SIMULATION DISTILLATION: PRETRAINING WORLD MODELS IN SIMULATION FOR RAPID REAL-WORLD ADAPTATION

Jacob Levy[1*] **Tyler Westenbroek**[2*] **Kevin Huang**[2] **Fernando Palafox**[1] **Patrick Yin**[2]
Shayegan Omidshafiei[3] **Dong-Ki Kim**[3] **Abhishek Gupta**[2] **David Fridovich-Keil**[1]
[1]University of Texas at Austin [2]University of Washington [3]FieldAI [*]Equal Contribution
https://sim-dist.github.io

Figure 1: Failures of zero-shot sim-to-real policies (left). `SimDist` rapidly overcomes the dynamics gap with minimal real-world interaction, achieving substantial gains on precise manipulation and quadrupedal locomotion with only 15–30 minutes of real data.

ABSTRACT

Simulation-to-real transfer remains a central challenge in robotics, as mismatches between simulated and real-world dynamics often lead to failures. While reinforcement learning offers a principled mechanism for adaptation, existing sim-to-real finetuning methods struggle with exploration and long-horizon credit assignment in the low-data regimes typical of real-world robotics. We introduce `Simulation Distillation` (`SimDist`), a sim-to-real framework that distills structural priors from a simulator into a latent world model and enables rapid real-world adaptation via online planning and supervised dynamics finetuning. By transferring reward and value models directly from simulation, `SimDist` provides dense planning signals from raw perception without requiring value learning during deployment. As a result, real-world adaptation reduces to short-horizon system identification, avoiding long-horizon credit assignment and enabling fast, stable improvement. Across precise manipulation and quadruped locomotion tasks, `SimDist` substantially outperforms prior methods in data efficiency, stability, and final performance.

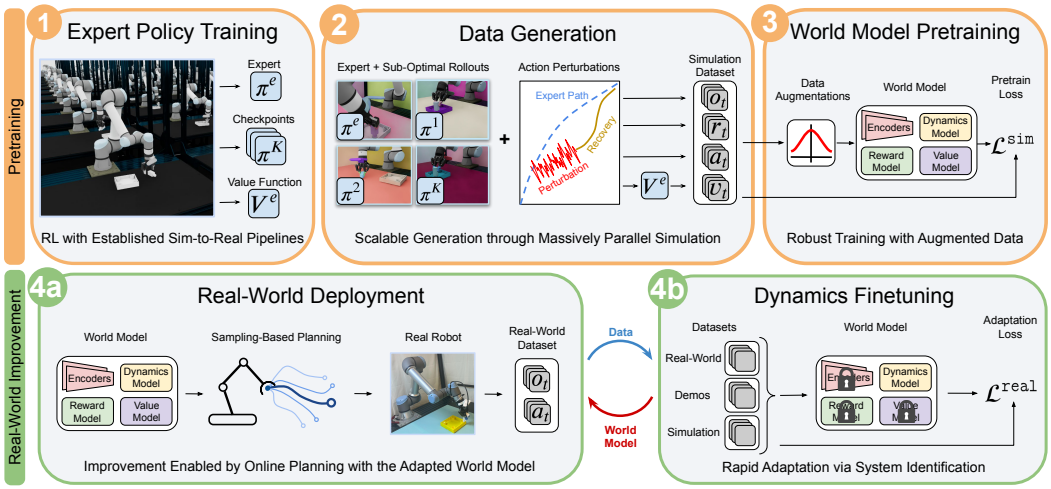

Figure 2: `SimDist` overview. 1) An expert policy, policy checkpoints, and a value function are trained in simulation using privileged state. 2) Large-scale training data are generated by combining expert and sub-optimal policies with contiguous action perturbations, yielding diverse trajectories with dense reward and value supervision. 3) A planning-oriented latent world model is pretrained on this data, learning representations, dynamics, rewards, and values from raw observations. 4a) At deployment, the learned representation and dense reward and value models are transferred to the real robot to enable planning with the latent dynamics. 4b) Real-world data is then used to finetune only the dynamics via supervised learning, with encoders, rewards, and values frozen. Deployment and finetuning are iterated, enabling rapid and stable real-world adaptation.

1 Introduction

Robotic systems must operate in complex, novel environments where success requires rapid deployment-time adaptation. Learning-based approaches (Intelligence et al., 2025a; Levine et al., 2016; Kim et al., 2024) have shown great promise in this setting, but training robots directly in the real world remains challenging due to safety, cost, and time constraints. As a result, simulators have emerged as a powerful tool for scaling robot learning, providing access to large volumes of interaction data and broad coverage over initial conditions and environment parameters (Rudin et al., 2022; Yin et al., 2026; Narang et al., 2022). While this strategy has enabled many recent success stories (Lee et al., 2020; Kumar et al., 2021; Singh et al., 2024), sim-to-real failures remain common due to the inevitable dynamics "gap" between the two domains.

Bootstrapping behavior in simulation and finetuning in the real world with limited on-domain data to overcome dynamics mismatch is an attractive paradigm for robotics (Smith et al., 2022a; Yin et al., 2025; Westenbroek et al., 2022; Wagenmaker et al., 2024). However, using existing end-to-end policy optimization approaches for this purpose often suffers performance collapse when finetuning in new domains (Zhou et al., 2024; Yin et al., 2025) due to catastrophic forgetting of the priors learned during pretraining. This reflects the difficulty of exploration and credit assignment in low-data regimes, particularly when learning from raw perception. Recent off-policy RL methods attempt to improve sample efficiency by aggressively bootstrapping from limited datasets (Ball et al., 2023; Chen et al., 2021b; Hiraoka et al., 2022a), but face brittle trade-offs between data efficiency and training stability. We argue that this behavior stems from a fundamental limitation of end-to-end model-free policy finetuning: by tightly entangling task representations, rewards, and dynamics, these methods force algorithms to relearn the full task structure when transferred to new domains. In this work we ask: *how can we retain physical priors and known task structure throughout finetuning?*

We argue that world models, rather than end-to-end policies, are the appropriate abstraction for sim-to-real transfer. Unlike policies, world models decompose decision making into state representations, reward and value functions, and environment-specific dynamics that are learned jointly but maintained as distinct components. This decomposition enables an approximate separation of *global* and *local* task structure. State encoders, rewards, and values capture global structure—such as relevant objects, semantic roles, and distances to goal configurations—while the dynamics model captures local, action-dependent behavior specific to the environment. Our key insight is that this global structure is largely invariant across simulation and reality for dynamic, high-precision prob-

lems, whereas local dynamics are not. We exploit this asymmetry to focus real-world adaptation on dynamics while preserving global task structure learned in simulation.

Concretely, we introduce `Simulation Distillation` (`SimDist`), a framework that bootstraps a world model in simulation and adapts in the real world through online planning and supervised finetuning of the latent dynamics model. At deployment, we freeze the state encoder, reward model, and value function learned in simulation, and adapt only the dynamics. The planner uses the frozen reward and value models to estimate long-horizon returns, enabling immediate behavioral improvement as dynamics predictions become more accurate, while avoiding difficult bootstrapping problems. By planning over the world model, the robot can reason counterfactually about trajectories it has not directly experienced, mitigating exploration challenges. Altogether, this sidesteps the instability of existing off-policy methods by reducing real-world adaptation to a simple, short-horizon supervised learning problem in latent space.

Enabling this reliable and efficient adaptation requires systematically distilling the structure of the simulator into the world model during pretraining. In particular, the dynamics model must generalize from only a small number of real-world samples and remain accurate along the many sub-optimal trajectories sampled by the planner. To achieve this, we build on standard teacher–student distillation frameworks, leveraging privileged, state-based expert policies in simulation to generate high-quality behavior at scale. Importantly, rather than just collecting expert data, we deliberately perturb actions during data generation. This introduces failures and recovery behaviors, yielding broad state–action coverage that keeps representations, rewards, values, and dynamics in-distribution during real-world planning and finetuning. Relying on privileged experts for pretraining enables massive parallelism and scalable training, in contrast to computationally intensive online model-based RL methods (Hafner et al., 2020; Hansen et al., 2022) whose performance often saturates on challenging robotics problems. We summarize our contributions as follows:

1) We introduce `SimDist`, a world-model–based paradigm for sim-to-real transfer that reduces real-world adaptation to a simple supervised dynamics learning problem, avoiding long-horizon credit assignment and enabling reliable, off-policy, and sample-efficient adaptation. `SimDist` integrates seamlessly with existing student–teacher distillation.

2) We instantiate `SimDist` on two precise, contact-rich manipulation tasks and two quadruped locomotion tasks in the real world, achieving effective autonomous improvement with just 15–30 minutes of real-world data.

3) We show that informative reward and value models trained in simulation can transfer zero-shot to the real world from raw perception, highlighting a promising direction for leveraging privileged simulation supervision even beyond the settings studied in this work.

We provide a more detailed discussion of related work in Appendix A.

2 PRELIMINARIES

Problem Setting. We consider the problem of controlling a robotic system operating in the real world under partial observability and unmodeled dynamics. Specifically, we assume the dynamics are of the form $s_{t+1} \sim p(\cdot|s_t, a_t)$ where $s_t \in \mathcal{S}$ is the underlying state of the system and $a_t \in \mathcal{A}$ is the robot action. The underlying state is not directly observable in the real world as the robot only has access to raw observations $o_t \in \mathcal{O}$. We model the control problem as a partially observable Markov decision process (POMDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{X}, p, r, \gamma)$, with user-specified reward function $r(s_t, a_t, s_{t+1})$ and discount factor $\gamma \in (0, 1]$, and seek to maximize the discounted return $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t, s_{t+1})\right]$. In general, dense informative reward functions are difficult to evaluate directly in the real world, given the difficulty in measuring the underlying state of the system. To render this problem more tractable, we make the following assumption:

Assumption 1. *We assume access to an approximate physics-based simulator $s_{t+1} \sim \hat{p}(\cdot \mid s_t, a_t)$ that provides privileged access to the underlying state s_t.*

Planning-Oriented Latent World Models. We build on common planning-oriented latent world models from works such as Hansen et al. (2022); Hafner et al. (2020). These approaches learn reward and value models that operate directly on raw observations, enabling planning in the real world using the algorithms outlined below. The primary novelty of `SimDist` lies in our systematic

framework for sim-to-real adaptation of this class of world models, introduced in the following section. We structure our world model according to (1), depicted in Fig. 3:

$$\begin{aligned}
&\text{Latent representation:} && z_t = E_\theta(o_t) \\
&\text{History representation:} && h_t = C_\theta(o_{t-H:t-1}, a_{t-H:t-1}) \\
&\text{Latent dynamics:} && \hat{z}_{t+1:t+T} = f_\theta(z_t, a_{t:t+T-1}, h_t) \\
&\text{Reward Prediction:} && \hat{r}_{t:t+T-1} = R_\theta(\hat{z}_{t:t+T}, a_{t:t+T-1}) \\
&\text{Value Prediction:} && \hat{v}_{t+1:t+T} = V_\theta(\hat{z}_{t:t+T}) \\
&\text{Base Policy:} && \hat{a}_{t:t+H} = \pi_\theta(z_t, h_t).
\end{aligned} \tag{1}$$

Here E_θ encodes a latent representation z_t of the underlying state s_t directly from raw observation o_t; h_t is an encoding of history over a window of H timesteps; $\hat{z}_{t+1:t+T}$ is a predicted sequence of T future latent states generated by the learned model f_θ; and $\hat{r}_{t:t+T-1}$ and $\hat{v}_{t+1:t+T}$ are reward and value estimates. We discuss key architecture decisions below.

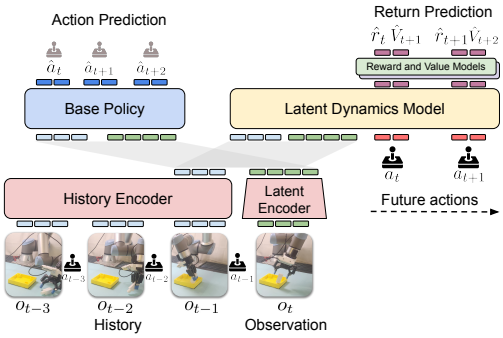

Figure 3: World model architecture. The current observation and an observation–action history are encoded into separate latents that condition a transformer-based dynamics model. Reward, value, and policy heads evaluate predicted trajectories and warm-start sampling-based planning.

Sampling-Based Planning. We control the robot in the real world with Model Predictive Path Integral (MPPI) (Williams et al., 2016) control, a sampling-based Model Predictive Control (MPC) method. At each control step, we sample a batch of candidate future action sequences and evaluate them using the world model (1), extracting the predicted future reward sequence $\hat{r}_{t:t+T-1}$ and terminal value $\hat{v}_{t+T}$, which are combined to compute the trajectory return $\mathcal{R}(a_{t:t+T-1}) = \gamma^T \hat{v}_{t+T} + \sum_{s=t}^{t+T-1} \gamma^{s-t} \hat{r}_s$. MPPI then computes the control action to execute by importance-weighting sampled trajectories according to their predicted returns. Similar to TD-MPC (Hansen et al., 2022), we warm-start sampling by seeding a subset of candidate action sequences with noise-corrupted outputs of $\hat{a}_{t:t+T-1}$ from the base policy π_θ.

3 Simulation Distillation for Efficient Real-World Adaptation

We now introduce `SimDist`, a framework for distilling physical priors and task structure into a world model that enables rapid real-world adaptation with online planning and dynamics adaptation.

3.1 Pretraining on Simulated Data

`SimDist` builds on existing sim-to-real data-generation pipelines that use privileged, state-based expert policies to collect large-scale datasets paired with raw perception. While prior approaches primarily use this data for imitation, planning-based adaptation requires world models that can reason counterfactually about trajectories far beyond the expert and real-world data distributions. In particular, dynamics must generalize from few real-world samples, and reward and value models must reliably discriminate between good and bad trajectories under off-policy action sequences. To meet these requirements, we deliberately inject sub-optimal actions during simulation rollouts, producing a broad training distribution.

Expert Policy Training. We first train a state-based expert policy $\pi^e(s_t)$ with PPO (Schulman et al., 2017). We also save the optimal state-based value function $V^e(s_t)$ and intermediate policy checkpoints $\{\pi^k\}_{k=1}^K$ learned during training.

Generating Diverse Trajectories and Dense Supervision. We generate diverse simulation rollouts (Fig. 2) by alternating between an expert policy π^e and a set of sub-optimal policies $\{\pi^k\}_{k=1}^K$, and by periodically injecting random action perturbations over short temporal windows. This generates diverse failure and recovery behaviors beyond the nominal expert manifold, enabling the world model to reliably produce predictions about counterfactual trajectories. This results in a dataset

$\mathcal{D}_{\texttt{sim}} = \{(o_t, a_t, r_t, v_t)\}_{t=0}^{N}$, where rewards r_t are computed from privileged simulator state and value targets v_t are provided by the expert value function $V^e(s_t)$. Crucially, this data-generation process is massively parallelizable and exploits the full training artifact of the simulator—including expert policies, intermediate checkpoints, and value functions—to produce rich supervision at scale. The full data generation procedure is provided in Appendix B.

World Model Pretraining. We pretrain the world model (1) on $\mathcal{D}_{\text{sim}}$ by applying the following loss to predictions made at each time step t. Specifically, we feed the history $o_{t-H:t}$ and future actions $a_{t:t+T-1}$ into the model, and then penalize the resulting predictions with the following loss:

$$\begin{aligned}\mathcal{L}_t^{\texttt{sim}}(\theta) = \sum_{i=0}^{T} \Bigg(& \underbrace{\|\hat{z}_{t+i+1} - \texttt{sg}(E_\theta(o_{t+i+1}))\|_2^2}_{\text{latent dynamics}} + c_1 \underbrace{(\hat{r}_{t+i} - r_{t+i})^2}_{\text{reward}} + \\ & + c_2 \underbrace{(\hat{v}_{t+i+1} - v_{t+i+1})^2}_{\text{value}} + c_3 \underbrace{\mathbb{1}_e(a_{t+i}) \|\hat{a}_{t+i} - a_{t+i}\|_2^2}_{\text{behavior cloning}} \Bigg),\end{aligned} \tag{2}$$

where $\texttt{sg}$ is the `stop-grad` operator, constants $c_{1:3}$ are determined by normalizing over the range for each target to ensure each loss term is roughly on the same scale, and $\mathbb{1}_e(a_t) = 1$ if a_t came from the uncorrupted expert policy and $\mathbb{1}_e(a_t) = 0$ otherwise. We apply various data augmentations discussed below to prevent overfitting to simulated observations an ensure that the model learns robust, transferable representations. In contrast to typical online MBRL approaches (Hansen et al., 2022; Hafner et al., 2020) which are computationally intensive and attempt to bootstrap behavior from scratch, offload behavior generation to the privileged expert enabling `SimDist` solve a simple, stationary objective over a fixed but vast and diverse simulated dataset.

3.2 Real World Transfer and Efficient Dynamics Adaptation

Our key insight is that global task structure is largely invariant to low-level sim-to-real dynamics gaps. For example, in Peg Insertion (Fig. 1), a meaningful latent state captures the locations of the peg and hole, while the value function encodes distance to the goal and motions leading to successful insertion. This structure persists across the sim-to-real gap, even though the low-level actions required to realize these behaviors differ between domains. Exploiting this decomposition, `SimDist` finetunes only the environment-specific dynamics model while freezing the encoder, reward, and value functions. Planning with frozen reward and value models enables immediate improvement as dynamics predictions become more accurate, without requiring reward or value bootstrapping in the real world. Because the world model can be trained on arbitrary trajectories, `SimDist` naturally supports off-policy learning and can incorporate prior data such as demonstrations.

Dynamics Adaptation Loss. When updating the dynamics model, we modify the pretraining loss for predictions made at each time-step t to simply:

$$\begin{aligned}&\mathcal{L}_t^{\texttt{real}}(\theta) = \textstyle\sum_{i=0}^{T} \|\hat{z}_{t+i+1} - \texttt{sg}(E_\theta(o_{t+i+1}))\|_2^2, \\ &\text{with } C_\theta, E_\theta, R_\theta, V_\theta, \pi_\theta \text{ frozen, } f_\theta \text{ finetunable.}\end{aligned} \tag{3}$$

We do not apply data augmentations when optimizing this objective, as pretraining yields a robust encoder E_θ. Because the encoder is frozen, it provides consistent latent targets $E_\theta(o_t)$ throughout finetuning, avoiding the need to boot strap a representation as in (2).

Iterative Improvement. The overall pipeline for `SimDist` is shown in Fig. 2. `SimDist` autonomously improves in the real-world by repeatedly collecting M on-policy rollouts under the planner, adding this data to the real-world data set $\mathcal{D}_{\texttt{real}}$, then finetuning f_θ to minimize prediction losses as in (3). Notably, because system identification can effectively learn from any real-world trajectories, `SimDist` is a simple off-policy reinforcement learning strategy which can easily incorporate diverse data sources such as demonstrations or play data into $\mathcal{D}_{real}$.

Remark 1. *In contrast to standard MBRL frameworks (Hafner et al., 2020; Hansen et al., 2022), which must jointly bootstrap latent representations, value functions, and policies from scarce in-domain data, `SimDist` offloads these challenging objectives to simulation, where diverse data is cheap and plentiful. As a result, real-world adaptation reduces to finetuning only the dynamics via a simple, iterative supervised learning procedure.*

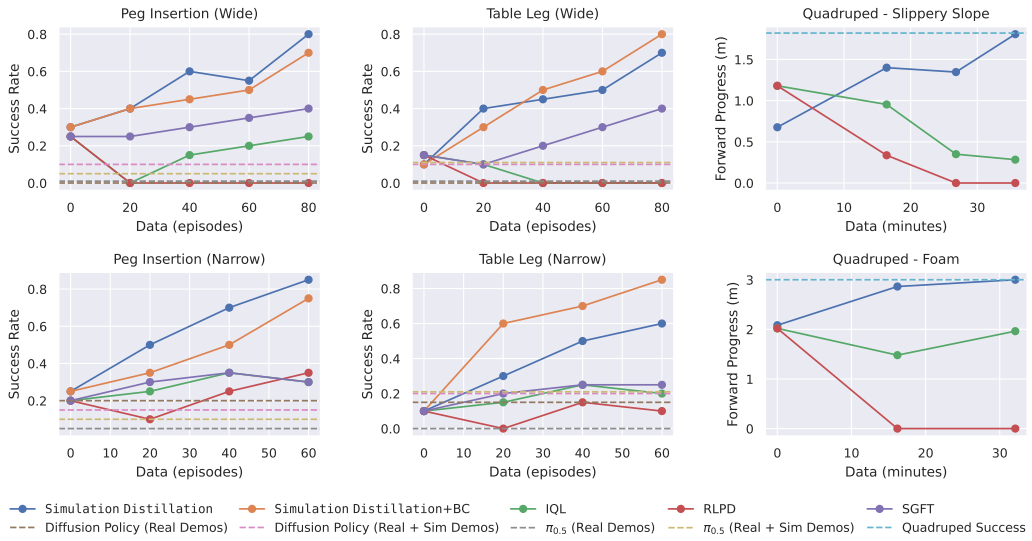

Figure 4: Real-world results. Success rate for two manipulation tasks, computed over 20 trials, and average forward progress for two quadruped locomotion tasks, averaged across all 15 trials (3 speeds, 5 trials each), as a function of real-world finetuning data. `SimDist` exhibits rapid and consistent improvement with limited data. In contrast, direct policy finetuning with the baselines shows limited or no improvement.

3.3 World Model Design Decisions

The following decisions are essential for enabling reliable real-time planning for `SimDist`.

Minimal History Representation. Observation histories $o_{t-H:t}$ can contain high-dimensional inputs such as images that are costly to process. To reduce inference cost, we split observations $o_t = (o_t^p, o_t^e)$ into proprioceptive and exteroceptive components and feed only $(o_{t-H:t}^p, a_{t-H:t}, o_t^e)$ into the history encoder C_θ. Using only the most recent high-dimensional observation substantially reduces planning latency and, empirically, improves training stability by reducing context length.

Chunked Prediction for Planning. Autoregressive world models (Hansen et al., 2022; Hafner et al., 2020; Chen et al., 2022) require sequential unrolling over the planning horizon, limiting parallelism when evaluating many rollouts. Inspired by Xiao et al. (2025), our dynamics model f_θ predicts T future latent states in a single forward pass using a transformer, enabling fully parallel prediction and substantially higher planning throughput.

Sequence-to-Sequence Return Modeling. Prior work often decodes rewards and values with per-timestep MLPs applied independently to each predicted latent state (Hansen et al., 2022; Hafner et al., 2020; Chen et al., 2022). We instead use transformer-based reward and value models that aggregate information over the entire predicted trajectory $\hat{z}_{t:t+T}$, yielding more accurate return estimates and improving planning performance (see Section 4.5).

4 Experiments

We evaluate the ability of `SimDist` to adapt in the real world on the four manipulation and quadruped tasks depicted in Fig. 1 and carefully ablate key design decisions in simulation. Additional details of the experiments can be found in the Appendix. We aim to answer 1) does `SimDist` outperform existing online RL methods and behavior cloning baselines? 2) what factors enable the planner to effectively improve performance with minimal real-world data? 3) which components of the architecture and pretraining procedure are crucial for the performance of `SimDist`?

4.1 Robotic Systems and Tasks

Manipulation System and Tasks. Our manipulation experiments use a UR5e robot with six-dimensional relative end-effector pose actions. Observations include joint states and three 224×224 RGB images from wrist-mounted, overhead, and side-view cameras. Images are encoded with a

pretrained ResNet-18, fused with proprioception, and mapped to a 64-dimensional latent state z. Training uses $100K$ trajectories, approximately 36% of which are optimal expert (see Appendix C for details). We use history and prediction horizons $H = T = 5$ and control the robot at $5\,\mathrm{Hz}$. Expert policies are trained following Yin et al. (2026). We evaluate two precise assembly tasks:

1) Peg Insertion. We construct a peg insertion task similar to Narang et al. (2022). This task requires picking the peg, aligning it with the hole, and insertion. We consider two difficulties with initial conditions drawn from a Narrow ($2\,\mathrm{cm} \times 2\,\mathrm{cm}$) or Wide ($35\,\mathrm{cm} \times 35\,\mathrm{cm}$) grid.

2) Table Leg. We use the assets from Heo et al. (2023) to generate a task where a table leg must be picked, aligned, and threaded into a hole on a table. We consider initial conditions drawn from Narrow ($2\,\mathrm{cm} \times 2\,\mathrm{cm}$) or Wide ($35\,\mathrm{cm} \times 35\,\mathrm{cm}$) grids.

Quadruped System and Tasks. We conduct quadrupedal experiments on a Unitree Go2 platform, with actions given as position targets for the 12 joints. Observations include proprioceptive signals and a local terrain height map, which is encoded by E_θ using a CNN and fused with low-dimensional observations via an MLP to produce the latent state z. The policy π_θ, reward R_θ, and value V_θ are additionally conditioned on desired forward, lateral, and base velocities. Pretraining uses 100M simulated trajectories, of which 55.7% come from an expert policy trained in IsaacLab (Mittal et al., 2025). We use history and prediction horizons $H = T = 25$ and plan at $50\,\mathrm{Hz}$ on a laptop with an RTX 4090M. See Appendix D for more details. We consider the following tasks:

1) Slippery Slope. The robot traverses over two panels inclined at 3.0° and 5.7°. The panels are covered with PTFE (Teflon), and the robot's feet are wrapped with thermoplastic, creating extremely low-friction contacts. A trial is successful if the robot traverses $1.82\,\mathrm{m}$, clearing both panels. We conduct five trials per commanded forward speed at 0.1, 0.3, and $0.5\,\mathrm{m\,s^{-1}}$.

2) Foam. The robot traverses two $5\,\mathrm{cm}$ thick overlapping memory foam pads whose compliant dynamics are not modeled in simulation. A trial is successful if the robot traverses $3.00\,\mathrm{m}$ to clear the foam. We conduct five trials per commanded forward speed at 0.2, 0.7, and $1.2\,\mathrm{m\,s^{-1}}$.

4.2 Real-World Learning Methods.

We evaluate `SimDist` against state-of-the-art real-world RL methods and additionally behavior cloning baselines on manipulation tasks. All RL method use the same encoders as `SimDist`, but an MLP policy head (no action chunking).

Manipulation. For each task, we collect 20 real-world demonstrations via teleoperation. We evaluate two variants of `SimDist`: (i) `SimDist`, which adapts only the latent dynamics model using on-policy rollouts, and (ii) `SimDist`+BC, which additionally finetunes the base policy π_θ using expert action labels. In both cases, the dynamics model is updated after every 20 real-world episodes. We compare against model-free RL baselines RLPD (Ball et al., 2023) and IQL (Kostrikov et al., 2022), trained from sparse rewards to reflect common manipulation settings where dense rewards are difficult to specify; these baselines update after every episode. We also include SGFT-SAC (Yin et al., 2025), a model-free method that transfers a simulated value function, isolating the benefit of full world-model adaptation beyond value transfer alone. Finally, we compare against behavior cloning baselines—Diffusion Policy (Chi et al., 2024) and $\pi_{0.5}$ (Intelligence et al., 2025b)—trained on real-world demonstrations, as well as variants trained on both real-world demonstrations and the simulated dataset used by `SimDist`. All manipulation methods except `SimDist` without demonstrations are given access to the same 20 real-world demonstrations.

Quadruped. For quadruped locomotion, we compare `SimDist` against the off-policy algorithm RLPD (Ball et al., 2023) and the offline-to-online finetuning method IQL (Kostrikov et al., 2022), whose value function is pretrained in simulation. All methods use the same learned reward model from `SimDist`, allowing us to isolate the effect of different adaptation strategies.

4.3 Real World Improvement Results

Figure 4 summarizes our results. Across all tasks, `SimDist` consistently outperforms prior approaches, achieving substantially higher success rates with far greater sample efficiency than online RL baselines, while autonomously improving well beyond the performance of behavior cloning methods. Across the board `SimDist` typically reaches scores around $2\times$ higher than any baseline.

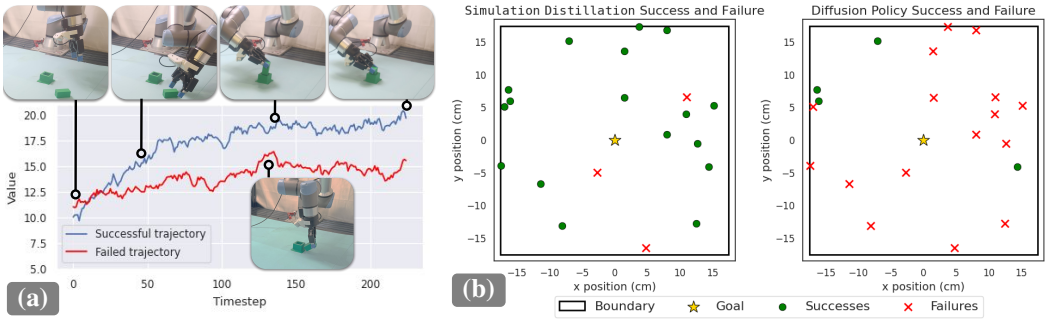

Figure 5: **(a)** Value predictions from `SimDist` along successful and failed real-world Peg trajectories starting from the same initial condition. The predicted values track task progress and clearly distinguish successful from failure. **(b)** Scatter-Plots showing successful and failed attempts at solving the Peg Wide task for Diffusion Policy (Right) and the final trained policy for `SimDist`(Left). The broad coverage of pretraining data for `SimDist` enables efficiently learning policies which are far more robust than baselines.

Standard RL finetuning methods frequently exhibit catastrophic forgetting, with performance collapsing during adaptation, whereas `SimDist` makes steady, monotonic progress throughout training. SGFT avoids catastrophic collapse by transferring value functions from simulation, but remains significantly more sample inefficient than `SimDist`, highlighting the importance of adapting a full world model. Interestingly, demonstrations provide a larger boost to `SimDist` on the Leg task, where the precise screwing motion was difficult to transfer. This highlights how `SimDist` can naturally leverage demonstrations when available.

Notably, the performance gap between `SimDist` and all baselines widens on more challenging tasks, particularly when comparing narrow versus wide ranges of initial conditions. This trend underscores the benefit of broad simulation pretraining, which enables `SimDist` to generalize across diverse initial states and adapt reliably in difficult real-world regimes. This effect is illustrated in Fig. 5b, which visualizes successful and failed initial conditions for `SimDist` and Diffusion Policy on Peg Hard, revealing the substantially greater robustness of policies learned by `SimDist`.

4.4 Analyzing Real-World Predictions and Effects on Planning

The planner used by `SimDist` can only improve behavior if it can reliably distinguishes trajectories with high and low returns. This requires both accurate dynamics predictions and successful transfer of reward and value models. We examine value transfer in Fig. 5a, which plots predicted values over time for successful and failed trajectories, both collected with teleoperation from the same initial condition. For the successful rollout, predicted value increases consistently over time, while remaining lower for the failed trajectory. Thus the transfered encoder E_θ and value function V_θ can **reliably discriminate between successful and failed trajectories**.

We next evaluate dynamics prediction accuracy on the Quadruped Slippery Slope task in Fig. 6a. We roll a held-out real-world trajectory through the world model and compute the latent dynamics loss (3) at each timestep, yielding an average loss of 0.076 for the pretrained model and 0.019 after finetuning. To visualize the impact of this improvement, we decode predicted latent states into predicted front-left foot positions in Fig. 6c. Before finetuning, the model incorrectly predicts stable contact on the PTFE surface and fails to anticipate slip; after finetuning, it accurately predicts future slippage, closely matching the real trajectory (Fig. 6b). This illustrates how broad simulation pretraining enables the dynamics model to generalize beyond the real-world training set.

Finally, improved dynamics predictions directly reshape planning behavior. As shown in Fig. 6d, trajectory samples generated with the finetuned model reflect the altered contact dynamics and lead the planner to select plans that account for real-world slip. In contrast, plans derived from the pretrained model are qualitatively inconsistent with the true dynamics. Together, these results show that finetuning corrects latent dynamics errors in a way that meaningfully changes the planner's trajectory distribution, explaining the rapid real-world performance gains observed in Fig. 4.

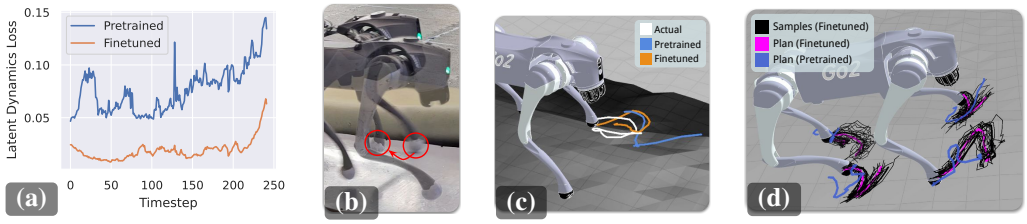

Figure 6: **(a)** Finetuning drastically lowers dynamics prediction loss during a quadruped PTFE trial. **(b)** Overlaid frames showing the front left foot slipping during a PTFE trial. **(c)** Foot-trajectory predictions from the world model at the same instant: the finetuned model correctly anticipates future slippage, while the pretrained model fails to do so. **(d)** Visualization of sampling-based planning. Candidate action sequences are evaluated with the world model to produce planned foot trajectories for each leg. Sampled trajectories are shown from the finetuned dynamics model, along with the resulting optimal plans under the finetuned and pretrained models. The finetuned model produces plans that account for real world dynamics mismatch, while the pretrained model generates qualitatively different trajectories.

4.5 Ablating Key Design Decisions

We ablate key components of `SimDist` by modifying world-model pretraining and evaluating the resulting closed-loop planning policies in simulation. Results are summarized in Table 1, where we evaluate policies under a wide range domain randomizations for both manipulation and quadrupeds.

Scaling Simulation Data. We study dataset scale by reducing pretraining data to 10% and 50% of the full dataset (100k for manipulation, 100M for quadrupeds). Performance drops sharply for both systems, indicating that large-scale pretraining is critical for reliable planning.

Data Diversity. We next evaluate the role of data diversity by comparing pretraining on expert-only trajectories to our mixed datasets with equal data volume. Expert-only data leads to a substantial performance drop, highlighting the importance of diverse, perturbed rollouts for learning dynamics and value estimates that remain accurate under off-policy planning.

Table 1: Ablation results in simulation, reporting success rates for manipulation tasks and average state-based reward per episode for the quadruped.

	Peg	Table	Quad
`SimDist`	0.90	0.85	22.78
50% data	0.72	0.61	22.73
10% data	0.06	0.02	19.38
Expert only	0.10	0.05	16.68
MLP R+V	0.82	0.60	19.47
Raw recon	0.32	0.21	23.34

Reward and Value Transformers. We ablate the use of sequence-to-sequence transformers for reward and value prediction, replacing them with per-timestep MLP decoders as in Hafner et al. (2020); Hansen et al. (2022). This degrades performance across tasks. We attribute this to the inability of per-step models to capture trajectory-level structure, which is essential for accurately evaluating candidate action sequences during planning.

Reconstruction Loss. Finally, we study the effect of adding an observation reconstruction loss to the dynamics objective, a common design choice in MBRL (Hafner et al., 2020; Bruce et al., 2024). While reconstruction slightly improves quadruped performance, it provides no qualitative benefit and leads to a steep performance drop for manipulation. We hypothesize that reconstructing high-dimensional visual observations encourages overfitting to task-irrelevant details, which interferes with learning compact latent dynamics suitable for planning.

5 Conclusion

We introduced `SimDist`, a framework for sim-to-real transfer and real-world adaptation that exploits the modular structure of world models to directly target the dynamics gap between simulation and reality. Across two precise manipulation tasks and two quadruped locomotion tasks, `SimDist` significantly outperforms prior approaches in both sample efficiency and robustness, achieving consistent improvement where standard RL finetuning methods often collapse. While our experiments focus on single-task settings with high-fidelity simulators, future work will scale `SimDist` to multi-task world models and broader domains. More broadly, freezing representations and task structure while adapting only dynamics offers a general and effective paradigm for efficient adaptation beyond sim-to-real transfer.

REFERENCES

Karl Johan Åström. Adaptive control. In *Mathematical System Theory: The Influence of RE Kalman*, pages 437–450. Springer, 1995.

Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pages 1577–1594. PMLR, 2023.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

Chang Chen, Yi-Fu Wu, Jaesik Yoon, and Sungjin Ahn. Transdreamer: Reinforcement learning with transformer world models. *arXiv preprint arXiv:2202.09481*, 2022.

Xinyue Chen, Che Wang, Zijian Zhou, and Keith W. Ross. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=AY8zfZm0tDd.

Xinyue Chen, Che Wang, Zijian Zhou, and Keith W Ross. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021b.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2024.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

Jared Di Carlo, Patrick M. Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, 2018. doi: 10.1109/IROS.2018.8594448.

Yunhai Feng, Nicklas Hansen, Ziyan Xiong, Chandramouli Rajagopalan, and Xiaolong Wang. Fine-tuning offline world models in the real world, 2023. URL https://arxiv.org/abs/2310.16029.

Tuomas Haarnoja, Sehoon Ha, Aurick Zhou, Jie Tan, George Tucker, and Sergey Levine. Learning to walk via deep reinforcement learning. *arXiv preprint arXiv:1812.11103*, 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

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

Minho Heo, Youngwoon Lee, Doohyun Lee, and Joseph J. Lim. Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation. In *Robotics: Science and Systems*, 2023.

Takuya Hiraoka, Takahisa Imagawa, Taisei Hashimoto, Takashi Onishi, and Yoshimasa Tsuruoka. Dropout q-functions for doubly efficient reinforcement learning. In *International Conference on Learning Representations*, 2022a.

Takuya Hiraoka, Takahisa Imagawa, Taisei Hashimoto, Takashi Onishi, and Yoshimasa Tsuruoka. Dropout q-functions for doubly efficient reinforcement learning. In *International Conference on Learning Representations*, 2022b. URL https://openreview.net/forum?id=xCVJMsPv3RT.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. Pi*0.6: a vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025a.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. pi0.5: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025b.

Petros A Ioannou and Jing Sun. *Robust adaptive control*, volume 1. PTR Prentice-Hall Upper Saddle River, NJ, 1996.

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, et al. Dreamgen: Unlocking generalization in robot learning through video world models. *arXiv preprint arXiv:2505.12705*, 2025.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization, 2021. URL https://arxiv.org/abs/1906.08253.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. Scalable deep reinforcement learning for vision-based robotic manipulation. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 651–673. PMLR, 29–31 Oct 2018. URL https://proceedings.mlr.press/v87/kalashnikov18a.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning, 2020. URL https://arxiv.org/abs/2006.04779.

JB Lanier, Kyungmin Kim, Armin Karamzade, Yifei Liu, Ankita Sinha, Kat He, Davide Corsi, and Roy Fox. Adapting world models with latent-state dynamics residuals, 2025. URL https://arxiv.org/abs/2504.02252.

Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47):eabc5986, 2020.

Kun Lei, Huanyu Li, Dongjie Yu, Zhenyu Wei, Lingxiao Guo, Zhennan Jiang, Ziyu Wang, Shiyu Liang, and Huazhe Xu. Rl-100: Performant robotic manipulation with real-world reinforcement learning. *arXiv preprint arXiv:2510.14830*, 2025.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016. URL http://jmlr.org/papers/v17/15-522.html.

Jacob Levy, Tyler Westenbroek, and David Fridovich-Keil. Learning to walk from three minutes of real-world data with semi-structured dynamics models. In *Conference on Robot Learning*, pages 2061–2079. PMLR, 2025.

Chenhao Li, Andreas Krause, and Marco Hutter. Offline robotic world model: Learning robotic policies without a physics simulator. *arXiv preprint arXiv:2504.16680*, 2025a.

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. *arXiv preprint arXiv:2503.00200*, 2025b.

Jianlan Luo, Zheyuan Hu, Charles Xu, You Liang Tan, Jacob Berg, Archit Sharma, Stefan Schaal, Chelsea Finn, Abhishek Gupta, and Sergey Levine. Serl: A software suite for sample-efficient robotic reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16961–16969. IEEE, 2024.

Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, Lukasz Wawrzyniak, Milad Rakhsha, Alain Denzler, Eric Heiden, Ales Borovicka, Ossama Ahmed, Iretiayo Akinola, Abrar Anwar, Mark T. Carlson, Ji Yuan Feng, Animesh Garg, Renato Gasoto, Lionel Gulich, Yijie Guo, M. Gussert, Alex Hansen, Mihir Kulkarni, Chenran Li, Wei Liu, Viktor Makoviychuk, Grzegorz Malczyk, Hammad Mazhar, Masoud Moghani, Adithyavairavan Murali, Michael Noseworthy, Alexander Poddubny, Nathan Ratliff, Welf Rehberg, Clemens Schwarke, Ritvik Singh, James Latham Smith, Bingjie Tang, Ruchik Thaker, Matthew Trepte, Karl Van Wyk, Fangzhou Yu, Alex Millane, Vikram Ramasamy, Remo Steiner, Sangeeta Subramanian, Clemens Volk, CY Chen, Neel Jawale, Ashwin Varghese Kuruttukulam, Michael A. Lin, Ajay Mandlekar, Karsten Patzwaldt, John Welsh, Huihua Zhao, Fatima Anes, Jean-Francois Lafleche, Nicolas Moënne-Loccoz, Soowan Park, Rob Stepinski, Dirk Van Gelder, Chris Amevor, Jan Carius, Jumyung Chang, Anka He Chen, Pablo de Heras Ciechomski, Gilles Daviet, Mohammad Mohajerani, Julia von Muralt, Viktor Reutskyy, Michael Sauter, Simon Schirm, Eric L. Shi, Pierre Terdiman, Kenny Vilella, Tobias Widmer, Gordon Yeoman, Tiffany Chen, Sergey Grizan, Cathy Li, Lotus Li, Connor Smith, Rafael Wiltz, Kostas Alexis, Yan Chang, David Chu, Linxi "Jim" Fan, Farbod Farshidian, Ankur Handa, Spencer Huang, Marco Hutter, Yashraj Narang, Soha Pouya, Shiwei Sheng, Yuke Zhu, Miles Macklin, Adam Moravanszky, Philipp Reist, Yunrong Guo, David Hoeller, and Gavriel State. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025. URL `https://arxiv.org/abs/2511.04831`.

Manfred Morari and Jay H Lee. Model predictive control: past, present and future. *Computers & chemical engineering*, 23(4-5):667–682, 1999.

Yashraj Narang, Kier Storey, Iretiayo Akinola, Miles Macklin, Philipp Reist, Lukasz Wawrzyniak, Yunrong Guo, Adam Moravanszky, Gavriel State, Michelle Lu, et al. Factory: Fast contact for robotic assembly. *arXiv preprint arXiv:2205.03532*, 2022.

Jack Parker-Holder, Philip Ball, Jake Bruce, Vibhavari Dasagi, Kristian Holsheimer, Christos Kaplanis, Alexandre Moufarek, Guy Scully, Jeremy Shar, Jimmy Shi, Stephen Spencer, Jessica Yung, Michael Dennis, Sultan Kenjeyev, Shangbang Long, Vlad Mnih, Harris Chan, Maxime Gazeau, Bonnie Li, Fabio Pardo, Luyu Wang, Lei Zhang, Frederic Besse, Tim Harley, Anna Mitenkova, Jane Wang, Jeff Clune, Demis Hassabis, Raia Hadsell, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 2: A large-scale foundation world model. 2024. URL `https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/`.

Samuel Pfrommer, Mathew Halm, and Michael Posa. Contactnets: Learning discontinuous contact dynamics with smooth, implicit representations. In *Conference on Robot Learning*, pages 2279–2291. PMLR, 2021.

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*, pages 91–100. PMLR, 2022.

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

Ritvik Singh, Arthur Allshire, Ankur Handa, Nathan Ratliff, and Karl Van Wyk. Dextrah-rgb: Visuomotor policies to grasp anything with dexterous hands. *arXiv preprint arXiv:2412.01791*, 2024.

Jean-Jacques E Slotine and Weiping Li. On the adaptive control of robot manipulators. *The international journal of robotics research*, 6(3):49–59, 1987.

Laura Smith, J Chase Kew, Xue Bin Peng, Sehoon Ha, Jie Tan, and Sergey Levine. Legged robots that keep on learning: Fine-tuning locomotion policies in the real world. In *2022 international conference on robotics and automation (ICRA)*, pages 1593–1599. IEEE, 2022a.

Laura Smith, Ilya Kostrikov, and Sergey Levine. A walk in the park: Learning to walk in 20 minutes with model-free reinforcement learning. *arXiv preprint arXiv:2208.07860*, 2022b.

Laura Smith, Yunhao Cao, and Sergey Levine. Grow your limits: Continuous improvement with real-world rl for robotic locomotion. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10829–10836. IEEE, 2024.

Andrew Wagenmaker, Kevin Huang, Liyiming Ke, Kevin Jamieson, and Abhishek Gupta. Overcoming the sim-to-real gap: Leveraging simulation to learn to explore for real-world rl. *Advances in Neural Information Processing Systems*, 37:78715–78765, 2024.

Tyler Westenbroek, Fernando Castaneda, Ayush Agrawal, Shankar Sastry, and Koushil Sreenath. Lyapunov design for robust and efficient robotic reinforcement learning, 2022. URL `https://arxiv.org/abs/2208.06721`.

Tyler Westenbroek, Jacob Levy, and David Fridovich-Keil. Enabling efficient, reliable real-world reinforcement learning with approximate physics-based models. In *Conference on Robot Learning*, pages 2478–2497. PMLR, 2023.

Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1433–1440. IEEE, 2016.

Wenli Xiao, Haoru Xue, Tony Tao, Dvij Kalaria, John M Dolan, and Guanya Shi. Anycar to anywhere: Learning universal dynamics model for agile and adaptive mobility. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8819–8825. IEEE, 2025.

Amber Xie, Oleh Rybkin, Dorsa Sadigh, and Chelsea Finn. Latent diffusion planning for imitation learning. *arXiv preprint arXiv:2504.16925*, 2025.

Jie Xu, Eric Heiden, Iretiayo Akinola, Dieter Fox, Miles Macklin, and Yashraj Narang. Neural robot dynamics. *arXiv preprint arXiv:2508.15755*, 2025.

Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 1(2):6, 2023.

Patrick Yin, Tyler Westenbroek, Simran Bagaria, Kevin Huang, Ching-an Cheng, Andrey Kobolov, and Abhishek Gupta. Rapidly adapting policies to the real world via simulation-guided fine-tuning. *arXiv preprint arXiv:2502.02705*, 2025.

Patrick Yin, Tyler Westenbroek, Zhengyu Zhang, Ignacio Dagnino, Eeshani Shilamkar, Numfor Mbiziwo-Tiapo, Simran Bagaria, Xinlei Liu, Galen Mullins, Andrey Kolobov, and Abhishek Gupta. Emergent dexterity via diverse resets and large-scale reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL `https://openreview.net/forum?id=nAO9LcV7nE`.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.

Jiahui Zhang, Yusen Luo, Abrar Anwar, Sumedh Anand Sontakke, Joseph J Lim, Jesse Thomason, Erdem Biyik, and Jesse Zhang. Rewind: Language-guided rewards teach robot policies without new demonstrations, 2025. URL `https://arxiv.org/abs/2505.10911`.

Zhiyuan Zhou, Andy Peng, Qiyang Li, Sergey Levine, and Aviral Kumar. Efficient online reinforcement learning fine-tuning need not retain offline data. *arXiv preprint arXiv:2412.07762*, 2024.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. *arXiv preprint arXiv:2504.02792*, 2025.

A Related Work

Real-World Reinforcement Learning. A growing body of work studies reinforcement learning on real-world robotic systems (Luo et al., 2024; Levy et al., 2025; Yin et al., 2025; Smith et al., 2022b; Westenbroek et al., 2022; 2023; Levine et al., 2016; Kalashnikov et al., 2018; Haarnoja et al., 2018b; Lanier et al., 2025). However, both model-free and model-based approaches remain challenging to apply reliably in low-data regimes and typically require sophisticated regularization. Model-free methods aggressively reuse off-policy data and rely on frequent critic updates (Ball et al., 2023; Hiraoka et al., 2022a; Luo et al., 2024; Chen et al., 2021b), often leading to value overestimation and unstable learning. Prior work mitigates these issues using conservative value estimation (Kumar et al., 2020), policy constraints (Lei et al., 2025; Zhang et al., 2025; Smith et al., 2024), or critic ensembles (Chen et al., 2021a; Haarnoja et al., 2018a; Hiraoka et al., 2022b). Model-based methods instead learn dynamics and reward models (Hafner et al., 2020; Hansen et al., 2022) to reason about unseen trajectories, but must carefully avoid exploiting model inaccuracies. Common strategies include uncertainty-aware dynamics (Chua et al., 2018; Levy et al., 2025; Janner et al., 2021) and penalizing out-of-distribution predictions (Yu et al., 2020; 2021; Feng et al., 2023). Rather than constraining learning, we show that bootstrapping a world model in simulation enables generalization beyond the real-world dataset.

Adaptation with Physics-based Models. Many lines of work leverage approximate physics-based models for adaptation and control, but typically rely on simplified, low-dimensional state representations that are brittle in partially observed, contact-rich settings. Classical adaptive control (Åström, 1995; Slotine and Li, 1987; Ioannou and Sun, 1996) and model-predictive control (Morari and Lee, 1999; Di Carlo et al., 2018) approaches use highly simplified dynamics, abstracting away complex contacts and interactions. Neural physics engines (Xu et al., 2025; Pfrommer et al., 2021) combine structured system identification with residual learning to adapt high-fidelity simulators using real-world data, but often assume access to object poses, contact labels, or reliable state estimates that degrade under partial observability. Closely related work learns world models from mixtures of simulated and real data (Li et al., 2025a; Xiao et al., 2025) or transfers value functions from simulation to guide real-world learning (Westenbroek et al., 2022; Yin et al., 2025), but similarly depends on low-dimensional state observations. We instead propose distilling simulator structure from raw perception to enable in-the-wild adaptation.

Generative World Models for Robotics. Recent work trains large world models on internet-scale video to learn broad physical priors for robotics (Yang et al., 2023; Bruce et al., 2024; Parker-Holder et al., 2024), sometimes augmented with simulation data (Yang et al., 2023). Translating video predictions into executable robot actions typically requires expert demonstrations, either by planning in video space and using inverse models to recover actions (Xie et al., 2025; Jang et al., 2025), or by combining video prediction with behavior cloning (Zhu et al., 2025; Li et al., 2025b; Cheang et al., 2024). While effective for reproducing demonstrated behaviors, these approaches remain fundamentally constrained by the real-world action distribution. In contrast, `SimDist` learns world models directly over a wide range low-level robot actions, enabling it to directly improve behaviors beyond the real-world data.

B Diverse Data Generation Details

Algorithm 1 details the data generation process. For each environment, we sample an action-noise covariance and contiguous noise intervals, during which Gaussian noise is added to policy actions. This induces temporally persistent deviations from nominal behavior, rather than isolated perturbations, encouraging exploration of dynamically consistent off-nominal states. In addition, each environment is assigned a randomly selected actor checkpoint from $\{\pi^k\}_{k=1}^K$ at reset, with earlier checkpoints inducing suboptimal or partially trained behaviors. During rollouts, we query the optimal state-based value function V^e at each visited state to generate value targets v_t for distilling an approximate optimal value function (Section 3.1). We also record an expert action flag e_t that distinguishes the expert actions from the expert policy π^e versus noised or earlier-checkpoint actions to support behavior cloning (Section 3.1).

Algorithm 1 Diverse Data Generation

Input: Expert policy π^e, policy checkpoints $\{\pi^k\}_{k=1}^K$, optimal value function V^e
Output: Simulation dataset $\mathcal{D}_{\texttt{sim}}$
for environment $j \in \{1 \dots N_E\}$ **do**
  Sample action noise variance: $\Sigma_j = \text{diag}(\sigma)$ with $\sigma_i \sim \mathcal{U}[\sigma_i^{\texttt{max}}, \sigma_i^{\texttt{min}}]; i \in \{1, \dots, \dim(\mathcal{A})\}$
for $t = 0$ to number of steps N_T **do**
  for environment $j \in \{1 \dots N_E\}$ **do**
    if s_t is terminal **then**
      Reset environment
      Sample noise intervals $\mathcal{T}_{\texttt{noise}}$
      Randomly select a policy π^k from $\{\pi^k\}_{k=1}^K$ and assign it to the environment.
    $\varepsilon_t \sim \mathcal{N}(0, \Sigma_j)$ if $t \in \mathcal{T}_{\texttt{noise}}$, else $\varepsilon_t \leftarrow 0$
    $a_t \leftarrow \pi^k(s_t) + \varepsilon_t$
    $e_t \leftarrow 1$ if (using expert π^e and $\varepsilon_t = 0$), else $e_t \leftarrow 0$
    $v_t \leftarrow V^e(s_t)$
    $(s_{t+1}, o_t, r_t) = \texttt{EnvStep}(s_t, a_t)$
    Add (o_t, a_t, e, r_t, v_t) to $\mathcal{D}_{\texttt{sim}}$

C Manipulation Experiment Details

Expert Policy Training. For our manipulation experts we use the expert policies π^e and value function V^e from (Yin et al., 2026), replicating the training from this work exactly. For the sake of brevity, we refer the reader to (Yin et al., 2026) for more details, including the exact rewards we use.

Data Generation. To generate the simulation dataset $\mathcal{D}_{\texttt{sim}}$, environments are reset with sub-optimal policies $\{\pi^k\}_{k=1}^K$ with probability 0.5, and Gaussian action perturbations are injected in contiguous intervals sampled from $\mathcal{U}[1, 5]$ steps, interleaved with noise-free intervals sampled from $\mathcal{U}[5, 10]$ steps. We generate 100k trajectories for each tasks, of which approximately 36% optimal actions. We save policies every 100 checkpoints up to checkpoint 1000.

World Model Structure. The encoder for the world model first passes each of the three camera images through a ResNet-18 encoder pretrained on imagenet, producing embeddings of size 3×512, which are stacked and concatenated with the robots 6 joint observations and then passed through an MLP to produce the latent z. Specific parameters are in Table 2.

Table 2: World model architectural parameters for manipulation.

Parameter	Value
Embedding dimension	64
All transformers MLP hidden size	256
Dynamics transformer layers	3
Dynamics transformer heads	4
Reward transformer layers	1
Reward transformer heads	1
Value transformer layers	1
Value transformer heads	1
Base policy transformer layers	4
Base policy transformer heads	8

World Model Pretraining. We pretrain the world model for two epochs over the full simulation dataset $\mathcal{D}_{\texttt{sim}}$. Using a batch size of 256 and approximately 200k gradient updates. Optimization is performed with Adam using an initial learning rate of 2×10^{-4}, which is annealed to 1×10^{-4} via a cosine decay schedule, with a linear warmup over the first 10,000 steps. We apply data augmentation by injecting zero-mean Gaussian noise into the proprioceptive observations, and visual augmentations such as color jitter, gaussian blurring and random cropping.

Hardware Deployment. We use MPPI as implemented in TD-MPC (Hansen et al., 2022) with the hyperparameters listed in Table 3.

Table 3: MPPI parameters for manipulation.

Parameter	Value
Candidate actions batch size	250
Noised base policy actions batch size	100
Solver iterations	3
Initial action standard deviation	1.0
Minimum action standard deviation	0.05
Base policy action standard deviation	0.1
Elites	64
Temperature	0.4
Momentum	0.0
Discount	0.99

D QUADRUPED EXPERIMENT DETAILS

Expert Policy Training. We train a state-based expert policy π^e and its associated optimal value function V^e using PPO in IsaacLab (Mittal et al., 2025). Both the policy and value networks are MLPs with three hidden layers of width 512 and operate on privileged simulator state variables listed in Table 4. The expert is trained with a dense state-based reward composed of the terms summarized in Table 5; full implementation details will be released with the public code. To improve robustness and coverage, we apply domain randomization over the parameters in Table 4 and randomize terrain conditions across steps, boxes, rough terrain, and slopes, using a curriculum that gradually increases terrain difficulty. Training is performed with 4096 parallel simulation environments over 5000 PPO iterations, with 24 environment steps per iteration, for a total of 490M environment steps. We save policy checkpoints at iterations $\{0, 50, 100, 150, 200, 250, 300, 400, 500, 1000, 2000\}$.

Data Generation. To generate the simulation dataset $\mathcal{D}_{\texttt{sim}}$, environments are reset with sub-optimal policies $\{\pi^k\}_{k=1}^K$ with probability 0.5, and Gaussian action perturbations are injected in contiguous intervals sampled from $\mathcal{U}[1, 50]$ steps, interleaved with noise-free intervals sampled from $\mathcal{U}[25, 500]$ steps. We run 4096 parallel environments for 25000 steps, yielding approximately 100M data points, of which 55.7% correspond to uncorrupted expert actions. Data generation takes approximately 7 hours with a single NVIDIA RTX 4500 Ada GPU.

World Model Structure. Figure 7 illustrates the world model architecture, and Table 7 lists the corresponding model parameters used in the quadruped experiments. The observation space for the quadruped is given in Table 6. The history encoder processes a history of proprioceptive observations (all observations except the height map) and actions by first projecting each input, assigning a type embedding to distinguish observations from actions, and interleaving the resulting embeddings to form the history representation h_t. The latent encoder E_θ encodes the local terrain height map using a CNN followed by spatial encoding, flattening, and projection; this representation is concatenated with the projected latest proprioceptive observation and passed through an MLP to produce the latent state embedding. Commands consist of the desired forward, lateral, and yaw velocities of the base $g_t := (v_t^x, v_t^y, \omega_t)$. The future commands $g_{t:t+T-1}$ are concatenated to the inputs of the base policy π_θ, reward model R_θ, and value model V_θ.

World Model Pretraining. We pretrain the world model for two epochs over the full simulation dataset $\mathcal{D}_{\texttt{sim}}$. Using a batch size of 512, this corresponds to approximately 3.69×10^5 gradient update steps. Optimization is performed with Adam using an initial learning rate of 2×10^{-4}, which is annealed to 1×10^{-4} via a cosine decay schedule, with a linear warmup over the first 10,000 steps. We apply data augmentation by injecting zero-mean Gaussian noise into the input observations (both proprioceptive and height map) during training. Pretraining requires approximately 28 hours on a single NVIDIA RTX 4500 Ada GPU.

Hardware Deployment. We use MPPI as implemented in TD-MPC (Hansen et al., 2022) with the hyperparameters listed in Table 8. To encourage straight-line locomotion during quadruped experiments, we compute commanded base velocities g_t from the robot's current base pose using a PD controller on position, and provide these commands to the world model.

Table 4: Privileged simulator state space for the quadruped, along with domain randomization ranges, where applicable.

State Variable	Dim.	Domain Rand.
Base linear velocity	3	-
Base angular velocity	3	-
Projected gravity vector	3	-
Commanded base twist	3	-
Joint angles	12	-
Joint speeds	12	-
Previous action	12	-
Cosine / sine of phase	2	-
Height map	$(21, 15)$	-
Foot force wrenches	$(4, 6)$	-
Foot heights	4	-
Base mass	1	$-1.0, +3.0$ kg
Static / dynamic friction	2	$[0.2, 1.2]$
Coefficient of restitution	1	$[0.0, 0.3]$
Joint stiffness	12	$\pm 10\%$
Joint damping	12	$\pm 10\%$
Joint friction	12	$[0.0, 0.05]$

Table 5: State-based reward terms used for quadruped expert policy training.

Term	Weight
Commanded x, y-velocity tracking reward	1.5
Commanded yaw rate tracking reward	0.75
Desired gait reward	0.05
Desired gait foot height reward	0.2
Base z-velocity penalty	-2.0
Base angular velocity penalty	-0.05
Base orientation penalty	-4.0
Deviation from default hip joint angles penalty	-0.25
Joint torque penalty	-2.0×10^{-4}
Joint acceleration penalty	-2.5×10^{-7}
Action rate penalty	-0.01

Detailed Results. Table 9 reports detailed real-world quadruped locomotion results on both tasks across commanded forward speeds. We report both success rate (successful trials out of five) and average forward progress (mean $\pm$ standard deviation) over all trials at each speed. The *Pretrained model* corresponds to zero-shot deployment of the simulation-pretrained world model without any real-world finetuning. While this model occasionally achieves partial forward progress, it fails to complete the task reliably, highlighting the severity of the sim-to-real dynamics gap. The *Single-step BC policy*, which serves as the initial policy for the RLPD and IQL baselines prior to finetuning, improves performance in some settings but remains inconsistent and rarely achieves full task completion.

Table 6: Observation space for the quadruped.

Observation	Dimension
Base linear velocity (local frame)	3
Base angular velocity (local frame)	3
Projected gravity vector	3
Joint angles	12
Joint speeds	12
Cosine and sine of phase	2
Height map	$(21, 15)$

Table 7: World model architectural parameters for the quadruped.

Parameter	Value
Embedding dimension	64
Proprioceptive observations MLP hidden dims	128, 128
CNN kernel size	3
CNN strides	2, 2, 2
CNN features	8, 16, 32
All transformers MLP hidden size	256
Dynamics transformer layers	2
Dynamics transformer heads	8
Reward transformer layers	1
Reward transformer heads	1
Value transformer layers	1
Value transformer heads	1
Base policy transformer layers	4
Base policy transformer heads	8

Table 8: MPPI parameters for the quadruped.

Parameter	Value
Candidate actions batch size	450
Noised base policy actions batch size	22
Solver iterations	8
Initial action standard deviation	2.0
Minimum action standard deviation	0.05
Base policy action standard deviation	0.05
Elites	64
Temperature	0.25
Momentum	0.0
Discount	0.99

After real-world finetuning (32.1 minutes of data for Foam and 35.7 minutes for Slippery Slope), `SimDist` consistently achieves the highest success rates and forward progress across all tested speeds. In contrast, both IQL and RLPD exhibit limited improvement despite access to the same real-world data budget. In particular, RLPD destabilized the robot during adaptation on the Foam task and is therefore not reported for that condition.

E Ablations Details

Each ablation corresponds to a separately pretrained world model. Across all ablations, the same planning hyperparameters and evaluation environments are used, and each configuration is evaluated on an identical set of randomized environments. Next, we discuss platform specific details.

Manipulation. For both manipulation tasks, we evaluate success rates over the initial condition and domain randomization described in the environments from (Yin et al., 2026).

Quadruped. The robot is commanded to walk forward at a specified target speed until episode termination. For each evaluated model, we instantiate environments covering all combinations of parameters listed in Table 10, resulting in 1080 distinct environments per model. All models are evaluated on the same fixed set of environments to ensure fair comparison. Each environment is run for a single episode, during which state-based reward is accumulated until termination. Episodes terminate either after 1000 simulation steps or upon failure, defined as body contact with the ground or violation of base orientation limits. Results are summarized in Table 1, where we report the average accumulated reward per episode across all environments.

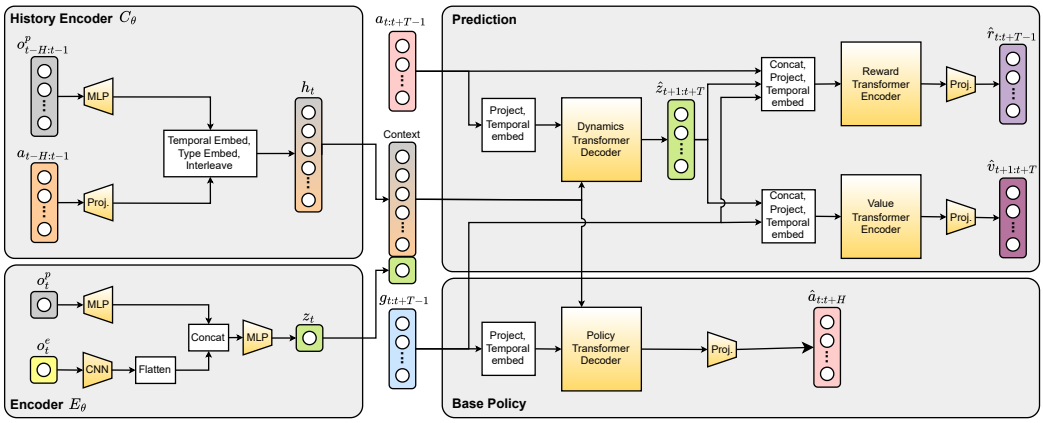

Figure 7: Detailed world model architecture for the quadruped.

Table 9: Real-world quadruped results for both tasks. Success is reported as successful trials out of five. Forward progress is reported as mean ± standard deviation (meters) across trials at each commanded speed. The *Pretrained model* corresponds to zero-shot deployment of the simulation-trained world model. The *Single-step BC policy* is the behavior cloning policy used to initialize IQL and RLPD prior to finetuning. `SimDist`, IQL, and RLPD results reflect performance after real-world finetuning using 35.7 minutes (Slippery Slope) and 32.1 minutes (Foam) of data. RLPD results on the Foam task are not reported, as the method destabilized the robot prior to evaluation.

	Slippery Slope					
Speed $\mathrm{m\,s^{-1}}$ →	0.1		0.3		0.5	
Method	Fwd. Prog.	Success	Fwd. Prog.	Success	Fwd. Prog.	Success
Pretrained model	0.70 ± 0.56	0/5	0.43 ± 0.13	0/5	0.90 ± 0.40	0/5
Single-step BC policy	1.49 ± 0.29	2/5	1.43 ± 0.25	1/5	0.62 ± 0.23	0/5
`SimDist` (ours)	$\mathbf{1.78 \pm 0.08}$	**4/5**	$\mathbf{1.82 \pm 0.00}$	**5/5**	$\mathbf{1.82 \pm 0.00}$	**5/5**
IQL	0.00 ± 0.00	0/5	0.39 ± 0.56	0/5	0.46 ± 0.42	0/5
RLPD	0.32 ± 0.01	0/5	0.34 ± 0.05	0/5	0.35 ± 0.01	0/5
	Foam					
Speed $\mathrm{m\,s^{-1}}$ →	0.2		0.7		1.2	
Method	Fwd. Prog.	Success	Fwd. Prog.	Success	Fwd. Prog.	Success
Pretrained model	2.98 ± 0.02	3/5	2.39 ± 0.71	2/5	1.70 ± 0.99	0/5
Single-step BC policy	2.07 ± 1.01	1/5	1.54 ± 0.81	1/5	2.45 ± 0.47	2/5
`SimDist` (ours)	$\mathbf{3.00 \pm 0.00}$	**5/5**	$\mathbf{3.00 \pm 0.00}$	**5/5**	$\mathbf{3.00 \pm 0.00}$	**5/5**
IQL	0.92 ± 0.48	1/5	2.25 ± 0.87	2/5	2.73 ± 0.34	3/5
RLPD	–	–	–	–	–	–

Table 10: Parameter values used to construct quadruped ablation environments in simulation. Each environment is defined by a unique combination of commanded forward speed, ground friction coefficient, terrain type, and terrain difficulty. All combinations are evaluated for each model, yielding 1080 environments per model.

Speed	Friction	Terrain	Terrain Difficulty
0.2	0.2	Boxes	0.2
0.4	0.4	Rough	0.4
0.6	0.6	Stairs Up	0.6
0.8	0.8	Stairs Down	0.8
1.0	1	Slope Up	1
1.2	1.2	Slope Down	