EASI: Evolutionary Adversarial Simulator Identification for Sim-to-Real Transfer

Haoyu Dong, Huiqiao Fu, Wentao Xu, Zhehao Zhou, Chunlin Chen*
Department of Control Science and Intelligent Engineering,
School of Management and Engineering, Nanjing University, China
{haoyudong,hqfu, wtxu, zhzhou}@smail.nju.edu.cn,
clchen@nju.edu.cn

Abstract

Reinforcement Learning (RL) controllers have demonstrated remarkable performance in complex robot control tasks. However, the presence of reality gap often leads to poor performance when deploying policies trained in simulation directly onto real robots. Previous sim-to-real algorithms like Domain Randomization (DR) requires domain-specific expertise and suffers from issues such as reduced control performance and high training costs. In this work, we introduce Evolutionary Adversarial Simulator Identification (EASI), a novel approach that combines Generative Adversarial Network (GAN) and Evolutionary Strategy (ES) to address sim-to-real challenges. Specifically, we consider the problem of sim-to-real as a search problem, where ES acts as a generator in adversarial competition with a neural network discriminator, aiming to find physical parameter distributions that make the state transitions between simulation and reality as similar as possible. The discriminator serves as the fitness function, guiding the evolution of the physical parameter distributions. EASI features simplicity, low cost, and high fidelity, enabling the construction of a more realistic simulator with minimal requirements for real-world data, thus aiding in transferring simulated-trained policies to the real world. We demonstrate the performance of EASI in both sim-to-sim and sim-to-real tasks, showing superior performance compared to existing sim-to-real algorithms.

1 Introduction

With the development of Reinforcement Learning (RL) , an increasing number of RL controllers have demonstrated remarkable performance in complex, high-dimensional robot control tasks[1, 2]. Such RL algorithms require minimal domain-specific knowledge, offering superior control performance and more natural robot locomotion compared to traditional control methods. However, training these RL controllers requires plenty of trial and error processes, it's unsafe and costly to complete training tasks in the real world. On the other hand, simulation provides a cheaper and safer way to obtain a vast amount of trial data which is more suitable for RL training. Yet, due to simulation inaccuracies, disparities between simulation and reality inevitably exist and simulated-trained policies transferring directly to the real environment often result in poor performance, which is also known as 'reality gap'.

To bridge the gap between simulation and reality, many studies focus on sim-to-real transfers to translate simulated-trained policy to reality. As one of the most popular sim-to-real methods, Domain Randomization (DR) [3–6] injects random noise into observations, actions, and physical parameters during simulated training, which helps agents adapt to different environments, making the policies

*Corresponding author

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

more robust. However, DR requires a good understanding of the specific domain to determine the appropriate ranges for these random variations. If these ranges are set inappropriately, it can hinder the successful transfer of policies. Additionally, DR often needs longer training time to ensure adaptability of the policy to various environments, and always results in suboptimal and conservative performance[6]. To tackle these challenges, recent attempts have been made to use real-world data for fostering policy transfer to real-world applications. One approach is optimizing simulator parameters to make the simulation more realistic [7–14]. Unfortunately, these methods still face challenges, such as the need to iteratively collect large amounts of high-cost real-world data, difficulty in applying to high-dimensional control tasks, and instability in the parameter optimization process.

In this work, we introduce Evolutionary Adversarial Simulator Identification (EASI), a novel sim-to-real method that frames sim-to-real transfer as a search problem and employs a unique collaboration between Generative Adversarial Network (GAN) and Evolutionary Strategy (ES). EASI searches for physical parameters to align state transitions between simulation and reality as closely as possible, ensuring that policies trained in simulation seamlessly transfer to the real world. Within EASI, ES acts as a generator in an adversarial competition with a neural network discriminator, which distinguishes state transitions originating from either simulation or reality. The discriminator assigns higher scores to state transitions more likely sampled from reality, acting as the fitness function for ES and guiding the evolution of physical parameter distributions. Concurrently, the discriminator's ability to differentiate between simulated and real state transitions improves through iterative training. EASI features simplicity, low cost, and high fidelity, requiring only a small amount of real-world data to bridge the gap between simulation and reality.

In the experiments, we test EASI on 4 sim-to-sim tasks (Go2, Cartpole, Ant, Ballbalance) and 2 sim-to-real tasks (Go2, Cartpole). We first use Uniform Domain Randomization (UDR) to train a rough initial policy over a wide parameter range, allowing the agent to collect data in the target environment controlled by the initial policy. Then, we employ EASI for simulator identification that makes the simulator most similar to the target domain. The final policy is trained based on these optimized simulator parameters. Our experiments demonstrate that EASI can stably and rapidly search for simulator parameters, and the policies trained within the optimized parameter distribution can effectively transfer to the target domain. Video and code are shown at our page. In particular, we make three main contributions in this work:

1. We propose a novel method, EASI, which employs a unique collaboration between GAN and ES to address sim-to-real challenges.
2. We model sim-to-real transfer as a search problem, searching for simulator parameters in an adversarial process to make state transitions between simulation and reality most similar.
3. We demonstrate the performance of EASI in both sim-to-sim and sim-to-real tasks, showing superior performance compared to existing sim-to-real algorithms.

2 Related Work

Transferring simulated-trained policies to the real world in a stable and cost-effective manner has long been a goal for sim-to-real transfers. Previous research has approached sim-to-real transfer from both zero-shot and few-shot perspectives.Zero-shot sim-to-real transfer directly trains a policy capable of operating in reality without using any real-world data. Domain Randomization (DR) is one of the most intuitive and widely applied zero-shot sim-to-real methods[3–6]. DR injects random noise into state observations, actions, and physical parameters to enhance the robustness of the policy, allowing the agent to adapt to various environments. However, DR requires specific domain prior knowledge and hand-engineering to determine the distribution of injected random noise, while also suffering from reduced control performance and high training costs. To improve DR, Akkaya et al.[15] used task performance as a metric to develop a curriculum that gradually increases the level of randomization. Mehta et al.[16] employed GAN to identify the most informative parts of the randomization parameter space, focusing more attention on these areas. Furthermore, some zero shot methods [17–19] introduced an adaptation module or estimator to predict robot state latent variables or explicitly estimate privileged information that the robot cannot access in reality, allowing the robot to adapt to different reality environments relying only on limited proprioception.

Few-shot sim-to-real transfer leverages real-world data to facilitate policy transfer. Domain adaptation is one of the approaches that allows policy to generalize to reality. Image adaptation methods [20–23]

used GAN to transform simulated RGB images in a way that is matched to the reality. Grounded Action Transformation (GAT) [24–26] suggested action transformation that applying the transformed actions in simulation has the same effects as applying the original actions had on reality. In robotic tasks, domain adaptation methods often require a substantial amount of costly real-world data for effective transfer. Inspired by classic system identification [27–29], some efforts focused on optimizing simulators so that the policies trained within them will better adapt to reality. Bayesian optimization was used to find the posterior distribution of simulator parameters based on real-world observations [13, 14, 30]. Chebotar et al.[8] and Chang et al.[31] treated the trajectory mismatch as the cost function and iteratively updated the simulator parameters. Memmel et al. [11] and Marija et al.[32] designed effective real-world exploration policies to sample more informative trajectories for system identification. TuneNet [7] employed supervised learning to train a neural network mapping simulation trajectories to parameter gradients toward the real world. Some approaches [9, 33] used RL to find proper parameters, treating trajectories as RL states and parameters as RL actions. SimGAN [10] employed GAN to distinguish between source and target domain dynamics and enhance the hybrid simulator to make it more realistic. However, most current simulation parameter optimization methods still suffer from instability, high cost of reality data collecting and are only suitable for low-dimensional control tasks.

In this work, EASI employs GAN to differentiate between state transitions in simulation and reality, using ES to search for parameters that align the simulation more closely with the real world based on the GAN's distinction results. EASI requires lower demands on real-world data while enabling faster and more stable searching of simulator parameters. This helps us obtain high-fidelity, low-cost simulators to provide a real world like environment for RL training.

3 Preliminary

3.1 Domain Randomization

Consider a RL problem defined by the Markov Decision Process (MDP) represented by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \gamma, \mathcal{P}(\xi) \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ represents the reward function, γ is the discount factor, $\mathcal{P}(\xi) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}_+$ represents the system's state-transition probability function under simulator parameters ξ. The state transition varies depending on the type of system and its specific parameters. We denote the real-world state transition as $\mathcal{P}_r(\xi_{real})$ and the simulation state transition as $\mathcal{P}_s(\xi_{sim})$. In RL process, an agent interacts with the environment and learns a policy $\pi(\mathbf{a}|\mathbf{s})$ to maximize the expected discounted return $\mathbb{E}_{p(\tau|\pi)}\left[\sum_{t=0}^{T-1} \gamma^t \mathcal{R}\left(\mathbf{s}_t, \mathbf{a}_t\right)\right]$, where $p(\tau|\pi) = p(\mathbf{s}_0) \prod_{t=0}^{T-1} p\left(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t\right) \pi(\mathbf{a}_t|\mathbf{s}_t)$ represents the likelihood of a trajectory $\tau = \{\mathbf{s}_0, \mathbf{a}_0, r_0, \mathbf{s}_1, \ldots, \mathbf{s}_{T-1}, \mathbf{a}_{T-1}, r_{T-1}, \mathbf{s}_T\}$ under π. Due to the existence of the reality gap, policies trained in simulation often perform poorly when directly transferred to the real environment, and the manifestation of the reality gap in MDP lies in the difference between $\mathcal{P}_r(\xi_{real})$ and $\mathcal{P}_s(\xi_{sim})$. To address this issue, DR samples $\xi_{sim} \sim \Xi \in \mathbb{R}^N$ and sets the simulator's parameters to ξ_{sim} at each simulation training round, allowing the RL-trained policy to adapt to various $\mathcal{P}_s(\xi_{sim})$, thus improving system robustness. However, to train policies that perform well in the real world using DR, it is necessary to ensure that there exists $\xi^* \in \Xi$ such that $\mathcal{P}_r(\xi_{real}) \approx \mathcal{P}_s(\xi^*)$, which means it needs specific domain knowledge to determine Ξ.

3.2 Evolutionary Strategy

Evolutionary Strategy (ES) is a method that mimics biological evolution to solve high-dimensional, continuous-value domain problems. ES algorithm addresses the following search problem: maximize a nonlinear fitness function that is a mapping from search space $\mathbb{R}^d$ to fitness value $\mathbb{R}$. In each iteration (generation), the individual from the previous generation undergoes recombination and mutation to reproduce the next generation of individuals. Each offspring is evaluated through the fitness function, and individuals with the higher scores form the next generation's population.

In this work, we use the discriminator as the fitness function of ES to evaluate the similarity of state transition, guiding ES to find ξ_{sim} that deceives the discriminator to minimize the distance between $\mathcal{P}_r(\xi_{real})$ and $\mathcal{P}_s(\xi_{sim})$.

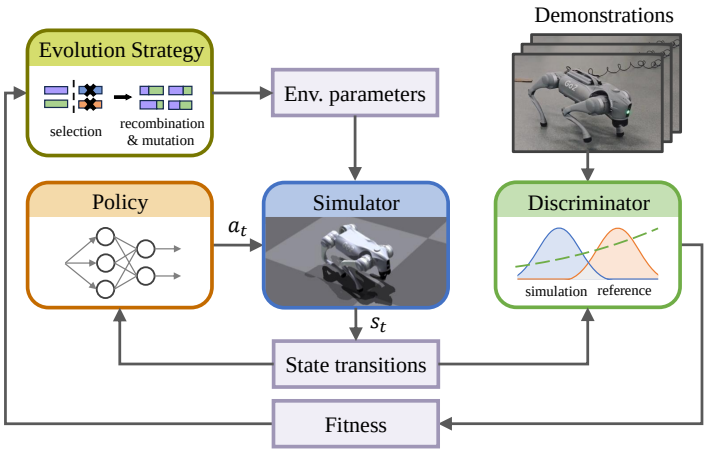


Figure 1: Schematic overview of EASI. ES acts as a generator in adversarial competition with a neural network discriminator distinguishing between simulation and reality state transitions. The discriminator serves as the fitness function, guiding the evolution of the physical parameter distributions.

3.3 Generative Adversarial Network

GAN [34] consists two neural networks, generator G and discriminator D, engaged in an adversarial process to generate new data with the same statistics as the training set. G and D play a two-player minimax game with value function $V(G, D)$, which can be described as

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z \sim p(z)}[log(1 - D(G(z)))], \tag{1}$$

where $p(z)$ is the prior on input noise variables, p_{data} is the training set. G and D continuously adjust parameters, with the ultimate goal of training a generator strong enough to let the discriminator unable to distinguish whether the outputs of the generator network are generated or sampled from training set.

4 Evolutionary Adversarial Simulator Identification

4.1 Imitate State Transitions

Modeling the processes in reality and simulation as MDPs, the primary difference between the two MDPs lies in their state transitions $\mathcal{P}_r(\xi_{real})$ and $\mathcal{P}_s(\xi_{sim})$. In this paper, we focus on minimizing the state transition gap between simulation and reality. For simulation with variable physical parameters, we aim to find the distribution of the parameters to achieve

$$\Xi = \arg\min_{\xi_{sim} \in \Xi} \|\mathcal{P}_r(\xi_{real}), \mathcal{P}_s(\xi_{sim})\|. \tag{2}$$

In EASI, we use a discriminator to distinguish whether state transitions come from simulation or reality, the discriminator $D(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ can be trained using

$$\max_D \mathbb{E}_{d^{\mathcal{M}}(\mathbf{s},\mathbf{a},\mathbf{s}')}[log(D(\mathbf{s}, \mathbf{a}, \mathbf{s}'))] + \mathbb{E}_{d^{\mathcal{B}}(\mathbf{s},\mathbf{a},\mathbf{s}')}[log(1 - D(\mathbf{s}, \mathbf{a}, \mathbf{s}'))], \tag{3}$$

where $d^{\mathcal{M}}(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ and $d^{\mathcal{B}}(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ denote the distribution of action $\mathbf{a}$ causing state transition from state $\mathbf{s}$ to $\mathbf{s}'$ in the demonstration dataset and in simulation respectively. As demonstrated in previous work [34], the discriminator serves as a measure of the Jensen-Shannon divergence between $\mathcal{P}_r(\xi_{real})$ and $\mathcal{P}_s(\xi_{sim})$.

It is important to acknowledge that, due to inherent design factors in the simulator, achieving state transitions that are precisely identical to those in the real world may be impossible. Nonetheless, our experiments have shown that EASI significantly enhances the simulator's realism, allowing policies trained in simulation to transfer more effectively to real world.

Algorithm 1: EASI

Input: Initial parameter distribution $\Xi^{(0)}$, motion control policy π_0, demonstration state transition $\mathcal{M}$, simulation state transition buffer $\mathcal{B}$

Output: Parameter distribution Ξ^*

for *generation* $i = 0, 1, 2, \cdots, G$ **do**
  Sample individuals by $\xi_j^{(i)} \sim \Xi^{(i)} \quad j = 1, 2, 3, \cdots, N$
  for *simulation environment* $j = 1, 2, 3, \cdots, N$ **do**
    Set simulator parameters to $\xi_j^{(i)}$
    Use π_0 to sample trajectory τ_j
    Store τ_j in $\mathcal{B}$
  end
  for *update step* $= 1, 2, 3, \cdots, n$ **do**
    $b^{\mathcal{M}} =$ sample batch of K state transitions$\{(\mathbf{s}_k, \mathbf{a}_k, \mathbf{s}'_k)\}_{k=1}^{K}$ from $\mathcal{M}$
    $b^{\mathcal{B}} =$ sample batch of K state transitions$\{(\mathbf{s}_k, \mathbf{a}_k, \mathbf{s}'_k)\}_{k=1}^{K}$ from $\mathcal{B}$
    Update D according to Equation 4 using $b^{\mathcal{M}}$ and $b^{\mathcal{B}}$
    Clip network weights of D
  end
  Calculate discriminator reward $r_j^{(i)} = \mathbb{E}_{\tau_j}[D(\mathbf{s}, \mathbf{a}, \mathbf{s}')] \quad j = 1, 2, 3, \cdots, N$
  Using ES to find next generation distribution $\Xi^{(i+1)} =$ ES $(\xi^{(i)}, r^{(i)})$
end

4.2 Wasserstein Loss

The standard GAN discriminator objective detailed in Equation 3 typically uses a binary cross-entropy loss function which tends to encounter the problem of vanishing gradients due to saturation regions [35]. Specifically, if the discriminator is overly powerful, it may excessively classify state transitions, leading to identical bad rewards for all transitions sampled from the simulation. This makes it difficult to identify which parameters yield more realistic state transitions. Additionally, due to inherent discrepancies between the simulator and reality, the state transitions generated by the simulator may not perfectly align with those in the real world, further complicating the challenges posed by a strong discriminator. To address such challenges, WGAN-style discriminator [36] is proposed to mitigate the issue of gradient vanishing and has proven effective for robot-related tasks [2]. In our work, we employ Wasserstein loss to update the discriminator using

$$\max_{D} \mathbb{E}_{d^{\mathcal{M}}(\mathbf{s},\mathbf{a},\mathbf{s}')}[D(\mathbf{s}, \mathbf{a}, \mathbf{s}')] - \mathbb{E}_{d^{\mathcal{B}}(\mathbf{s},\mathbf{a},\mathbf{s}')}[D(\mathbf{s}, \mathbf{a}, \mathbf{s}')]. \tag{4}$$

This WGAN-style discriminators is an efficient approximation to the Earth-Mover distance which more effectively and smoothly measures the distance between two probability distributions. We clip the network weights to $[-0.01, 0.01]$ as the original WGAN did to ensure the Lipschitz continuity of $D(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ in the training process. Using Wasserstein loss, we could better evaluate the similarity between simulation and real world state transitions.

4.3 Evolutionary Strategy for Parameter Search

Our goal is to find a parameter distribution (e.g. Gaussian distribution) that makes the simulator most similar to the demonstration. The distance between $P_r(\xi_{real})$ and $P_s(\xi_{sim})$ is measured by $D(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ and what we need to do is to find the best parameter distribution Ξ^* that get the highest reward of $D(\mathbf{s}, \mathbf{a}, \mathbf{s}')$, which could be described as

$$\Xi^* = \arg\min_{\Xi} \max_{D} \mathbb{E}_{d^{\mathcal{M}}(\mathbf{s},\mathbf{a},\mathbf{s}')}[D(\mathbf{s}, \mathbf{a}, \mathbf{s}')] - \mathbb{E}_{d^{\mathcal{B}}(\mathbf{s},\mathbf{a},\mathbf{s}')}[D(\mathbf{s}, \mathbf{a}, \mathbf{s}')]. \tag{5}$$

In EASI, we employ the ES as the generator, with the goal of creating a parameter distribution for the simulator that closely mimics the state transitions observed in the demonstration data. EASI uses the ES to select parameters that receive higher rewards from the discriminator, designating these parameters as elites. The elites then undergo recombination and mutation to produce the next

generation of parameters. The discriminator evaluates whether the state transitions in the generated trajectories are sufficiently similar to those of the expert demonstrations, assigning scores based on the degree of similarity. Simultaneously, the discriminator is trained using the trajectories collected during simulation and demonstration, guided by Equation 4. With the selection process driven by the ES, the state transitions produced by the simulator increasingly resemble those in the demonstration data, and the discriminator can better distinguish the gap between simulated and demonstration state transitions. Ultimately, this algorithm yields a set of simulator parameters that align the state transitions in the simulated environment with those in the real world. This optimized parameter set is then utilized to train policies that can be directly applied in real world. The schematic overview of the EASI architecture is shown in Fig. 1, and the pseudo-code is shown in Algorithm 1.

It's notable that when sampling trajectory from simulation and real world, the motion control policy we use does not necessarily match the final desired policy and the robot may be controlled by MPC[37], PID[38], RL, or even manually operated by humans, as long as it can collect state transition containing enough state transition information we need. This characteristic of our algorithm significantly reduces the cost of collecting real data. With our algorithm, using a rough policy run in the real world to collect suboptimal trajectories is enough to find a proper parameter for the simulator.

5 Experiment

In this section, we conduct experiments in various tasks to validate the performance of EASI and answer three questions:

1. Can EASI enhance the simulator's similarity to real-world?
2. Can EASI improve performance in sim-to-real transfer tasks?
3. How much real-world data is required for EASI?

Task Setup We test EASI in 4 sim-to-sim tasks illustrated in Fig. 2 and 2 sim-to-real tasks presented in Fig. 3. **(1) Cartpole** environment includes an inverse pendulum connected to a 1-DoF cart. The goal of Cartpole task is to keep the pendulum on the cart balanced for as long as possible. **(2) Go2** environment is a 12-DoF Unitree Go2 quadruped robot, with each leg capable of 3-DoF. The goal of Go2 task is to keep the quadruped run forward and track a specified speed in a natural locomotion. **(3) Ant** environment is an 8-DOF quadruped robot consisting of four legs attached to a common base. The goal of Ant task is controlling the ant run as fast as possible. **(4) Ballbalance** environment consists a table with three legs and a ball placed on the table. Each leg has 1 DoF to control the target angle of the leg joint. The goal of Ballbalance is to stabilize the ball at a target position on the table.

We utilize Isaac Gym [39] as the simulator. In the simulator identification process, we create 300 parallel environments for EASI to evaluate the similarity between reality and simulator with different

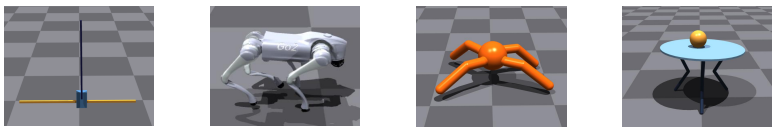


Figure 2: Experiment tasks in simulation: Cartpole, Go2, Ant, and Ballbalance, presented in order from left to right.

Figure 3: *Left*: Cartpole task in reality, aiming to keep the pendulum on the cart balanced for as long as possible. *Right*: Go2 task in reality, aiming to keep running forward and track a specified speed.

parameters. For Cartpole, Ant, and Ballbalance, we use SAC [40] to train a policy (4 layer MLP) as the locomotion controller. As for Go2, we employ Ess-InfoGAIL [41] to train a quadruped controller.

Baselines **(1) Uniform Domain Randomization (UDR)**: Uniformly sampling parameters from Ξ_{UDR} at the beginning of each training iteration. **(2) Oracle**: Directly training in the pseudo-real environment in sim-to-sim experiments, representing the ideal upper bound for sim-to-sim transfer tasks. **(3) FineTune**: Fine-tune the UDR policy using real-world data, enabling the UDR to adapt to the real world in advance to improve transfer performance. **(4) GARAT**: GARAT[26] adjusts actions from the policy such that, after adjustment, the actions applied in the simulator result in state transitions more similar to those in the real demonstration.

EASI Implementation There are no specific requirements for the selection of the ES in EASI, mainstream ES such as CMA-ES[42], NES[43] etc. can be used. In this paper, we employ $(\mu/\mu_I, \lambda)$-ES [44] as the generator with the setting $\mu = 150$ and $\lambda = 300$, which means every generation has 300 individuals and 150 elites are chosen to undergo recombination and mutation to generate the next generation. For the collection of the demonstration dataset, we use UDR to train a rough policy, and then use the policy to control agents in the target domain collecting trajectories.

EASI can quickly accomplish parameter searching tasks. The discriminator can directly evaluate the similarity between simulator and reality by leveraging a single trajectory sampled from the simulator with a specific parameter. Furthermore, in Isaac Gym, we can parallelly collect trajectories in hundreds of environments with different parameters which means we could evaluate hundreds of parameters parallelly. In our experiment, running on a PC equipped with Intel i5-13600KF and RTX 4060 Ti, EASI completed the evolutionary adversarial searching process in **less than 10 minutes**.

5.1 Sim-to-Sim Policy Transfer

To validate the theoretical capability of EASI for simulator identification, we conduct sim-to-sim experiments in 4 tasks (Cartpole, Go2, Ant, and Ballbalance). Initially, a set of parameters ξ_r is selected as the target parameters for the simulator, which is regarded as the pseudo-real environment. For UDR, we use $\Xi_{UDR} = U[\frac{1}{k} \times \xi_r, k \times \xi_r]$ as the uniform parameter distribution and train an initial policy π_0 with UDR. Depending on the difficulty of the task, we set $k = 3$ for the Cartpole, Ant, and Ballbalance tasks, and $k = 2$ for Go2 task. Then π_0 is used to collect 200 trajectories in the pseudo-real environment as demonstration state transition $\mathcal{M}$. During the EASI parameter optimization process, two approaches are used for selecting the initial parameter distribution: Within Distribution (WD) and Out of Distribution (OOD). In the WD experiment, the initial parameter distribution $\Xi^{(0)}$ is consistent with the domain randomization distribution Ξ_{UDR}, and the target parameter ξ_r is included within the initial parameter distribution $\Xi^{(0)}$. In the OOD experiment, we deliberately set incorrect initial parameter distribution $\Xi^{(0)}$ (e.g. set as $U[0, \xi_r]$ or $U[\xi_r, 2 \times \xi_r]$) so that the target parameter is outside the initial parameter distribution. Taking Go2 and Ant environment as examples, there are 8 parameters for Go2 and 11 parameters for Ant to identify. Here we choose 2 of the parameters in each task to demonstrate the generating process of the parameter. Fig. 4 represents the convergence process of parameters, where the x-axis represents the generations of ES, and the y-axis represents the parameter values. It can be observed from the figure that at the beginning, parameters are uniformly distributed within $\Xi^{(0)}$. As the parameter evolution progresses, the parameter distribution quickly adjusts to the vicinity of the target parameters. This experiment showcases EASI's capability to adjust parameter distributions, even when the initial

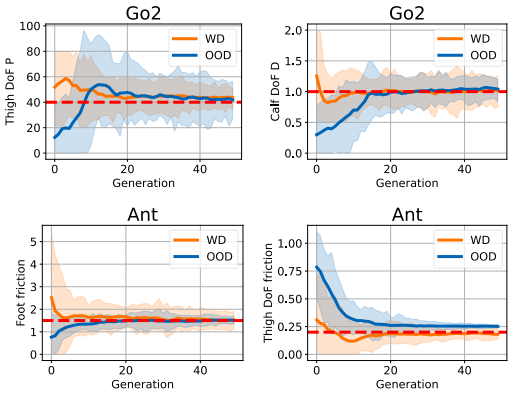


Figure 4: Evolution process of parameters. The red dashed line represents the target parameter. WD means target parameter within initial parameter distribution, and OOD means target parameter out of initial parameter distribution.

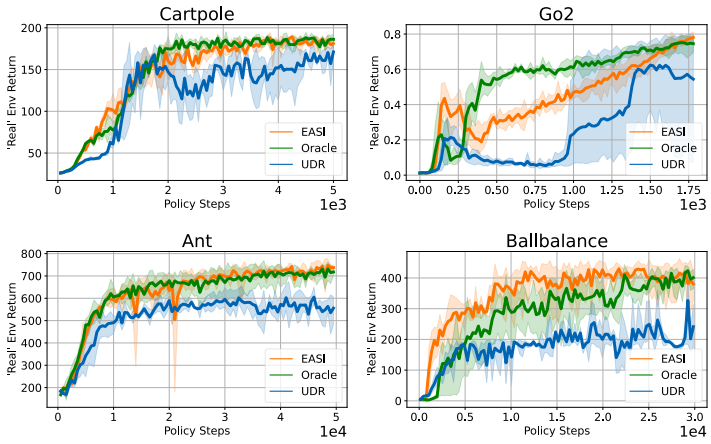


Figure 5: Performance on the pseudo-real environment over the process of training.

distribution is incorrect. This implies that by specifying a rough initial parameter range, EASI can effectively align the parameter distribution within the desired range.

After parameter optimization, we utilize the converged parameter distribution Ξ^* to train new policies. In each task, EASI runs three times with different random seeds to avoid randomness. Figure 5 depicts the policy's performance in the pseudo-real environment throughout the training process. The X-axis represents the number of optimization steps for the policy during training in the simulation, and the Y-axis indicates the policy's performance when tested in the pseudo-real environment. In all tasks, our method outperforms UDR and even demonstrates comparable performance to the policy trained directly in the pseudo-real environment.

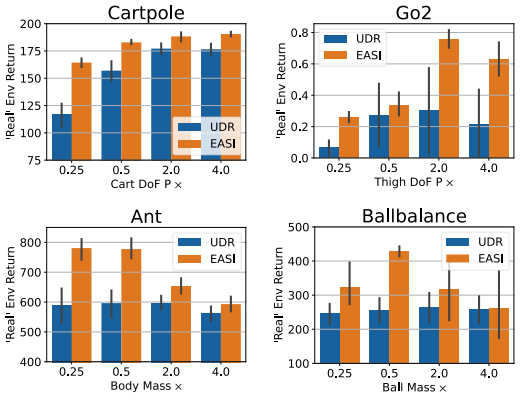


Figure 6: Policy performance in target environments with different parameters.

Subsequently, we conduct a test to evaluate the performance of EASI across various target environments. We fix the initial parameter distribution $\Xi^{(0)} = U[\frac{1}{k} \times \xi_r, k \times \xi_r]$ and adjust one of the parameters ξ_r^i to 1/4, 1/2, 2, and 4 times its original value, denoted as ξ_r'. We choose the proportional parameter of the PD controller for the Cart DoF in Cartpole, the proportional parameter of the PD controller for the Thigh Dof in Go2, the body mass of Ant, and ball mass of Ballbalance as the parameters to be adjusted. We use the simulation with parameter ξ_r' as the target environment and utilize EASI to identify. The performance of trained policies in different target environments are shown in Fig.6. It can be seen that the policies trained with UDR are overly conservative or fail to adapt to the target environment due to the gap between source and target environments. On the contrary, simulators optimized by EASI are more similar to the target environment, thus policies trained

Table 1: Policy performance in target environments with different parameters.

Target Environment	UDR	FineTune	GARAT	EASI
CartPole Cart DoF P $\times$ 0.25	116.8±13.0	135.1±7.8	144.4±1.2	**164.4±4.6**
CartPole Cart DoF P $\times$ 0.5	157.0±10.5	171.2±8.4	161.6±4.5	**182.9±2.2**
Ant Body Mass $\times$ 0.25	588.9±65.9	590.2±18.6	327.7±15.9	**780.7±41.5**
Ant Body Mass $\times$ 0.5	596.4±52.3	640.8±30.0	228.4±19.2	**777.4±37.5**

in these simulations are more adaptable to target environments and achieve higher performance in the target domain. In specific scenarios, we compare EASI with other sim-to-real methods, with detailed results presented in Table 1. The experimental results demonstrate that policies trained with EASI consistently achieve better performance in the target environment, highlighting EASI's effectiveness in facilitating a seamless transfer of policies to the target environment.

5.2 Sim-to-Real Policy Transfer

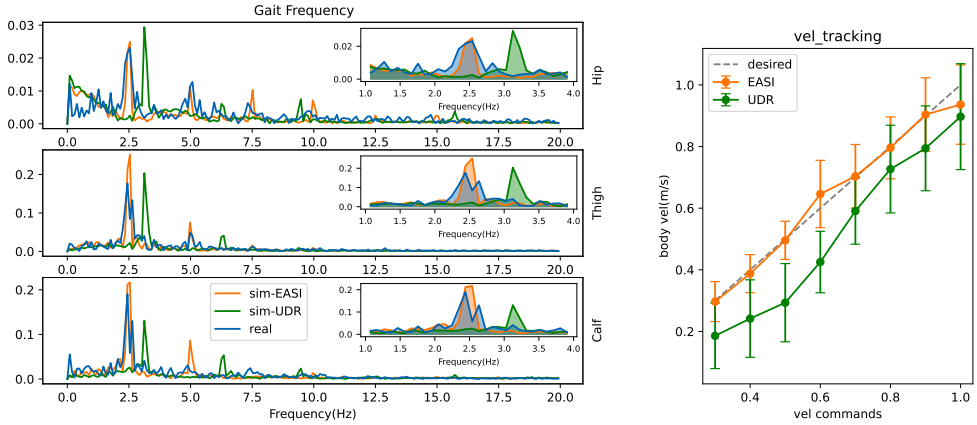
Cartpole In real-world control, using Cartpole as an example, we run the policy on a PC receiving state information from robot and sending action commands to the robot. The control frequency for the Cartpole task is 50Hz. We evaluate the performance of the policy by the angle error of pole and the velocity of cart, which reflect the accuracy and stability of the control policy.

Table 2: Policy performance in reality

Algotirhm	Angle Error$\times 10^{-2}$	Cart Vel $\times 10^{-1}$
UDR	2.65$\pm$0.19	1.63$\pm$0.03
EASI	1.67$\pm$0.16	1.21$\pm$0.08

We train an initial policy using UDR and sample 50 trajectories including approximately 10000 state transitions in real-world environment. For EASI, initial parameter distribution and other algorithm settings remain the same as sim-to-sim tasks. Table 2 presents the performance of different algorithms in real-world Cartpole tasks, and it can be seen that the policies trained in the simulator optimized by EASI achieve more accurate and stable control in the Cartpole task and show significant improvements compared to previous sim-to-real method.

Go2 To further evaluate the performance of EASI in real-world scenarios, we conduct an experiment using the Unitree Go2 quadruped robot. We search for 7 parameters: the PD parameters for the hip, thigh, and calf joints, as well as the body mass. Unlike the sim-to-sim experiment, we do not include ground friction in our search to allow the robot to adapt to various ground types it might encounter. We train the initial policy using UDR in simulation and use the initial policy to collect real-world data, sampling 2000 steps at a control frequency of 50Hz, which provides about 40 seconds of data. Using these real-world samples, we then employ EASI for parameter search.



(a) Movement spectrum of the joint.

(b) Velocity tracking performance.

Figure 7: (a) Using the same policy, there is a significant difference in the motion spectrum of the Go2's joints between the simulation and the real environment due to the reality gap. After optimizing with EASI, the motion spectrum of the Go2's joints in the simulation becomes closer to that in the real environment. (b) Comparison of speed tracking in real environments between policies trained with EASI parameter distribution and trained with UDR parameter distribution.

After parameter search with EASI, the simulator become more realistic. In this experiment, we use the same initial policy and speed command (v=1.4m/s) to control the robot's movement in both

simulation and reality environments and plot the frequency spectrum of the robot's joint movements in Fig. 7(a). It is notable that the final output of EASI is the distribution of simulator parameters Ξ^*. To elucidate the differences in simulation with optimized parameter distribution Ξ^* and original parameter distribution $\Xi^{(0)}$ (consistent with Ξ_{UDR}), we utilize the means of these distributions ξ^* and $\xi^{(0)}$ as the parameters for the simulator, referred to as sim-EASI and sim-UDR, respectively. Despite being controlled by the same policy, there are significant differences in the joint movement frequencies between the sim-UDR and real environments, indicating a substantial reality gap between simulator and the real world. After parameter optimization with EASI, the differences in the joint movement frequency spectrum have been significantly reduced, indicating that the sim-EASI more closely resembles real-world.

Subsequently, the real robot's ability to follow speed commands is tested. Using the parameter distribution Ξ^* obtained from EASI as the simulator parameter distribution to retrain a policy before testing on the real quadruped robot. During the testing phase, we evaluate the robot's performance in the same scenarios using various speed commands. The results in Fig. 7(b) show that policies trained with EASI parameters have better speed tracking capabilities compared to those trained with origin parameters. This also proves that policies trained in the EASI-optimized simulator can perform better in the real world.

5.3 Real World Data Requirement

We design a data budget experiment based on sim-to-sim experiments to test the impact of demonstration data quantity on EASI. In this experiment, we limit the number of trajectories sampled in the target domain to 100, 50, and 1 respectively, and conduct EASI using the limited trajectories.

Table 3: Performance in target environments with varying numbers of reference trajectories

Trajectories	UDR	1	50	100	200
Cartpole	161.3±15.3	185.1±3.1	182.7±2.7	181.1±4.4	182.9±4.3
Ant	557.4±55.6	724.2±7.5	747.0±25.0	703.1±63.1	724.0±31.5
Ballbalance	226.9±61.7	360.1±53.8	421.3±26.1	428.5±16.9	393.1± 60.5
Go2	0.62±0.25	0.44±0.28	0.80±0.06	0.72±0.05	0.78±0.02

The experimental results of the data budget are presented in Table 3. The results demonstrate EASI's data efficiency, as only a small amount of real-world data is needed for EASI to identify parameters that align the simulation closely with the target environment and enhance the policy's adaptability in the real world.

6 Limitations

Even though EASI requires relatively low amounts of real-world data, it still necessitates collecting corresponding state transition data for each specific scenario and maintaining certain standards regarding sensor error ranges. When sensors exhibit significant noise, additional techniques for imitation learning from imperfect demonstrations are needed. Furthermore, even within the same scenario, substantial differences in task styles can reduce EASI's performance. To achieve optimal results, it is recommended to collect expert data separately for each task.

7 Conclusion

In this work, we introduce EASI, a novel sim-to-real method that combines GAN with ES. EASI uses ES as a generator in an adversarial competition with a neural network discriminator to find physical parameter distributions that align state transitions between simulation and reality. The discriminator serves as the fitness function for ES, guiding the evolution of parameter distributions. We conducted a series of experiments to validate EASI's effectiveness and data efficiency in both sim-to-sim and sim-to-real tasks, demonstrating its exceptional performance. We believe EASI is a general sim-to-real method characterized by simplicity, low cost, and high fidelity. In future work, we will evaluate EASI's performance on a broader range of robotic platforms and industrial scenarios.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Natural Science Foundation of China (Nos. 62073160 & 72394363) and the Nanjing University Integrated Research Platform of the Ministry of Education-Top Talents Program.

References

[1] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. Amp: adversarial motion priors for stylized physics-based character control. *ACM Trans. Graph.*, 40(4), jul 2021.

[2] Chenhao Li, Marin Vlastelica, Sebastian Blaes, Jonas Frey, Felix Grimminger, and Georg Martius. Learning agile skills via adversarial imitation of rough partial demonstrations. In *6th Annual Conference on Robot Learning*, 2022.

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

[4] 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 *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, 2017.

[5] Stephen James, Andrew J. Davison, and Edward Johns. Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 334–343. PMLR, 13–15 Nov 2017.

[6] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[7] Adam Allevato, Elaine Schaertl Short, Mitch Pryor, and Andrea Thomaz. Tunenet: One-shot residual tuning for system identification and sim-to-real robot task transfer. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 445–455. PMLR, 30 Oct–01 Nov 2020.

[8] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979, 2019.

[9] Yuqing Du, Olivia Watkins, Trevor Darrell, Pieter Abbeel, and Deepak Pathak. Auto-tuned sim-to-real transfer. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1290–1296, 2021.

[10] Yifeng Jiang, Tingnan Zhang, Daniel Ho, Yunfei Bai, C. Karen Liu, Sergey Levine, and Jie Tan. Simgan: Hybrid simulator identification for domain adaptation via adversarial reinforcement learning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2884–2890, 2021.

[11] Marius Memmel, Andrew Wagenmaker, Chuning Zhu, Dieter Fox, and Abhishek Gupta. ASID: Active exploration for system identification in robotic manipulation. In *The Twelfth International Conference on Learning Representations*, 2024.

[12] Peide Huang, Xilun Zhang, Ziang Cao, Shiqi Liu, Mengdi Xu, Wenhao Ding, Jonathan Francis, Bingqing Chen, and Ding Zhao. What went wrong? closing the sim-to-real gap via differentiable causal discovery. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 734–760. PMLR, 06–09 Nov 2023.

[13] Fabio Ramos, Rafael Carvalhaes Possas, and Dieter Fox. Bayessim: adaptive domain randomization via probabilistic inference for robotics simulators. *arXiv preprint arXiv:1906.01728*, 2019.

[14] Fabio Muratore, Christian Eilers, Michael Gienger, and Jan Peters. Data-efficient domain randomization with bayesian optimization. *IEEE Robotics and Automation Letters*, 6(2):911–918, 2021.

[15] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving rubik's cube with a robot hand, 2019.

[16] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J. Pal, and Liam Paull. Active domain randomization. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1162–1176. PMLR, 30 Oct–01 Nov 2020.

[17] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots, 2021.

[18] Gwanghyeon Ji, Juhyeok Mun, Hyeongjun Kim, and Jemin Hwangbo. Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion. *IEEE Robotics and Automation Letters*, 7(2):4630–4637, 2022.

[19] I Made Aswin Nahrendra, Byeongho Yu, and Hyun Myung. Dreamwaq: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning, 2023.

[20] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, Yunfei Bai, Matthew Kelcey, Mrinal Kalakrishnan, Laura Downs, Julian Ibarz, Peter Pastor, Kurt Konolige, Sergey Levine, and Vincent Vanhoucke. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4243–4250, 2018.

[21] Ignasi Clavera, Anusha Nagabandi, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019.

[22] Xingshuo Jing, Kun Qian, Tudor Jianu, and Shan Luo. Unsupervised adversarial domain adaptation for sim-to-real transfer of tactile images. *IEEE Transactions on Instrumentation and Measurement*, 72:1–11, 2023.

[23] Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12619–12629, 2019.

[24] Josiah P. Hanna and Peter Stone. Grounded action transformation for robot learning in simulation. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 3834–3840. AAAI Press, 2017.

[25] Haresh Karnan, Siddharth Desai, Josiah P. Hanna, Garrett Warnell, and Peter Stone. Reinforced grounded action transformation for sim-to-real transfer, 2020.

[26] Siddarth Desai, Ishan Durugkar, Haresh Karnan, Garrett Warnell, Josiah P. Hanna, and Peter Stone. An imitation from observation approach to transfer learning with dynamics mismatch. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2020. Curran Associates Inc.

[27] K.J. Åström and P. Eykhoff. System identification—a survey. *Automatica*, 7(2):123–162, 1971.

[28] Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification, 2017.

[29] 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, 2018.

[30] Rika Antonova, Jingyun Yang, Priya Sundaresan, Dieter Fox, Fabio Ramos, and Jeannette Bohg. A bayesian treatment of real-to-sim for deformable object manipulation. *IEEE Robotics and Automation Letters*, 7(3):5819–5826, 2022.

[31] Peng Chang and Taşkin Padif. Sim2real2sim: Bridging the gap between simulation and real-world in flexible object manipulation. In *2020 Fourth IEEE International Conference on Robotic Computing (IRC)*, pages 56–62, 2020.

[32] Marija Jegorova, Joshua Smith, Michael Mistry, and Timothy Hospedales. Adversarial generation of informative trajectories for dynamics system identification. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7109–7115, 2020.

[33] Allen Z. Ren, Hongkai Dai, Benjamin Burchfiel, and Anirudha Majumdar. Adaptsim: Task-driven simulation adaptation for sim-to-real transfer. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2023.

[34] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[35] Martin Arjovsky and Leon Bottou. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*, 2017.

[36] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 214–223. JMLR.org, 2017.

[37] Max Schwenzer, Muzaffer Ay, Thomas Bergs, and Dirk Abel. Review on model predictive control: An engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117(5):1327–1349, 2021.

[38] Rakesh P Borase, DK Maghade, SY Sondkar, and SN Pawar. A review of pid control, tuning methods and applications. *International Journal of Dynamics and Control*, 9:818–827, 2021.

[39] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021.

[40] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.

[41] Huiqiao Fu, Kaiqiang Tang, Yuanyang Lu, Yiming Qi, Guizhou Deng, Flood Sung, and Chunlin Chen. Ess-infoGAIL: Semi-supervised imitation learning from imbalanced demonstrations. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[42] Nikolaus Hansen. The cma evolution strategy: A tutorial, 2023.

[43] Daan Wierstra, Tom Schaul, Jan Peters, and Juergen Schmidhuber. Natural evolution strategies. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 3381–3387, 2008.

[44] Hans-Georg Beyer. *The Theory of Evolution Strategies*. Springer Berlin Heidelberg, 2001.

Supplementary

A Full parameter search

To explore the performance of EASI as the number of simulation parameters increases, we conducted a sim-to-sim experiment with the Go2 environment, searching for 25 parameters. In the experiment, we searched for the PD parameters of all motors in the Go2 robot and the mass of the robot. The results of the parameter search are shown in Fig. 8 (Due to space limitations, 10 out of 25 results are displayed).

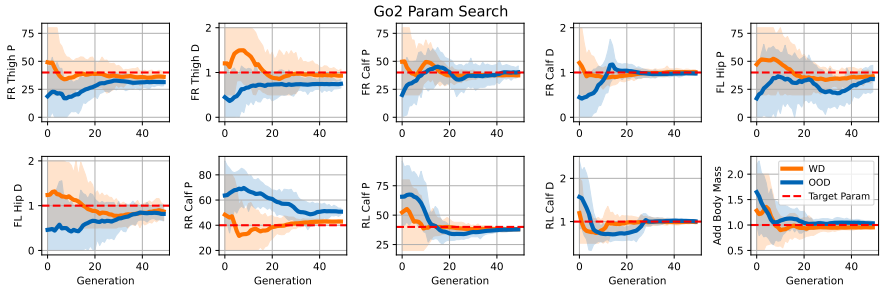


Figure 8: The convergence process for different parameters being searched in the Go2 environment. WD means target parameter within initial parameter distribution, while OOD means out of initial parameter distribution.

B Hyperparameter sensitivity analysis

Since EASI uses Evolutionary Strategies (ES) as the simulation parameter generator, it is much more stable in the training process compared to the original GAN algorithm and less sensitive to hyperparameters. In EASI, we select two hyperparameters for sensitivity analysis in Table 4 and Table 5 ,the evaluation metric is the L2 error between the searched parameter and the target parameters.

We first analyze the hyperparameter μ/λ. In each evolution, we test λ individuals and select the best-performing μ elite individuals for recombination and mutation. Generally, a higher μ/λ value helps maintain diversity in the population but may slow down the evolution process. Conversely, if μ/λ is too low, a lack of diversity can lead to local optima. To enhance parameter search, we recommend using a larger μ/λ and increasing the number of evolution steps, which will improve EASI's performance but also increase computational time.

Next, we analyze another hyperparameter epoch_disc, which refers to the number of times the discriminator is trained before each generation of evolution. Insufficient training may prevent the discriminator from achieving optimal performance, while excessive training can cause overfitting. In our experiments, we found that varying epoch_disc had only a minor effect on the final search results.

Table 4: Hyperparameter sensitivity analysis for EASI (μ/λ).

μ/λ	0.1	0.3	0.5	0.7	0.9
Param Search Error	1.35±0.03	0.63±0.11	0.34±0.20	1.99±0.13	1.44±0.04

Table 5: Hyperparameter sensitivity analysis for EASI (epoch_disc).

epoch_disc	2	5	10	15	20
Param Search Error	0.60±0.08	0.19±0.14	0.34±0.20	0.82±0.06	0.46±0.04

C Broader Impacts

In our work, we have proposed EASI for simulation trained strategy transfer to reality. This could potentially drive the application of robotics technology in the real world, thereby positively impacting societal operational efficiency. Sim-to-real techniques reduce the need for extensive physical trials, minimizing costs associated with hardware testing and prototyping. This can accelerate the development of robotic technologies, making them more accessible to a broader range of industries.

However, there is a risk that developers may become overly reliant on simulation results, potentially leading to systems that perform well in simulated environments but fail in the real world due to unmodeled complexities. Additionally, it is important to recognize that the capabilities of simulations have limits, and we cannot expect simulations to perfectly mirror reality. Resources should still be invested in designing better robot structures and developing more advanced simulators.

Moreover, there are growing concerns about the computational resources required for current training schemes. The increasing demand for computational power undoubtedly consumes significant amounts of energy and resources. We are continually exploring more efficient algorithms to address this challenge.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the main contributions, including the key method and scope of the research, which are supported by the theoretical and experimental results presented in the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a dedicated "Limitations" section where we discuss the potential weaknesses of our approach.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper primarily focuses on the innovative combination of existing methods and their application in sim-to-real scenarios, and does not involve proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The method proposed in this paper is easy to reproduce, and we introduce detailed information of method implementation and experiment settings in Section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We are in the process of organizing the experiment code and we will provide open access to well-documented code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: All relevant training and test details are described in Sections 5.1, 5.2, 5.3, and supplementary.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Error bars are reported for key results. All figures and tables include error bars capturing the variability due to different random seeds.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper specifies the type of compute resources used, including CPU and GPU specifications, etc. These details are provided in Section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research adheres to the NeurIPS Code of Ethics, with careful consideration given to ethical standards throughout the study.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The potential societal impacts, both positive and negative, are discussed in the Appendix C.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The method proposed in this paper is mainly applied to robot control tasks and poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All external assets used in this research are properly credited, and their licenses and terms of use are respected. This information is provided in the references and acknowledgment sections.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: New assets, including datasets and code, are thoroughly documented, and this documentation is provided alongside the assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.