

# Can Context Bridge the Reality Gap? Sim-to-Real Transfer of Context-Aware Policies

Marco Iannotta<sup>a</sup>, Yuxuan Yang<sup>a</sup>, Johannes A. Stork<sup>a</sup>, Erik  
Schaffernicht<sup>b</sup>, Todor Stoyanov<sup>a</sup>

<sup>a</sup>*AASS Research Centre, Örebro University, , Örebro, Sweden*

<sup>b</sup>*Technology Transfer Center Kitzingen, Technical University of Applied  
Sciences Würzburg-Schweinfurt, , Kitzingen, Germany*

---

## Abstract

Sim-to-real transfer remains a major challenge in reinforcement learning (RL) for robotics, as policies trained in simulation often fail to generalize to the real world due to discrepancies in environment dynamics. Domain Randomization (DR) mitigates this issue by exposing the policy to a wide range of randomized dynamics during training, yet leading to a reduction in performance. While standard approaches typically train policies agnostic to these variations, we investigate whether sim-to-real transfer can be improved by conditioning the policy on an estimate of the dynamics parameters — referred to as context. To this end, we integrate a context estimation module into a DR-based RL framework and systematically compare SOTA supervision strategies. We evaluate the resulting context-aware policies in both a canonical control benchmark and a real-world pushing task using a Franka Emika Panda robot. Results show that context-aware policies outperform the context-agnostic baseline across all settings, although the best supervision strategy depends on the task.

*Keywords:* Robotics, Reinforcement Learning, Sim-to-Real

---

## 1. Introduction

Reinforcement learning (RL) has achieved significant success in developing robot controllers capable of solving complex tasks [1]. However, training RL policies directly on physical robots demands extensive interactions with

the real environment, making training expensive and dangerous for the robot and its surroundings. To address these limitations, physics simulation engines are widely used as a safer and more efficient alternative for policy training. Once a policy has been trained in simulation, it is transferred to the physical robot—a process known as *sim-to-real* transfer [2, 1, 3]. Although promising, this paradigm is hindered by the *reality* or *sim-to-real gap*, which refers to the discrepancy between the simulated and real-world environments [4, 5]. This gap often leads to a significant decrease in performance when the policy is deployed in reality, posing a substantial challenge.

Domain Randomization (DR) is a widely adopted approach to mitigate the reality gap [6]. The core idea is to expose the policy to a broad distribution of simulated environments during training by randomizing various simulation parameters that affect the environment dynamics. Learning to perform robustly across this diverse range of scenarios makes the policy less reliant on a precise match between the simulated and the real environment. Standard approaches in DR typically train policies that are *agnostic* to the dynamics parameters randomized in simulation, i.e., the policy is trained to perform robustly under all variations, without explicitly incorporating knowledge of the randomized parameters and relying solely on the observed state. Instead, Yu et al. [7] propose coupling the RL policy with an online system identification model (OSI) trained to infer dynamics parameters from recent trajectories. These inferred parameters condition the policy alongside the observed state, explicitly informing the control policy on the underlying dynamics. Although this approach has shown promising results in simulation, it has never been validated on a real robot, leaving the effectiveness of dynamics-aware policies in real-world scenarios untested.

In parallel to this line of research, Zero-Shot Generalization (ZSG) has emerged as a key topic in RL and robotics [9]. ZSG aims to train policies that can generalize to novel environments, without requiring environment-specific training. Within this area, some works have explored *context*-aware policies — policies that receive, alongside the state, an explicit input referred to as the *context*, which captures structured information about the environment or task. The assumption is that this context information, whether provided or inferred, enables the policy to adapt to varying conditions [10]. These approaches typically rely on an estimator trained to infer a context representation from experience, in a way analogous to OSI. The key difference lies in the supervision strategy used during training, employing proxy tasks to enable context inference without direct supervision.

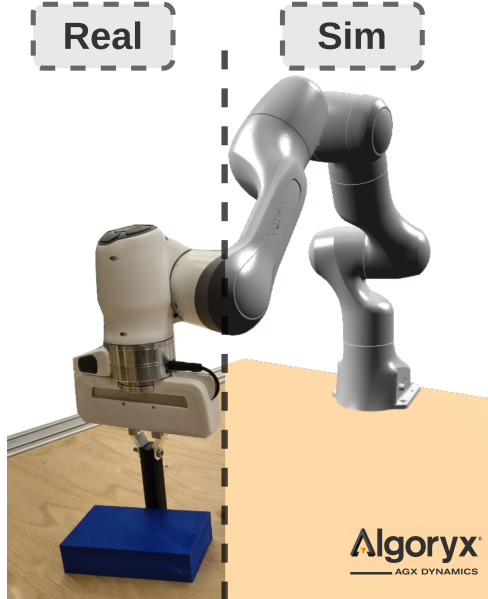


Figure 1: Setup employed for the experimental task evaluated in Sec. 4.2 — pushing a box to a desired location — featuring the Franka Emika Panda robot and its digital twin in the AGX Dynamics simulation [8].

Despite sharing the objective of enabling policies to adapt to unseen dynamics, OSI and context-aware policies for ZSG have evolved in parallel, without direct comparison. In this work, we aim to bridge these two lines of research and to investigate whether context-aware policies can enhance the sim-to-real transfer of robot control policies. We focus on how the supervision strategy used to train the context estimator affects the policy’s ability to generalize to unseen dynamics. To this end, we conduct an empirical study of the main supervision strategies proposed in the literature, evaluated within a unified framework on both a canonical control benchmark and a real-robot task with a Franka Emika Panda, where the goal is to push a box to a desired location (Fig. 1). In our benchmark scenarios, DR results in the poorest transfer performance under equivalent interaction samples. In contrast, context-aware policies achieve better generalization, although no single method consistently outperforms the others in all settings. This suggests that incorporating context information is beneficial, but that the optimal supervision strategy may be task-dependent.

This work makes two main contributions. First, we empirically demonstrate that incorporating context information and applying techniques from

contextual RL results in improved sim-to-real transfer of manipulation policies. Second, we analyze how different supervision strategies impact zero-shot generalization of context-conditioned policies, highlighting their task-dependent behavior and discussing practical implications for real-world deployment.

## 2. Related Work

### 2.1. *Sim-To-Real Transfer*

Domain Randomization is a common approach for transferring robot controllers trained with RL in simulation to reality [5]. The policy is trained across a diverse range of simulated environments by varying parameters that influence the environment dynamics, such as physical properties like mass and friction coefficients. DR ensures that the policy is exposed to a broad spectrum of variations, enhancing its ability to perform robustly when deployed in the real world.

Several works have shown the effectiveness of DR in addressing the reality gap. Matas et al. [11] train a controller in simulation for solving different deformable object manipulation tasks. Experimental results indicate that randomizing extrinsic camera parameters aids sim-to-real transfer due to the controller’s sensitivity to position changes, but excessive randomization can hinder transfer success. Van Baar et al. [12] show that DR-trained controllers require fewer fine-tuning steps for sim-to-real transfer in a robotic Marble Maze task. OpenAI et al. [13] propose automatic domain randomization, where the environment parameters used during training are sampled from a changing distribution, rather than a fixed one. [5] presents a comprehensive overview of approaches leveraging DR for sim-to-real transfer. Common to most approaches is that the trained policies are not explicitly aware of the dynamics parameters randomized in simulation. Instead, Yu et al. [7] propose conditioning the control policy on these parameters, estimated by an online system identification model. This model is trained in a supervised way using the ground truth parameters known in simulation and is designed to infer these parameters online at test time. Despite promising results in simulation, the approach is not evaluated on a physical system, leaving its effectiveness for sim-to-real transfer unverified. In this work, we evaluate both the classical context-agnostic DR approach and a context-aware alternative that estimates dynamic parameters through a learned context encoder, inspired by the system identification framework proposed in [7].

## 2.2. Context-Aware Policies for Zero-Shot Generalization

Recent approaches for ZSG of control policies build on the framework of Contextual RL (cRL) [14, 15], which assumes variations in the environment can be represented by a *context* and used to guide a generalizable agent in adapting its behavior accordingly. Benjamins et al. [10] show that optimal behavior in cRL requires context information. This is validated by comparing the performance of a context-aware policy, which incorporates the known ground-truth context through simple state concatenation, with a context-agnostic policy. Both policies are trained on various context-encoded versions of common RL environments, where dynamics parameters are randomized during training. The results show that the context-conditioned policy often significantly outperforms the context-agnostic one, especially when the task is sensitive to changes in environment dynamics, underscoring the potential benefits of incorporating context.

In the more practically relevant case when a ground-truth context is not available, several approaches have been proposed to train a context estimator to generate a latent representation from recent transitions. This is similar to the system identification model introduced in [7], with the key difference being how the model (i.e., the context estimator) is supervised. We identified two supervision strategies in the literature that can be applied to sim-to-real, where dynamics parameters are only available during training in simulation. Evans et al. [16] propose coupling the context estimator with a second model, referred to as the predictor, which utilizes the latent context to predict the next state of the environment. In contrast, Ndir et al. [17] propose training the context estimator based on the control policy loss, forcing the context to capture information relevant to the current policy.

In this work, we explore whether such latent context representations offer advantages for sim-to-real transfer and which supervision strategy yields the best results. We train all policies in a unified framework and evaluate them both in simulation and on a physical setup, providing direct insight into their effectiveness for real-world deployment.

## 3. Sim-to-real Transfer through Explicit Context Estimation

We aim to investigate how conditioning RL policies on a representation of the environment’s dynamics can improve sim-to-real transfer. In particular, we explore the use of a *context* vector  $\mathbf{c} \in \mathbb{R}^c$ , where  $c \in \mathbb{N}^+$  denotes the

context dimensionality, to capture unobservable but task-relevant environment properties, such as mass and friction, which vary between simulation and reality. By leveraging this context information, the policy can adapt its behavior based on the specific dynamics of the deployment environment, rather than relying solely on robustness learned through domain randomization — assuming simulation can approximate real-world dynamics through parameter tuning.

The approach consists of two main components: the *control policy*  $\pi$  and the *context estimator*  $\phi$ . The context estimator infers a context representation  $\hat{\mathbf{c}} \in \mathbb{R}^{\hat{c}}$  from a set of  $N$  transitions  $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ , where  $\hat{c} \in \mathbb{N}^+$ ,  $\mathbf{s}$  is the state,  $\mathbf{a}$  the control policy action, and  $\mathbf{s}'$  the next state:

$$\phi: \{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i)\}_{i=1}^N \mapsto \hat{\mathbf{c}}. \quad (1)$$

Depending on the supervision strategy,  $\hat{\mathbf{c}}$  may correspond to an estimate of the ground-truth context vector  $\mathbf{c}$  or a latent representation learned through a proxy task.

The control policy  $\pi$  receives both the observable state  $\mathbf{s} \in \mathcal{S}$  and the inferred context representation  $\hat{\mathbf{c}}$  as input, returning an action  $\mathbf{a} \in \mathcal{A}$ , corresponding to a robot command:

$$\pi: (\mathbf{s}, \hat{\mathbf{c}}) \mapsto \mathbf{a}. \quad (2)$$

### 3.1. Training and Evaluation

Line 1 outlines the training procedure for a generic off-policy RL algorithm. We jointly train the control policy  $\pi$  and the context estimator  $\phi$ , following the approach proposed in [17]. A key advantage of this approach is that both  $\pi$  and  $\phi$  learn from data generated by the same policy  $\pi$ , eliminating the need for an auxiliary data collection policy as required in other methods. Moreover, by using data from the same policy, we avoid distributional mismatch, ensuring that the context estimator is trained on state-action distributions that reflect those encountered during policy execution.

At the beginning of each episode, the agent receives an initial observation  $\mathbf{s}_1$ , while the episodic context  $\mathbf{c}_e$ , available in simulation, is retrieved. At each time step  $t$ , we sample a set  $\mathcal{T}_e$  of  $N$  transitions sharing context  $\mathbf{c}_e$  from the replay buffer. Based on this set, the context estimator  $\phi$  infers the context representation  $\hat{\mathbf{c}}_e$ . The control policy then selects an action  $\mathbf{a}_t$  based on the observable state  $\mathbf{s}_t$  and the inferred context  $\hat{\mathbf{c}}_e$ . After action execution, the

---

**Algorithm 1:** Training loop.

---

```
1 Initialize control policy  $\pi$ , context estimator  $\phi$ , and replay buffer  $\mathcal{R} = \emptyset$ .
2 for  $e = 1, \dots, \textit{episodes}$  do
3   Receive observation  $\mathbf{s}_1$  and context  $\mathbf{c}_e$ 
4   for  $t = 1, \dots, \textit{steps and !done}$  do
5     Sample set  $\mathcal{T}_e$  of  $N$  transitions with context  $\mathbf{c}_e$  from  $\mathcal{R}$ 
6     Infer context representation  $\hat{\mathbf{c}}_e = \phi(\mathcal{T}_e)$ 
7     Select action  $\mathbf{a}_t = \pi(\mathbf{s}_t, \hat{\mathbf{c}}_e)$ 
8     Execute  $\mathbf{a}_t$ , observe next state  $\mathbf{s}'_t$  and reward  $r_t$ 
9      $\mathcal{R} = \mathcal{R} \cup (\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}'_t, \mathbf{c}_e)$ 
10    Sample mini-batch  $\mathcal{B}$  from  $\mathcal{R}$ 
11    for  $(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}'_i, \mathbf{c}_i) \in \mathcal{B}$  do
12      Sample set  $\mathcal{T}_i$  of  $N$  transitions with context  $\mathbf{c}_i$  from  $\mathcal{R}$ 
13      Infer context representation  $\hat{\mathbf{c}}_i = \phi(\mathcal{T}_i)$ 
14    end
15    Compute loss  $\mathcal{L}_\pi$  and update  $\pi$ 
16    Compute loss  $\mathcal{L}_\phi^*$  according to the selected supervision strategy
      and update  $\phi$ 
17  end
18 end
```

---

agent observes the next state  $\mathbf{s}'_t$  and receives a reward  $r_t$ , while the tuple  $(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}'_t, \mathbf{c}_e)$  is stored in the replay buffer.

To optimize the control policy and context estimator, we first sample a mini-batch  $\mathcal{B}$  from the replay buffer. For each transition in  $\mathcal{B}$ , we sample a set  $\mathcal{T}_i$  of  $N$  transitions sharing the context  $\mathbf{c}_i$ , following the same procedure used for action selection. The context estimator  $\phi$  then infers the context representation  $\hat{\mathbf{c}}_i = \phi(\mathcal{T}_i)$  for each sampled transition. Lastly, we compute the losses for the control policy and the context estimator and update them accordingly. The context estimator loss is computed based on the selected supervision strategy (sec. 3.3), while the control policy loss remains unchanged, depending only on the selected RL algorithm.

During evaluation, we follow the same procedure for selecting action  $\mathbf{a}_t$ , except that context estimation is performed using  $N$  transitions sampled from the current episode.

### 3.2. Context Estimator Architectures

Similarly to [16], we investigate two architectures for the context estimator. The first employs a feed-forward neural network combined with average pooling (FF+AVG). Each transition is processed through a shared estimator, and the resulting embeddings are averaged to produce a fixed-length representation. This approach is simple and computationally efficient, though it treats all transitions equally, regardless of how informative they are. The second architecture leverages a recurrent neural network, specifically an LSTM [18], to aggregate context information. Although the transitions are inherently unordered, we treat them as a sequence and extract the latent representation by applying a linear projection to the final hidden state of the LSTM. This formulation allows the model to learn how to weigh and extract relevant information across different transitions, potentially improving its ability to capture complex environmental variations.

### 3.3. Context Estimator Supervision Strategies

We evaluate three strategies to supervise the context estimator training — one based on regressing the ground truth context, and two on proxy tasks inspired by cRL research.

**Ground-Truth (GT) Supervision.** Based on [7], the context estimator is trained to directly regress the ground-truth context available in simulation. The loss is defined as the mean squared error (MSE) between the estimated and the ground-truth context vectors:

$$\mathcal{L}_\phi^{GT} = \mathbb{E}_{(\mathbf{s}, \mathbf{a}, \mathbf{s}', \mathbf{c}) \sim \mathcal{B}} [\|\hat{\mathbf{c}} - \mathbf{c}\|^2]. \quad (3)$$

**Proxy Task Forward Dynamics Prediction (FP).** Based on [16], the context estimator is trained end-to-end in conjunction with a prediction model on a forward prediction task. The context estimator  $\phi$  infers a context representation  $\hat{\mathbf{c}}$  in a learned latent space (Eq. (1)). This latent vector is used to condition the prediction model  $p_f$ , which is implemented as a feed-forward neural network and infers the next state  $\mathbf{s}'$  from current state  $\mathbf{s}$  and action  $\mathbf{a}$ :

$$p_f: (\mathbf{s}, \mathbf{a}, \hat{\mathbf{c}}) \mapsto \mathbf{s}'. \quad (4)$$

$\phi$  and  $p_f$  are updated jointly by minimizing the mean-squared error between the predicted and ground-truth next state, with gradient back-propagated through both models:

$$\mathcal{L}_{\phi}^{FP} = \mathbb{E}_{(\mathbf{s}, \mathbf{a}, \mathbf{s}', \mathbf{c}) \sim \mathcal{B}} \left[ \|p_f(\mathbf{s}, \mathbf{a}, \hat{\mathbf{c}}) - \mathbf{s}'\|^2 \right]. \quad (5)$$

**Proxy Task Policy Loss (PL).** Based on [17], the context estimator is trained by directly minimizing the policy loss, effectively using the policy’s objective as a supervision signal. Being the control policy explicitly conditioned on the context representation  $\hat{\mathbf{c}}$  (Eq. (2)),  $\mathcal{L}_{\pi}$  is differentiable with respect to the parameters of  $\phi$ . Consequently, the gradients of the control policy loss can be backpropagated directly into the context estimator:

$$\mathcal{L}_{\phi}^{PL} = \mathcal{L}_{\pi}, \quad (6)$$

where  $\mathcal{L}_{\pi}$  denotes the policy loss associated with the selected RL algorithm. This implicit supervision drives  $\phi$  to learn latent context representations that maximise the policy’s expected return.

## 4. Evaluation

We evaluate the different supervision strategies on both sim-to-sim and sim-to-real transfer tasks. In Sec. 4.1, we consider a classic control task commonly used in RL, assessing the zero-shot generalization of policies across different simulated domains. Although this setup does not involve a robot or actual sim-to-real transfer, it offers a low-cost and reproducible benchmark for comparison in controlled domain conditions. In Sec. 4.2, we assess the sim-to-real transfer on a physical robot to evaluate the effectiveness of the strategies in real-world settings. In all experiments, we employ the Soft Actor-Critic (SAC) algorithm [19] for training the policies, in its implementation provided by *Stable-Baselines3* [20]. We design the actor and critic networks with 2 layers of 256 neurons each, while the context estimator and the predictor with 2 layers of 16 neurons each.

**Baselines.** We compare the supervision strategies described in Sec. 3.3 with two baseline approaches. The first, referred to as *Oracle*, augments the policy’s input by directly appending the ground truth context to the observable state. While this approach is not applicable to sim-to-real transfer, since the ground-truth context is not available in real-world settings, it serves as a reference profile to assess the performance of the other policies in simulation. In contrast, the second baseline, referred to as *Agnostic*, restricts the policy’s input solely to the observable state, reflecting the conventional

domain randomization approach that does not incorporate explicit context information.

**Evaluation.** To assess the generalization capability of the trained policies, we generate three distinct context sets: training, validation, and test. These sets are obtained using Latin hypercube sampling with different seeds to ensure coverage and variability within each context space. All policies are trained on the same training set by iteratively cycling through the available contexts, with a different context assigned to each training episode in a round-robin fashion. In contrast to standard RL tasks, which typically allow periodic evaluation on a limited number of episodes in a fixed environment, our framework requires performance assessment across a broad spectrum of context values. Consequently, frequent policy evaluation during training on such an extensive validation set is computationally infeasible. To address this challenge, we adopt a sparser evaluation strategy. After the training, we select a limited number of checkpoints, evenly spaced over a predefined interval of training steps. Among these, the best-performing checkpoint on the validation set is then evaluated on the test set, and we report the corresponding results in the tables. We provide the details regarding the training, validation, and test sets, as well as the checkpoint selection interval, in each experiment’s description.

#### 4.1. Classic Control Task

**Description.** We use the CARL library [10], which provides contextual extensions to standard RL environments by enabling systematic variation of domain parameters such as mass, friction, and damping. For our experiments, we select the *Pendulum* environment and consider gravity magnitude  $g$ , pendulum length  $l$ , and mass  $m$  as context parameters.

We conduct two main experiments. The first experiment investigates how the context dimensionality affects generalization. To this end, we evaluate all possible combinations of the three context parameters, ranging from  $1D$ , where each parameter is considered in isolation, to  $3D$ , where all three parameters vary jointly. This results in a total of seven context space combinations. In line with [10], we define the bounds of each context feature as 0.1 to 2 times its standard value. To enable a fair comparison across different context dimensions, we maintain a consistent sampling density by increasing the number of samples exponentially with the number of dimensions. Specifically, we use 7, 49, and 343 samples for the  $1D$ ,  $2D$ , and  $3D$  spaces, respectively. We run  $10^5$  training steps in the case of the  $1D$  and  $2D$

context spaces, and  $2 \times 10^5$  for the  $3D$  one. For all the supervision strategies but  $GT$ , we fix the dimension of the context representation to the number of context dimensions plus one. This choice follows common practice in related work, where a slightly over-parameterized latent space is used to provide additional representational capacity, which can improve policy performance and generalization. We select 50 checkpoints over the final 25% of training steps for evaluation, and we run 3 episodes per context for both evaluation and test. In Table 1, we report the average and the best test performance in terms of reward return, computed over 10 replicates obtained by varying the seeds for both policy initialization and environment setup.

The second experiment investigates the effect of varying the dimensions of the context representation. We fix the context to the  $3D$  case and train the  $FP$  and  $PL$  policies using context representation dimensions ranging from 2 to 6. As in the first experiment, we generate three sets for training, validation, and testing, performing validation and testing as previously described. Table 2 shows the average and the best results on the test set over 10 replicates.

**Analysis.** Across all evaluated context configurations reported in Table 1, conditioning on explicit contextual information — either via a ground-truth oracle or through a learned estimator — results in markedly improved performance compared to the *Agnostic* baseline. *Oracle* establishes an empirical upper bound on performance, exhibiting very low variance across runs. In no case does a learned estimator exceed *Oracle*’s performance. While the best-performing runs across methods can occasionally approach *Oracle*’s return, none consistently achieve superior performance.

Among the learned supervision strategies, both  $GT$  and  $PL$  exhibit comparable and consistently superior performance. We conduct a Welch’s t-test to evaluate the statistical significance of the observed performance differences between  $GT$  and  $PL$ , due to its robustness to unequal variances across samples. The resulting p-values exceed the conventional significance threshold ( $p > 0.05$ ), indicating that the differences are not statistically significant. This suggests that, under the conditions considered, weak supervision via policy loss is equally effective as fully supervised regression, despite the latter’s access to direct context targets.

Table 1: Test reward on Pendulum with varying context. Numbers encode mean  $\pm$  std (best) across seeds.

| Policy          |        | 1D Context                             |                                        |                                        | 2D Context                             |                                        |                                        | 3D Context                             |
|-----------------|--------|----------------------------------------|----------------------------------------|----------------------------------------|----------------------------------------|----------------------------------------|----------------------------------------|----------------------------------------|
| estimator       |        | g                                      | l                                      | m                                      | g, l                                   | g, m                                   | l, m                                   | g, l, m                                |
| <b>Oracle</b>   | -      | $-280 \pm 1$ (-279)                    | $-160 \pm 2$ (-154)                    | $-148 \pm 3$ (-144)                    | $-354 \pm 13$ (-333)                   | $-358 \pm 5$ (-346)                    | $-240 \pm 7$ (-229)                    | $-384 \pm 7$ (-376)                    |
| <b>Agnostic</b> | -      | $-355 \pm 36$ (-307)                   | $-348 \pm 35$ (-274)                   | $-175 \pm 7$ (-163)                    | $-633 \pm 25$ (-595)                   | $-490 \pm 18$ (-474)                   | $-530 \pm 42$ (-467)                   | $-575 \pm 24$ (-543)                   |
| <b>GT</b>       | FF+AVG | $-313 \pm 16$ (-289)                   | $-199 \pm 23$ (-171)                   | $-374 \pm 179$ (-157)                  | $-488 \pm 37$ (-447)                   | $-463 \pm 39$ (-407)                   | $-458 \pm 78$ (-358)                   | $-522 \pm 36$ (-470)                   |
|                 | LSTM   | $-310 \pm 11$ (-300)                   | $-191 \pm 9$ (-175)                    | <b><math>-173 \pm 17</math></b> (-151) | $-464 \pm 66$ (-378)                   | $-405 \pm 40$ (-362)                   | $-298 \pm 41$ (-242)                   | $-448 \pm 31$ (-389)                   |
| <b>FP</b>       | FF+AVG | $-362 \pm 129$ (-296)                  | $-396 \pm 152$ (-209)                  | $-176 \pm 12$ (-156)                   | $-611 \pm 156$ (-417)                  | $-491 \pm 71$ (-410)                   | $-534 \pm 152$ (-412)                  | $-596 \pm 79$ (-509)                   |
|                 | LSTM   | $-326 \pm 22$ (-298)                   | $-216 \pm 14$ (-188)                   | $-199 \pm 28$ (-167)                   | $-496 \pm 42$ (-428)                   | $-400 \pm 33$ (-371)                   | $-409 \pm 80$ (-314)                   | $-465 \pm 31$ (-401)                   |
| <b>PL</b>       | FF+AVG | $-430 \pm 137$ (-300)                  | $-271 \pm 40$ (-199)                   | $-361 \pm 167$ (-151)                  | $-522 \pm 32$ (-448)                   | $-450 \pm 52$ (-379)                   | $-465 \pm 115$ (-281)                  | $-511 \pm 61$ (-410)                   |
|                 | LSTM   | <b><math>-308 \pm 16</math></b> (-298) | <b><math>-181 \pm 24</math></b> (-163) | $-176 \pm 46$ (-150)                   | <b><math>-461 \pm 71</math></b> (-376) | <b><math>-390 \pm 21</math></b> (-360) | <b><math>-291 \pm 39</math></b> (-235) | <b><math>-441 \pm 81</math></b> (-379) |

Table 2: Test reward on Pendulum with varying latent context dimension. Numbers encode mean  $\pm$  std (best) across seeds.

| Policy    |        | Latent Context Dimensions              |                                        |                                        |                                        |                                        |
|-----------|--------|----------------------------------------|----------------------------------------|----------------------------------------|----------------------------------------|----------------------------------------|
| estimator |        | 2                                      | 3                                      | 4                                      | 5                                      | 6                                      |
| <b>FP</b> | FF+AVG | $-608 \pm 68$ (-490)                   | $-628 \pm 62$ (-496)                   | $-596 \pm 79$ (-509)                   | $-589 \pm 69$ (-510)                   | $-563 \pm 74$ (-471)                   |
|           | LSTM   | $-496 \pm 25$ (-455)                   | $-522 \pm 71$ (-456)                   | $-465 \pm 31$ (-401)                   | $-449 \pm 29$ (-407)                   | $-460 \pm 47$ (-378)                   |
| <b>PL</b> | FF+AVG | $-495 \pm 42$ (-432)                   | <b><math>-497 \pm 43</math></b> (-457) | $-511 \pm 61$ (-410)                   | $-467 \pm 47$ (-398)                   | $-493 \pm 43$ (-426)                   |
|           | LSTM   | <b><math>-466 \pm 39</math></b> (-411) | $-499 \pm 93$ (-425)                   | <b><math>-441 \pm 81</math></b> (-379) | <b><math>-439 \pm 37</math></b> (-380) | <b><math>-438 \pm 35</math></b> (-392) |

Analysis of performance across increasing context dimensionality reveals a degradation in both return and stability for all methods, with broader confidence intervals and growing divergence from *Oracle*. This is particularly pronounced in the *3D* case, underscoring the difficulty of accurately inferring context embeddings in high-dimensional settings.

Ablation results provided in Table 2 further examine the sensitivity of estimator performance to the dimensionality of the latent embedding space. For both *FP* and *PL*, performance improves as the latent dimension increases from 2 to 5, after which it either saturates or exhibits minor deterioration. This observation aligns with prevailing heuristics that advocate for latent spaces whose dimensionality slightly exceeds that of the ground-truth context.

Finally, across all methods and context settings, estimators incorporating recurrent architectures (LSTM) consistently achieve higher average performance than their feed-forward counterparts employing average pooling. In some cases, particularly for low-dimensional contexts, the best-performing feed-forward models can reach performance levels comparable to those of LSTM-based models. However, such outcomes are rare and highly sensitive to initialization and training variability. The significantly higher variance observed with feed-forward models suggests that strong performance is difficult to obtain reliably, whereas LSTM architectures yield robust results more consistently.

#### 4.2. Pushing Task

**Description.** We evaluate sim-to-real transfer on a planar pushing task, where a robot arm uses its end-effector to push a box from an initial to a desired end position (Fig. 2). This task has been extensively studied in the context of sim-to-real transfer, due to its sensitivity to latent physical properties such as mass and friction [21, 22]. We model the problem in the commercial physics engine AGX Dynamics [8], using a high-fidelity digital twin of the Franka Emika Panda robot employed for the real-world experiments (Fig. 1). We use a Cartesian Impedance Controller [23] to learn a policy that directly operates in the Cartesian space, enabling compliant interaction with the environment through force-aware motion control. We define observation and action at time-step  $t$  as follows:

$$\mathbf{s}^t = [x_{ee}^t, y_{ee}^t, x_o^t, y_o^t, \theta_o^t, \Delta x_T^{t-1}, \Delta y_T^{t-1}], \quad (7)$$

$$\mathbf{a}^t = [\Delta x_T^t, \Delta y_T^t], \quad (8)$$

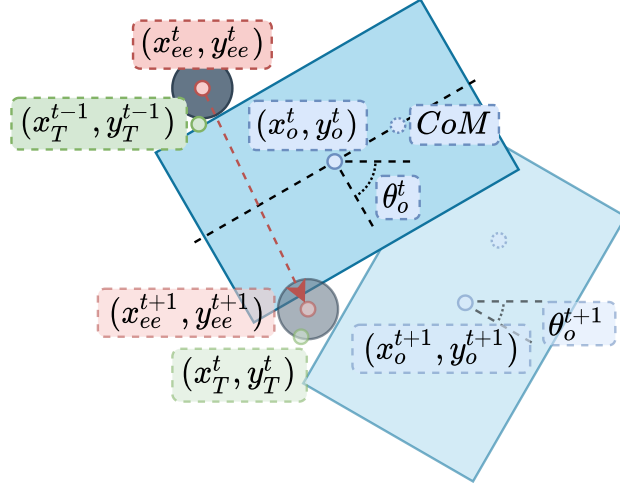


Figure 2: Illustration of the planar pushing task, showing the object and end-effector positions at two consecutive time steps. Solid blue dots represent the object positions, while dashed blue dots represent its center of mass. Red circles represent the position of the cylindrical tool mounted on the robot hand, which is used to push the object. Green circles represent the target end-effector positions issued to the Cartesian Impedance Controller to command the robot end-effector.

where:

- $x_{ee}^t$  and  $y_{ee}^t$  denote the end-effector 2D position;
- $x_o^t, y_o^t, \theta_o^t$  denote the object 2D position and planar orientation;
- $\Delta x_T^t, \Delta y_T^t$  denote the target control action, defined as the displacement relative to the previous action, computed as  $\Delta v^t = v^t - v^{t-1}$  for  $v \in \{x_T, y_T\}$ .

We employ the following reward function, which smoothly penalizes the distance to the goal:

$$r = -\log \left( 1 + \frac{d}{\delta} \right) + \mathbb{I}_{\text{fail}} \cdot r_{\text{fail}} \quad (9)$$

where:

- $d$  is the Euclidean distance between the object and the goal position;
- $\delta$  is a normalization constant;

Table 3: Randomized parameters for the pushing task

| Parameter                       | Sampling Distribution                               |
|---------------------------------|-----------------------------------------------------|
| Box Mass                        | $\mathcal{U}([0.1, 1])$ kg                          |
| Box-Tool Friction               | $\mathcal{U}([0.1, 0.5])$                           |
| Box-Table Friction              | $\mathcal{U}([0.2, 0.8])$                           |
| Box Center of Mass              | $\mathcal{U}([-0.04, 0.04])$ m (rel. to centroid)   |
| Starting Robot Position ( $x$ ) | $\mathcal{U}([0.45, 0.55])$ m                       |
| Starting Robot Position ( $y$ ) | $\mathcal{U}([0.25, 0.35])$ m                       |
| Starting Box Position ( $x$ )   | $\mathcal{U}([-0.03, 0.03])$ m (rel. to robot)      |
| Starting Box Position ( $y$ )   | $\mathcal{U}([-0.1025, -0.0675])$ m (rel. to robot) |
| Starting Box Orientation        | $\mathcal{U}([-0.5236, 0.5236])$ rad                |
| Action Duration                 | $\mathcal{U}(\{0.04 + 0.005 i\}_{i=0}^4)$ s         |
| Box Position Noise              | $\mathcal{N}(0, 0.003^2)$ m                         |
| Box Orientation Noise           | $\mathcal{N}(0, 0.05^2)$ rad                        |

- $r_{\text{fail}}$  is a fixed penalty applied upon failure (i.e., kinematic infeasibility due to unreachable positions).

The task is considered successful when the object is within a fixed threshold of 3 cm from the target location. We truncate episodes after 250 steps.

We conduct two experiments using a box of fixed dimensions 17 cm  $\times$  10.5 cm and a cylindrical tool, measuring 10.8 cm in length and 3 cm in diameter, which is mounted on the robot hand and used to push the object. In the first experiment, the context includes the mass of the box, the friction coefficient between the box and the tool, and the friction coefficient between the box and the table. In the second experiment, we additionally include the box’s center of mass, varying its position along the longer axis of the box. To enhance sim-to-real transfer, we randomize the starting box pose and end-effector position within a pre-defined workspace region, and we perturb the box pose with Gaussian noise to simulate the inaccuracies introduced by the tracking system used in the real-world setup. Additionally, we randomize action duration by discretely sampling the number of simulation steps, while keeping a fixed simulation step of 0.005 s to ensure consistent and stable physics solver performance. Table 3 shows the full list of randomized parameters and corresponding sampling distributions.

Table 4: Test reward on pushing. Numbers show mean  $\pm$  std (best) across seeds.

| Policy          | Without Center of Mass                    |                                             |                                           |                                             | With Center of Mass                       |                                             |                                           |                                             |
|-----------------|-------------------------------------------|---------------------------------------------|-------------------------------------------|---------------------------------------------|-------------------------------------------|---------------------------------------------|-------------------------------------------|---------------------------------------------|
|                 | Sim                                       |                                             | Real                                      |                                             | Sim                                       |                                             | Real                                      |                                             |
|                 | Reward                                    | Success Rate                                | Reward                                    | Success Rate                                | Reward                                    | Success Rate                                | Reward                                    | Success Rate                                |
| <b>Oracle</b>   | $-286 \pm 3$<br>(-283)                    | $0.97 \pm 0.03$<br>(1.00)                   | -<br>-                                    | -<br>-                                      | $-299 \pm 12$<br>(-282)                   | $0.95 \pm 0.02$<br>(0.97)                   | -<br>-                                    | -<br>-                                      |
| <b>Agnostic</b> | $-287 \pm 8$<br>(-276)                    | $0.93 \pm 0.14$<br>(0.94)                   | $-359 \pm 2$<br>(-357)                    | $0.69 \pm 0.03$<br>(0.72)                   | $-381 \pm 22$<br>(-362)                   | $0.68 \pm 0.07$<br>(0.76)                   | $-541 \pm 32$<br>(-502)                   | $0.32 \pm 0.08$<br>(0.40)                   |
| <b>GT</b>       | $-273 \pm 10$<br>(-266)                   | $0.97 \pm 0.02$<br>(0.99)                   | $-340 \pm 24$<br>(-323)                   | $0.71 \pm 0.13$<br>(0.82)                   | $-403 \pm 32$<br>(-378)                   | $0.67 \pm 0.11$<br>(0.82)                   | $-570 \pm 25$<br>(-536)                   | $0.33 \pm 0.05$<br>(0.40)                   |
| <b>FP</b>       | <b><math>-269 \pm 13</math></b><br>(-256) | <b><math>0.99 \pm 0.05</math></b><br>(1.00) | <b><math>-315 \pm 13</math></b><br>(-297) | <b><math>0.78 \pm 0.04</math></b><br>(0.82) | <b><math>-351 \pm 24</math></b><br>(-326) | <b><math>0.78 \pm 0.06</math></b><br>(0.86) | <b><math>-496 \pm 34</math></b><br>(-448) | <b><math>0.48 \pm 0.07</math></b><br>(0.58) |
| <b>PL</b>       | $-277 \pm 16$<br>(-257)                   | $0.97 \pm 0.01$<br>(0.99)                   | $-322 \pm 17$<br>(-302)                   | $0.78 \pm 0.10$<br>(0.92)                   | $-475 \pm 180$<br>(-340)                  | $0.59 \pm 0.39$<br>(0.88)                   | $-592 \pm 89$<br>(-485)                   | $0.31 \pm 0.24$<br>(0.63)                   |

We train and evaluate all policies in simulation, and test them in simulation and on the physical robot. Given the increased complexity of this task compared to the pendulum, we adopt a sampling strategy that mirrors the common 80/10/10 ratio for training, validation, and test. Specifically, we sample 400 context values for training, 50 for validation, and 50 for testing in simulation from the defined context space. We train each policy for  $10^6$  steps and evaluate 200 checkpoints over the second half of training steps, running 2 episodes per context value for both evaluation and test in simulation. Based on findings in Sec. 4.1, we adopt an LSTM as the context estimator architecture and we fix the dimension of the context representation to the number of context dimension plus one — resulting in 4 and 5 dimensions for the variation without and with the center of mass, respectively. On the real robot, we evaluate the policies using 12 context configurations obtained by combining 3 surface materials with 4 box variants. For the setting without center of mass variation, the 4 box variants differ in mass only, achieved by filling the box with different materials, ranging from 283 g to 824 g. For the other setting, the 4 variants differ in both mass and mass distribution, achieved by using internal separators to distribute the filling material non-uniformly, with mass ranging from 276 g to 803 g (Fig. 3). For each of the 12 context configurations, we run 5 episodes, resulting in a total of 60 episodes per test. We employ FoundationPose [24] to track the box pose during real-world experiments. Table 4 shows the average and best results on the test sets over 3 replicates.

**Analysis.** The transfer from simulation to the real robot leads to a performance degradation across all evaluated policies, highlighting the inherent challenge of zero-shot sim-to-real transfer. Nonetheless, policies trained with access to contextual information consistently outperform the *Agnostic* baseline, regardless of the context supervision strategy. Importantly, the relative performance ranking observed in simulation is reflected in the real-world evaluations, indicating consistency between simulated and physical deployments.

In the scenario without center-of-mass variation, all context-aware policies outperform the *Agnostic* baseline. Among these, *FP* yields the highest average return and exhibits minimal variability across seeds, indicating consistent performance. *PL* achieves slightly lower return metrics than *FP*, both on average and in the best-performing seed, but attains the highest success rate for its top-performing policy. As illustrated in Fig. 4, this discrepancy arises because *FP* completes successful episodes in fewer steps and ends unsuccessful ones closer to the goal, resulting in higher overall returns despite a



Figure 3: Context configurations used for real-world evaluation of the pushing task with center of mass variation, obtained by combining 3 surface materials 4 box variants. The surface materials differ in their friction coefficients, while the boxes vary in both mass and mass distribution, achieved by redistributing the filling material using internal separators.

lower success rate. *GT* performs the worst among the context-aware methods both in average and best return metrics, while achieving a similar success rate when compared to *FP*.

In the task variant including the center of mass variation, the performance gap between the *Agnostic* baseline and context-aware methods becomes more pronounced. Notably, *GT* performs worse than the *Agnostic* baseline, making this the only scenario where a context-conditioned policy underperforms the context-agnostic counterpart in our experiments. *FP* maintains consistent performance across seeds, exhibiting limited degradation despite the increased task complexity introduced by the center-of-mass variation. It achieves the highest return, both in terms of average performance and best-performing seed, mirroring the trend observed in the simpler task variant. However, as in the previous scenario, it does not attain the highest success rate. *PL*, on the other hand, reaches competitive performance in one of the seeds but shows substantially higher variance compared to the other methods. This variability is attributed to one of the three training runs failing to converge to a competent policy, even in simulation. Despite this, *PL* yields the highest success rate among all evaluated methods.

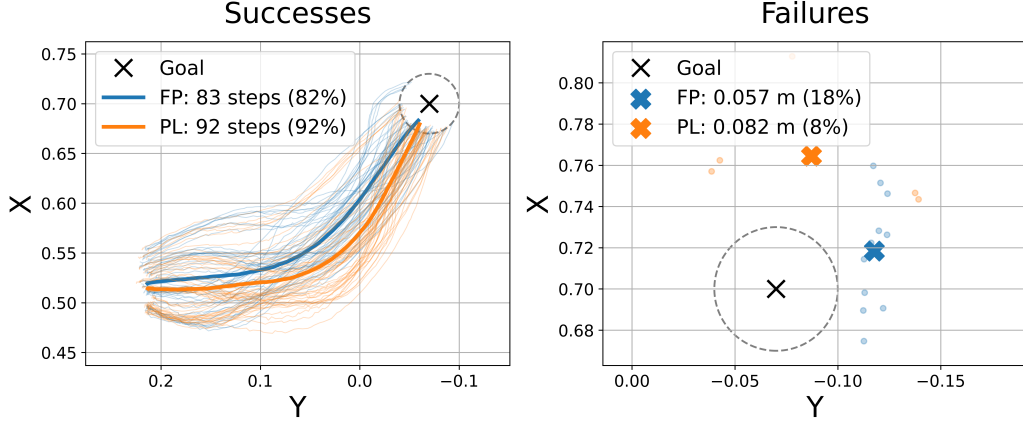


Figure 4: Comparison of the best-performing *FP* and *PL* policies on the pushing task without center of mass variation. Dashed circles denote the success threshold. **Left.** Successful box trajectories (faded lines) and their average ones (bold lines); legend values denote the average number of steps and the percentage of successful trials. **Right.** End box positions of failed trajectories (faded dots) and their centroids (bold crosses); legend values denote the average final distance to the goal and the percentage of failures.

#### 4.3. Discussion

**1) How does contextual information influence zero-shot generalization and sim-to-real transfer?** Conditioning policies on contextual cues, either via a ground-truth oracle or a learned estimator, consistently yields substantial performance gains compared to a context-agnostic baseline. We observe these benefits across both simulation and real-robot experiments: context-aware controllers achieve higher returns and success rates under domain shifts, with the gap widening as the context dimensionality increases. While domain randomization alone offers some robustness, it fails to match the stability and peak performance of context-informed methods. *Oracle* appears to define an empirical upper bound, and no learned strategy surpasses it, particularly in high-dimensional settings where inferring accurate embeddings remains challenging. Although previous works [10, 17] have suggested that some forms of learned supervision may surpass this upper bound, our results do not support this conclusion.

**2) Which supervision strategy is most effective?** Results suggest that the effectiveness of a supervision strategy depends on the considered task. This finding underscores the need for further investigation to determine whether consistent correlations between task characteristics and supervision strategies can be identified. Establishing such correlations would facilitate more principled method selection, reducing the need for exhaus-

tive empirical comparisons. Notably, policies trained using weak supervision through policy loss demonstrate competitive performance across both tasks, despite some variability. Moreover, models incorporating recurrent architectures, such as LSTMs, consistently exhibit improved performance and reduced variance compared to feed-forward counterparts.

**3) What are the main challenges in evaluating generalization across varying contexts for sim-to-real?** Assessing generalization across varying contexts requires distinct training, validation, and test sets, each sampled from the relevant context space. However, this presents two key challenges: computational cost and limitations when testing on real hardware. On the computational side, frequent evaluation of policies across the full validation set during training incurs significant time and resource overhead. This is particularly problematic in high-dimensional contexts, where the number of required samples grows exponentially. While reducing the evaluation frequency or using a sparser subset of contexts can alleviate the computational burden, it may lead to suboptimal model selection or misleading conclusions. On the real robot, additional challenges arise due to the uncertainty in context parameters, such as friction, which are not easily measurable. This can result in the testing of policies on a non-representative subset of the context space. In response to this challenge, we recommend considering multiple context configurations when testing the policies on the real robot to ensure more reliable assessments of sim-to-real transfer.

## 5. Conclusion

In this work, we examine the potential of context-aware policies to improve sim-to-real transfer in robotic control and analyze how different supervision strategies for learning contextual representations affect zero-shot generalization. We conduct a systematic evaluation across simulated and real-world tasks using a Franka Emika Panda robot. Experimental results demonstrate that conditioning policies on contextual information consistently enhances robustness to domain shifts, compared to standard domain randomization. Among the evaluated approaches, weak supervision through policy loss achieves competitive performance across tasks. Nonetheless, no single strategy consistently outperforms the others, indicating that the effectiveness of supervision methods may be task-specific. We also highlight key challenges in evaluating generalization, particularly the computational burden of validation across high-dimensional context spaces and the difficulty of assessing

performance on physical systems with unobservable parameters. Addressing these limitations is essential for reliable benchmarking and deployment.

Future research should aim to identify principled criteria for selecting appropriate supervision strategies based on task properties and further investigate methods for improving context estimation and policy robustness in real-world environments. Moreover, future work should also compare explicit context estimation with memory-based approaches, where recurrent policies infer dynamics through internal memory states derived from past observations [25, 22].

### **Declaration of Generative AI and AI-assisted technologies in the writing process**

During the preparation of this work, the authors used ChatGPT in order to improve readability and language. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

### **Declaration of competing interest**

The authors affirm that there are no known competing interests or financial relationships that could be perceived as potential conflicts of interest.

### **Acknowledgement**

This work was supported in part by Industrial Graduate School Collaborative AI & Robotics (CoAIRob), in part by the Swedish Knowledge Foundation under Grant Dnr:20190128, and the Knut and Alice Wallenberg Foundation through Wallenberg AI, Autonomous Systems and Software Program (WASP).

### **References**

- [1] J. Kober, J. Bagnell, J. Peters, Reinforcement Learning in Robotics: A Survey, The International Journal of Robotics Research 32 (09 2013). doi:10.1177/0278364913495721.
- [2] D. Floreano, P. Husbands, S. Nolfi, Evolutionary Robotics, 2008. doi:10.1007/978-3-540-30301-5\_62.

- [3] M. Breyer, F. Furrer, T. Novkovic, R. Siegwart, J. Nieto, Flexible Robotic Grasping with Sim-to-Real Transfer based Reinforcement Learning, arXiv preprint arXiv:1803.04996 (2018).
- [4] N. Jacobi, P. Husbands, I. Harvey, Noise and the Reality Gap: The Use of Simulation in Evolutionary Robotics, in: Proceedings of the Third European Conference on Advances in Artificial Life, Springer-Verlag, Berlin, Heidelberg, 1995, p. 704–720.
- [5] E. Salvato, G. Fenu, E. Medvet, F. A. Pellegrino, Crossing the Reality Gap: A Survey on Sim-to-Real Transferability of Robot Controllers in Reinforcement Learning, IEEE Access 9 (2021) 153171–153187. doi:10.1109/ACCESS.2021.3126658.
- [6] X. Chen, J. Hu, C. Jin, L. Li, L. Wang, Understanding Domain Randomization for Sim-to-real Transfer, in: International Conference on Learning Representations, 2022.
- [7] W. Yu, J. Tan, C. K. Liu, G. Turk, Preparing for the Unknown: Learning a Universal Policy with Online System Identification, in: N. M. Amato, S. S. Srinivasa, N. Ayanian, S. Kuindersma (Eds.), Robotics: Science and Systems XIII, 2017. doi:10.15607/RSS.2017.XIII.048.
- [8] AGX Dynamics.  
URL <https://www.algoryx.se/agx-dynamics/>
- [9] R. Kirk, A. Zhang, E. Grefenstette, T. Rocktäschel, A Survey of Zero-shot Generalisation in Deep Reinforcement Learning, J. Artif. Int. Res. 76 (May 2023). doi:10.1613/jair.1.14174.
- [10] C. Benjamins, T. Eimer, F. Schubert, A. Mohan, S. Döhler, A. Biedenkapp, B. Rosenhahn, F. Hutter, M. Lindauer, Contextualize Me – The Case for Context in Reinforcement Learning, Transactions on Machine Learning Research (2023).
- [11] J. Matas, S. James, A. J. Davison, Sim-to-Real Reinforcement Learning for Deformable Object Manipulation, in: Conference on Robot Learning (CoRL), 2018, pp. 734–743.
- [12] J. v. Baar, A. Sullivan, R. Cordorel, D. Jha, D. Romeres, D. Nikovski, Sim-to-real transfer learning using robustified controllers in robotic tasks involving complex dynamics, in: 2019 International Conference on Robotics and Automation (ICRA), IEEE Press, 2019, p. 6001–6007. doi:10.1109/ICRA.

2019.8793561.

URL <https://doi.org/10.1109/ICRA.2019.8793561>

- [13] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, L. Zhang, Solving Rubik’s Cube with a Robot Hand, arXiv preprint arXiv:1910.07113 (2019).
- [14] A. Hallak, D. Di Castro, S. Mannor, Contextual markov decision processes, arXiv preprint arXiv:1502.02259 (2015).
- [15] A. Modi, N. Jiang, S. Singh, A. Tewari, Markov Decision Processes with Continuous Side Information, in: F. Janoos, M. Mohri, K. Sridharan (Eds.), *Proceedings of Algorithmic Learning Theory*, Vol. 83, 2018.
- [16] B. Evans, A. Thankaraj, L. Pinto, Context is Everything: Implicit Identification for Dynamics Adaptation, in: *2022 International Conference on Robotics and Automation (ICRA)*, IEEE Press, 2022, pp. 2642–2648. doi:10.1109/ICRA46639.2022.9812119.
- [17] T. C. Ndir, A. Biedenkapp, N. Awad, Inferring Behavior-Specific Context Improves Zero-Shot Generalization in Reinforcement Learning, in: *Seventeenth European Workshop on Reinforcement Learning*, 2024.
- [18] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, *Neural Comput.* 9 (8) (Nov. 1997). doi:10.1162/neco.1997.9.8.1735.
- [19] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, in: J. Dy, A. Krause (Eds.), *Proceedings of the 35th International Conference on Machine Learning*, Vol. 80 of *Proceedings of Machine Learning Research*, PMLR, 2018, pp. 1861–1870.
- [20] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, N. Dormann, Stable-Baselines3: Reliable Reinforcement Learning Implementations, *Journal of Machine Learning Research* 22 (268) (2021) 1–8.
- [21] J. Del Aguila Ferrandis, J. Moura, S. Vijayakumar, Nonprehensile Planar Manipulation through Reinforcement Learning with Multimodal Categorical Exploration, in: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 5606–5613. doi:10.1109/IROS55552.2023.10341629.

- [22] L. Bergmann, D. Leins, R. Haschke, K. Neumann, Precision-Focused Reinforcement Learning Model for Robotic Object Pushing, arXiv preprint arXiv:2411.08622 (2024).
- [23] N. Hogan, Impedance Control: An Approach to Manipulation, in: 1984 American Control Conference, 1984, pp. 304–313.
- [24] B. Wen, W. Yang, J. Kautz, S. Birchfield, FoundationPose: Unified 6D Pose Estimation and Tracking of Novel Objects, in: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024, pp. 17868–17879. doi:10.1109/CVPR52733.2024.01692.
- [25] X. B. Peng, M. Andrychowicz, W. Zaremba, P. Abbeel, Sim-to-real transfer of robotic control with dynamics randomization, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE Press, 2018, p. 1–8. doi:10.1109/ICRA.2018.8460528.  
URL <https://doi.org/10.1109/ICRA.2018.8460528>