DEAL: Diffusion Evolution Adversarial Learning for Sim-to-Real Transfer

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

Abstract

Training Reinforcement Learning (RL) controllers in simulation offers cost-efficiency and safety advantages. However, the resultant policies often suffer significant performance degradation during real-world deployment due to the reality gap. Previous works like System Identification (Sys-Id) have attempted to bridge this discrepancy by improving simulator fidelity, but encounter challenges including the collapse of high-dimensional parameter identification, low identification accuracy, and unstable convergence dynamics. To address these challenges, we propose a novel Sys-Id framework that combines Diffusion Evolution with Adversarial Learning (DEAL) to iteratively infer physical parameters with limited real-world data, which makes the state transitions between simulation and reality as similar as possible. Specifically, our method iteratively refines physical parameters through a dual mechanism: a discriminator network evaluates the similarity of state transitions between parameterized simulations and target environment as fitness guidance, while diffusion evolution adaptively modulates noise prediction and denoising processes to optimize parameter distributions. We validate DEAL in both simulated and real-world environments. Compared to baseline methods, DEAL demonstrates state-of-the-art stability and identification accuracy in high-dimensional parameter identification tasks, and significantly enhances sim-to-real transfer performance while requiring minimal real-world data.

1 Introduction

In recent years, Reinforcement Learning (RL) controllers have achieved significant success in robotic control through iterative trial-and-error learning [1–5]. This enables robust control of complex robots even in unstructured environments. However, such learning processes demand extensive data collection, and conducting training directly in the real world poses challenges related to efficiency and safety. Simulation-based training offers a practical alternative by enabling large-scale data generation in parallelized and risk-free environments [6]. Despite this, policies trained in simulation often experience substantial performance degradation when deployed on real-world systems due to the sim-to-real gap.

To mitigate the performance degradation caused by the sim-to-real gap, previous research has explored various sim-to-real transfer techniques [7]. Among these, Domain Randomization (DR) is one of the most widely used approaches [8–12]. It improves policy robustness by randomizing environmental and robotic physical parameters within predefined ranges during training and injecting noise into observations and actions. This paradigm aims to expose agents to sufficiently diverse simulated scenarios, thereby enhancing robustness and facilitating zero-shot transfer to reality. However, DR

*Corresponding authors

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

often relies heavily on expert knowledge to define suitable randomization ranges, tends to produce overly conservative behaviors [13], and may lead to unstable or prolonged training. Recent advances in simulator fidelity optimization [14–22] have introduced few-shot approaches that leverage limited real-world data to learn target environment parameter distributions, thereby narrowing the reality gap and aligning simulator physical dynamics with the reality. Nevertheless, these approaches still encounter challenges including high-dimensional system identification collapse, low identification accuracy, inherent instability during the alignment process.

To address these challenges, we introduce a novel Sys-Id framework for sim-to-real transfer, which innovatively combines Diffusion Evolution with Adversarial Learning (DEAL) to narrow the reality gap. As diffusion models have demonstrated exceptional capabilities in image and video synthesis [23–27], existing works primarily utilize them as high-dimensional data generators. In DEAL, Diffusion Evolution (DE) [28] is employed both as a high-dimensional optimizer for parameter search and as a powerful generator within a Generative Adversarial Network (GAN) [29]. This dual role enables the generation of physical parameters that bring simulated state transitions closer to those observed in the real world. Meanwhile, a discriminator network evaluates each evolving simulator by scoring its fitness based on the similarity between simulated state transitions and real-world demonstrations, thereby guiding the evolution of parameter distributions. DEAL builds upon this adversarial learning framework but avoids the need for costly real-world data to train a noise prediction module, as required in traditional diffusion models. Instead, it estimates optimal parameters by weighting the entire parameter population according to their fitness scores, predicts noise based on the current parameters and the estimated optimum, and approaches the target parameters through an iterative denoising process. To further improve performance, DEAL introduces automatic noise adaptation to balance exploration and exploitation and applies parameter normalization to reduce crossover errors caused by parameter scale imbalances. These enhancements enable DEAL to achieve stable and accurate high-dimensional parameter identification, significantly improving sim-to-real transfer with minimal real-world data and reduced computational cost.

In the experiments, we evaluate DEAL on five sim-to-sim tasks (AllegroHand, Humanoid, Go2, Cartpole, Ant) and two sim-to-real tasks (Cartpole, Go2). First, we evaluate DEAL's parameter identification capabilities, particularly in high-dimensional settings. Using a policy trained with Uniform Domain Randomization (UDR), we collect demonstrations in target environment and conduct parameter searches with DEAL to redefine the simulator. The policy is then retrained in the enhanced simulator and its transfer performance is tested in the target domain. We further assess DEAL's adaptability by expanding the search scale and analyzing its dependence on target-domain demonstrations. Finally, we complete the challenging sim-to-real transfer task. Experimental results show that DEAL achieves state-of-the-art stability and identification accuracy in high-dimensional parameter identification tasks, effectively bridging the sim-to-real gap with limited real-world data. In particular, the contributions of this work are threefold:

1. We introduce a novel methods, DEAL, which innovatively combines Diffusion Evolution with Adversarial Learning to narrow the reality gap.
2. We develop a automatic noise adaptation mechanism to balance exploration and exploitation, and propose a parameter normalization framework to counteract search errors caused by parameter scale imbalances.
3. We demonstrate the effectiveness of DEAL in both sim-to-sim and sim-to-real tasks, showing superior performance compared to existing baselines.

2 Related Work

Transferring simulation-trained policies to the reality in a stable and cost-effective manner has long been a goal for sim-to-real research. Previous work has approached sim-to-real transfer from both policy adaptation and system identification perspectives.

Policy adaptation optimization focuses on training policies that are robust to dynamics discrepancies without relying on real-world fine-tuning. Domain randomization (DR) is a foundational technique that injects randomness into simulation parameters to encourage generalization [8–10, 13]. However, DR often requires manual tuning of randomization ranges, leading to suboptimal performance or excessive training cost. Recent advancements mitigate these limitations by integrating curriculum learning [30] or GAN-guided subspace prioritization [31], Dropo further improves upon DR by

leveraging off-policy data or human demonstrations to learn a more effective randomization distribution [11]. Other methods introduce latent adaptation modules [32, 33] and employ Concurrent Policy Optimization [34] to estimate environment variables in real time, but these approaches struggle with high-dimensional or partially observable dynamics. Domain adaptation techniques, such as image-to-image translation [35], grounded action transformation (GAT) [36–38] and ASAP [39], aim to align simulation-reality action spaces or visual inputs. However, they typically require costly real-world data and often lack generalizability across tasks.

System identification optimization calibrates simulation parameters to match real-world dynamics using limited real-world trajectories. Inspired by classical Sys-ID frameworks [40–42], recent work has focused on identifying simulation parameters that better align with real-world dynamics. Bayesian approaches like BayesSim [21] and its extensions [22, 43] formalize system identification as an inference problem, iteratively estimating posterior distributions over simulation parameters. However, these methods suffer from high computational costs and poor scalability in high-dimensional parameter spaces. Some data-efficient approaches adopt alternative strategies, ASID [19] designs exploration policies to collect informative real-world trajectories, while TuneNet [15] employs supervised learning to map simulation trajectories to parameter gradients, bypassing iterative optimization. RL-based methods reframe the parameter search problem as a policy learning task, where trajectories serve as states and parameters as actions [17, 44], these methods struggle with sparse rewards in complex tasks. For dynamics alignment, adversarial training frameworks like SimGAN [18] and EASI [14] utilize a discriminator to distinguish simulated and real-world state transitions. EASI optimizes by selecting fixed elite parameters through manual sorting in evolutionary search, ignoring the information of the entire population. This reduces parameter population diversity and leads to a tendency to get trapped in local optima, which limits both the amount and accuracy of the searched parameters and often produces multiple sub-optimal solutions in system identification. In contrast, DEAL employs soft probabilistic weighting over the entire parameter population, preserving diversity and improving convergence stability. Its iterative denoising process continuously refines parameters without relying on a noise prediction module, enabling stable and precise high-dimensional parameter search with minimal real-world demonstrations and lower computation cost, while effectively enhancing transfer capabilities in complex robotic tasks.

3 Approach

In this section, we first provide a brief background to DE and GAN, then describe in detail how DEAL integrates DE with GAN, and demonstrate how it narrow the reality gap by aligning simulator's physcial parameters using limited real-world data.

3.1 Background

Diffusion Evolution Diffusion models [24, 25] operate in two phases: diffusion and denoising. In the diffusion phase, Gaussian noise is progressively added to the original data, while a noise prediction module ϵ_θ is trained to predict the added noise. The denoising phase iteratively estimates the original data and performs directed denoising to recover it. Diffusion Evolution [28] (DE) reinterprets this framework through evolutionary principles: denoising mimics evolution, diffusion emulate reversed evolution and add random perturbations act as mutations. This algorithm uses the iterative denoising process from diffusion models to refine solutions in a parameter space, instead of recovering a data distribution, it shifts an initial random population toward an optimized solution distribution. Similar to the relationship between energy and probability in statistical physics, evolutionary tasks can be connected to generative tasks by mapping fitness to probability density, diffusion models are directly predicting the original data x_0 from noisy versions of x_0 at each time step, DE can estimate the optimal point by weighting the current population individuals x based on the corresponding fitness probabilities:

$$\hat{x}_0(x_t, \alpha, t) = \frac{1}{Z} \sum_{x \in X_t} g[f(x)] \mathcal{N}(x_t; \sqrt{\alpha_t} x, 1 - \alpha_t) x, \tag{1}$$

where x_t is the corrupted data at timestep $t \sim [0, T]$, diffusion schedule α_t governs the noise , x is a sample from the current data distribution X_t, the probability mapping function $g(\cdot)$, typically implemented as Softmax, transforms fitness evaluated by $f(\cdot)$ into probabilities, and the Gaussian distribution indicates the conditional probability of the current data point given any sample point as

the optimum, the weighted results of these are normalized by Z. Given the design of the diffusion process, i.e., $x_t = \sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\epsilon$, the noise ϵ can be estimated without the need for noise prediction module ϵ_θ by:

$$\hat{\epsilon}(x_t, \alpha, t) = \frac{x_t - \sqrt{\alpha_t}\,\hat{x}_0(x_t, \alpha, t)}{\sqrt{1-\alpha_t}}. \tag{2}$$

Under DDIM [25] framework, this step-by-step denoising process can be described as:

$$x_{t-1} = \sqrt{\alpha_{t-1}}\hat{x}_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\hat{\epsilon} + \sigma_t\omega, \tag{3}$$

where ω is the random perturbations controlled by the noise schedule σ_t in denoising phase. More proofs about DE can be found in the Appendix A.1.

Generative Adversarial Network The Generative Adversarial Network (GAN) [29] framework employs two neural networks — a generator G and a discriminator D — that engage in an adversarial training process to synthesize data matching the statistical properties of the training distribution. These networks optimize through a minimax game defined by:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z\sim p_z(z)}[log(1-D(G(z)))], \tag{4}$$

where $p_z(z)$ denotes the noise prior distribution, and x represents samples from the real data distribution p_{data}. Through iterative updates, G learns to generate synthetic data indistinguishable from real samples under D's evaluation, thus achieving distributional alignment. For our specialized objective, we use WGAN [45] in this work, the discriminator optimization objective is formulated as:

$$\max_D \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim d^{\mathcal{T}}}[D(\mathbf{s},\mathbf{a},\mathbf{s}')] - \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim d^{\mathcal{S}}}[D(\mathbf{s},\mathbf{a},\mathbf{s}')], \tag{5}$$

where $d^{\mathcal{S}}$ and $d^{\mathcal{T}}$ denote trajectories from the simulator and reality respectively, this objective incentivizes D to discriminate between simulator and reality state-action transitions dynamics. The discriminator used in DEAL adopts a fully connected MLP with input $(\mathbf{s},\mathbf{a},\mathbf{s}')$, two hidden layers of 256 units with ReLU activation, and a scalar output. To further stabilize the training process, we use Weight Clipping [45] to located the weights of discriminator in a compact interval to satisfy the Lipschitz continuity condition, ensuring the effective computation of the Wasserstein distance.

3.2 DEAL

The schematic overview of the DEAL architecture is shown in Fig. 1, and the pseudo-code is shown in Algorithm 1. Through adversarial co-optimization, this framework progressively minimizes the distribution discrepancies until convergence, achieving a calibrated simulator that becomes indistinguishable from the reality to the discriminator.

Diffusion Evolution Adversarial Learning In this work, by modeling the processes in reality and simulation as MDPs, the system identification objective is to narrow the gap between the state transitions of the target (reality) and source (simulator) domains. Due to different physical dynamic parameters, there's a reducible reality gap between the target domain state transition distribution $\mathcal{P}_t(\theta^{target})$ and the source domain one $\mathcal{P}_s(\theta)$, θ denotes the physical parameters of the robot or the external environment, as detailed in Appendix A.7. Our method aims to search a parameter distribution θ which is modeled as a Gaussian distribution to minimize this gap. Specifically, the goal of DEAL is to minimize the discrepancy between state transition distributions from different domains as follows:

$$\min_{\theta\in\mathcal{U}} \|\mathcal{P}_t(\theta^{target}), \mathcal{P}_s(\theta)\|. \tag{6}$$

The similarity of state transitions between parameterized simulations and reality can be measured by a fitness function $f(\cdot)$, which is replaced by a discriminator network in this work. The discriminator evaluates how similar the state transitions in the trajectories are to those in the target domain and gives a corresponding score based on the level of similarity. With DE serves as the generator and combined with Equation (5), the overall search objective can be described as:

$$\theta^* = \arg\min_{\theta\in\mathcal{U}} \max_D \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim d^{\mathcal{T}}(\theta^{target},\pi_0)}[D(\mathbf{s},\mathbf{a},\mathbf{s}')] - \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim d^{\mathcal{S}}(\theta,\pi_0)}[D(\mathbf{s},\mathbf{a},\mathbf{s}')], \tag{7}$$

where $d^{\mathcal{S}}(\theta, \pi_0)$ represents the trajectory collected by π_0 in the simulator parameterized by θ. In this adversarial process, DE acts as the generator to generate and optimize the θ, making the simulation trajectory increasingly similar to the real world trajectory $d^{\mathcal{T}}(\theta^{target}, \pi_0)$ from the discriminator's perspective, while the discriminator aims to distinguish between them as much as possible.

In this frame, firstly, we randomly sample physical parameters θ_T from the initial search range $\mathcal{U}$ to initialize each simulator, then use the policy π_0 trained via UDR to collect trajectories that reflect the current environment's dynamics. After sampling state-action transition $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ sequence $b^{\mathcal{T}}$ and $b^{\mathcal{S}}$ from the target and source domain trajectories respectively, then fed them to the discriminator to evaluate each parameterized simulator's fitness, which represent the similarity of state transitions between the evolving simulator and target environment, it is used to update discriminator according to Equation (5) and guide DE for parameter updates. During the parameter update phase, by mapping the fitness through Softmax to obtain a probability distribution, DE can estimates the optimal parameters $\hat{\theta}_0$ by weighting the current population individuals based on the corresponding fitness probabilities and predicts noise $\hat{\epsilon}$ used for directed evolution. Finally, according to Equation (3), the next generation of parameters can be generated by:

$$\theta_{t-1} = \sqrt{\alpha_{t-1}}\hat{\theta}_0 + \sqrt{1 - \alpha_{t-1} - \sigma_t^2}\hat{\epsilon} + \sigma_t \omega. \tag{8}$$

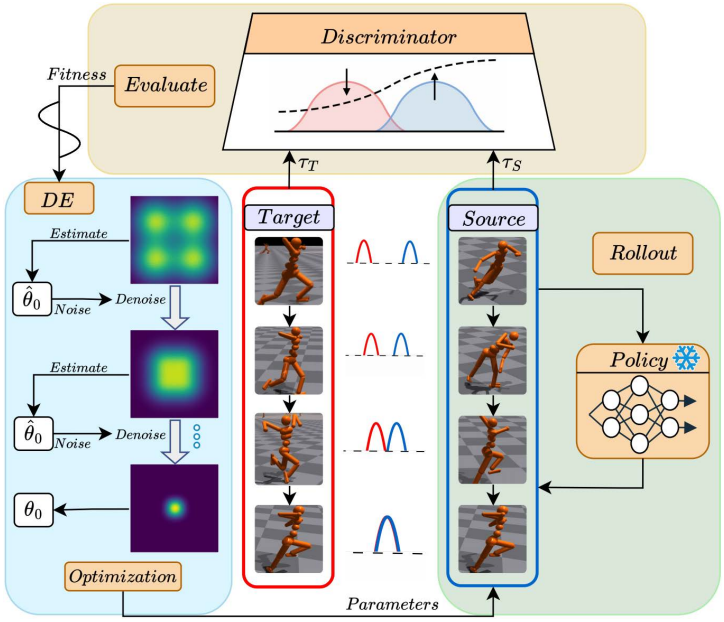

Figure 1: Schematic overview of DEAL. The framework iteratively optimizes parameters through a dual mechanism: a discriminator evaluates the similarity of state transitions sampled in source domain trajectories τ_S and target domain trajectories τ_T as fitness, while DE estimates the optimal parameter $\hat{\theta}_0$ based on the fitness probabilities, adaptively updates noise predictions $\hat{\epsilon}$ and performs denoising to optimize parameter distributions until convergence.

Automatic Noise Adaptaion and Parameter Normalization To encourage DEAL to search for the optimum, we use Equation (9) to finetune the noise impact factor δ base on fitness, then we let $\hat{\sigma}_t = \delta\sigma_t$ to adaptively adjust the noise schedule σ_t, which can influence the current random perturbation ω in search process. Specifically, this method adjusts the level of ω to balance exploration and exploitation, when the parameter population has low fitness, it increases ω to encourage exploration and generate more mutations in search of better optima. Conversely, when fitness is high, indicating the population is in a good evolutionary state, it reduces ω to focus on exploitation of the current optimal region.

$$\delta = \lambda \cdot \exp\left(-\mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim d^{\mathcal{S}}(\mathbf{s},\mathbf{a},\mathbf{s}')}[D(\mathbf{s},\mathbf{a},\mathbf{s}')]\right) \text{ and } \hat{\sigma}_t = \delta\sigma_t, \tag{9}$$

where $d^{\mathcal{S}}$ represents trajectories collected in source domain and λ is the influence coefficient, D is the discriminator quantifies the fitness of each evolving simulator. Combined with Equation (8), the

Algorithm 1: DEAL

Input: Population size N, Parameter dimension K, Search steps T, Sample batch size M, UDR policy π_0, Search range $\mathcal{U}$, Target domain demonstration $\mathcal{D}_\mathcal{T}$, Source domain demonstration $\mathcal{D}_\mathcal{S}$, Diffusion schedule α_t, Noise schedule σ_t, Noise factor δ, Random perturbations ω, Probability mapping function $g(\cdot)$.

Output: Optimal parameter distribution θ^*= θ^0

Initialize parameters: $\theta^T = [\theta_1^T, \theta_2^T, ..., \theta_N^T] \leftarrow \mathcal{U}$

for $t \in [T, T-1, ..., 1]$ **do**

  $\forall j \in [1, N]$: Store $Rollout(\pi_0, Sim(\theta_j^{(t)}))$ in $\mathcal{D}_\mathcal{S}$

  for *update and clip network weights step* $= 0, 1, 2, \cdots, n$ **do**

    $b^\mathcal{T} = (\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i)_{i=1}^M \leftarrow Sample(\mathcal{D}_\mathcal{T})$, $b^\mathcal{S} = (\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i)_{i=1}^M \leftarrow Sample(\mathcal{D}_\mathcal{S})$

    Update D according to Equation (5) using $b^\mathcal{T}$ and $b^\mathcal{S}$

  end

  Evaluate fitness and Map to Probability:

  $\forall j \in [1, N]$: $fitness = \mathbb{E}_{\tau_j \sim Rollout(\pi_0, Sim(\theta_j^{(t)}))}[D(\mathbf{s}, \mathbf{a}, \mathbf{s}')]$, $p_j \leftarrow g(fitness)$

  Calculate noise factor δ and Adjust noise schedule σ_t according to Equation (9)

  for *env* $j = 1, 2, \cdots, N$ **do**

    $Z \leftarrow \sum_{i=1}^N p_i \cdot \mathcal{N}(\theta_t^{(j)}; \sqrt{\alpha_t}\theta_t^{(i)}, 1-\alpha_t)$

    $\hat{\theta}_0 \leftarrow \frac{1}{Z}\sum_{i=1}^N p_i \cdot \mathcal{N}(\theta_t^{(j)}; \sqrt{\alpha_t}\theta_t^{(i)}, 1-\alpha_t)\theta_t^{(i)}$

    $\hat{\epsilon} \leftarrow \dfrac{\theta_t^{(j)} - \sqrt{\alpha_t}\hat{\theta}_0}{\sqrt{1-\alpha_t}}$, $\omega \leftarrow \mathcal{N}(0, I^K)$

    $\theta_{t-1}^{(j)} \leftarrow \sqrt{\alpha_{t-1}}\hat{\theta}_0 + \sqrt{1-\alpha_{t-1}-\hat{\sigma}_t^2}\hat{\epsilon} + \hat{\sigma}_t\omega$

  end

end

generation equation can be rewritten as:

$$\theta_{t-1} = \sqrt{\alpha_{t-1}}\hat{\theta}_0 + \sqrt{1-\alpha_{t-1}-\hat{\sigma}_t^2}\hat{\epsilon} + \hat{\sigma}_t\omega. \tag{10}$$

This mechanism balances exploration and exploitation for DEAL, encouraging thorough parameter space search while converging toward high-fidelity simulations.

Previous learning based Sys-Id methods [14, 16, 18, 22] have crossover errors due to the parameter scale imbalances. For instance, when searching for motor 's stiffness and damping, their values can differ by tens to hundreds of times, such parameters have unbalanced values and the optimization step lengths don't match may lead to crossover errors during system identification. In this work, we normalize the parameters within their initial search range $\mathcal{U}$, this approach allows for a percentage based adjustment of parameters, effectively reducing search errors caused by parameter scale mismatches.

4 Experiments

In this section, we compare DEAL with several baselines across multiple tasks to demonstrate its effectiveness. The key aspects are as follows:

1. We search for parameters in multiple tasks, including high-dimensional scenarios, then compare the search errors of DEAL and other baselines.
2. After enhancing the simulator across various tasks, we evaluate the transfer performance of the retrained policies using DEAL and other baselines.
3. We search for parameters with larger search scales to examine the search adaptability and assess the demonstration requirements of DEAL and other baselines.
4. Finally, we complete the challenging sim-to-real transfer and demonstrate the performance improvements brought by DEAL.

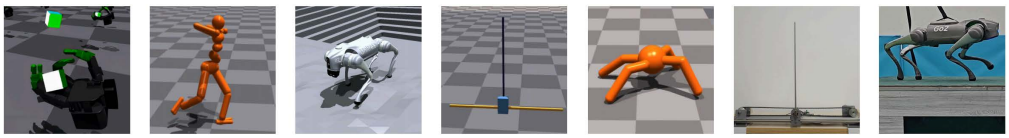

Figure 2: Experiment tasks in simulation (1∼5): AllegroHand, Humanoid, Go2, Cartpole and Ant. Experiment tasks in reality (6∼7): Cartpole, Go2. Presented in order from left to right.

Task Setup We test DEAL in 5 sim-to-sim tasks (AllegroHand, Humanoid, Go2, Cartpole, Ant) and 2 sim-to-real task (Cartpole, Go2), all environments are shown in th Fig. 2. **(1)AllegroHand** environment is a AllegroHand robot designed to interact with a cube, it features 16 DoF distributed across the hand and fingers. The task is to use the hand to manipulate an object to align its orientation with the target orientation. **(2) Humanoid** environment contain a humanoid robot has 21 DoF, with 3 for the abdomen, 6 each for the right and left legs, and 3 each for the right and left arms. Its tasks include walking stably and performing various movements to achieve specific speeds/directions. **(3) Go2** environment is a 12-DoF Unitree GO2 quadruped, with each leg having 3-DoF. The task is to follow randomly chosen x, y, yaw target velocities and climb platforms. **(4) Cartpole** environment includes an inverse pendulum connected to a 1-DoF cart. The task is to keep the pendulum on the cart balanced for as long as possible. **(5) Ant** environment is an 8-DOF quadrupedal robot consisting of four legs attached to a common base. The task is controlling the ant run as fast as possible.

Experiment Detail Our experiments employed Isaac Gym [46] as the simulator. In Isaac Gym, we can parallelly collect trajectories in hundreds of environments with different parameters which means we could evaluate hundreds of parameters parallelly. During parameter search, we instantiat 200 parallel environments to assess reality-to-simulation alignment across varying physical parameters. For Cartpole, Ant, Humanoid and AllegroHand, we implement Soft Actor-Critic (SAC) [47] to train a neural network as RL controller. For Go2, we adopt RMA [33] to develop controllers tracking velocity commands and climbing platforms, and train canter controllers using Ess-InfoGAIL [48] for bio-inspired running and command following. In our experiment, running on a PC equipped with Intel i5-14600KF and RTX 4060 Ti, DEAL can complete the entire search process within a few minutes, the search computation cost can be found in the Appendix A.2. Additionally, we provide the simulation data budgets for each task, the detailed numbers are shown in Appendix A.3.

Baselines We design various baselines to demonstrate the efficiency of our method. **(1) EASI [14]**: A learning-based Sys-Id method combines CMA-ES [49] and GAN [29]. **(2) Bayes Optimization (BO)**: Modification of Bayesian Sys-Id approach—BayRn [22] in search tasks with its return evaluated by a discriminator. **(3) DEAL\FN**: DEAL without using Automatic Noise Adaptation base on fitness **(4) DEAL\PN**: DEAL without using Parameter Normalization **(5) Uniform Domain Randomization(UDR)**: Uniformly sampling parameters from $\mathcal{U}$ at the beginning of each training iteration. **(6) Oracle**: Direct training in the target environment in sim-to-sim experiments, representing the ideal upper bound for sim-to-sim transfer tasks.

4.1 Parameter Search Capability

In this experiment, we selected five tasks and conducted a thorough parameter search for each. The parameter list is detailed in the Appendix A.7 and mainly covers friction coefficients, restitution coefficients, masses of rigid bodies and motor properties such as stiffness, damping and friction. When the parameters of target domain are unknown, we typically train a policy across a wide range of simulation parameters in hope that the policy can thus handle possible real-world variations in dynamics or observations. In this experiment, the UDR policy π_0 of Ant and Go2 was trained within the range of [1/3$\times\theta_t$,3$\times\theta_t$], the remain tasks were trained within the range of [1/5$\times\theta_t$,5$\times\theta_t$], where θ_t denoting the parameters of the target domain. The initial search range $\mathcal{U}$ for parameters of each task is the same as its training range, we performed 50 steps of parameter search for each task with limited 'real-world' demonstration collected by π_0. To avoid randomness, we presented the experimental outcomes using the average values from multiple random seeds. The evaluation metric is the average percentage error of the searched parameters relative to the true parameters, defined as follows:

$$\epsilon_p = \mathbb{E}\left(\left|\frac{\theta - \theta^{target}}{\theta^{target}}\right|\right). \tag{11}$$

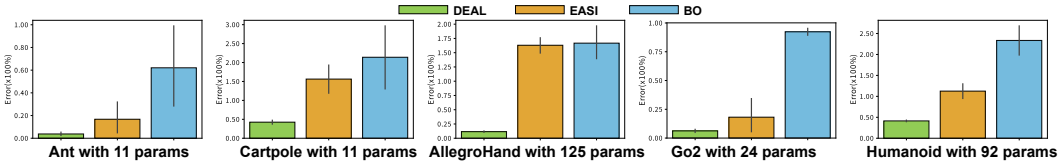

Figure 3: Average search errors for each method, consisting of the error percentage of all parameters.

As shown in Fig. 3, when comparing the final search results of the three methods with the target parameter errors, it is evident that DEAL exhibits significantly higher search accuracy than other baselines across all environments, especially in high-dimensional environments. The baselines often suffer from high-dimensional parameter identification collapse and fail to converge to precise solutions. Despite using less computation cost and the same demonstration in this experiment, DEAL achieve optimal performance which demonstrates its powerful search capability and stable performance.

4.2 Sim-to-Sim Transfer

In this experiment, we utilize the simulator updated from the previous experiment for new policy training, then transfer these newly trained policies to target environment to compare the performance with UDR and Oracle. Notably, the target environment in this experiment is the simulator set with the true target parameter. This is done to compare the performance improvements brought by various enhanced simulators to the policy. As shown in Fig. 4 left, the simulator improved by DEAL delivers the greatest performance boost to newly trained policies for all tasks, it is close to or only slightly inferior to the performance of Oracle whose policy is trained directly in the target domain. Moreover, in high dimensional tasks like Humanoid and AllegroHand, due to the high-dimensional parameter identification collapse of the baselines, the simulator undergoes negative optimization, the policies trained in such simulator perform worse than those trained by UDR and may even cause training to crash, fail to converge, and lead to abnormal behavior. For Go2 task, after training an initial policy

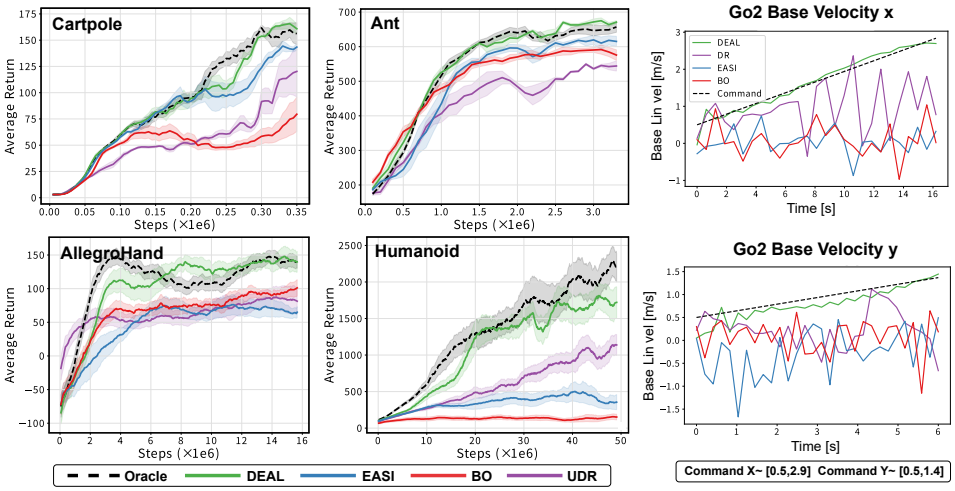

Figure 4: **Left**: The average return in the target environment for the policies trained at each stage with DEAL and other sim-to-real baselines. **Right**: The speed tracking display of Go2 under training with DEAL and other sim-to-real baselines. up: speed in the robot's x-axis direction, down: speed in the robot's y-axis direction.

to collect target-domain trajectories, we retrain the policy in the simulator optimized by DEAL and other baselines. Then, we compare the adaptability of the final policy in target environment, this experiment examines the speed-tracking capability of policies, whether the controller can control the Go2 robot to move forward in the specified direction and speed, and to ascend steps. As shown

in Fig. 4 right, DEAL achieves the best performance as the robot's movement accurately tracks the speed commands in both forward and vertical directions. However, the policies derived from other baselines fail to sustain tracking of velocity commands when they are progressively increased, ultimately leading to more severe speed collapse and destabilizing oscillations in robot motion than UDR. This experiment demonstrates DEAL significantly enhances transfer performance across all tasks.

4.3 Parameter Search Adaptability and Data Requirements

Table 1: Average search error percentage ($\times 100\%$) (See Appendix A.8 for error bars).

	Cartpole			Humanoid			AllegroHand		
Method	ξ =10	ξ =15	ξ =20	ξ =10	ξ =15	ξ =20	ξ =10	ξ =15	ξ =20
DEAL	**0.85**	**1.74**	**2.63**	**0.81**	**1.65**	**2.50**	**0.83**	**1.69**	**2.57**
DEAL\PN	1.90	2.85	3.76	2.40	3.53	4.58	2.60	3.80	4.92
DEAL\FN	1.67	2.96	4.28	1.57	2.79	4.03	1.59	2.80	4.06
EASI	4.03	6.12	8.40	2.94	4.64	6.03	4.25	6.82	9.32
BO	5.20	8.52	11.78	4.67	7.29	10.06	3.75	6.32	9.16

In this experiment, we search for parameters on larger search scales and determine the requirements for demonstration of DEAL and other baselines. As shown in Table 1, denoting θ_t as the target parameters, DEAL attain the minimal search errors in the initial search ranges of $[\frac{1}{\xi} \times \theta_t, \xi \times \theta_t]$ across all tasks, and maintain estimation accuracy even under severely limited parameter distribution priors and extreme initial values during large-scale searches. Notably, UDR policies trained with a range of $[\frac{1}{10} \times \theta_t, 10 \times \theta_t]$ have started to fail in the target domain, producing low quality even all failed trajectories that cause other search baselines' errors to surge. Such scenarios are prevalent in real world deployment as the initial UDR policies rarely perform well at the outset. This highlights DEAL's strong search adaptability and stable performance. Furthermore, due to the lack of parallel data collection in reality, the scarcity of data and the high cost of collection mean few shot methods must minimize data requirements. Fig. 5 reveals DEAL's superior data efficiency: While existing Sys-Id methods suffer significant performance degradation with sparse trajectory samples, DEAL sustains robust identification accuracy under data-scarce conditions.

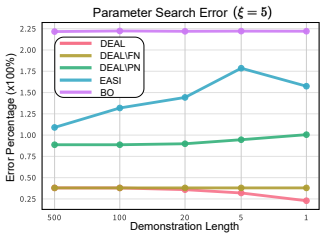

Figure 5: Average search errors of DEAL and other baselines when given different demonstration lengths in Humanoid task.

4.4 Sim-to-Real Transfer

In this experiment, we first deploy the UDR-trained policy to collect real-world trajectories in a few seconds, then optimize the simulator with DEAL, retrain the policy and redeploy to compare sim-to-real performance. In the Cartpole task, the controller outputs a target cart position to maintain the pole in balance. Stability is measured by pole-angle errors and cart-velocity, both of which are reduced after DEAL optimization as shown in Table 2, demonstrating a improved sim-to-real performance. In the Go2 environment, the controller outputs target position signals for each joint at a frequency of 50 Hz. These signals are then converted into motor torques by the motor's built-in PD controller operating at 1000 Hz and applied to the corresponding joints. In this section, we train a climbing policy based on RMA [33] for the experiment of climbing onto a high platform, and train a canter policy using Ess-InfoGAIL [48] for the quadruped robot's running experiments. Firstly, we conduct repetitive high-platform climbing experiments to evaluate the policy performance. As shown in Fig. 6, both UDR and Unitree's built-in RL policies are unable to reproduce the climbing behavior trained in simulation, always fail to control Go2 to climb the platform. After optimized by DEAL, the Go2 robot can fluently and successfully complete the task, the probability of success has been significantly

Table 2: Cartpole sim-to-real performance.

Method	Angle Error $\times 10^{-2}$	Cart Vel $\times 10^{-1}$
UDR	3.655±1.122	1.480±0.367
DEAL	1.372±0.382	1.214±0.118

improved. Subsequently, we deploy the canter policy and compare the running performance of the controllers trained with UDR and DEAL. As shown in Fig. 7, the UDR policy fails to reproduce the straight-running gait observed in simulation, resulting in significant deviations in both the magnitude and direction of the velocity. In contrast, the policy retrained with DEAL accurately tracks the velocity commands and maintains a forward speed close to that achieved in simulation during straight-line locomotion. These results indicate DEAL indeed narrows the sim-to-real gap and enhances the performance of the retrained policy in reality.

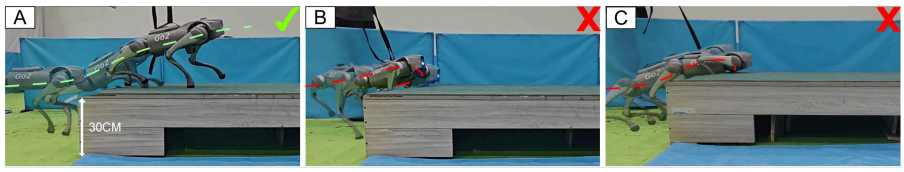

Figure 6: **(A)** Deploy with the climbing controller after being optimized with DEAL.**(B)** Deploy with the climbing controller trained with UDR. **(C)** Deploy with Unitree's built-in RL controller.

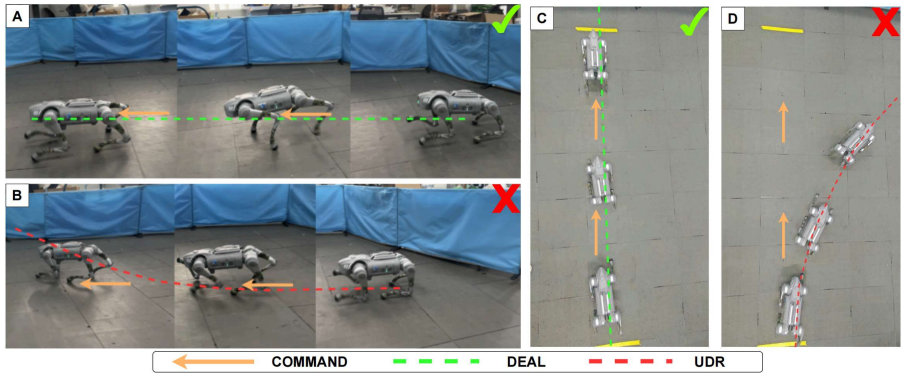

Figure 7: **(A, C)** Deploy with the canter controller after being optimized with DEAL. A: side-view imaging, C: top-view imaging. **(B, D)** Deploy with the canter controller trained with UDR. B: side-view imaging, D: top-view imaging.

5 Limitations

Although DEAL demonstrates remarkable performance in high-dimensional parameter optimization tasks under limited demonstration data, its applicability may face challenges in environments with diverse state transition distributions. This potential limitation arises from the inherent binary classification nature of the GAN discriminator, which is designed to distinguish between one target environment transition and one simulator transition at a time. As a result, the discriminator may struggle to provide multimodal fitness evaluations during optimization, suggesting an area for future improvement in handling more complex or multimodal environmental dynamics.

6 Conclusion

In this work, we propose DEAL, a novel Sys-Id framework that combines diffusion evolution with adversarial learning for sim-to-real transfer. DEAL optimizes simulator parameters through a dual mechanism: a discriminator evaluates the similarity of state transitions between evolving simulators and reality as fitness guidance, while DE adaptively refines parameter distributions to narrow the reality gap by using fitness-driven denoising. Extensive experiments in simulation and real-world demonstrate DEAL's superior performance in high-dimensional parameter identification and robust transfer performance across challenging tasks, outperforming baselines with minimal real-world data and lower computation cost. We believe that DEAL advances sim-to-real transfer and offers a promising approach for deploying RL controllers in the real world.

Acknowledgments and Disclosure of Funding

This work was supported in part by the Major Science and Technology Project of Jiangsu Province under Grant BG2024041, National Key Research and Development Program of China under Grant 2023YFD2001003, and the Fundamental Research Funds for the Central Universities under Grant 011814380048.

References

[1] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[2] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science robotics*, vol. 7, no. 62, p. eabk2822, 2022.

[3] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, no. 7976, pp. 982–987, 2023.

[4] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "Anymal parkour: Learning agile navigation for quadrupedal robots," *Science Robotics*, vol. 9, no. 88, p. eadi7566, 2024.

[5] T. Miki, J. Lee, L. Wellhausen, and M. Hutter, "Learning to walk in confined spaces using 3d representation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 8649–8656.

[6] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 91–100.

[7] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," in *2020 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2020, pp. 737–744.

[8] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3803–3810.

[9] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. 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)*, 2017, pp. 23–30.

[10] S. James, A. J. Davison, and E. Johns, "Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 334–343.

[11] G. Tiboni, K. Arndt, and V. Kyrki, "Dropo: Sim-to-real transfer with offline domain randomization," *Robotics and Autonomous Systems*, vol. 166, p. 104432, 2023.

[12] J. Dao, K. Green, H. Duan, A. Fern, and J. Hurst, "Sim-to-real learning for bipedal locomotion under unsensed dynamic loads," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 10 449–10 455.

[13] O. M. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020. [Online]. Available: https://doi.org/10.1177/0278364919887447

[14] H. Dong, H. Fu, W. Xu, Z. Zhou, and C. Chen, "Easi: Evolutionary adversarial simulator identification for sim-to-real transfer," *Advances in Neural Information Processing Systems*, vol. 37, pp. 6603–6624, 2024.

[15] A. Allevato, E. S. Short, M. Pryor, and A. Thomaz, "Tunenet: One-shot residual tuning for system identification and sim-to-real robot task transfer," in *Proceedings of the Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, L. P. Kaelbling, D. Kragic, and K. Sugiura, Eds., vol. 100. PMLR, 30 Oct–01 Nov 2020, pp. 445–455.

[16] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8973–8979.

[17] Y. Du, O. Watkins, T. Darrell, P. Abbeel, and D. Pathak, "Auto-tuned sim-to-real transfer," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1290–1296.

[18] Y. Jiang, T. Zhang, D. Ho, Y. Bai, C. K. Liu, S. Levine, and J. Tan, "Simgan: Hybrid simulator identification for domain adaptation via adversarial reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2884–2890.

[19] M. Memmel, A. Wagenmaker, C. Zhu, P. Yin, D. Fox, and A. Gupta, "Asid: Active exploration for system identification in robotic manipulation," *arXiv preprint arXiv:2404.12308*, 2024.

[20] P. Huang, X. Zhang, Z. Cao, S. Liu, M. Xu, W. Ding, J. Francis, B. Chen, and D. Zhao, "What went wrong? closing the sim-to-real gap via differentiable causal discovery," in *Proceedings of The 7th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, J. Tan, M. Toussaint, and K. Darvish, Eds., vol. 229. PMLR, 06–09 Nov 2023, pp. 734–760.

[21] F. Ramos, R. C. Possas, and D. Fox, "Bayessim: adaptive domain randomization via probabilistic inference for robotics simulators," *arXiv preprint arXiv:1906.01728*, 2019.

[22] F. Muratore, C. Eilers, M. Gienger, and J. Peters, "Data-efficient domain randomization with bayesian optimization," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 911–918, 2021.

[23] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[24] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[25] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.

[26] G. Batzolis, J. Stanczuk, C.-B. Schönlieb, and C. Etmann, "Conditional image generation with score-based diffusion models," *arXiv preprint arXiv:2111.13606*, 2021.

[27] D. Epstein, A. Jabri, B. Poole, A. Efros, and A. Holynski, "Diffusion self-guidance for controllable image generation," *Advances in Neural Information Processing Systems*, vol. 36, pp. 16 222–16 239, 2023.

[28] Y. Zhang, B. Hartl, H. Hazan, and M. Levin, "Diffusion models are evolutionary algorithms," *arXiv preprint arXiv:2410.02543*, 2024.

[29] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[30] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang, "Solving rubik's cube with a robot hand," 2019.

[31] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull, "Active domain randomization," in *Proceedings of the Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, L. P. Kaelbling, D. Kragic, and K. Sugiura, Eds., vol. 100. PMLR, 30 Oct–01 Nov 2020, pp. 1162–1176.

[32] I. M. A. Nahrendra, B. Yu, and H. Myung, "Dreamwaq: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning," 2023.

[33] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," 2021.

[34] G. Ji, J. Mun, H. Kim, and J. Hwangbo, "Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4630–4637, 2022.

[35] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, and V. Vanhoucke, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 4243–4250.

[36] J. P. Hanna and P. Stone, "Grounded action transformation for robot learning in simulation," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, ser. AAAI'17. AAAI Press, 2017, p. 3834–3840.

[37] H. Karnan, S. Desai, J. P. Hanna, G. Warnell, and P. Stone, "Reinforced grounded action transformation for sim-to-real transfer," 2020.

[38] S. Desai, I. Durugkar, H. Karnan, G. Warnell, J. P. Hanna, and P. 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: Curran Associates Inc., 2020.

[39] T. He, J. Gao, W. Xiao, Y. Zhang, Z. Wang, J. Wang, Z. Luo, G. He, N. Sobanbab, C. Pan *et al.*, "Asap: Aligning simulation and real-world physics for learning agile humanoid whole-body skills," *arXiv preprint arXiv:2502.01143*, 2025.

[40] K. Åström and P. Eykhoff, "System identification—a survey," *Automatica*, vol. 7, no. 2, pp. 123–162, 1971.

[41] W. Yu, J. Tan, C. K. Liu, and G. Turk, "Preparing for the unknown: Learning a universal policy with online system identification," 2017.

[42] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," 2018.

[43] R. Antonova, J. Yang, P. Sundaresan, D. Fox, F. Ramos, and J. Bohg, "A bayesian treatment of real-to-sim for deformable object manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 5819–5826, 2022.

[44] A. Z. Ren, H. Dai, B. Burchfiel, and A. Majumdar, "Adaptsim: Task-driven simulation adaptation for sim-to-real transfer," in *Proceedings of the Conference on Robot Learning (CoRL)*, 2023.

[45] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17. JMLR.org, 2017, p. 214–223.

[46] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[47] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018.

[48] H. Fu, K. Tang, Y. Lu, Y. Qi, G. Deng, F. Sung, and C. Chen, "Ess-infoGAIL: Semi-supervised imitation learning from imbalanced demonstrations," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. [Online]. Available: https://openreview.net/forum?id=jxhUNLoi4m

[49] N. Hansen, "The cma evolution strategy: A tutorial," *arXiv preprint arXiv:1604.00772*, 2016.

A Appendix

A.1 Proof of Diffusion Evolution

In the diffusion phase, let x_0 denote the original data point and x_T becomes entirely Gaussian noise. This process can be expressed as:

$$x_t = \sqrt{\alpha_t} x_0 + \sqrt{1-\alpha_t}\epsilon, \tag{12}$$

where the total noise $\epsilon \sim \mathcal{N}(0, I)$ added to the data x_0 at time step $t \in [0, T]$ is controlled by α_t, which decreases $\alpha_0 = 1$ to $\alpha_T \approx 0$. Meanwhile, a noise prediction module ϵ_θ is trained to minimize this loss function in diffusion phase:

$$\mathcal{L} = \mathbb{E}_{x_0 \sim p_{data}, \epsilon \sim \mathcal{N}(0,I)} \left\| \epsilon_\theta \left(\sqrt{\alpha_t} x_0 + \sqrt{1-\alpha_t}\epsilon, t \right) - \epsilon \right\|^2, \tag{13}$$

where p_{data} denotes the distribution of the training data. Under the DDIM framework [25], the denoising phase is defined as:

$$x_{t-1} = \sqrt{\alpha_{t-1}} \left(\frac{x_t - \sqrt{1-\alpha_t}\epsilon_\theta(x_t, t)}{\sqrt{\alpha_t}} \right) + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\epsilon_\theta(x_t, t) + \sigma_t \omega, \tag{14}$$

where σ_t controls the amount of random perturbations $\omega \sim \mathcal{N}(0, I)$ added during the denoising phase. Given the diffusion process by Equation (12), x_0 can be expressed by the noise ϵ, and vise versa:

$$x_0 = \frac{x_t - \sqrt{1-\alpha_t}\epsilon}{\sqrt{\alpha_t}}, \text{ and } \epsilon = \frac{x_t - \sqrt{\alpha_t} x_0}{\sqrt{1-\alpha_t}}, \tag{15}$$

by substituting Equation (15) to Equation (14) to estimate $\hat{x}_0$ and $\hat{\epsilon}$, the denosing phase can be rewritten as:

$$x_{t-1} = \sqrt{\alpha_{t-1}}\hat{x}_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\hat{\epsilon} + \sigma_t \omega. \tag{16}$$

Since the denoising step in diffusion models requires an estimation of x_0, DE derive it from sample x and the corresponding fitness $f(x)$. The estimation of x_0 can be expressed as a conditional probability $p(x_0 = x|x_t)$. Using Bayes' theorem and $p(x_0 = x) = g[f(x)]$ yields:

$$p(x_0 = x|x_t) = \frac{p(x_t|x_0 = x)p(x_0 = x)}{p(x_t)} = \frac{p(x_t|x)g[f(x)]}{p(x_t)}. \tag{17}$$

Here, $p(x_t|x_0 = x)$ can be computed easily by $\mathcal{N}(x_t; \sqrt{\alpha_t}x, 1-\alpha_t)$ given the design of the diffusion process. Since deep-learning-based diffusion models are trained using mean squared error loss, the x_0 estimated by x_t should be the weighted average of the sample x. Hence, the estimation function of x_0 becomes:

$$\hat{x}_0(x_t, \alpha, t) = \sum_{x \sim p_{\text{eval}}(x)} p(x_0 = x|x_t)x = \sum_{x \sim p_{\text{eval}}(x)} g[f(x)] \frac{\mathcal{N}(x_t; \sqrt{\alpha_t}x, 1-\alpha_t)}{p(x_t)} x, \tag{18}$$

where p_{eval} is the evaluation sample on which we compute the fitness score, here given by the current population $X_t = (x_t^{(1)}, x_t^{(2)}, ..., x_t^{(N)})$ of N individuals. Equation (18) has three weight terms: The first term $g[f(x)]$ assigns larger weights to high fitness samples. For each individual sample x_t, the second Gaussian term $\mathcal{N}(x_t; \sqrt{\alpha_t}x, 1-\alpha_t)$ makes each individual only sensitive to local neighbors of evaluation samples. The third term $p(x_t)$ is a normalization term. Hence, $\hat{x}_0$ can be simplified to:

$$\hat{x}_0(x_t, \alpha, t) = \frac{1}{Z} \sum_{x \in X_t} g[f(x)]\mathcal{N}(x_t; \sqrt{\alpha_t}x, 1-\alpha_t)x, \tag{19}$$

where Z is the normalization term:

$$Z = p(x_t) = \sum_{x \in X_t} g[f(x)]\mathcal{N}(x_t; \sqrt{\alpha_t}x, 1-\alpha_t). \tag{20}$$

When substituting Equation (19) into Equation (15) we can express noise prediction $\hat{\epsilon}$ as:

$$\hat{\epsilon}(x_t, \alpha, t) = \frac{x_t - \sqrt{\alpha_t}\,\hat{x}_0(x_t, \alpha, t)}{\sqrt{1-\alpha_t}}. \tag{21}$$

After we successfully estimate the optimal point $\hat{x}_0$ and the noise $\hat{\epsilon}$, the next generation of parameters can be generated by using Equation (16).

Previous black-box optimization methods such as evolution strategies (ES) typically model the parameter distribution as a Gaussian, updating its mean and covariance in each sample iteration. However, the new samples are randomly drawn from the updated distribution and lack explicit control over the convergence trajectory, often resulting in slow convergence and low final accuracy. Instead of blindly perturbing parameters in each iteration, DE performs a re-weighted posterior estimation of the optimum based on observed samples. This allows the search process to focus more quickly on regions close to real data distribution.

Specifically, DEAL updates the parameter distribution using the Equation (10), where the estimated optimal parameter $\hat{\theta}_0$ is given by Equation (1), which can be rewritten as follows:

$$\hat{\theta}_0(\theta_t, \alpha, t) = \frac{1}{Z} \sum_{\theta \in \Theta_t} \underbrace{g[f(\theta)]}_{p(\theta_0=\theta)} \underbrace{\mathcal{N}(\theta_t; \sqrt{\alpha_t}\theta, 1-\alpha_t)}_{p(\theta_t|\theta_0=\theta)} \theta, \quad g(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}. \tag{22}$$

In Equation (22), $p(\theta_t \mid \theta_0 = \theta)$ acts as a neighborhood weight around parameter θ_t. $p(\theta_0 = \theta)$ represents a probability derived by applying a softmax function $g(\cdot)$ to the discriminator's output, reflecting the likelihood that a state transition under parameter θ resembles one from the real world. This formulation effectively uses a softmax-style weighting to emphasize high-fitness regions, guiding $\hat{\theta}_0$ rapidly toward the true optimal parameters. As a result, the estimate $\hat{\theta}_0$ serves as a proxy for the gradient direction of the WGAN's objective function $\mathcal{L}$ in Equation (23) as follows:

$$\underbrace{\mathcal{L}}_{\searrow} = \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim d^{\mathcal{T}(\theta^{target},\pi_0)}}[D(\mathbf{s},\mathbf{a},\mathbf{s}')] - \underbrace{\mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim d^{\mathcal{S}}(\theta,\pi_0)}[D(\mathbf{s},\mathbf{a},\mathbf{s}')]}_{\nearrow}. \tag{23}$$

The diffusion noise schedule also helps to stablize the convergence by determining the update step at each iteration. As $t \to 0$, $\alpha_t \to 1$ and $\sigma_t \to 0$, reducing the variance of $\mathcal{N}(\theta_t; \sqrt{\alpha_t}\theta, 1-\alpha_t)$ and encouraging fine-grained, local updates in the later stages. The update $\theta_{t-1} \to \hat{\theta}_0$ serves as a denoising step, similar to the reverse process in diffusion models. With a well-trained discriminator, this process enables stable convergence toward the true optimum.

Additionally, unlike traditional ES that rely on elitism or truncation, which can lose global information and lead to premature convergence to local optima, DEAL utilizes the entire population in Equation (22). It assigns soft weights based on discriminator-derived fitness to estimate the gradient direction of the WGAN's objective function. This leads to more targeted and stable updates, offering improved convergence stability, efficiency and accuracy over previous methods.

A.2 Computation cost

The computation time cost presented in Table 3 is based on searches conducted over 50 steps on a PC equipped with an Intel i5-14600KF and an RTX 4060 Ti.

Table 3: Average computation time cost.

Task	DEAL	EASI	BO
Ant	**1 min 38 s**	1 min 43 s	7 min 05 s
Cartpole	**1 min 15 s**	1 min 35 s	6 min 55 s
Go2	**5 min 03 s**	5 min 26 s	21 min 17 s
Humanoid	**1 min 43 s**	1 min 52 s	42 min 54 s
AllegroHand	**54 min 05 s**	54 min 15 s	125 min 04 s

A.3 Data budget

All tasks run 200 parallel environments, but the trajectory lengths differ across tasks. Table 4 below lists the data budget allocated to DEAL and the baseline methods for each task.

Table 4: Data Budget (200 parallel environments $\times$ trajectory length).

Data budget	Ant	CartPole	Humanoid	Go2	AllegroHand
For DEAL and baselines	200×200	200×200	200×200	200×250	200×600

A.4 Impact of trajectories quality

Current learning-based system identification methods do require the collection of trajectories with a certain quality in the real world. However, even trajectories from very poor policies still encode physical dynamics information. When using the same policy to search parameters, the failed behavior should be reproduced by adjusting those parameters. To address the concern regarding DEAL's reliance on trajectory quality, we train the UDR policies for the CartPole and Humanoid in simulation, and take the policies at various training checkpoints to collect a 200-step trajectory in the target environment. We then search for parameters within $[\frac{1}{3} \times \theta^{target}, 3 \times \theta^{target}]$ to reflect the impact of the policies performance on the search results. In the Tables 5, 50 denotes the policy after 50 training iterations, and the policy only starts to converge in the last row.

Table 5: Average search results at each checkpoint.

Iterations(CartPole)	Avg. Search Error (%)	Iterations(Humanoid)	Avg. Search Error (%)
50	8.84 ± 2.96	5e3	13.59 ± 5.37
100	7.87 ± 2.35	1e4	14.70 ± 4.80
250	7.19 ± 2.96	2e4	13.08 ± 5.00
1000	7.34 ± 2.88	2.5e4	10.24 ± 4.13
2500	5.10 ± 2.02	3e4	10.78 ± 3.40

Contrary to concerns about its reliance on trajectory quality, DEAL demonstrates strong robustness and can still accurately search parameters even when using low-quality trajectories collected from early-stage policies. As training progresses and trajectory quality improves, DEAL's search performance improves accordingly.

A.5 Comparison with model-based methods

To assess the improvement of DEAL in finding the optimal parameters in comparison with a non-convex optimization problem formulation. We further compare DEAL with several strong model-based baselines in the case of a simple robotic system with non-linear dynamics, including least-squares estimation and Extended Kalman Filter(EKF). In this section, we conduct parameter search experiments on the CartPole task using the same trajectory data for all three methods, the results are reported in the Table 6.

Table 6: Comparison with model-based methods.

Avg. Search Error (%)	CartPole
DEAL	**6.9±1.1**
Model-based EKF	37.4±8.8
Least-squares	53.0±19.6

As shown in the table, EKF suffers from significant bias due to linearization errors when estimating nonlinear parameters. The least-squares estimation exhibits relatively large search errors, and the results are highly sensitive to random initialization. In contrast, DEAL demonstrates stable and accurate search performance in such nonlinear systems.

A.6 Broader impacts

In our work, we propose DEAL, a novel framework that leverages diffusion evolution and adversarial learning to align simulated and real-world dynamics with minimal real-world demonstration. By

narrowing the reality gap, DEAL can accelerate the deployment of RL controllers in robotics, reducing the need for costly and time-consuming hardware trials. This has the potential to democratize advanced robotic applications across industries—manufacturing, logistics, healthcare by lowering both development time and resource requirements. At the same time, overreliance on simulator-guided optimization carries the risk that unmodeled real-world complexities—sensor degradation, unanticipated contacts may still lead to failures during long-term operation. Thus, while DEAL substantially improves sim-to-real transfer, it should be complemented by continued investment in robust robot design, sensor redundancy, and online adaptation schemes. Future work should explore hardware–software co-design to further to address the sim-to-real challenge.

A.7 Physical parameter list

The parameters that are symmetrically distributed on the robot's limbs should be chosen to be the same or similar values to ensure successful training, however, during the parameter search process, they are still treated as independent parameters for searching.

Table 7: Parameter list in the Ant Environment.

Parameter	Target Value	Unit
Contact Friction	1.5	-
Contact Restitution	0.01	-
Body-Leg Motor Friction	0.2	-
Body-Leg Motor Damping	0.3	$\mathrm{N \cdot m \cdot s/rad}$
Body-Leg Motor Armature	0.1	$\mathrm{kg \cdot m^2}$
Foot-Leg Motor Friction	0.1	-
Foot-Leg Motor Damping	0.2	$\mathrm{N \cdot m \cdot s/rad}$
Foot-Leg Motor Armature	0.1	$\mathrm{kg \cdot m^2}$
Foot Mass	0.1	kg
Leg Mass	0.2	kg
Body Mass	1.0	kg

Table 8: Parameter list in the Cartpole Environment.

Parameter	Target Value	Unit
Pole Length	0.3	m
Pole Mass	0.1	kg
Cart Mass	0.3	kg
Pole DOF_Friction	0.1	-
Pole DOF_Damping	0.1	$\mathrm{N \cdot m \cdot s/rad}$
Pole DOF_Amature	0.2	$\mathrm{kg \cdot m^2}$
Cart DOF_Friction	0.1	-
Cart PID_P	0.1	$\mathrm{N \cdot m/rad}$
Cart PID_D	0.1	$\mathrm{N \cdot m \cdot s/rad}$
Cart EffortLimit	0.2	N
Cart Vel	1.0	m/s

Table 9: Parameter list in the Go2 Environment.

Parameter	Target Value	Unit
Hip Damping$\times 4$	20	$\mathrm{N \cdot m \cdot s/rad}$
Hip Stiffness$\times 4$	0.5	$\mathrm{N \cdot m/rad}$
Calf Damping$\times 4$	20	$\mathrm{N \cdot m \cdot s/rad}$
Calf Stiffness$\times 4$	0.5	$\mathrm{N \cdot m/rad}$
Thigh Damping$\times 4$	20	$\mathrm{N \cdot m \cdot s/rad}$
Thigh Stiffness$\times 4$	0.5	$\mathrm{N \cdot m/rad}$

Table 10: Parameter list in the AllegroHand Environment.

Parameter	Target Value	Unit
Contact Friction$\times 22$	1.0	-
Contact Restitution$\times 22$	0.01	-
Dof Friction$\times 16$	0.01	-
Dof Damping$\times 16$	0.1	$\mathrm{N \cdot m \cdot s/rad}$
Dof Armature$\times 16$	0.001	$\mathrm{kg \cdot m^2}$
Dof Stiffness$\times 16$	3.0	$\mathrm{N \cdot m/rad}$
Allegro_Mount Mass	0.47	kg
Index_Link_0 Mass	0.012	kg
Index_Link_1 Mass	0.065	kg
Index_Link_2 Mass	0.036	kg
Index_Link_3 Mass	0.031	kg
Middle_Link_0 Mass	0.012	kg
Middle_Link_1 Mass	0.065	kg
Middle_Link_2 Mass	0.036	kg
Middle_Link_3 Mass	0.031	kg
Ring_Link_0 Mass	0.012	kg
Ring_Link_1 Mass	0.065	kg
Ring_Link_2 Mass	0.036	kg
Ring_Link_3 Mass	0.031	kg
Thumb_Link_0 Mass	0.018	kg
Thumb_Link_1 Mass	0.012	kg
Thumb_Link_2 Mass	0.038	kg
Thumb_Link_3 Mass	0.060	kg

Table 11: Parameter list in the Humanoid Environment.

Parameter	Target Value	Unit
Contact Friction$\times 4$	1.0	-
Contact Restitution$\times 4$	0.01	-
Abdomen_Y Friction	0.01	-
Abdomen_Y Damping	5.0	$\mathrm{N \cdot m \cdot s/rad}$
Abdomen_Y Armature	0.02	$\mathrm{kg \cdot m^2}$
Abdomen_Y Stiffness	20.0	$\mathrm{N \cdot m/rad}$
Abdomen_Z Friction	0.01	-
Abdomen_Z Damping	5.0	$\mathrm{N \cdot m \cdot s/rad}$
Abdomen_Z Armature	0.01	$\mathrm{kg \cdot m^2}$
Abdomen_Z Stiffness	20.0	$\mathrm{N \cdot m/rad}$
Abdomen_X Friction	0.01	-
Abdomen_X Damping	5.0	$\mathrm{N \cdot m \cdot s/rad}$
Abdomen_X Armature	0.01	$\mathrm{kg \cdot m^2}$
Abdomen_X Stiffness	10.0	$\mathrm{N \cdot m/rad}$
Hip_X Friction$\times 2$	0.01	-
Hip_X Damping$\times 2$	5.0	$\mathrm{N \cdot m \cdot s/rad}$
Hip_X Armature$\times 2$	0.01	$\mathrm{kg \cdot m^2}$
Hip_X Stiffness$\times 2$	10.0	$\mathrm{N \cdot m/rad}$
Hip_Z Friction$\times 2$	0.01	-
Hip_Z Damping$\times 2$	5.0	$\mathrm{N \cdot m \cdot s/rad}$
Hip_Z Armature$\times 2$	0.01	$\mathrm{kg \cdot m^2}$
Hip_Z Stiffness$\times 2$	10.0	$\mathrm{N \cdot m/rad}$
Hip_Y Friction$\times 2$	0.01	-
Hip_Y Damping$\times 2$	5.0	$\mathrm{N \cdot m \cdot s/rad}$
Hip_Y Armature$\times 2$	0.01	$\mathrm{kg \cdot m^2}$
Hip_Y Stiffness$\times 2$	20.0	$\mathrm{N \cdot m/rad}$
Knee Friction$\times 2$	0.01	-
Knee Damping$\times 2$	0.1	$\mathrm{N \cdot m \cdot s/rad}$
Knee Armature$\times 2$	0.007	$\mathrm{kg \cdot m^2}$
Knee Stiffness$\times 2$	5.0	$\mathrm{N \cdot m/rad}$
Ankle_X Friction$\times 2$	0.01	-
Ankle_X Damping$\times 2$	1.0	$\mathrm{N \cdot m \cdot s/rad}$
Ankle_X Armature$\times 2$	0.006	$\mathrm{kg \cdot m^2}$
Ankle_X Stiffness$\times 2$	2.0	$\mathrm{N \cdot m/rad}$
Ankle_Y Friction$\times 2$	0.01	-
Ankle_Y Damping$\times 2$	1.0	$\mathrm{N \cdot m \cdot s/rad}$
Ankle_Y Armature$\times 2$	0.006	$\mathrm{kg \cdot m^2}$
Ankle_Y Stiffness$\times 2$	2.0	$\mathrm{N \cdot m/rad}$
Shoulder1 Friction$\times 2$	0.01	-
Shoulder1 Damping$\times 2$	5.0	$\mathrm{N \cdot m \cdot s/rad}$
Shoulder1 Armature$\times 2$	0.01	$\mathrm{kg \cdot m^2}$
Shoulder1 Stiffness$\times 2$	10.0	$\mathrm{N \cdot m/rad}$
Shoulder2 Friction$\times 2$	0.01	-
Shoulder2 Damping$\times 2$	5.0	$\mathrm{N \cdot m \cdot s/rad}$
Shoulder2 Armature$\times 2$	0.01	$\mathrm{kg \cdot m^2}$
Shoulder2 Stiffness$\times 2$	10.0	$\mathrm{N \cdot m/rad}$
Elbow Friction$\times 2$	0.01	-
Elbow Damping$\times 2$	1.0	$\mathrm{N \cdot m \cdot s/rad}$
Elbow Armature$\times 2$	0.006	$\mathrm{kg \cdot m^2}$
Elbow Stiffness$\times 2$	2.0	$\mathrm{N \cdot m/rad}$

A.8 Error bars

These error bars are based on calculations from 95% CI.

Table 12: Average search error percentage($\times 100\%$).

	Cartpole		
Method	ξ =10	ξ =15	ξ =20
DEAL	**0.85±0.06**	**1.74±0.10**	**2.63±0.13**
DEAL\PN	1.90±0.11	2.85±0.15	3.76±0.17
DEAL\FN	1.67±0.10	2.96±0.06	4.28±0.12
EASI	4.03±0.50	6.12±0.35	8.40±0.46
BO	5.20±0.04	8.52±0.16	11.78±0.25

Table 13: Average search error percentage($\times 100\%$).

	Humanoid		
Method	ξ =10	ξ =15	ξ =20
DEAL	**0.81±0.02**	**1.65±0.04**	**2.50±0.05**
DEAL\PN	2.40±0.06	3.53±0.07	4.58±0.15
DEAL\FN	1.57±0.02	2.79±0.02	4.03±0.03
EASI	2.94±0.85	4.64±1.50	6.03±1.82
BO	4.67±0.01	7.29±0.01	10.06±0.01

Table 14: Average search error percentage($\times 100\%$).

	AllegroHand		
Method	ξ =10	ξ =15	ξ =20
DEAL	**0.83±0.02**	**1.69±0.03**	**2.57±0.05**
DEAL\PN	2.60±0.03	3.80±0.05	4.92±0.05
DEAL\FN	1.59±0.01	2.80±0.17	4.06±0.20
EASI	4.25 ± 0.12	6.82±0.20	9.32±0.25
BO	3.75 ±0.01	6.32±0.03	9.16±0.06

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 in Section 5 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: [Yes]

Justification: We have provided proofs of the relevant theories used in the paper in the main text and Appendix A.1.

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 4.

   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 Section 4 and Appendix A.7.

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 4 and Appendix A.2.

   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 A.6.

    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.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We did not use LLMs as a component of the method in this study.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.