# Towards Transferring Tactile-based Continuous Force Control Policies from Simulation to Robot

**Luca Lach**
IRI, CSIC-UPC

**Robert Haschke**
Bielefeld University

**Davide Tateo**
TU Darmstadt

**Jan Peters**
TU Darmstadt

**Helge Ritter**
Bielefeld University

**Júlia Borràs**
IRI, CSIC-UPC

**Carme Torras**
IRI, CSIC-UPC

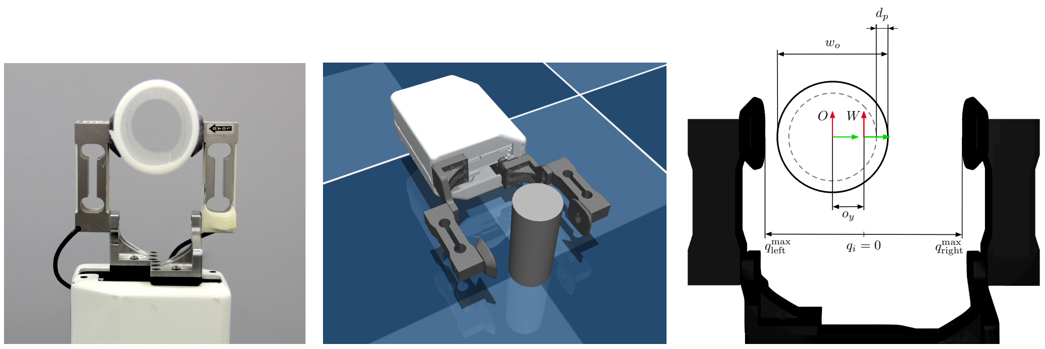
## Abstract

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 that of 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 have 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. An evaluation in which we compare against a baseline and perform an ablation study shows that our approach outperforms the hand-modeled baseline and that our 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`

## 1 Introduction

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 (Zhang et al. [2020], Ding et al. [2020, 2021], Peng et al. [2018], Lach et al. [2023]) or dexterous manipulation tasks (Mao et al. [2023], Melnik et al. [2021]). Low-level tasks like force control are commonly modeled by hand (Romano et al. [2011], Tahara et al. [2010], Li et al. [2012], Lach et al. [2022]), while those that use learning either rely on classical learning methods, do not investigate sim-to-real transfer, or both (Perrusquía et al. [2019], Luo et al. [2019]). 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) with two main control objectives: I) reaching and maintaining a given goal force, and II) minimizing object movements while closing and holding the object.

In the following, we propose a simulation environment based on MuJoCo (Todorov et al. [2012]), 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 an inductive bias to learn policies for subsequent sim-to-real transfer. Lastly, we compare our policy to a hand-modeled force controller from Lach et al. [2022], 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,



(a) Real-life grasp (b) Simulation environment (c) Schematic overview

Figure 1: Overview of the grasping scenario we consider in real life and simulation.

all methods and their evaluation and 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.

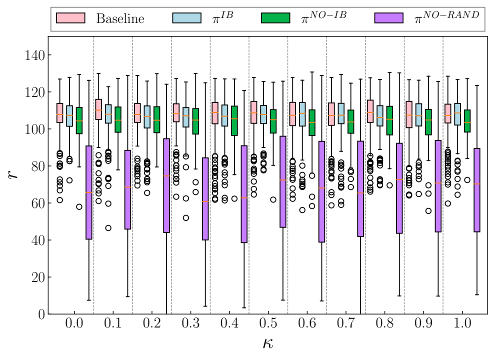
## 2 Related Work

**Grasp Force Control** In their review of human grasping, Johansson and Flanagan [2009] highlight the importance of tactile sensations, and divide the human grasping sequence into distinct phases, where tactile events often mark phase transitions. Many works from the robotics community presenting hand-modeled gripper controllers modeled their controllers accordingly (Romano et al. [2011], Lach et al. [2022], Hsiao et al. [2010], Patel et al. [2018]). In our work, we adapt this scheme through the inductive bias we apply to our policy actions. Learning grasp force control has also become popular in recent years. In Merzic et al. [2018], the authors learn a grasping policy that controls forces on rigid objects in simulation, while Wu et al. [2019] focuses on increasing grasp success using tactile feedback. Others have learned to control grasping forces for more complex tasks like door opening (Kang et al. [2023]), high-precision assembly tasks (Luo et al. [2019]), or surface tracking (Zhang et al. [2020]). Although these works learn force control behaviors, none of them investigate the potential of learning them in simulation and transferring them to the real robot afterward.

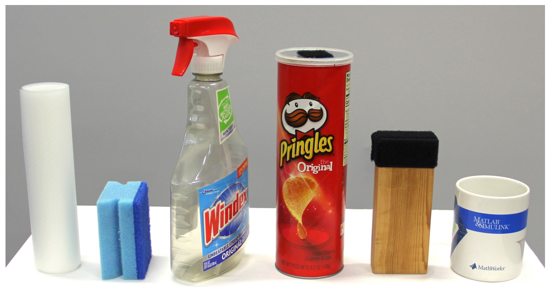
**Sim-to-real transfer** Sim-to-real transfer is widely used in robotics (Zhao et al. [2020], Ju et al. [2022]) to avoid the time-consuming and labor-intensive task of real-world data collection. In the domain of optical tactile sensors, Church and Lloyd [2021] and Lin et al. [2022] presented Tactile Gym, a simulation environment containing the TacTip (Ward-Cherrier et al. [2018]), DIGIT (Lambeta et al. [2020]) and DigiTac (Lepora et al. [2022]) sensors, and propose a domain adaptation approach by training adversarial networks on real-world data to generate tactile feedback in simulation. Approaches based on Finite Element Methods (FEM), are capable of generating accurate simulation data of complex tactile sensors by simulating their deformation (Narang et al. [2021a,b], Sferrazza and D'Andrea [2019], Sferrazza et al. [2020], Sferrazza and D'Andrea [2021]. Due to their high computational cost, FEM is typically not well-suited for data-driven approaches like DRL, unless some simplifying assumptions can be made (Bi et al. [2021]). In contrast to these studies, we focus on low-level force control tasks that solely require force measurements as inputs. Ding et al. [2021] use MuJoCo to simulate a self-made tactile sensor array to open up a cabin door. They again employ domain randomization for transferring the policy, but note that they binarized the sensor readings due to the low sensitivity of the built-in MuJoCo touch sensor. Although Akkaya et al. [2019] also mentions 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.

## 3 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. The control frequency was set to 25 Hz to match that of a real TIAGo. At each simulation step $t$, the environment executes $q_i^{\text{des}} = q_i + u_i$,



(a) Comparison of rewards for different $\kappa$



(b) Evaluation objects ordered by stiffness

Figure 2: Evaluation of our method in simulation (left) and on real-world objects (right).

where $u_i = \Delta q_i^{\text{des}}$ is the control signal given by a policy or user. It refers to the desired position delta, which is then added to the joint's current position and forwarded to the controllers.

Fig. 1c shows a schematic overview of the grasping scenario including all parameters required to define it. The gripper is depicted in its fully open state ($q_i = q_i^{\text{max}} = 0.045$), with an object located between the fingers somewhere on the grasping axis. $W$ and $O$ refer to the world and object frames, $d_p$ the maximum object deformation, $o_y$ the object displacement, and $w_o$ to the object width, where the two latter parameters are sampled upon episode initialization.

Next, we tune the simulation actuators and force sensors to be similar to the behavior of the real robot. We also identify parameter ranges in which the simulation behaves realistic, yet different from our robot, so that our domain randomization will be highly diverse but not unrealistic. To this end, we perform several grasps on the real robot, where we command different $\Delta q$ (since TIAGo is position-controlled), which remain constant for each trial. We then record the joint and force trajectories, repeat the experiments in simulation, and then tune the simulation parameters to match the real-world trajectories. We use MuJoCo's solver impedance parameter `width` to model $\frac{\partial f}{\partial q}$, depending on the object stiffness, and introduce a scaling parameter $f_\alpha$ that is multiplied with the simulation force to model lower forces for softer objects at the same position deltas. In order to avoid unrealistic parameter combinations, e.g. soft objects with high $f_\alpha$, we introduce $\kappa \in [0, 1]$, which we use to interpolate on the intervals of `width` $\in [0.003, 0.01]$ and $f_\alpha \in [0.5, 5]$. Lastly, we found that the actuator bias parameter $b_2 \in [-13, -6]$ generates realistic motor behavior.

## 4 Learning Methods

As the task of force control is a sequential decision-making problem, we can model it as a Markov Decision Process (MDP), allowing us to solve the problem using RL algorithms. The agent receives the following observation at each time step $o(t) = (q_i, f_i, \Delta f_i, a_i(t-1), h_i)^T$, where subscript $i$ indicates that the observation is given for all joints $i$, $\Delta f_i = f^{\text{goal}} - f_i$ refers to the difference of the current force to the goal force, $a_i$ the last action taken by the agent, and $h_i$ being the `had_contact` flag, which switches from 0 to 1 upon contact acquisition and stays 1 even if contact is lost afterward. We add gaussian noise to the joint position and fingertip force and stack the observation $k = 3$ times. The action space is simply a two-vector with one desired position 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 $\Delta q^{\text{max}}$, effectively denormalizing $a_i$. Additionally, we define a contact-state dependent inductive bias $\phi_i$ that acts as a scaling factor for the individual actions at each time step, yielding $a_i' = \phi_i \, a_i$. Thereby, our policy imitates the human grasping phases from Johansson and Flanagan [2009] and is safer to execute on the real robot as erratic joint movements are less likely.

Our reward function contains three terms, one for each controller objective and another one to foster a smooth control behavior. We give a continuous reward that is highest when $\Delta f = 0$, a sparse reward that penalizes object movements with -1, and we additionally penalize the difference between the last and current action. Finally, all individual rewards are weighted and a single, scalar reward is computed per time step. The heavy movement penalty will be given frequently during the early, exploratory

Table 1: Real-world experimental results for six household objects. Our method $\pi^{\text{IB}}$ performs strongest, closely followed by the baseline. The ablated policies showed weaker sim-to-real transfer.

| Model | Metric | Rubber Mat | Sponge | Spray | Pringles | Wood | Mug | Average |
|---|---|---|---|---|---|---|---|---|
| Baseline | Reward | $92 \pm 15$ | $96 \pm 14$ | $\mathbf{114 \pm 5}$ | $\mathbf{118 \pm 7}$ | $\mathbf{108 \pm 7}$ | $115 \pm 10$ | $107 \pm 10$ |
| | Obj.Mov. | $1.6 \pm 0.8$ | $\mathbf{3.1 \pm 2.1}$ | $1.9 \pm 0.6$ | $\mathbf{0.9 \pm 0.4}$ | $1.0 \pm 0.9$ | - | $\mathbf{1.7 \pm 0.9}$ |
| $\pi^{\text{IB}}$ | Reward | $\mathbf{96 \pm 15}$ | $\mathbf{103 \pm 7}$ | $109 \pm 17$ | $115 \pm 16$ | $105 \pm 15$ | $\mathbf{120 \pm 10}$ | $\mathbf{109 \pm 13}$ |
| | Obj.Mov. | $\mathbf{1.5 \pm 0.8}$ | $3.2 \pm 1.4$ | $\mathbf{1.8 \pm 0.8}$ | $1.2 \pm 0.6$ | $\mathbf{0.7 \pm 0.7}$ | - | $\mathbf{1.7 \pm 0.9}$ |
| $\pi^{\text{NO-IB}}$ | Reward | $77 \pm 10$ | $74 \pm 26$ | $88 \pm 33$ | $98 \pm 34$ | $101 \pm 16$ | $106 \pm 17$ | $89 \pm 23$ |
| | Obj.Mov. | $2.8 \pm 1.4$ | $4.0 \pm 0.9$ | $2.4 \pm 0.7$ | $1.9 \pm 0.6$ | $1.7 \pm 0.4$ | - | $2.6 \pm 0.8$ |
| $\pi^{\text{NO-RAND}}$ | Reward | $82 \pm 12$ | $97 \pm 15$ | $57 \pm 37$ | $53 \pm 39$ | $49 \pm 33$ | $40 \pm 41$ | $66 \pm 30$ |
| | Obj.Mov. | $2.6 \pm 1.6$ | $3.6 \pm 1.3$ | $2.6 \pm 0.9$ | $1.7 \pm 0.7$ | $1.8 \pm 0.9$ | - | $2.5 \pm 1.1$ |

phase of training. Therefore, we employ a learning curriculum that increases the penalty weight and amount of domain randomization during training, thereby increasing environment complexity.

## 5 Experimental Evaluation

For the experiments, we use Proximal Policy Optimization (PPO) from Schulman et al. [2017] 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 in terms of force reward and object movements. We compare our policy with inductive bias $\pi^{\text{IB}}$ to a Python implementation of the grasp force controller presented in Lach et al. [2022], a policy $\pi^{\text{NO-IB}}$ trained without the inductive bias, and a policy $\pi^{\text{NO-RAND}}$ trained with neither inductive bias nor domain randomization. We evaluated both models on 11 stiffness values $\kappa$ (spaced out evenly in $[0, 1]$, including interval borders) for 200 simulation trials each, summing up to 2200 trials per model and 8800 trials in total. Note, that during these trials all other environment parameters except $\kappa$ were randomly sampled for each trial. Fig. 2a shows a box plot of cumulative episode rewards for each model at each value for $\kappa$. It shows that all models using domain randomization can successfully control grasping forces while minimizing object movements. $\pi^{\text{NO-RAND}}$ however performed significantly worse, even on $\kappa = 0.5$ which it was trained on since actuator parameters were varied during evaluation while they were fixed during training.

To assess whether our policies are general enough to be transferred to the real robot, we evaluate them on TIAGo using six test objects of varying stiffness (see Fig. 2b), performing 20 grasping trials per object and method, yielding $6 \times 4 \times 20 = 480$ trials in total. In each trial, the object is offset to one finger, and the total object displacement is measured afterward as we did not have access to object velocity measurements. The force reward was calculated identically to the simulation, but since the object velocity is not included, the rewards between real and simulation are not directly comparable. Table 1 shows the results of the real-world evaluation, where $\pi^{\text{IB}}$ shows the strongest overall performance, slightly outperforming the baseline. $\pi^{\text{NO-IB}}$ and $\pi^{\text{NO-RAND}}$ both exhibit larger object movements, showing the importance of the inductive bias. $\pi^{\text{NO-RAND}}$ performed poorly in terms of reward, which shows that domain randomization is a crucial component for sim-to-real transfer. The evaluation clearly shows that domain randomization is crucial for successful zero-shot policy transfer, and that domain knowledge in the form of inductive biases further facilitates the transfer. Our proposed simulation environment has shown to generate realistic forces, such that the transfer was possible for continuous control policies.

## 6 Conclusion

In this work, we presented a DRL method to train grasp force controllers for 2-DoF grippers in simulation and subsequently 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 highly varying stiffnesses while minimizing object movements during the grasp. Conclusively, our results show that continuous force control policies can be learned end-to-end is simulation and slightly outperform hand-modeled controllers on real robots. An interesting direction for future work is the integration of grasp force control in more complex tasks for which DRL methods are used.

## Acknowledgments

This work was supported by the European Union's Horizon 2020 Marie Curie Actions under grant no. 813713 NeuTouch, the Horizon Europe research and innovation program under grant no. 101070600 SoftEnable, the German Research Center for AI (DFKI), and Hessian.AI. Research presented in this paper has been supported by the German Federal Ministry of Education and Research (BMBF) within the collaborative KIARA project (grant no. 13N16274).

## References

Tie Zhang, Meng Xiao, Yan-biao Zou, Jia-dong Xiao, and Shou-yan Chen. Robotic Curved Surface Tracking with a Neural Network for Angle Identification and Constant Force Control based on Reinforcement Learning. *International Journal of Precision Engineering and Manufacturing*, 21 (5):869–882, 2020. ISSN 2234-7593, 2005-4602.

Zihan Ding, Nathan F. Lepora, and Edward Johns. Sim-to-Real Transfer for Optical Tactile Sensing, 2020.

Zihan Ding, Ya-Yen Tsai, Wang Wei Lee, and Bidan Huang. Sim-to-Real Transfer for Robotic Manipulation with Tactile Sensory, 2021.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3803–3810, 2018.

Luca Lach, Niklas Funk, Robert Haschke, Severin Lemaignan, Helge Joachim Ritter, Jan Peters, and Georgia Chalvatzaki. Placing by Touching: An empirical study on the importance of tactile sensing for precise object placing. In *IROS23*, 2023.

Xiaofeng Mao, Yucheng Xu, Ruoshi Wen, Mohammadreza Kasaei, Wanming Yu, Efi Psomopoulou, Nathan F. Lepora, and Zhibin Li. Learning Fine Pinch-Grasp Skills using Tactile Sensing from Real Demonstration Data, 2023.

Andrew Melnik, Luca Lach, Matthias Plappert, Timo Korthals, Robert Haschke, and Helge Ritter. Using Tactile Sensing to Improve the Sample Efficiency and Performance of Deep Deterministic Policy Gradients for Simulated In-Hand Manipulation Tasks. *Frontiers in Robotics and AI*, 8: 538773, 2021. ISSN 2296-9144.

Joseph M Romano, Kaijen Hsiao, Günter Niemeyer, Sachin Chitta, and Katherine J Kuchenbecker. Human-inspired robotic grasp control with tactile sensing. *IEEE Trans. on Robotics*, 27(6): 1067–1079, 2011.

Kenji Tahara, Suguru Arimoto, and Morio Yoshida. Dynamic object manipulation using a virtual frame by a triple soft-fingered robotic hand. In *Proc. ICRA*, 2010.

Qiang Li, Robert Haschke, Helge Ritter, and Bram Bolder. Towards unknown objects manipulation. *IFAC Proceedings Volumes*, 45(22):289–294, 2012.

Luca Lach, Severin Lemaignan, Francesco Ferro, Helge Ritter, and Robert Haschke. Bio-Inspired Grasping Controller for Sensorized 2-DoF Grippers. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11231–11237. IEEE, 2022.

Adolfo Perrusquía, Wen Yu, and Alberto Soria. Position/force control of robot manipulators using reinforcement learning. *Industrial Robot: the international journal of robotics research and application*, 46(2):267–280, 2019. ISSN 0143-991X, 0143-991X.

Jianlan Luo, Eugen Solowjow, Chengtao Wen, Juan Aparicio Ojea, Alice M. Agogino, Aviv Tamar, and Pieter Abbeel. Reinforcement Learning on Variable Impedance Controller for High-Precision Robotic Assembly. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3080–3087. IEEE, 2019.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

Roland S Johansson and J Randall Flanagan. Coding and use of tactile signals from the fingertips in object manipulation tasks. *Nature Reviews Neuroscience*, 10(5):345–359, 2009.

Kaijen Hsiao, Sachin Chitta, Matei Ciocarlie, and E. Gil Jones. Contact-reactive grasping of objects with partial shape information. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

Radhen Patel, Rebecca Cox, and Nikolaus Correll. Integrated proximity, contact and force sensing using elastomer-embedded commodity proximity sensors. *Autonomous Robots*, 42:1443–1458, 2018.

Hamza Merzic, Miroslav Bogdanovic, Daniel Kappler, Ludovic Righetti, and Jeannette Bohg. Leveraging Contact Forces for Learning to Grasp, 2018.

Bohan Wu, Iretiayo Akinola, Jacob Varley, and Peter Allen. Mat: Multi-fingered adaptive tactile grasping via deep reinforcement learning. In *3rd Conference on Robot Learning (CoRL 2019),*, 9 2019.

Gyuree Kang, Hyunki Seong, Daegyu Lee, and D. Hyunchul Shim. A Versatile Door Opening System with Mobile Manipulator through Adaptive Position-Force Control and Reinforcement Learning, 2023.

Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: A Survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 737–744, 2020.

Hao Ju, Rongshun Juan, Randy Gomez, Keisuke Nakamura, and Guangliang Li. Transferring policy of deep reinforcement learning from simulation to reality for robotics. *Nature Machine Intelligence*, 4(12):1077–1087, 2022. ISSN 2522-5839.

Alex Church and John Lloyd. Tactile Sim-to-Real Policy Transfer via Real-to-Sim Image Translation. *5th Conference on Robot Learning (CoRL 2021)*, 2021.

Yijiong Lin, John Lloyd, Alex Church, and Nathan F. Lepora. Tactile Gym 2.0: Sim-to-real Deep Reinforcement Learning for Comparing Low-cost High-Resolution Robot Touch, 2022.

Benjamin Ward-Cherrier, Nicholas Pestell, Luke Cramphorn, Benjamin Winstone, Maria Elena Giannaccini, Jonathan Rossiter, and Nathan F. Lepora. The TacTip Family: Soft Optical Tactile Sensors with 3D-Printed Biomimetic Morphologies. *Soft Robotics*, 5(2):216–227, 2018. ISSN 2169-5172, 2169-5180.

Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian Yang, Benjamin Maloon, Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, Dinesh Jayaraman, and Roberto Calandra. DIGIT: A Novel Design for a Low-Cost Compact High-Resolution Tactile Sensor with Application to In-Hand Manipulation. *IEEE Robotics and Automation Letters*, 5(3): 3838–3845, 2020. ISSN 2377-3766, 2377-3774.

Nathan F. Lepora, Yijiong Lin, Ben Money-Coomes, and John Lloyd. DigiTac: A DIGIT-TacTip Hybrid Tactile Sensor for Comparing Low-Cost High-Resolution Robot Touch. *IEEE Robotics and Automation Letters*, 7(4):9382–9388, 2022. ISSN 2377-3766, 2377-3774.

Yashraj Narang, Balakumar Sundaralingam, Miles Macklin, Arsalan Mousavian, and Dieter Fox. Sim-to-Real for Robotic Tactile Sensing via Physics-Based Simulation and Learned Latent Projections, 2021a.

Yashraj S. Narang, Balakumar Sundaralingam, Karl Van Wyk, Arsalan Mousavian, and Dieter Fox. Interpreting and Predicting Tactile Signals for the SynTouch BioTac, 2021b.

Carmelo Sferrazza and Raffaello D'Andrea. Design, Motivation and Evaluation of a Full-Resolution Optical Tactile Sensor. *Sensors*, 19(4):928, 2019. ISSN 1424-8220.

Carmelo Sferrazza, Thomas Bi, and Raffaello D'Andrea. Learning the sense of touch in simulation: A sim-to-real strategy for vision-based tactile sensing, 2020.

Carmelo Sferrazza and Raffaello D'Andrea. Sim-to-real for high-resolution optical tactile sensing: From images to 3D contact force distributions, 2021.

Thomas Bi, Carmelo Sferrazza, and Raffaello D'Andrea. Zero-Shot Sim-to-Real Transfer of Tactile Control Policies for Aggressive Swing-Up Manipulation. *IEEE Robotics and Automation Letters*, 6(3):5761–5768, 2021. ISSN 2377-3766, 2377-3774.

Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, 2017.

Siddharth Mysore, Bassel Mabsout, Renato Mancuso, and Kate Saenko. Regularizing Action Policies for Smooth Control with Reinforcement Learning, 2021.

Sanmit Narvekar and Peter Stone. Learning Curriculum Policies for Reinforcement Learning, 2018.

Sanmit Narvekar. Curriculum learning for reinforcement learning domains. *Journal of Machine Learning Research*, 21, 2020.

## Appendix

### Force Control Simulation

The following three constraints are imposed on the sampling of the environment parameters from Fig. 1c:

$$|o_y| + r_o < q^{\text{max}} \tag{1}$$

$$r_o - d_p > |o_y| \tag{2}$$

$$d_p < r_o \tag{3}$$

where $r_o = \frac{1}{2}w_o$ is the object radius. The joints are controlled by

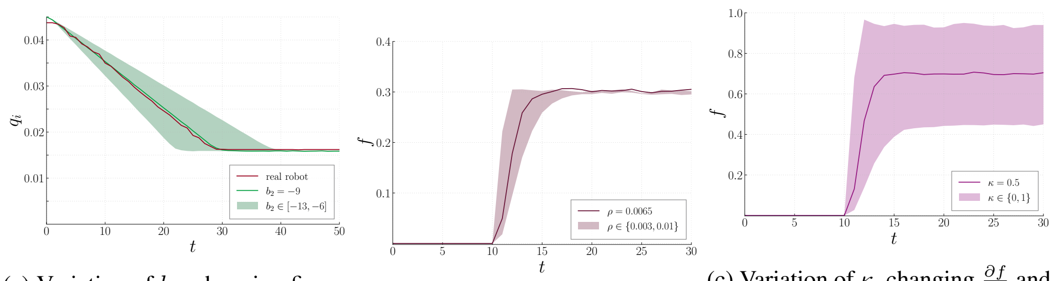
$$q_i^{\text{des}} = q_i + u_i \tag{4}$$

at each simulation step $t$, where $u_i = \Delta q_i^{\text{des}}$ is the control signal the policy or a user passes to the environment. Using position deltas instead of positions directly 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 with $|u_i| \leq \Delta q^{\text{max}}$, the maximum joint velocity is bounded and prevents the policy from executing erratic and potentially dangerous movements. Choosing $\Delta q^{\text{max}}$ is straightforward by executing different gripper closing trials with increasing values for $\Delta q^{\text{max}}$ and setting it to the value resulting in the highest velocity within safety limits. In all experiments, we set $\Delta q^{\text{max}} = 0.003$.

After defining the control, we tuned the actuators in MuJoCo to match the behavior of TIAGo. First, *gainprm* and *ctrlrange* were set to $[0.0, 0.045]$ and $[100, 0, 0]$ to mimic the real joint actuation range $[0, 0.045]$, which reflects the desired finger position in centimeters (see Fig. 1c). Then, the third parameter of *biasprm*, $b_2$, was tuned to match the finger's closing velocity and acceleration. In order 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 real behavior best, and $b_2 \in [-13, -6]$ to result in realistic actuator behavior for our domain randomization, which is shown in Fig. 3a.

After identifying realistic actuator parameters, the contact forces need to be modeled to imitate real-world objects as well. In MuJoCo, the softness of contact constraints can be changed with the *solimp* parameters, which allow more object penetration, resulting in different $\frac{\partial f}{\partial q}$. We set *solimp*'s first parameters, $d_{\text{min}} = 0$ to allow constraints to be maximally soft and use its *width* parameter, which we refer to as $\rho$, to vary the constraint softness for each trial.

In order to find a range of realistic values for $\rho$, we executed real-life grasping trials on a soft and a rigid object (Sponge and Wood from Fig. 2b). Then, we conducted the same experiment in simulation, compared $\frac{\partial f}{\partial q}$ for both experiments, and tuned $\rho$ until the changes in force w.r.t. the joint position matched the real-world trials for both objects. Fig. 3b shows force trajectories for the determined interval $\rho \in [0.003, 0.01]$.



(a) Variation of $b_2$, changing finger velocity

(b) Variation of $\rho$, changing $\frac{\partial f}{\partial q}$

(c) Variation of $\kappa$, changing $\frac{\partial f}{\partial q}$ and $f(T)$

Figure 3: Results from the real-world grasping trials conducted to determine realistic simulation parameters. Each plot shows the effect of varying one of the parameters discussed in Sec. 3, where $b_2$ changes actuator behavior and $\kappa$ affects the forces by changing $\rho$ and $f_\alpha$.

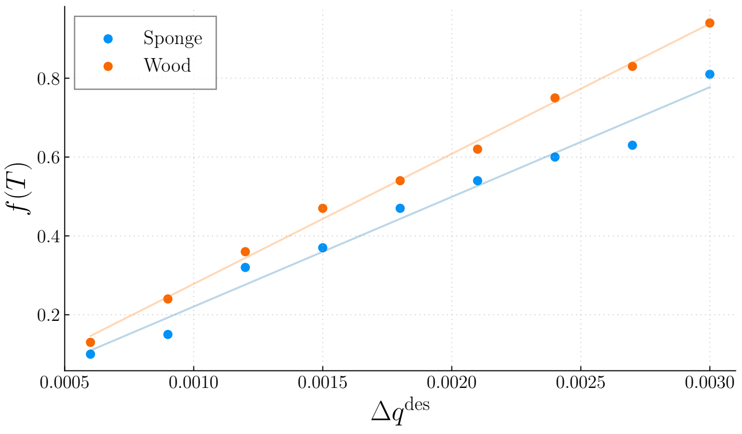


Figure 4: Comparison of final forces $f(T)$ of Sponge and Wood for grasps with different $\Delta q^{\text{des}}$. The regression slopes are 278 and 330.

Note, that changing the constraint stiffness $\rho$ results in different $\frac{\partial f}{\partial q}$, but not in different final forces. In reality, however, the force exerted on a stiff object is higher than one exerted on a soft object for the same $\Delta q^{\text{des}}$. To also model this behavior, we introduced a force scaling factor $f_\alpha$, such that stiffer objects generate higher forces more quickly, and softer objects slower. We conducted another set of grasping trials on the same objects, where a small, constant $\Delta q^{\text{des}}$ was commanded to both fingers and after the object was being held for a short amount of time, the final force $f(T)$ was noted. The gripper was then opened again, $\Delta q^{\text{des}}$ increased by $3 \times 10^{-4}$ and the experiment repeated until $\Delta q^{\text{des}}$ = $\Delta q^{\max}$. Then, we repeated the same experiment in simulation and regressed $f(T)$ to $\Delta q^{\text{des}}$ for both experiments. The experiment data from the robot trials are shown in Fig. 4. $f_\alpha$ is then the factor needed to match the regression slope of the simulation experiments with that of the real robot. This way, we determined $f_\alpha \in [0.5, 5]$.

Lastly, to determine both values for $\rho$ and $f_\alpha$, we define a unified stiffness factor $\kappa \in [0, 1]$ which we use to interpolate within their respective intervals. This way, we prevent unrealistic object configurations, e.g. a soft object with a high $f_\alpha$.

## Learning Methods

The `had_contact` flag is defined as:

$$c_i(t) = \begin{cases} 1 & \text{if } f_i(t) > f_\theta \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$$h_i(t) = c_i(t) \vee h_i(t-1) \tag{6}$$

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 $f_\theta$, and $h_i(0) = 0$. $h_i$ is 1 once $c_i$ was 1 before within the episode, even if the finger lost contact again ($c_i = 0$). We provide this flag to avoid having a recurrent policy (since they can be 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.

Additionally, we define a contact-state dependent inductive bias:

$$\phi_i = \begin{cases} \max(0.9, 1 - \frac{|\Delta f_i|}{f^{\text{goal}}}) & \text{if } h_i = h_j = 1 \\ 0.1 & \text{if } h_i = 1 \wedge h_i \neq h_j \\ 1 & \text{else} \end{cases} \tag{7}$$

which mimics the human grasping phases Johansson and Flanagan [2009] and is commonly used in other controllers (Romano et al. [2011], Lach et al. [2022]).

|  | Initial | Final |
|---|---|---|
| $\alpha_2$ | $0$ | $1.0$ |
| $\dot{o}_y^{\max}$ | $2 \times 10^{-4}$ | $5 \times 10^{-5}$ |
| $W_o$ | $[0.020, 0.025]$ | $[0.015, 0.035]$ |
| $O_y$ | $[0.0, 0.0]$ | $[-0.040, 0.040]$ |

Table 2: Annealed parameters with their initial and final values.

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

$$r^{\text{force}} = 1 - \tanh\left(\sum_i |\Delta f_i|\right) \tag{8}$$

$$r^{\text{obj}} = \begin{cases} -1 & \text{if } \dot{o}_y > \dot{o}_y^{\max} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$r^{\text{act}} = -\sum_i |a_i(t-1) - a_i(t)| \tag{10}$$

The term $r^{\text{force}}$ reflects the first control objective in the reward function, namely to reach and maintain the target force. $\tanh$ normalizes the force delta in $[0, 1]$ and subtracting it from 1 yields the highest reward if $\sum \Delta f_i = 0$. $r^{\text{obj}}$ expresses the second controller objective as a reward function by penalizing object movements with a sparse reward of -1 upon constraint violation. In equation 9, $\dot{o}_y$ refers to the current object velocity and $\dot{o}_y^{\max}$ to the object velocity threshold. $\dot{o}_y$ has shown to always be noisy during finger contact, hence we use the threshold $\dot{o}_y^{\max} = 0.00005$. The third reward term, $r^{\text{act}}$, penalizes high changes in policy actions at consecutive time steps to encourage a smooth control behavior. Other, more involved procedures have been proposed before (Mysore et al. [2021]), but for our purposes this simple constraint has shown to be sufficient. Finally, the total reward per time step is defined as

$$r = \alpha_1 r^{\text{force}} + \alpha_2 r^{\text{obj}} + \alpha_3 r^{\text{act}} \tag{11}$$

Our reward function as it is given in equation 11 poses a difficult problem for learning control policies: during exploration, it is highly 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 local minima where it avoids contact with the object since it will receive a large negative reward from $r^{\text{obj}}$ before being rewarded by $r^{\text{force}}$. We, therefore, employ a learning curriculum, a common approach to gradually increase the task complexity over the course of training by annealing certain environment parameters (Narvekar and Stone [2018], Narvekar [2020]).

The parameters controlling the likelihood of these high negative rewards are $\alpha_2$, the scaling factor for $r^{\text{obj}}$, the range of possible object displacements $O_y$, the range of object radii $W_o$, and the object velocity threshold $\dot{o}_y^{\max}$. For the scalar parameter $\alpha_2$ and $\dot{o}_y^{\max}$, a starting value and a final value are defined. During training, the parameters are linearly interpolated at each time step from their initial value at $t = 0$ until they reach their final value at $t = s_{\text{end}}$, where $s_{\text{end}}$ is a hyperparameter. For the intervals $O_y$ and $W_o$, the initial and final interval borders are given. During the annealing phase, both borders are also linearly interpolated from their respective initial to their final values. All annealed parameters and their initial and final values are shown in Table 2.