I. Akinola, J. Varley, and P. Allen, "Mat: Multi-fingered adaptive tactile grasping via deep reinforcement learning," in *CoRL 2019*, 2019.

[38] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis, "Sim-To-Real via Sim-To-Sim: Data-Efficient Robotic Grasping via Randomized-To-Canonical Adaptation Networks," in *2019 CVPR*, 2019.

[39] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas *et al.*, "Solving rubik's cube with a robot hand," *arXiv:1910.07113*, 2019.

[40] B. Ward-Cherrier, N. Pestell, L. Cramphorn, B. Winstone, M. E. Giannaccini, J. Rossiter, and N. F. Lepora, "The TacTip Family: Soft Optical Tactile Sensors with 3D-Printed Biomimetic Morphologies," *Soft Robotics*, 2018.

[41] M. Lambeta, P.-W. Chou, S. Tian, B. Yang, B. Maloon, V. R. Most, D. Stroud, R. Santos, A. Byagowi, G. Kammerer, D. Jayaraman, and R. Calandra, "DIGIT: A Novel Design for a Low-Cost Compact High-Resolution Tactile Sensor with Application to In-Hand Manipulation," *IEEE RAL*, 2020.

[42] N. F. Lepora, Y. Lin, B. Money-Coomes, and J. Lloyd, "DigiTac: A DIGIT-TacTip Hybrid Tactile Sensor for Comparing Low-Cost High-Resolution Robot Touch," *IEEE RAL*, 2022.

[43] M. Yang, Y. Lin, A. Church, J. Lloyd, D. Zhang, D. A. W. Barton, and N. F. Lepora, "Sim-to-Real Model-Based and Model-Free Deep Reinforcement Learning for Tactile Pushing," *arxiv:2307.14272*, 2023.

[44] Y. Lin, A. Church, M. Yang, H. Li, J. Lloyd, D. Zhang, and N. F. Lepora, "Bi-Touch: Bimanual Tactile Manipulation with Sim-to-Real Deep Reinforcement Learning," *arxiv:2307.06423*, 2023.

[45] E. Donlon, S. Dong, M. Liu, J. Li, E. Adelson, and A. Rodriguez, "GelSlim: A High-Resolution, Compact, Robust, and Calibrated Tactile-sensing Finger," *arxiv:1803.00628*, 2018.

[46] J. Xu, T. Chen, L. Zlokapa, M. Foshey, W. Matusik, S. Sueda, and P. Agrawal, "An End-to-End Differentiable Framework for Contact-Aware Robot Design," in *Robotics: Science and Systems XVII*. Robotics: Science and Systems Foundation, 2021.

[47] W. Yuan, S. Dong, and E. Adelson, "GelSight: High-Resolution Robot Tactile Sensors for Estimating Geometry and Force," *Sensors*, 2017.

[48] W. Chen, Y. Xu, Z. Chen, P. Zeng, R. Dang, R. Chen, and J. Xu, "Bidirectional Sim-to-Real Transfer for GelSight Tactile Sensors With CycleGAN," *IEEE RAL*, 2022.

[49] Z. Si, Z. Zhu, A. Agarwal, S. Anderson, and W. Yuan, "Grasp Stability Prediction with Sim-to-Real Transfer from Tactile Sensing," *arxiv:2208.02885*, 2022.

[50] Y. Narang, B. Sundaralingam, M. Macklin, A. Mousavian, and D. Fox, "Sim-to-Real for Robotic Tactile Sensing via Physics-Based Simulation and Learned Latent Projections," *arxiv:2103.16747*, 2021.

[51] Y. S. Narang, B. Sundaralingam, K. Van Wyk, A. Mousavian, and D. Fox, "Interpreting and Predicting Tactile Signals for the SynTouch BioTac," *arxiv:2101.05452*, 2021.

[52] J. A. Fishel and G. E. Loeb, "Sensing tactile microvibrations with the BioTac — Comparison with human sensitivity," in *BioRob 2012*, 2012.

[53] C. Sferrazza and R. D'Andrea, "Design, Motivation and Evaluation of a Full-Resolution Optical Tactile Sensor," *Sensors*, 2019.

[54] C. Sferrazza, T. Bi, and R. D'Andrea, "Learning the sense of touch in simulation: A sim-to-real strategy for vision-based tactile sensing," *arxiv:2003.02640*, 2020.

[55] C. Sferrazza and R. D'Andrea, "Sim-to-real for high-resolution optical tactile sensing: From images to 3D contact force distributions," *arxiv:2012.11295*, 2021.

[56] T. Bi, C. Sferrazza, and R. D'Andrea, "Zero-Shot Sim-to-Real Transfer of Tactile Control Policies for Aggressive Swing-Up Manipulation," *IEEE RAL*, 2021.

[57] S. Mysore, B. Mabsout, R. Mancuso, and K. Saenko, "Regularizing Action Policies for Smooth Control with Reinforcement Learning," *arxiv:2012.06644*, 2021.

[58] S. Narvekar and P. Stone, "Learning Curriculum Policies for Reinforcement Learning," *arxiv:1812.00285*, 2018.

[59] S. Narvekar, "Curriculum learning for reinforcement learning domains," *JMLR*, 2020.

[60] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arxiv:1707.06347*, 2017.

[61] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *JMLR*, 2021.