Rethinking the Suitability of Reinforcement Learning Algorithms Under Practical Transfer Constraints

Hany Hamed, Abhishek Naik, Colin Bellinger, A. Rupam Mahmood

Keywords: Zero-Shot Transfer, Evaluation Methodology, Domain Randomization

Summary

This paper provides evidence that sample efficiency alone is not enough to assess reinforcement-learning algorithms for transfer-oriented applications. First, while SAC and TD-MPC2 generally learn with fewer interactions, massively parallel PPO can produce a performant policy faster in terms of wall-clock time under commonly used training setups. Further, the effects of domain randomization do not appear to be algorithm-specific and depend on the task, algorithm, training coverage, and evaluation regime. Therefore, we recommend practitioners to consider the combination of interaction efficiency, wall-clock performance, and robustness under dynamics mismatch when selecting and evaluating RL algorithms for transfer-oriented applications.

Contribution(s)

1. Highlighting the gap between conventional evaluation criteria and practical transfer constraints.
   Context: Wall-clock time is a primary operational constraint for many RL practitioners. Simulator interactions are relatively inexpensive in modern simulation-training pipelines and can be generated at scale. This can enable practitioners to obtain a useful and reliable policy within a limited time budget. We show that the ranking of commonly used algorithms under *time-based* evaluation criteria can differ from that suggested by conventional *interaction-based* sample-efficiency criteria.
2. Through a controlled comparison across five domain-randomization levels and two zero-shot transfer regimes, we find that PPO, SAC, and TD-MPC2 can all benefit from domain randomization, with no consistent advantage for any algorithm; its effect remains task-, algorithm-, coverage-, and regime-dependent.
   Context: PPO is widely used with domain randomization in sim-to-real pipelines, whereas the behavior of SAC and TD-MPC2—used widely for their relatively high sample efficiency—is not understood well under comparable randomized training conditions. This paper provides a comprehensive empirical analysis of these algorithms to help practitioners make informed choices when working in transfer-oriented applications.

Rethinking the Suitability of Reinforcement Learning Algorithms Under Practical Transfer Constraints

Hany Hamed[1,3,7], **Abhishek Naik**[4], **Colin Bellinger**[2,5,*], **A. Rupam Mahmood**[1,3,6]
h.hamed.elanwar@gmail.com, armahmood@ualberta.ca,
abhishek.naik@nrc-cnrc.gc.ca, colin.bellinger@uottawa.ca

[1]**University of Alberta**
[2]**University of Ottawa**
[3]**Alberta Machine Intelligence Institute (Amii)**
[4]**National Research Council Canada**
[5]**Vector Institute**
[6]**CIFAR Canada AI Chair**
[7]**Generative Bionics**
[*]Work done while at the National Research Council Canada

Abstract

Transfer-oriented reinforcement learning requires evaluating algorithms along dimensions that go beyond standard sample efficiency. We focus on two dimensions: *practical efficiency*, which asks whether conclusions about algorithm suitability change under wall-clock rather than interaction-based budgets, and *robustness under dynamics mismatch*, which asks how different learning paradigms respond to variability in the training distribution induced by domain randomization. We provide two insights to reinforcement-learning practitioners. First, comparing the sample efficiency of different algorithms is often an insufficient criteria in transfer-oriented settings. The wall-clock time required to train a decent policy is an important consideration for practitioners, and we find that the sample-inefficient PPO algorithm can result in a performant policy faster than the relatively more sample efficient algorithms like SAC and TD-MPC2—validating the common knowledge about massively parallel training paradigms. Second, domain randomization can help different kinds of algorithms learn robust policies. In particular, despite PPO, SAC, and TD-MPC2 representing different RL paradigms—on-policy, off-policy, and model-based learning and planning—we found that domain randomization affects all three algorithms in a similar way. To the best of our knowledge, this is the first controlled comparison of the effect of domain-randomization coverage on PPO, SAC, and TD-MPC2 under the same transfer protocol. Taken together, these two insights highlight the importance of evaluating RL algorithms not only by sample efficiency, but also by practical considerations such as training time and the algorithms' ability to produce usable policies.

1 Introduction

Reinforcement learning (RL) has made significant progress in recent years, particularly in continuous-control domains, where methods such as SAC and TD-MPC2 have demonstrated strong performance under standard evaluation protocols (Haarnoja et al., 2018; Hansen et al., 2024). These methods are commonly developed and assessed with an emphasis on sample efficiency and final performance, reflecting a broader focus within the RL community on interaction-based evaluation. Transfer-oriented applications, however, require evaluating algorithms along dimensions that go beyond standard sample efficiency. In particular, algorithm suitability depends both on *practical*

efficiency—whether a performant policy can be obtained within a realistic wall-clock budget—and on *robustness under dynamics mismatch*—whether the learned policy remains effective when the evaluation dynamics differ from those encountered during training.

These two dimensions expose potential gaps between conventional evaluation protocols and the practical requirements of transfer-oriented workflows. In sim-to-sim and sim-to-real applications, Proximal Policy Optimization (PPO; Schulman et al., 2017) is widely regarded as sample inefficient under conventional benchmarks, yet it remains widely used in practical transfer pipelines. At the same time, domain randomization (DR; Tobin et al., 2017) is commonly used to improve robustness to dynamics mismatch, although it remains unclear whether different RL paradigms respond differently to the variability it introduces. Together, these observations raise two related questions: *do conclusions about algorithm suitability change when training is evaluated under wall-clock rather than interaction-based budgets, and do different learning paradigms respond differently to increasing variability in the training distribution?*

The first question concerns *practical efficiency*. Transfer-oriented workflows are often governed by constraints that are not well captured by interaction-based evaluation alone. In many simulated training pipelines, environment interactions are relatively inexpensive and can be generated at scale, whereas wall-clock time remains a primary operational constraint. Practitioners often care about obtaining a reliable and sufficiently performant policy within a limited training-time budget and therefore value properties such as scalability under parallel simulation, stability, reproducibility, and time-to-performance. Under such conditions, the ranking of algorithms may differ from that suggested by conventional sample-efficiency comparisons (e.g., see Henderson et al., 2018). In particular, an algorithm that requires more interactions may nevertheless produce a usable policy sooner if it can exploit massively parallel simulation more effectively.

The second question concerns *robustness under dynamics mismatch*. Suitability for transfer depends not only on how quickly a policy can be obtained, but also on how well it generalizes when the deployment dynamics differ from the nominal training dynamics. Domain randomization addresses this mismatch by exposing policies to a distribution of environment dynamics during training. Although PPO has repeatedly been paired with domain randomization in sim-to-real pipelines (Handa et al., 2023; Andrychowicz et al., 2020), the relative robustness of different RL paradigms under matched randomized dynamics has not been systematically established. It therefore remains unclear whether on-policy, off-policy, and model-based methods respond differently to increasing domain-randomization coverage, or whether transfer robustness is governed more directly by how the training distribution is constructed.

To study these questions, we conduct a systematic comparison of PPO, Soft Actor-Critic (SAC; Haarnoja et al., 2018), and TD-MPC2 (Hansen et al., 2024) in zero-shot sim-to-sim transfer across a diverse suite of continuous-control tasks. We evaluate these methods under both interaction-based and wall-clock budgets to examine how conclusions about practical efficiency change across evaluation axes. In parallel, we vary the coverage of the training distribution through domain randomization to study how the three learning paradigms respond to dynamics mismatch.

Our results show that conclusions about algorithm suitability depend strongly on the evaluation axis. Under interaction-based evaluation, SAC and TD-MPC2 are generally more sample efficient than PPO. Under wall-clock evaluation, however, PPO with parallel environments often reaches competitive performance earlier than the SAC and TD-MPC2 configurations considered in this study. In the transfer experiments, domain randomization does not systematically favor or disadvantage any particular algorithmic paradigm. Instead, its effect depends on the task, algorithm, training-distribution coverage, and evaluation regime. We support these findings through a controlled comparison of PPO, SAC, and TD-MPC2 across 18 continuous-control tasks, five domain-randomization levels, and two zero-shot sim-to-sim evaluation regimes. Together, these results show that algorithm suitability for transfer depends both on the evaluation axis and on how the training distribution is constructed.

2 Related Work

Algorithmic trade-offs in continuous control. PPO, SAC, and TD-MPC2 represent three major algorithmic paradigms in modern continuous-control RL. PPO emphasizes simple and stable optimization motivated by on-policy learning (Schulman et al., 2017). SAC improves sample efficiency through off-policy learning through a large replay buffer and was explicitly motivated in part by the need for greater stability and sample efficiency in real-time robot learning (Haarnoja et al., 2018). TD-MPC and TD-MPC2 pursue further gains in interaction efficiency by combining learned dynamics model with online planning, and report strong results on standard continuous-control benchmarks (Hansen et al., 2022; 2024). On the other hand, PPO is predominantly used in robotic training pipelines, which leverage its ability to exploit massive simulator parallelism (Rudin et al., 2022). Such parallelism has not been widely employed with SAC or TD-MPC2, even when they are used for sim-to-real transfer, examples of which are few and far between (e.g., Rizzardo et al., 2023; Yin et al., 2025). Using massive parallelization with SAC is an active research area with signs of progress observed in a few notable works (Li et al., 2023; Raffin, 2025; Huang et al., 2026). Narendra et al. (2025) noted diminishing stability of TD-MPC2 when utilized with a higher-degree of parallelism and attributed the difficulty to the off-policy nature of these algorithms. Instead of introducing new algorithmic innovations, our paper focuses on comparing these three existing algorithms in their canonical and commonly used form and shifts the emphasis from sample efficiency to suitability under transfer-oriented constraints, where training time becomes a primary concern.

Evaluation axes. Interaction-based evaluation is the most common paradigm in the RL literature that presents new algorithmic ideas. The motivation for this paradigm is *sample efficiency*: does the new algorithmic idea result in faster learning and good performance with a fixed budget of agent-environment interactions across a range of representative problems (e.g., Schulman et al., 2017; Sutton & Barto, 2018; Wan et al., 2021; Vasan et al., 2024)? On the other hand, some papers introduce (what we call) *infrastructural* insights of running existing algorithm more efficiently on the underlying hardware (e.g., using parallel threads, distributed computation). These tend to use time-based evaluations to showcase the benefit of, say, parallelization over a serial approach (e.g., Nair et al., 2015; Mnih et al., 2016; Heess et al., 2017; Horgan et al., 2018; Zakka et al., 2025). Such time-based evaluation is also relevant in *transfer* settings: when RL is used to solve specific problems (virtual—like a particular game such as AlphaGo—or real—such as a robotic picking task), the best learned policy is transferred to the deployment setting, whether the computer playing against a human, or the robot performing the picking task.

Transfer under dynamics mismatch. A large body of work studies transfer from simulation to deployment by broadening the training distribution to reduce the reality gap. Early results showed that visual randomization can enable perception and control policies trained only in simulation to transfer without target-domain images (Sadeghi & Levine, 2017; Tobin et al., 2017). For control, dynamics randomization became a standard approach for zero-shot transfer, with successful demonstrations in robotic manipulation and locomotion (Peng et al., 2018; Tan et al., 2018). Subsequent work asked how the randomization distribution itself should be chosen, either through automatic curricula such as automatic domain randomization or through adaptive schemes such as Active Domain Randomization (Akkaya et al., 2019; Mehta et al., 2020). More recent theory formalizes domain randomization as learning over a family of parameterized MDPs and studies when randomized training can reduce the sim-to-real gap (Chen et al., 2021). Our work is complementary to this literature: rather than proposing a new randomization scheme, we ask whether changing the breadth of the training distribution alters the relative suitability of different RL algorithms.

3 Problem Setting

In transfer-oriented reinforcement learning, algorithm suitability depends on two complementary dimensions. The first is *practical efficiency*: how quickly a useful policy can be obtained under the operational constraints of a transfer pipeline. The second is *robustness under dynamics mismatch*:

Table 1: **Training coverage vs. evaluation regimes.** Relationship between the support of the training context distribution $p_{\texttt{train}}$ and the fixed evaluation regimes. "In-support" indicates evaluation contexts lie within $\text{supp}(p_{\texttt{train}})$, "Out-of-support" indicates they lie outside, and "Partially out-of-support" indicates mixed coverage depending on context dimensions and regime.

Evaluation Regime	Nominal	Narrow	Moderate	Broad	Extensive
Nominal-Centered	Out-of-support (except nominal point)	In-support	In-support	In-support	In-support
Shifted	Out-of-support	Out-of-support	Partially out-of-support	In-support	In-support

how well a policy trained in simulation generalizes when deployed under environment dynamics that differ from those seen during training. These dimensions are not fully captured by conventional evaluation protocols that prioritize sample efficiency alone. We therefore first distinguish between interaction-based and wall-clock-based evaluation axes, and then formalize transfer as generalization across context distributions. This formulation allows us to study how shaping the training distribution through domain randomization affects robustness under dynamics mismatch.

3.1 Practical Transfer Constraints and Evaluation Axes

In simulated transfer settings, environment interactions are often inexpensive and can be generated at scale using modern compute resources such as parallel environments or GPU-accelerated simulation. As a result, performance is no longer limited solely by the number of interactions collected, but by the time required to produce a policy that performs reliably under dynamics mismatch. Conventional reinforcement learning evaluations emphasize interaction-based performance, measuring returns as a function of environment samples and prioritizing sample efficiency. However, in transfer-oriented pipelines where simulations are cheap and parallelization is available, interaction budgets can often be expanded without significant operational cost. In contrast, wall-clock time remains a critical constraint, governing iteration speed, experimentation cycles, and deployment readiness.

To capture this distinction, we consider two complementary evaluation axes. The **interaction-based axis** measures performance as a function of collected experience and reflects conventional notions of *sample efficiency*. This perspective prioritizes learning progress per environment interaction and is widely used in algorithm benchmarking. The **time-based axis** measures performance as a function of *wall-clock training time* and reflects practical considerations in simulation-driven transfer workflows. Under time constraints, factors that are less visible in interaction-based evaluation may become decisive, including training stability, variability across runs, and the best performance achieved within a fixed time budget. Furthermore, an algorithm's ability to exploit environment parallelism directly influences throughput and thus time-to-performance, making parallelizability an important practical consideration.

3.2 Transfer as Generalization Across Contexts

We model transfer under dynamics mismatch using a contextual Markov decision process (CMDP). Let the environment dynamics depend on a latent context variable $\theta \in \Theta$, which parameterizes physical properties such as mass, friction, damping, actuator strength, or latency. The transition dynamics are therefore defined as $p(s'|s, a, \theta)$. Training occurs under contexts sampled from a distribution $\theta \sim p_{\texttt{train}}$ while deployment occurs under $\theta \sim p_{\texttt{eval}}$. Zero-shot transfer corresponds to evaluating a policy trained under $p_{\texttt{train}}$ directly on contexts drawn from $p_{\texttt{eval}}$, without further adaptation. Transfer performance therefore depends on how well the learned policy generalizes across context distributions in which the dynamics mismatch arises when $p_{\texttt{train}} \neq p_{\texttt{eval}}$.

3.3 Training Distributions

In addition to evaluating algorithm suitability under practical constraints, we examine the role of domain randomization (DR), a widely adopted technique for improving transfer robustness. In

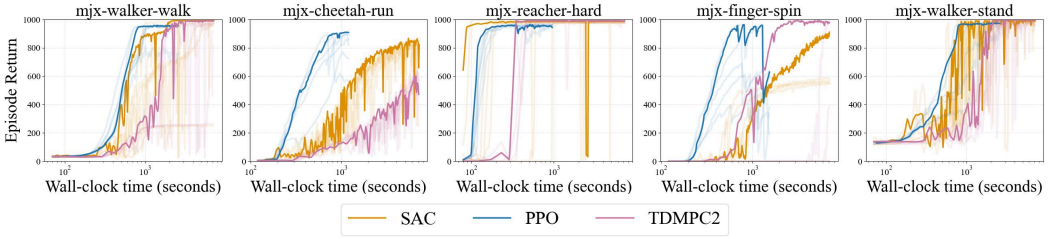

Figure 1: **Wall-clock learning curves.** Performance is plotted against wall-clock training time on a logarithmic scale. Light curves show all six seeds. The dark curve shows the best-performing run, obtained by ranking runs according to the sum of returns over the full recorded learning curve.

transfer-oriented workflows, DR is commonly used to introduce controlled variability in environment dynamics during training, with the goal of preparing policies for deployment under mismatch.

From the CMDP perspective, this corresponds to shaping the training distribution $p_{\texttt{train}}$ by exposing the policy to variations in physical parameters such as mass, friction, damping, length, actuator strength, and latency. Rather than treating DR solely as a robustness mechanism, we study it as a means of introducing distributional variability during training and assess whether such variability alters the relative suitability of different learning paradigms. To this end, we consider training regimes with increasing variability in contexts, allowing us to analyze how the breadth of training exposure influences zero-shot generalization under mismatch.

3.4 Evaluation Distributions

To assess transfer under dynamics mismatch, we evaluate policies across fixed evaluation regimes whose relationship to the training distribution varies depending on training coverage. We consider two evaluation regimes. The first, termed **Nominal-Centered Evaluation**, consists of contexts that remain close to nominal dynamics. The second, termed **Shifted Evaluation**, consists of contexts that introduce structured deviations from nominal parameters.

Importantly, these regimes are defined independently of the training distribution. As the coverage of the training distribution $p_{\texttt{train}}$ expands through domain randomization, the same evaluation contexts may transition from being out-of-support to partially or fully supported. Transfer performance therefore depends not only on the evaluation regime itself, but on its relationship to the support of the training distribution. Table 1 summarizes this relationship across the training coverage regimes considered in this work. While Nominal-Centered evaluation remains within the support of all, Shifted evaluation spans varying degrees of mismatch depending on training coverage, ranging from fully out-of-support under nominal training to fully supported under extensive coverage. This formulation allows us to analyze transfer performance as a function of the interaction between training coverage and evaluation mismatch, rather than treating evaluation difficulty as a fixed property.

4 Results and Analysis

We study algorithm suitability in transfer-oriented reinforcement learning along two complementary dimensions. The first concerns *practical efficiency*: whether conclusions about algorithm quality change when methods are evaluated under the wall-clock constraints that arise in transfer-oriented workflows, rather than only under conventional interaction-based metrics. The second concerns *robustness under dynamics mismatch*: whether different RL paradigms respond differently to increasing variability in the training distribution, as induced by domain randomization. These two dimensions allow us to examine suitability for transfer not only in terms of how quickly a useful policy can be obtained, but also in terms of how reliably it generalizes under mismatch.

Algorithms. We consider three reinforcement learning algorithms that represent distinct learning paradigms commonly used in continuous control and transfer settings. PPO (Schulman et al., 2017) represents an on-policy method that relies on policy gradient optimization and is widely adopted in transfer-oriented applications due to its stability and scalability under parallelized training. SAC

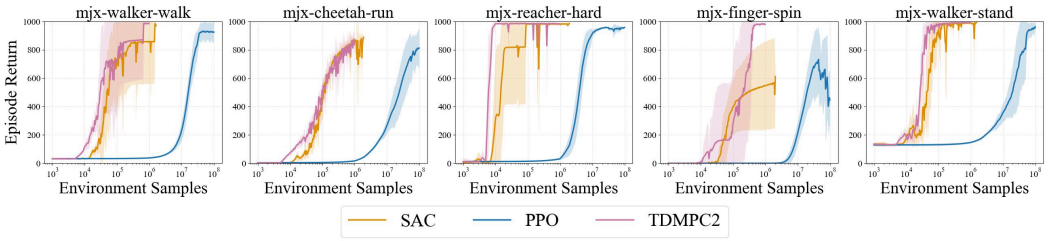

Figure 2: **Interaction-based learning curves.** Performance is plotted against the environment interactions on a logarithmic scale, for the same training runs shown in Fig. 1. The dark curve shows the *mean* across six seeds, with shaded regions indicating standard deviation. Both the x-axis and the plotting statistics correspond to the conventional interaction-based evaluation protocol.

(Haarnoja et al., 2018) represents an off-policy method that improves sample efficiency through entropy-regularized actor-critic learning and experience replay. TD-MPC2 (Hansen et al., 2024) represents a model-based method that leverages learned dynamics for planning and has demonstrated strong performance in interaction-based benchmarks. These algorithms span on-policy, off-policy, and model-based approaches, allowing us to examine how learning paradigm influences suitability under practical transfer constraints.

Environments. We evaluate the algorithms on 18 continuous control tasks from the DeepMind Control Suite (DMC; Tunyasuvunakool et al., 2020) implemented using MuJoCo XLA (MJX; MuJoCo XLA Authors, 2026). To enable controlled analysis of transfer, we parameterize the environment dynamics through multiplicative variations of key physical properties relative to their nominal values, including gravity, body mass, damping, actuator strength, friction, and link length. This allows us to instantiate the contextual transfer setting introduced in Section 3 and to vary the breadth of the training distribution through domain randomization.

To structure the analysis, we ask four questions. First, how do the algorithms compare under the wall-clock constraints relevant to transfer-oriented training? Second, how does this comparison relate to their conventional interaction-based ranking? Third, does domain randomization systematically favor or disadvantage particular RL paradigms? Fourth, how does the extent of training coverage affect transfer robustness and training stability?

4.1 Wall-clock evaluation under practical training constraints

Figure 1 reports performance as a function of wall-clock time. The light curves show the individual runs, while the dark curve highlights the best-performing run, determined by ranking runs according to the sum of returns over the complete learning curve. Displaying the individual runs preserves performance variation that may be hidden by pointwise aggregation (Tanaka & Mahmood, 2026). Figure 2 provides a complementary interaction-based view, following the conventional protocol of reporting the mean and one standard deviation across six seeds.

Under this evaluation axis, PPO often reaches strong performance earlier than SAC and TD-MPC2. This behavior is particularly evident on `cheetah-run` and `finger-spin`, where PPO reaches high returns within a shorter training time. PPO benefits from collecting experience across 2,048 parallel environments, allowing it to rapidly accumulate interactions despite its lower sample efficiency. The observed wall-clock advantage therefore reflects PPO under its commonly adopted massively parallel configuration relative to the SAC and TD-MPC2 default configurations considered in this study.

4.2 Interaction-based evaluation reproduces the conventional ranking

In Figure 2, we provide the complementary interaction-based evaluation, reporting the mean and one standard deviation across six seeds. Under this conventional protocol, SAC and TD-MPC2 generally reach strong performance with fewer environment interactions than PPO. This difference is particularly clear on `reacher-hard`, `finger-spin`, and `walker-stand`. PPO learns substantially

later than SAC and TD-MPC2 and finishes below TD-MPC2, although its final mean return is comparable to SAC. More generally, final performance remains task-dependent, with the algorithms approaching similar returns on some tasks and SAC or TD-MPC2 retaining an advantage on others.

These results reproduce the conventional ranking in terms of sample efficiency and clarify the wall-clock observations in Section 4.1. PPO does not reach strong performance earlier because it uses interactions more efficiently; rather, massive environment parallelism allows it to generate those interactions within a shorter training time. Thus, interaction-based and wall-clock evaluation provide complementary views of algorithm suitability. Complete results across all 18 tasks are provided in the supplementary material.

4.3 Effect of domain randomization and training coverage

While previous sections examine algorithm suitability through the lens of practical efficiency, transfer performance also depends on how the training distribution is constructed. We now turn to this second dimension by analyzing the effect of domain randomization on robustness under dynamics mismatch. Domain randomization (DR) is widely used to improve robustness in such settings by exposing policies to variability in environment dynamics during training. One possible explanation for the widespread practical use of PPO is that alternative methods may behave less reliably when trained under domain randomization. To examine this possibility, we evaluate algorithms trained under multiple domain randomization regimes using the shifted evaluation environments described in Section 3.4. Figures 3 and 4 report zero-shot transfer performance for policies trained without domain randomization and under four levels of increasing coverage.

Across the representative tasks, domain randomization does not systematically favor any particular algorithmic paradigm. PPO, SAC, and TD-MPC2 can all benefit from randomized training, but the effect varies across tasks and coverage levels. For example, randomization substantially improves SAC on `walker-walk` and `walker-stand`, while Narrow randomization achieves strong performance across all three algorithms on `finger-spin`. Conversely, broader randomization reduces performance for several algorithm–task pairs. These results provide no evidence that SAC or TD-MPC2 is systematically less compatible with domain randomization.

Transfer performance also does not improve monotonically with training coverage. Under Nominal-Centered evaluation, Narrow randomization often improves performance over nominal training, while further broadening can reduce these gains. Under Shifted evaluation, Broad and Extensive training distributions overlap with the evaluation contexts, yet they do not consistently outperform Narrow or Moderate randomization. On `finger-spin`, for example, Narrow randomization performs strongly even though the Shifted evaluation contexts remain outside its training support.

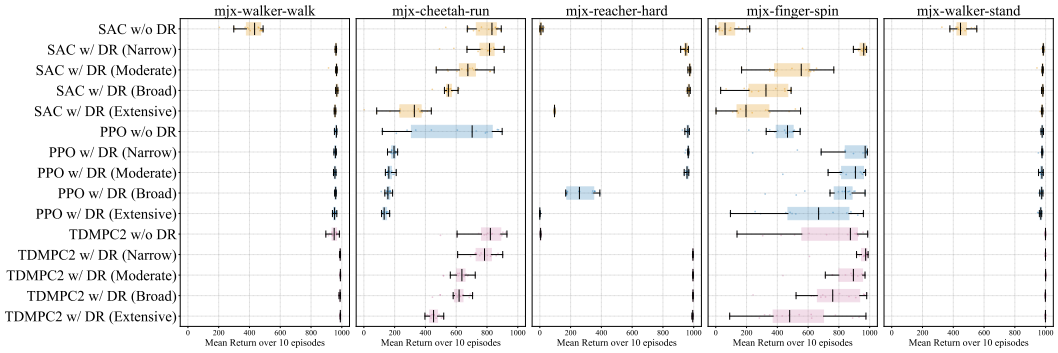

Figure 3: **Zero-shot transfer under Nominal-Centered evaluation.** Each box summarizes performance across 15 fixed evaluation environments, with the return in each environment averaged over 10 episodes. The center line denotes the median, the box spans the first and third quartiles, and the whiskers extend to the most extreme observations within 1.5 times the interquartile range. Points beyond the whiskers are shown individually.

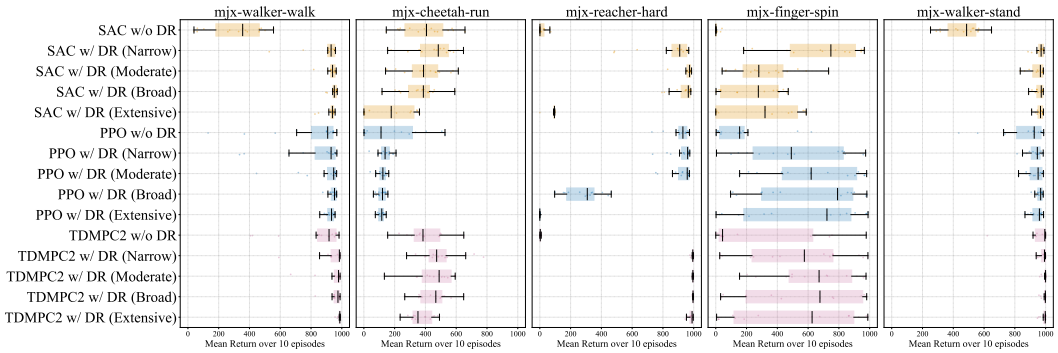

Figure 4: **Zero-shot transfer under Shifted evaluation.** Each box summarizes performance across 15 fixed evaluation environments, with the return in each environment averaged over 10 episodes.

These observations show that covering the evaluation distribution is not sufficient to ensure better transfer. Broader randomization increases exposure to diverse dynamics, but also increases the difficulty of the learning problem and reduces the concentration of experience within particular regions. Domain randomization therefore introduces a trade-off between distributional coverage and effective learning, with the preferred coverage depending on both the task and the algorithm.

5 Conclusions and Future Work

This study examines algorithm suitability in transfer-oriented reinforcement learning along two dimensions beyond standard sample efficiency: practical efficiency and robustness under dynamics mismatch. First, although SAC and TD-MPC2 are generally more sample efficient, PPO with massively parallel environments often produces a performant policy within a shorter wall-clock time under the configurations considered in this study. This finding supports the practical value of massively parallel training and shows that time-to-train is an important consideration alongside interaction-based sample efficiency. Second, domain randomization can improve robustness for PPO, SAC, and TD-MPC2, without consistently favoring any one of them. Its effect instead depends on the task, algorithm, training coverage, and evaluation regime, and broader randomization does not necessarily lead to better transfer. Taken together, these two findings highlight the importance of evaluating RL algorithms not only by sample efficiency, but also by practical considerations such as training time and the algorithms' ability to produce usable policies.

This study has several limitations and avenues for future work. First, our experiments focus on zero-shot sim-to-sim transfer in DeepMind Control Suite tasks, which do not capture the full complexity of real physical systems. The immediate next step is to extend these experiments to the sim-to-real setting. A precursor to this step could also involve a sim-to-sim transfer to a different simulator in which the same control problems are implemented.

Next, the wall-clock comparison at present reflects the common training configurations: PPO uses large-scale environment parallelization while SAC and TD-MPC2 are evaluated with their standard single-environment setups. An obvious extension of this work is to consider parallel-environment variants of the other algorithm. Recent work by Raffin (2025) and Seo et al. (2025) would be a good starting point for parallel variants of SAC and could help answer interesting questions such as whether on-policy algorithms like PPO scale better than off-policy and model-based methods like SAC and TD-MPC2 in the parallel-environment paradigm—and if so, why?

Acknowledgments

We are grateful to the National Research Council Canada's (NRC) AI for Design Challenge Program and the Canada CIFAR Chairs program for the financial support for this project. We also thank the anonymous reviewers for constructive feedback. Finally, the computational resources used in the paper were provided by the Digital Research Alliance of Canada.

A Implementation Details

Algorithms. We use widely adopted reference implementations for each algorithm. SAC is implemented using the CleanRL (Huang et al., 2022) continuous control implementation[1], TD-MPC2 is implemented using the official codebase released by the authors[2], and PPO is adopted from Brax (Freeman et al., 2021) following its configuration for training with massively parallel $2,048$ environments[3]. SAC and TD-MPC2 are trained using their standard configurations as provided in their respective implementations.

Environments. We evaluate the algorithms on tasks from the DeepMind Control Suite (DMC) implemented using MuJoCo XLA (MJX; MuJoCo XLA Authors, 2026). The following domains and tasks are included in our study:

`walker-walk`, `walker-run`, `walker-stand`, `cheetah-run`, `reacher-easy`, `reacher-hard`, `acrobot-swingup`, `pendulum-swingup`, `cartpole-balance`, `cartpole-balance_sparse`, `cartpole-swingup`, `cartpole-swingup_sparse`, `ball_in_cup-catch`, `finger-spin`, `finger-turn_easy`, `finger-turn_hard`, `hopper-stand`, `hopper-hop`.

Training computation. All experiments were run on compute nodes equipped with NVIDIA A100 GPUs. To construct the wall-clock evaluation axis, we averaged the elapsed time per environment interaction over a representative training window and used it to convert training steps into wall-clock time.

To study transfer under dynamics mismatch, we apply domain randomization to key physical parameters of the environments. Specifically, we randomize body mass, joint damping, actuator strength, friction, and link length. Each parameter is scaled multiplicatively relative to its nominal value.

We consider four domain randomization regimes of increasing coverage: *narrow*, *moderate*, *broad*, and *extensive*. For each regime, scaling factors are sampled uniformly within the ranges listed below.

Table A.1: **Domain randomization ranges for training and evaluation regimes.** Multiplicative scaling factors applied to environment parameters relative to nominal values. The shifted evaluation regime samples parameters from ranges that exclude the nominal-centered region.

Parameter	Evaluation		Training			
	Nominal-centered	Shifted	Narrow	Moderate	Broad	Extensive
Mass	$[0.85, 1.15]$	$[0.70, 0.85] \cup [1.15, 1.30]$	$[0.85, 1.15]$	$[0.80, 1.20]$	$[0.70, 1.30]$	$[0.50, 1.80]$
Damping	$[0.70, 1.50]$	$[0.50, 0.70] \cup [1.50, 2.00]$	$[0.70, 1.50]$	$[0.60, 1.70]$	$[0.50, 2.00]$	$[0.30, 3.00]$
Actuator	$[0.85, 1.15]$	$[0.70, 0.85] \cup [1.15, 1.30]$	$[0.85, 1.15]$	$[0.80, 1.20]$	$[0.70, 1.30]$	$[0.50, 1.60]$
Friction	$[0.70, 1.30]$	$[0.50, 0.70] \cup [1.30, 1.50]$	$[0.70, 1.30]$	$[0.60, 1.40]$	$[0.50, 1.50]$	$[0.30, 2.00]$
Length	$[0.95, 1.05]$	$[0.85, 0.95] \cup [1.05, 1.15]$	$[0.95, 1.05]$	$[0.90, 1.10]$	$[0.85, 1.15]$	$[0.80, 1.20]$

For evaluation, we construct fixed test sets by sampling 15 environments for each of the two evaluation regimes, Nominal-Centered and Shifted. These environments are sampled once and reused across all algorithms and training configurations to ensure consistent comparisons. For the reported zero-shot sim-to-sim transfer results, we select the best-performing run among the training seeds.

[1] https://github.com/vwxyzjn/cleanrl/blob/master/cleanrl/sac_continuous_action.py
[2] https://github.com/nicklashansen/tdmpc2
[3] https://github.com/google/brax

References

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.

Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

Xiaoyu Chen, Jiachen Hu, Chi Jin, Lihong Li, and Liwei Wang. Understanding domain randomization for sim-to-real transfer. In *International Conference on Learning Representations*, 2021.

C Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax: A differentiable physics engine for large-scale rigid-body simulation. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor–critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pp. 1856–1865, 2018.

Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, et al. DeXtreme: Transfer of agile in-hand manipulation from simulation to reality. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5977–5984, 2023.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In *International Conference on Machine Learning*, pp. 8387–8406, 2022.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, S. M. Ali Eslami, Martin A. Riedmiller, and David Silver. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver. Distributed prioritized experience replay. In *International Conference on Learning Representations*, 2018.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and João G.M. Araújo. CleanRL: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022. URL https://jmlr.org/papers/v23/21-1342.html.

Weidong Huang, Zhehan Li, Hangxin Liu, Biao Hou, Yao Su, and Jingwen Zhang. Towards bridging the gap between large-scale pretraining and efficient finetuning for humanoid control. *arXiv preprint arXiv:2601.21363*, 2026.

Zechu Li, Tao Chen, Zhang-Wei Hong, Anurag Ajay, and Pulkit Agrawal. Parallel Q-learning: Scaling off-policy reinforcement learning under massively parallel simulation. In *International Conference on Machine Learning*, pp. 19440–19459, 2023.

Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J Pal, and Liam Paull. Active domain randomization. In *Conference on Robot Learning*, pp. 1162–1176, 2020.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937, 2016.

MuJoCo XLA Authors. MuJoCo XLA (MJX), 2026. URL https://mujoco.readthedocs.io/en/stable/mjx.html. Accessed: June 24, 2026.

Arun Nair, Praveen Srinivasan, Sam Blackwell, Cagdas Alcicek, Rory Fearon, Alessandro De Maria, Vedavyas Panneershelvam, Mustafa Suleyman, Charles Beattie, Stig Petersen, Shane Legg, Volodymyr Mnih, Koray Kavukcuoglu, and David Silver. Massively parallel methods for deep reinforcement learning. *arXiv preprint arXiv:1507.04296*, 2015.

Aditya Narendra, Dmitry Makarov, and Aleksandr Panov. M3PO: Massively multi-task model-based policy optimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9747–9752, 2025.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3803–3810, 2018.

Antonin Raffin. Getting SAC to work on a massively parallel simulator: An RL journey with off-policy algorithms. Blog post, February 2025. URL https://araffin.github.io/post/sac-massive-sim/.

Carlo Rizzardo, Fei Chen, and Darwin Caldwell. Sim-to-real via latent prediction: Transferring visual non-prehensile manipulation policies. *Frontiers in Robotics and AI*, 9:1067502, 2023.

Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In *Conference on Robot Learning*, pp. 91–100, 2022.

Fereshteh Sadeghi and Sergey Levine. CAD2RL: Real single-image flight without a single real image. In *Robotics: Science and Systems*, 2017.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Younggyo Seo, Carmelo Sferrazza, Juyue Chen, Guanya Shi, Rocky Duan, and Pieter Abbeel. Learning sim-to-real humanoid locomotion in 15 minutes. *arXiv preprint arXiv:2512.01996*, 2025.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. In *Robotics: Science and Systems*, 2018.

Haruto Tanaka and A Rupam Mahmood. Performance variation in deep reinforcement learning. *arXiv preprint arXiv:2606.06746*, 2026.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 23–30, 2017.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638. DOI: 10.1016/j.simpa.2020.100022. URL https://www.sciencedirect.com/science/article/pii/S2665963820300099.

Gautham Vasan, Mohamed Elsayed, Seyed Alireza Azimi, Jiamin He, Fahim Shahriar, Colin Bellinger, Martha White, and Rupam Mahmood. Deep policy gradient methods without batch updates, target networks, or replay buffers. In *Advances in Neural Information Processing Systems*, volume 37, pp. 845–891, 2024.

Yi Wan, Abhishek Naik, and Richard S Sutton. Learning and planning in average-reward markov decision processes. In *International Conference on Machine Learning*, pp. 10653–10662, 2021.

Patrick Yin, Tyler Westenbroek, Ching-An Cheng, Andrey Kolobov, and Abhishek Gupta. Rapidly adapting policies to the real-world via simulation-guided fine-tuning. In *International Conference on Learning Representations*, 2025.

Kevin Zakka, Baruch Tabanpour, Qiayuan Liao, Mustafa Haiderbhai, Samuel Holt, Jing Yuan Luo, Arthur Allshire, Erik Frey, Koushil Sreenath, Lueder A. Kahrs, Carmelo Sferrazza, Yuval Tassa, and Pieter Abbeel. MuJoCo playground. *arXiv preprint arXiv:2502.08844*, 2025.

Supplementary Materials

The following content was not necessarily subject to peer review.

B Full Results

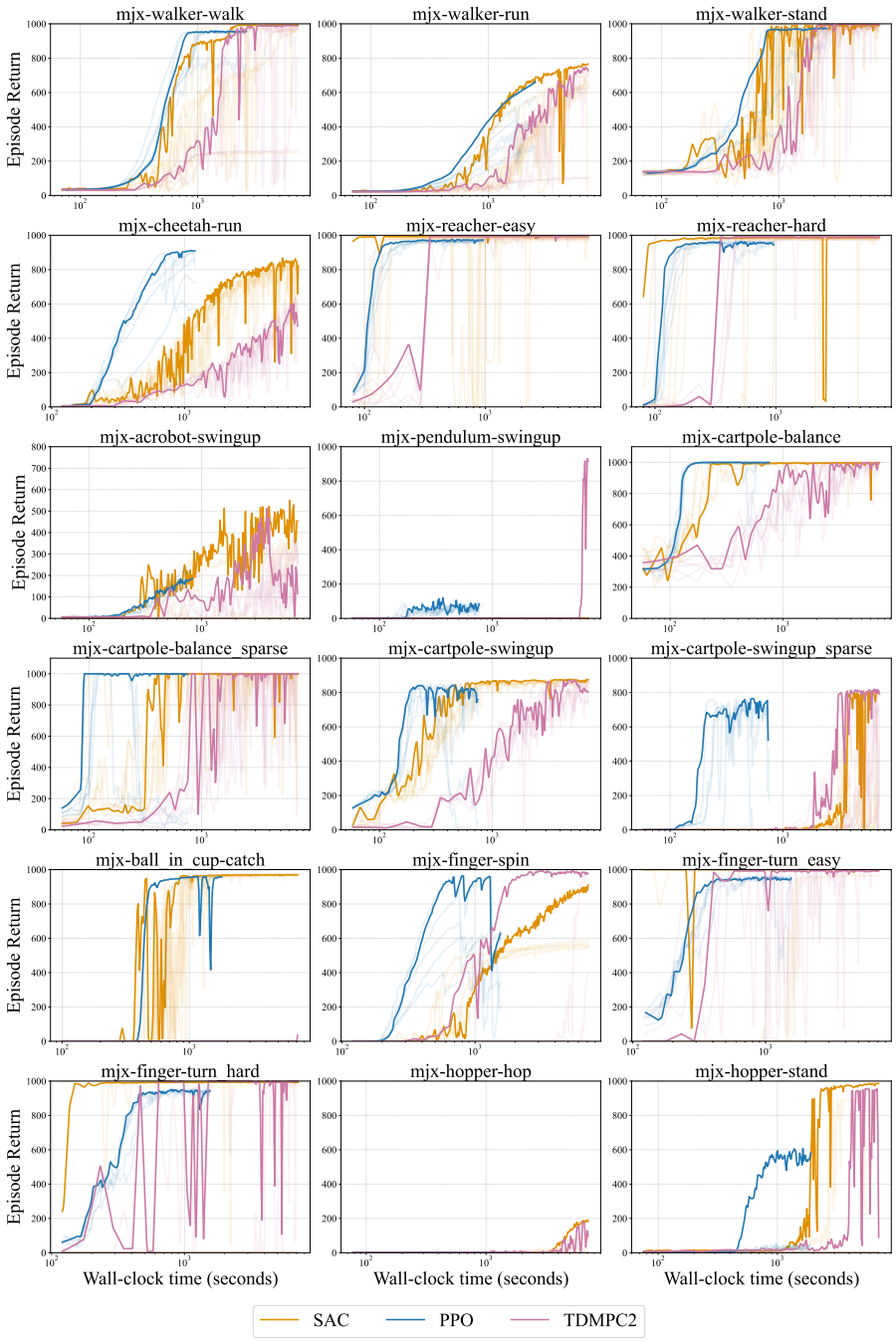

Figure B.1: **Learning curves under wall-clock evaluation.** Training runs as a function of wall-clock time (log-scale); the curves emphasize the best-performing seed, while the shaded regions indicate the individual runs across seeds.

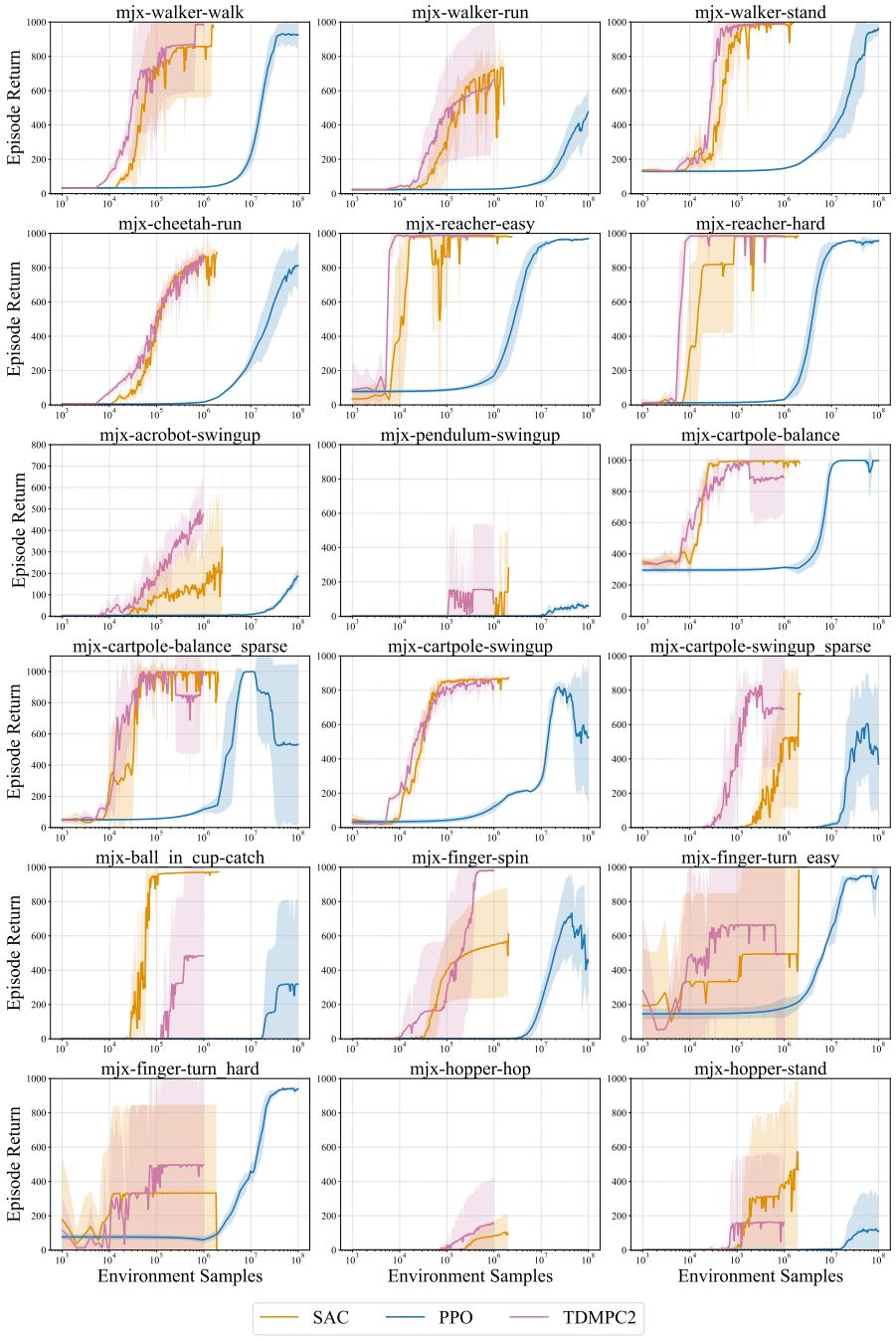

Figure B.2: **Learning curves under interaction-based evaluation.** Training runs as a function of environment interactions (log-scale, up to 40M steps), corresponding to the conventional interaction-based evaluation protocol. Curves show the mean across six seeds, with shaded regions indicating one standard deviation.

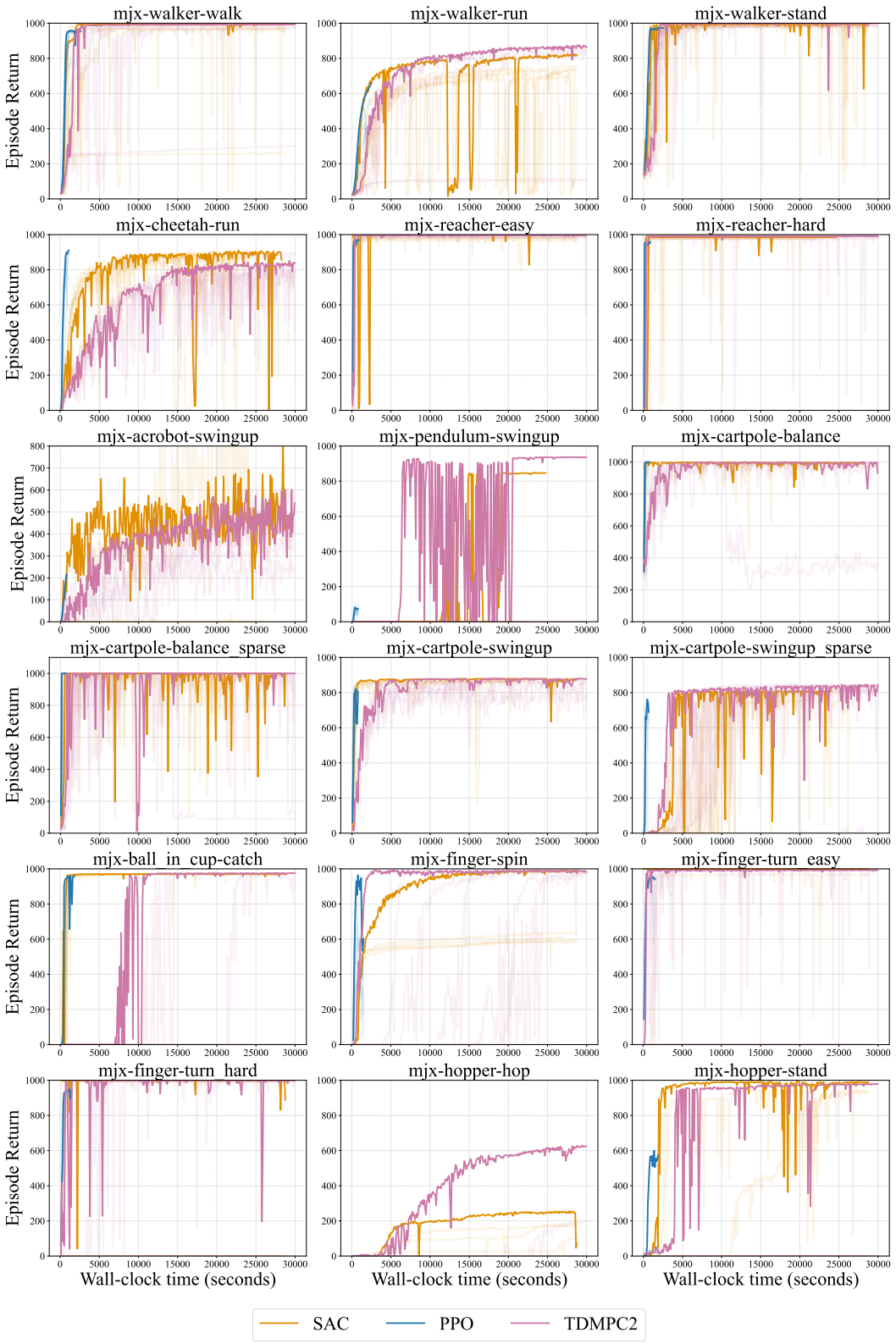

Figure B.3: **Learning curves under wall-clock evaluation.** Training runs as a function of wall-clock time (log-scale, **for an extended period of time**); the curves emphasize the best-performing seed, while the shaded regions indicate the individual runs across seeds.

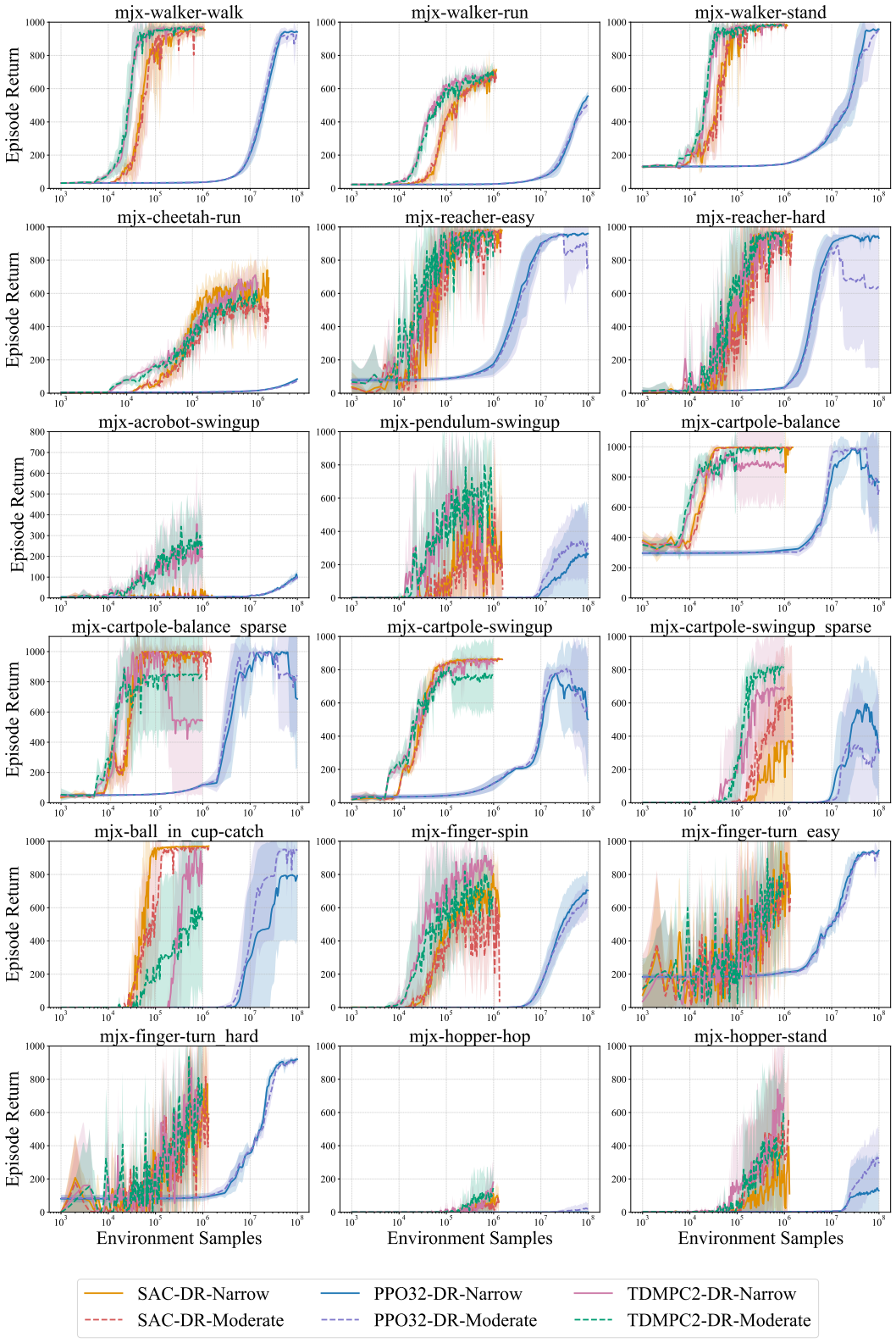

Figure B.4: **Learning curves under interaction-based evaluation with domain randomization (Narrow and Moderate).**

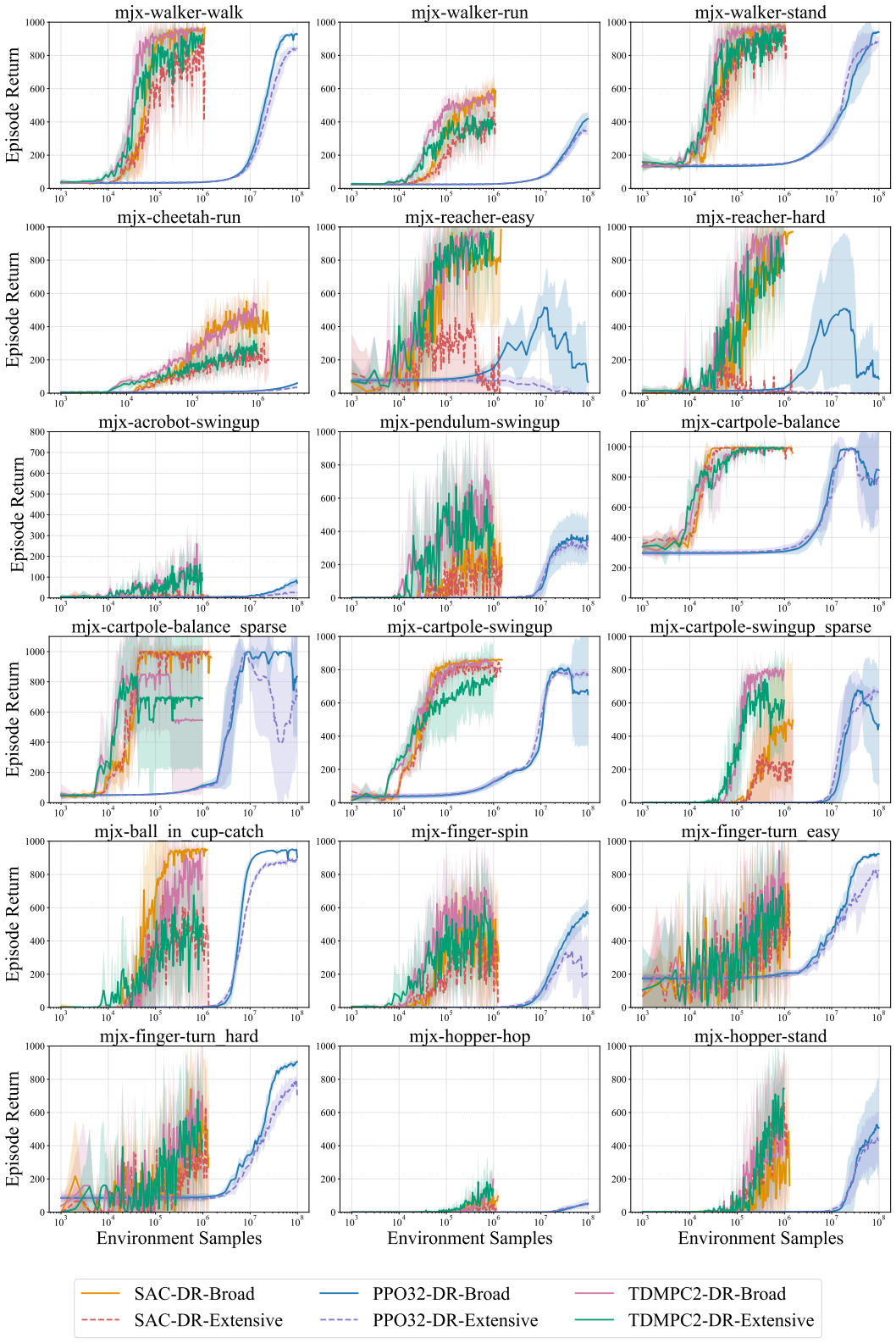

Figure B.5: **Learning curves under interaction-based evaluation with domain randomization (Broad and Extensive).**

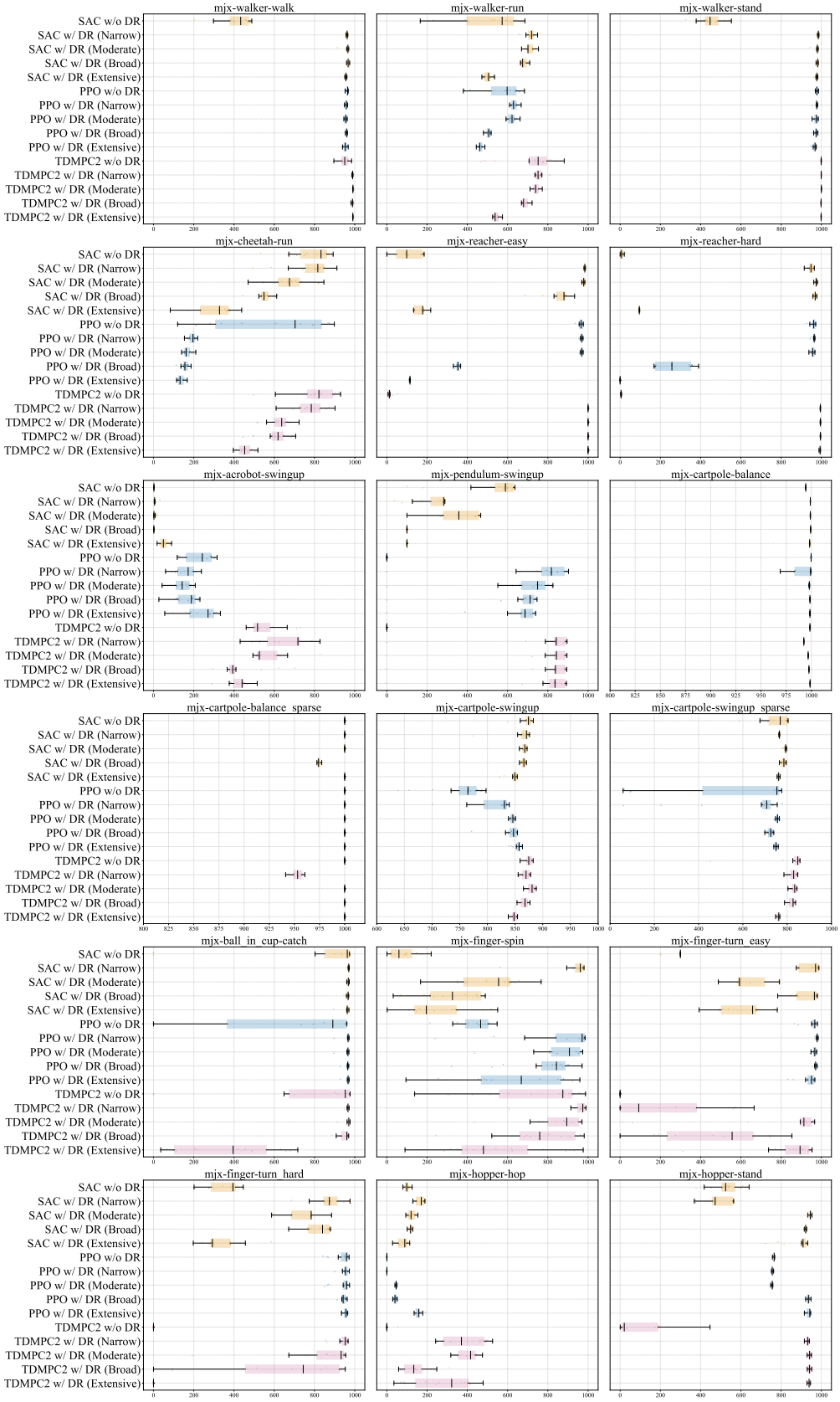

Figure B.6: **Effect of domain randomization regimes on zero-shot transfer performance for the Nominal-Centered evaluation set.**

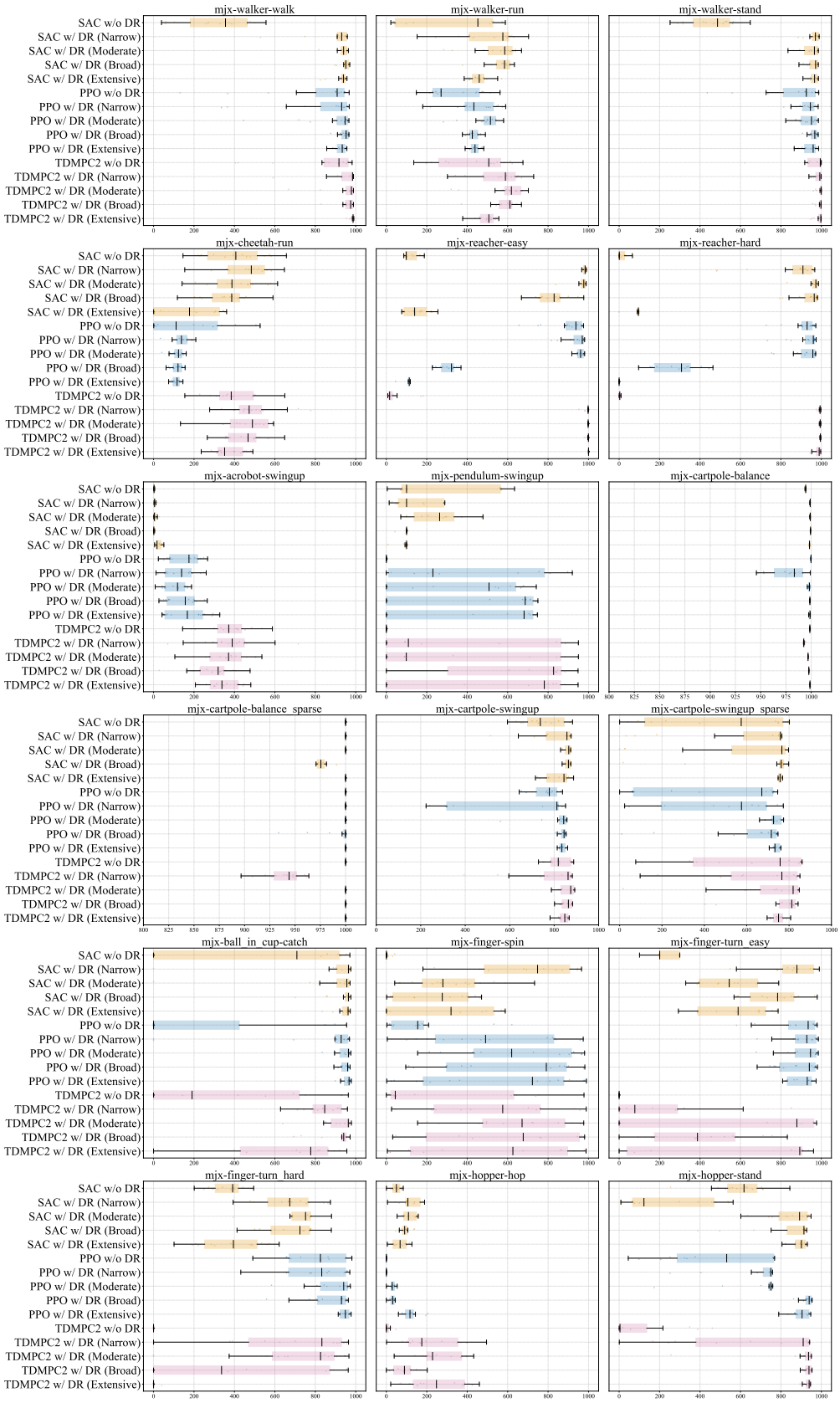

Figure B.7: **Effect of domain randomization regimes on zero-shot transfer performance for the Shifted evaluation set.**