The Role of Domain Randomization in Training Diffusion Policies for Whole-Body Humanoid Control

Oleg Kaidanov^{1,2}, **Firas Al-Hafez**^{1}, **Yusuf Süvari**^{1}, **Boris Belousov**^{2}, **Jan Peters**^{1,2,3}

^{1}TU Darmstadt ^{2}German Research Center for AI (DFKI) ^{3}Hessian.AI
`oleg.kaidanov@dfki.de`

Abstract: Humanoids have the potential to be the ideal embodiment in environments designed for humans. Thanks to the structural similarity to the human body, they benefit from rich sources of demonstration data, e.g., collected via teleoperation, motion capture, or even using videos of humans performing tasks. However, distilling a policy from demonstrations is still a challenging problem. While Diffusion Policies (DPs) have shown impressive results in robotic manipulation, their applicability to locomotion and humanoid control remains underexplored. In this paper, we investigate how *dataset diversity and size* affect the performance of DPs for humanoid whole-body control. In a simulated IsaacGym environment, we generate synthetic demonstrations by training Adversarial Motion Prior (AMP) agents under various Domain Randomization (DR) conditions, and we compare DPs fitted to datasets of different size and diversity. Our findings show that, although DPs can achieve stable walking behavior, successful training of locomotion policies requires significantly larger and more diverse datasets compared to manipulation tasks, even in simple scenarios. Videos can be found at `https://sites.google.com/view/dps-for-humanoid-control`.

Keywords: Humanoid Control, Locomotion, Diffusion Policies

1 Introduction

Humanoid robots hold great promise as ideal embodiments for human-centered environments due to their structural resemblance to the human body, which enables them to leverage rich datasets like motion capture for learning control policies. As more companies begin to develop, produce, and commercialize humanoid robots, there is a growing demand for robust and general whole-body motion policies. While diffusion models, in particular Diffusion Policies (DPs), have recently achieved significant success in robot arm manipulation [1, 2], research on whole-body control — encompassing simultaneous locomotion and arm movements — remains relatively limited [3, 4]. Most existing approaches that use diffusion models for robot control rely on small real-world datasets, which are often sparse and difficult to collect [5, 6, 7]. In contrast, training Reinforcement Learning (RL) policies for locomotion and whole-body control in randomized simulators has demonstrated substantial robustness and success [8, 9]. Recent work, such as DiffuseLoco [10], has shown that diffusion models can integrate multiple source policies into a unified model for robot control. However, while the focus has largely been on the development of the diffusion framework, the impact of the source dataset used for training remains underexplored.

To address the insufficient understanding of the role of dataset characteristics on training DPs, we investigate different types of Domain Randomization (DR) — such as perturbations, dynamic variations, and terrain changes — as well as varying dataset sizes. We collect datasets with distinct randomization strategies and sizes to train separate DPs. Our findings reveal that DR is crucial for training successful DPs: even large datasets without sufficient randomization struggle to generalize to non-randomized environments.

Contributions. Our contributions are twofold. First, we present the first ablation study on the impact of DR in dataset generation for training DPs in humanoid control. This includes not only commonly used DR techniques but also the introduction of a novel approach. Second, we analyze the effect of dataset size on training, exploring how varying amounts of data interact with different randomization techniques. Notably, while only a few trajectories may suffice for manipulation tasks, training DPs for whole-body control demands substantially more data to achieve robust performance.

2 Related Work

Imitation Learning for Humanoid Control. One prominent approach in whole-body control is Imitation Learning (IL) from motion capture data. DeepMimic [11] introduced an RL framework where physics-based characters learn skills by mimicking reference motions from a set of motion clips. By combining imitation rewards with task-specific objectives, DeepMimic allows characters to both mimic reference motions and achieve specific goals. Most imitation learning approaches are based on Generative Adversarial Imitation Learning (GAIL) [12], in which a discriminator is learned and used as a reward signal to an RL policy. Various adaptations have been proposed [13, 14, 15, 16], with Adversarial Motion Prior (AMP) [14] achieving notable success. AMP leverages the discriminator's learned reward as a style reward, supplemented with additional handcrafted rewards for task-specific objectives. This approach encourages the agent to replicate the style of the dataset while effectively accomplishing the given tasks. Escontrela et al. [17] demonstrated that AMP can produce natural locomotion strategies for quadrupedal robots using only a few seconds of motion capture data from a German Shepherd. Extending these ideas to humanoid robots, however, presents additional challenges due to their higher degrees of freedom and balance requirements. He et al. [18] proposed Human-to-Humanoid (H2O), a framework for real-time whole-body teleoperation of humanoid robots using an RGB camera. They introduced a "sim-to-data" process to filter and select feasible motions from a large human motion dataset. While their approach scales to a large number of motions, the dataset still consists of retargeted human motions without added variability. Our work builds upon this foundation by utilizing diffusion policy and versatile DR.

Diffusion Policies in Robot Learning. Recent advancements in robotics have seen the integration of diffusion models into policy learning for legged locomotion tasks. Diffusion models, known for their ability to capture complex, multimodal distributions [19], have been effectively utilized to model the stochasticity and adaptability required for locomotion in varied environments. Kang et al. [19] propose Efficient Diffusion Policies (EDP) for offline RL, aiming to overcome the computational inefficiency of previous diffusion-based approaches like Diffusion-QL. While their focus is on improving training efficiency and compatibility with various offline RL algorithms, the data collection in their experiments relies on existing offline datasets from benchmarks like D4RL [20], without specific emphasis on DR or data diversity.

Ren et al. [21] introduce Diffusion Policy Policy Optimization (DPPO), a framework for fine-tuning diffusion-based policies using policy gradient methods. Their approach demonstrates improved performance and training stability over other RL methods for diffusion policies. While their work focuses on fine-tuning pre-trained policies and does not delve deeply into the data collection process, they use DR during sim-to-real transfer. By adding noise to observations and actions, they simulate imperfect conditions, thereby improving the robustness of their policies in real-world deployments. However, the application of DR, in their approach compared to ours, is not extensively explored.

BiRoDiff introduced by Mothish et al. [22] presents a real-time controller based on diffusion models for bipedal robots. They collect their source dataset by deploying a deep reinforcement learning policy trained via Proxmimal Policy Optimization (PPO) on their custom-made bipedal robot, Stoch BiRo. The dataset comprises observation-action pairs collected from walking on flat ground and slopes with specific inclinations. Their framework emphasizes generalization to unseen terrains, demonstrating the capability of diffusion policies to handle multiple walking behaviors with different velocities on various terrains. However, their approach primarily focuses on the generalization aspect without delving deeply into the challenges posed by dynamic whole-body control.

Huang et al. proposed DiffuseLoco [10], a framework that leverages diffusion models to learn multi-skill locomotion policies from offline datasets. DiffuseLoco showcases the potential of diffusion policies in capturing diverse locomotion skills, including agile maneuvers like hopping and bipedal walking. To collect the source dataset, they generate data from multiple single-skill control policies obtained through various methods, including reinforcement learning and central pattern generators. They focus on offline learning, which allows them to scalably incorporate diverse skills. While their work highlights the scalability and versatility of diffusion models in robotics, it also addresses the importance of DR in handling dynamic tasks involving whole-body motions.

While these works collectively advance the field of diffusion-based policies for locomotion, there is a gap in addressing the challenges posed by dynamic whole-body control tasks. DR has been shown to be crucial in bridging the simulation-to-reality gap, particularly for tasks involving high dynamics and complex interactions [23, 24]. Our approach builds upon the strengths of diffusion-based policies and explicitly integrates DR to handle the complexities of dynamic whole-body control tasks. By introducing variability during training, our method enhances the robustness and adaptability of the learned policies, ensuring better performance in real-world scenarios with diverse and unpredictable conditions.

3 Framework for Offline Dataset Generation

To generate datasets for training Diffusion Policies, we first train robust RL policies using AMP [25]. This method integrates goal-conditioned RL with IL. In our case, the goal is a velocity command. For the imitation component, we leverage 10 motion capture sequences from the AMASS dataset [26], which include walking in various directions (forward, backward, sideways), in-place rotations, walking in circles, and "8-figure" walking.

The policy is trained under extensive Domain Randomization (cf. Table 2), incorporating several widely adopted regularization rewards (cf. Table 1) to enhance the stability and smoothness of the resulting motions [8]. Having trained the RL policy, we utilize it to collect a total of 24 distinct datasets (3 dataset sizes across 8 environment setups), which are used to train diffusion models both individually and collectively. The specific environment randomization configurations are detailed in Sec. 3.2. Once the diffusion models are trained, we assess their performance across two distinct evaluation setups: i) without DR on a flat surface, and ii) with dynamics randomization on a complex, uneven terrain.

3.1 Reinforcement Learning Environment

We model the environment as a Markov Decision Process (MDP), defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, p_0, \gamma)$. $\mathcal{S}$ denotes the state space, $\mathcal{A}$ represents the action space, $\mathcal{T}$ describes the transition function, $\mathcal{R}$ indicates the reward function, p_0 is the distribution of the initial state, and γ is the discount factor. The objective of RL is to determine the optimal parameters θ of a parameterized policy $\pi_\theta : \mathcal{S} \to \mathcal{A}$ that maximizes the expected discounted return: $J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} \gamma^t r_t\right]$.

We use the AMP algorithm in a simulated Unitree H1 humanoid robot environment. Actions are represented by a 19-dimensional vector $\mathbf{a}_t \in \mathcal{A} = \mathbb{R}^{19}$, which indicates the desired positional changes for each actuated joint, later processed by a Proportional-Derivative (PD) controller. Similar to prior work [10], we give the critic privileged information such as base linear and angular velocities, while the policy receives a reduced observation vector without velocities. The privileged observations represented as $\mathbf{o}_t \in \mathbb{R}^{69}$, include the current linear and angular velocities of the humanoid, the orientation of the gravity vector relative to the robot's base frame, joint positions and velocities, target command, and previous actions. A command vector $\mathbf{c}_t$ is used to define the target velocities along the x-, y-, and yaw-axis in the robot's base frame.

To encourage the robot to effectively track the desired command velocities $\hat{v}_{t,x}$, $\hat{v}_{t,y}$, and the global yaw rate $\hat{\omega}_t$, we define a task reward function similar to [17], that assesses how closely the robot's actual motion aligns with these targets. Specifically, we want to minimize the difference between

Table 1: Regularization Rewards

Name	Value	Weight
DoF lower limit	$-\max(\theta - \theta_{\text{lim,low}}, 0)$	-4.0
DoF upper limit	$\min(\theta - \theta_{\text{lim,up}}, 0)$	-4.0
DoF velocities	$\|\dot{\theta}\|$	-3.0e-5
DoF acceleration	$\|\ddot{\theta}\|$	-1.0e-7
Non-flat base orientation	$\|\mathbf{g}^{xy}\|$	-1.0

Table 2: Dynamics Randomization

Parameter	Range
Body friction	[0.7, 1.3]
Added base mass [kg]	[-2.0, 2.0]
Link mass multiplier	[0.8, 1.2]
PD gains multiplier	[0.8, 1.2]
COM displacement [m]	[-0.15, 0.15]
External perturbation [m/s]	[0, 0.6]
Joint friction coeff.	[0.01, 1.15]
Joint damping coeff.	[0.3, 1.5]

the robot's current linear and angular velocities $v_{t,x}$, $v_{t,y}$, ω_t and the desired values. The reward function r_g at each timestep is defined as

$$r_\text{g} = w_v \exp\left(-\|\hat{v}_{t,xy} - v_{t,xy}\|\right) + w_\omega \exp\left(-|\hat{\omega}_t - \omega_t|\right). \tag{1}$$

The desired velocities are sampled uniformly from the ranges: $\hat{v}_{t,x} \in (-1, 1)$ m/s, $\hat{v}_{t,y} \in (-0.7, 0.7)$ m/s, $\hat{\omega}_t \in (-1.57, 1.57)$ rad/s. This task reward function is combined with regularization rewards defined in Table 1 and a style reward function learned by a discriminator in AMP.

3.2 Environment Randomizations During Data Collection

DR is used in RL to enhance an agent's robustness by training it across a distribution of similar environments, rather than just a single environment. This approach has proven crucial for addressing the sim-to-real gap in RL, enabling zero-shot transfer of policies from simulation to the real world. In this work, we aim to apply DR to randomize the source dataset used for training the DP. To achieve this, we introduce various randomization clusters below, including both commonly used techniques and newly proposed ones. While we apply all DR during the training of the AMP policy, we implement each one separately during the collection of the source dataset used to train the DP. This approach enables us to identify the effects and importance of each randomization method.

Dynamics randomization. During dynamics randomization, we randomize the simulator's parameters, as initially proposed by [27]. A detailed overview of randomized parameters of the simulator is given in Table 2.

Perturbation randomization. We apply force perturbations of random amplitude to the humanoid's torso, capping the maximum amplitude at 0.6 m/s. These perturbations are introduced at 3-second intervals. The primary objective of perturbation randomization is to destabilize the robot's state, compelling it to adapt and learn effective recovery strategies from unstable conditions. By exposing the humanoid to various disturbances, we aim to enhance its resilience and improve its ability to maintain balance and perform tasks in dynamic environments.

Terrain randomization. Instead of simulating a flat ground only, we randomize the terrain the humanoid is walking on. To do so, we utilize terrains supported in IsaacGym which include terrains with obstacles as well as bumpy surfaces.

Initial state randomization. Reference state initialization has proven important for imitating human movements [11, 14]. Hence, we sample an initial state from the expert dataset at the beginning of each episode to initialize the simulation. To further diversify the source dataset, we sample random joint positions and velocities in a limited range at a 50% chance.

Humanoid scale randomization. Al-Hafez et al. [28, 29] recently proposed an idea of using multiple body scales during training. Here we aim to investigate how data collected from humanoids of different scales impacts the performance of the Diffusion Policy. We initially trained three separate RL policies, each for a different scale. However, we later discovered that our RL policy, originally trained with a single scale, remains robust to a range of variations in the humanoid scale. As a result, we opted to use this policy for data collection. When scaling is applied, the mass and inertia of all links are scaled by factors of k^3 and k^5, respectively, where k is the scaling factor. To account for these changes, we scale the PD gains of the motors and the torque limits by a factor of k^4 for each scale. During data collection, humanoids of scales 0.8, 1.0, and 1.2 are used.

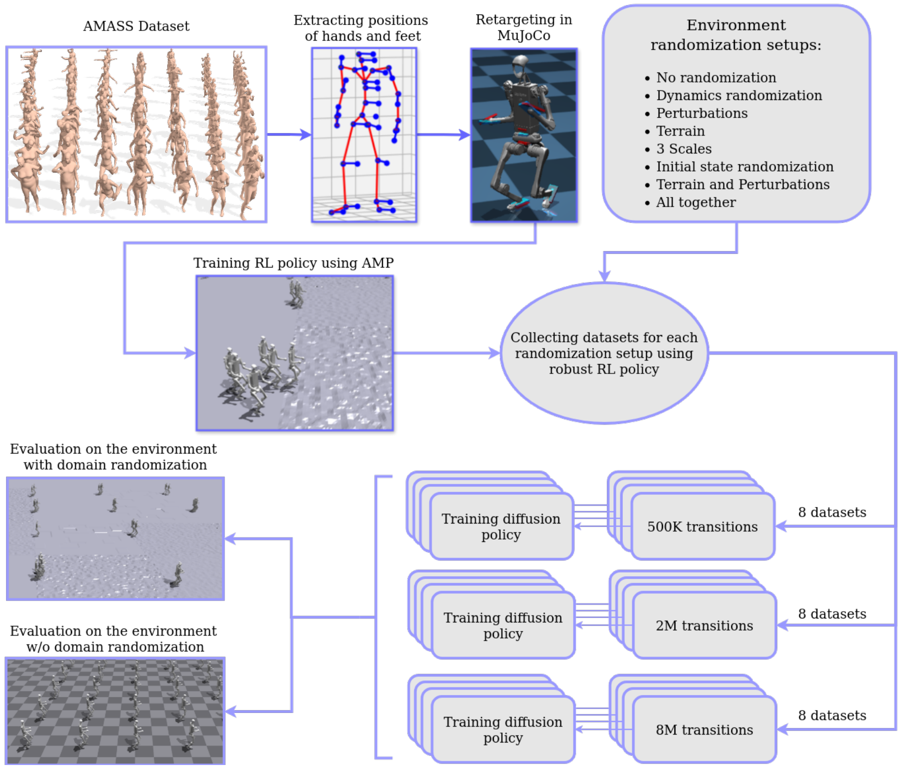
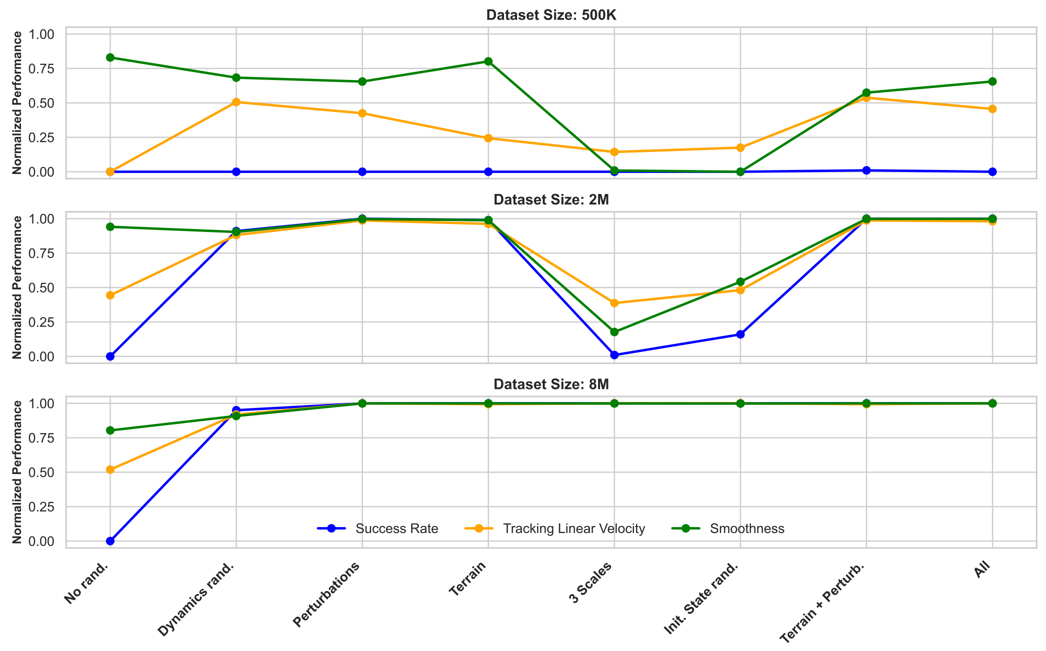

AMASS Dataset
Extracting positions of hands and feet
Retargeting in MuJoCo
Environment randomization setups:
• No randomization
• Dynamics randomization
• Perturbations
• Terrain
• 3 Scales
• Initial state randomization
• Terrain and Perturbations
• All together
Training RL policy using AMP
Collecting datasets for each randomization setup using robust RL policy
Evaluation on the environment with domain randomization
Evaluation on the environment w/o domain randomization
Training diffusion policy
500K transitions
8 datasets
Training diffusion policy
2M transitions
8 datasets
Training diffusion policy
8M transitions
8 datasets

Figure 1: Proposed method. First, a robust and stable RL policy is trained using AMP under extensive Domain Randomization. This policy is then used for data collection, to subsequently train Diffusion Policies. We generate different datasets, each with different DR applied during data collection, and we train DPs on each dataset separately. Finally, performance of each DP is evaluated on two environments: with and without DR.

4 Diffusion Models for Whole-Body Humanoid Control

In a recent work, Huang et al. [10] utilized diffusion models for learning skill transitions from a set of separately trained RL skill policies. Similarly, we adopt an encoder-decoder architecture, incorporating two 2-layer MLP encoders and six Transformer decoder layers, each having an 8-head cross-attention layer. The MLP encoders embed the previous state-action transitions and goal information, which — along with the diffusion timestep embedding — serve as conditional inputs. Ground-truth actions, after the addition of noise, are passed through the Transformer decoder layers. In each layer, cross-attention weights are computed between the noisy actions and the conditional information. Finally, the model minimizes the mean square error loss between the predicted and sampled noise. As proposed by Huang et al. [10], we employ receding horizon control, where predictions are made for n future steps, but only the first predicted action is executed.

Figure 1 illustrates the overall setup. Motion sequences are collected from the AMASS dataset, from which the global positions and rotations of the hands and feet of a human subject are extracted. These keypoints are then processed using the MuJoCo simulator to solve the inverse kinematics problem, taking into account the humanoid morphology and potential collisions. This approach generates motion sequences consistent with humanoid structure, although not necessarily aligned with the physical dynamics of the real world or robot. These sequences, however, provide valuable style information, significantly reducing the need for manual reward shaping. During training, this style information — together with similarly structured observations from the simulator — is provided to a discriminator within the AMP framework. The discriminator evaluates how closely the humanoid motions in simulation resemble the preprocessed mocap data. Once the RL policy is trained, it serves as a data generator for subsequent training of diffusion-based policies.



Dataset	Metric	No rand.	Dynamics rand.	Perturbations	Terrain	3 Scales	Init. State rand.	Terrain + Perturb.	All
500K	Success Rate	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.01 ± 0.0	0.0 ± 0.0
	Tracking lin. vel.	1.73 ± 0.09	0.92 ± 0.11	1.05 ± 0.13	1.34 ± 0.06	1.5 ± 0.18	1.45 ± 0.12	0.87 ± 0.19	1.0 ± 0.18
	Smoothness	49.36 ± 4.06	89.31 ± 13.48	97.02 ± 15.08	57.0 ± 4.65	273.25 ± 81.04	275.73 ± 33.22	119.02 ± 37.88	96.86 ± 27.21
2M	Success Rate	0.0 ± 0.0	0.91 ± 0.0	$\mathbf{1.0 \pm 0.0}$	0.99 ± 0.0	0.01 ± 0.0	0.16 ± 0.01	$\mathbf{1.0 \pm 0.0}$	$\mathbf{1.0 \pm 0.0}$
	Tracking lin. vel.	1.02 ± 0.11	0.32 ± 0.05	0.15 ± 0.01	0.19 ± 0.07	1.11 ± 0.15	0.96 ± 0.24	0.15 ± 0.01	0.16 ± 0.01
	Smoothness	18.93 ± 2.56	29.17 ± 10.0	3.61 ± 0.27	5.84 ± 19.5	227.38 ± 63.88	127.82 ± 60.74	3.24 ± 0.33	3.13 ± 0.17
8M	Success Rate	0.0 ± 0.0	0.95 ± 0.0	$\mathbf{1.0 \pm 0.0}$	$\mathbf{1.0 \pm 0.0}$	$\mathbf{1.0 \pm 0.0}$	$\mathbf{1.0 \pm 0.0}$	$\mathbf{1.0 \pm 0.0}$	$\mathbf{1.0 \pm 0.0}$
	Tracking lin. vel.	0.9 ± 0.16	0.26 ± 0.07	$\mathbf{0.13 \pm 0.0}$	0.14 ± 0.01	$\mathbf{0.13 \pm 0.0}$	0.13 ± 0.01	0.14 ± 0.0	$\mathbf{0.13 \pm 0.0}$
	Smoothness	56.51 ± 28.58	27.94 ± 6.12	2.97 ± 0.12	2.99 ± 0.1	3.11 ± 0.1	3.22 ± 0.21	$\mathbf{2.9 \pm 0.05}$	2.92 ± 0.07

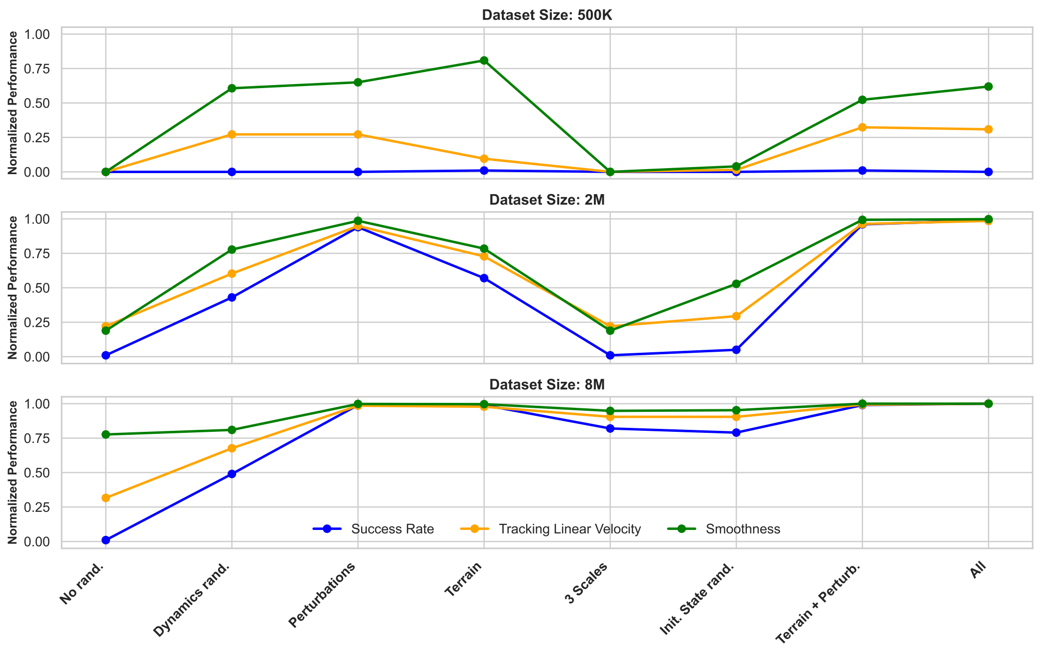
Figure 2: Evaluation of Diffusion Policies in a non-randomized target environment. **Top**: A plot displaying the normalized performances of all configurations, with tracking performance and smoothness inverted for unified metrics (higher values indicate better performance). **Bottom**: A table presenting detailed results, including the success rate (higher is better), tracking performance (lower is better), and smoothness (lower is better).

5 Results

We gathered a total of 24 datasets, which comprise 8 distinct environment setups and 3 dataset sizes: 500K, 2M, and 8M transitions of observations and actions. Each diffusion policy is trained with 3 different random seeds. To assess the performance of the trained DP across various environments, we create two validation environments: one without DR and another with DR. Each evaluation lasts for 10 seconds, which corresponds to 500 simulation steps. The commanding velocity is set to 1 m/s in the forward direction. We evaluate the performance using three metrics: success rate, tracking performance, and smoothness. The success rate is defined as the number of environments that do not terminate before the end of the episode. The tracking performance is measured by the Euclidean distance between the current and commanded velocities. Smoothness is the sum of squared L^2 norms between two consecutive actions. Training times on an Nvidia V100-16GB GPU are as follows: approximately 1 hour and 25 minutes for a dataset of 500,000 samples, about 5 hours for a dataset of 2 million samples, and roughly 23 hours and 17 minutes for a dataset of 8 million samples.

5.1 Evaluation on Non-Randomized Target Environment

First, we evaluate DPs in a non-randomized target environment. All DPs are trained on datasets generated with various randomizations during data collection. Figure 2 presents the results for this setting. As observed, the DP does not achieve stable walking, even when DR is applied with



Dataset	Metric	No rand.	Dynamics rand.	Perturbations	Terrain	3 Scales	Init. State rand.	Terrain + Perturb.	All
	Success Rate	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.01 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.01 ± 0.0	0.0 ± 0.0
500K	Tracking lin. vel.	1.5 ± 0.22	1.13 ± 0.27	1.13 ± 0.12	1.37 ± 0.12	1.5 ± 0.22	1.48 ± 0.16	1.06 ± 0.17	1.08 ± 0.16
	Smoothness	277.23 ± 85.87	111.22 ± 21.34	99.39 ± 13.14	55.98 ± 10.53	277.23 ± 85.87	266.3 ± 37.24	134.12 ± 30.89	107.77 ± 27.75
	Success Rate	0.01 ± 0.0	0.43 ± 0.0	0.94 ± 0.0	0.57 ± 0.01	0.01 ± 0.0	0.05 ± 0.0	0.96 ± 0.0	0.99 ± 0.0
2M	Tracking lin. vel.	1.2 ± 0.15	0.68 ± 0.47	0.21 ± 0.11	0.51 ± 0.38	1.2 ± 0.15	1.1 ± 0.2	0.19 ± 0.06	0.16 ± 0.03
	Smoothness	225.54 ± 48.11	64.53 ± 42.46	7.41 ± 10.93	62.68 ± 78.58	225.54 ± 48.11	132.6 ± 51.38	5.54 ± 6.31	4.21 ± 1.11
	Success Rate	0.01 ± 0.0	0.49 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.82 ± 0.0	0.79 ± 0.0	0.99 ± 0.0	$\mathbf{1.0 \pm 0.0}$
8M	Tracking lin. vel.	1.07 ± 0.16	0.58 ± 0.44	0.16 ± 0.05	0.17 ± 0.05	0.27 ± 0.21	0.27 ± 0.19	0.15 ± 0.02	$\mathbf{0.14 \pm 0.02}$
	Smoothness	64.77 ± 29.29	55.73 ± 34.79	4.15 ± 2.31	4.61 ± 7.17	17.89 ± 37.34	16.58 ± 26.01	$\mathbf{3.62 \pm 0.9}$	3.68 ± 0.69

Figure 3: Evaluation of Diffusion Policies in a randomized target environment. Evaluation of Diffusion Policies in a non-randomized target environment. **Top**: A plot displaying the normalized performances of all configurations, with tracking performance and smoothness inverted for unified metrics (higher values indicate better performance). **Bottom**: A table presenting detailed results, including the success rate (higher is better), tracking performance (lower is better), and smoothness (lower is better).

a dataset size of 500K samples. Starting from 2M samples, some configurations achieve robust walking, although DR becomes crucial for success. It is evident that some randomizations are more significant than others: while the configurations with dynamics, perturbations, and terrain randomization achieve a success rate close to 1.0, scale and initial state randomizations do not yield successful policies. Nevertheless, with a large dataset size of 8M, most randomizations produce strong results, with the notable exception of dynamics randomization, which achieves a success rate of 95%.

For reference, we report the performance of the source RL policy in the same fixed target environment presented in Figure 2. Comparing Table 3 with the 8M+All setting from Figure 2, we see that our best Diffusion Policy can match the performance of the source RL policy.

Table 3: Source RL policy performance in a fixed target environment (100 runs)

	Success Rate	1.0
Baseline	Tracking lin. vel.	0.12 ± 0.0
	Smoothness	2.89 ± 0.02

5.2 Evaluation on Randomized Target Environment

Second, we evaluate DPs in a randomized target environment. We generate this environment by applying milder domain randomization to provide a fair setup for all configurations. To do so, we applied only terrain and dynamics randomization, where we reduced the range of dynamics randomization. Detailed parameters are shown in Table 4.

Once more, all DPs are trained on datasets generated with various randomizations during data collection. When comparing these results to those from the non-randomized target environment, the impact of each randomization cluster becomes clearer. In the results using 500K samples, no configuration achieves stable walking. However, starting from 2M samples, perturbations yield strong results, while other configurations perform poorly. With a larger dataset size of 8M samples, perturbation and terrain randomization demonstrate the strongest outcomes, whereas the other configurations perform less effectively, with dynamics randomization only achieving a success rate of 50%.

Table 4: DR during evaluation

Parameter	**Range**
Body friction	[0.8, 1.2]
Added base mass [kg]	[-1., 1]
Link mass multiplier	[0.9, 1.1]
PD gains multiplier	[0.9, 1.1]
COM displacement [m]	[-0.1, 0.1]
Joint friction coeff.	[0.01, 0.5]
Joint damping coeff.	[0.3, 1.]

Comparing DPs performance to the source RL policy, we again find that our best DP is able to match the RL policy performance, despite target environment randomizations as can be seen in Table 5.

Table 5: Source RL policy performance in 100 randomized target environments

	Success Rate	1.0
Baseline	Tracking lin. vel.	0.13 ± 0.02
	Smoothness	3.45 ± 0.36

6 Discussion

Our results reveal two key findings. First, we find that DR is essential across all dataset sizes, even when the DP is evaluated in environments equivalent to those used for collecting the training dataset (see 5.1). Second, we note that the dataset size also plays a significant role when applying DR. This contrasts sharply with previous work in DP for manipulation, where often only a few expert trajectories are sufficient to accomplish a task. This finding emphasizes the importance of both DR and dataset size in complex and dynamic tasks, such as whole-body humanoid control.

We also found that not all randomizations are of equal importance. For some of the randomization setups, an increase of the dataset size from 2M to 8M leads to significantly better performance (cf. Figure 2 and Figure 3, columns “3 Scales” and “Init state rand.”). In other cases, however, the improvement in performance is marginal, and the overall performance is low (cf. Figure 3, columns “No rand.” and Dynamics), highlighting the fact that the dataset size alone cannot compensate for the lack of diversity in the training data.

In the settings, where the data was collected with external perturbations and on different terrains (cf. Figure 2 and Figure 3, columns Perturbations, Terrain+Perturbations and All), DPs could achieve high success rate, tracking performance, and smoothness on the datasets of size 2M. Increasing the dataset size to 8M only slightly improved the results.

Another interesting finding is that RL policies trained under extensive DR perform well on both smaller and larger body scales compared to the original setup. Despite the highly nonlinear changes in the robot dynamics parameters, such as the masses and inertias of the links, the policy maintains almost the same performance across different scales when the PD gains are scaled by a factor of k^4.

7 Conclusion

We evaluated the effects of dataset size and diversity on training of DPs for whole-body humanoid control. In particular, we investigated which randomization configurations have the largest impact on the training performance. We found that terrain and perturbation randomization are the most important configurations to ensure high data coverage and therefore good generalization performance of the trained DPs. Additionally, we found that the dataset size plays a crucial role when learning highly dynamic tasks such as whole-body humanoid control. We believe that our findings will provide helpful insights for further research efforts in whole-body humanoid control using DP.

Acknowledgments

This research was supported by the Research Clusters “The Adaptive Mind” and “Third Wave of AI,” funded by the Excellence Program of the Hessian Ministry of Higher Education, Science, Research and the Arts, and by the grant for establishing a DFKI lab at TU Darmstadt. We acknowledge computing time on the Lichtenberg II HPC at TU Darmstadt, funded by the BMBF and the State of Hesse. Partial support was also provided by the DFG under grant SE1042/41-1.

References

[1] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[2] T. Pearce, T. Rashid, A. Kanervisto, D. Bignell, M. Sun, R. Georgescu, S. V. Macua, S. Z. Tan, I. Momennejad, K. Hofmann, and S. Devlin. Imitating human behaviour with diffusion models. In *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2023.

[3] Y. Ze, Z. Chen, W. Wang, T. Chen, X. He, Y. Yuan, X. B. Peng, and J. Wu. Generalizable humanoid manipulation with improved 3d diffusion policies. *arXiv preprint arXiv:2410.10803*, 2024.

[4] T. He, Z. Luo, X. He, W. Xiao, C. Zhang, W. Zhang, K. Kitani, C. Liu, and G. Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024.

[5] M. Reuss, M. Li, X. Jia, and R. Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

[6] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy. *arXiv preprint arXiv:2403.03954*, 2024.

[7] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024.

[8] 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*, pages 91–100. PMLR, 2022.

[9] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath. Real-world humanoid locomotion with reinforcement learning. *Science Robotics*, 9(89):eadi9579, 2024.

[10] X. Huang, Y. Chi, R. Wang, Z. Li, X. B. Peng, S. Shao, B. Nikolic, and K. Sreenath. Diffuse-loco: Real-time legged locomotion control with diffusion from offline datasets. *arXiv preprint arXiv:2404.19264*, 2024.

[11] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics*, 37 (4):143:1–143:18, 2018. doi:10.1145/3197517.3201311.

[12] J. Ho and S. Ermon. Generative adversarial imitation learning. In *Proceeding of the Thirtieth Conference on Neural Information Processing Systems*, Barcelona, Spain, Dec. 2016.

[13] J. Fu, K. Luo, and S. Levine. Learning robust rewards with adversarial inverse reinforcement learning. In *Proceeding of the International Conference on Learning Representations*, Vancouver, Canada, Apr. 2018.

[14] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics*, 40(4):1–20, 2021. doi:10.1145/3450626.3459670.

[15] D. Garg, S. Chakraborty, C. Cundy, J. Song, and S. Ermon. Iq-learn: Inverse soft-q learning for imitation. In *Proceeding of the Thirty-fifth Conference on Neural Information Processing Systems*, Sydney, Australia, Dec. 2021.

[16] F. Al-Hafez, D. Tateo, O. Arenz, G. Zhao, and J. Peters. LS-IQ: Implicit reward regularization for inverse reinforcement learning. In *Proceeding of the International Conference on Learning Representations*, Kigali, Rwanda, May 2023.

[17] A. Escontrela, X. B. Peng, W. Yu, T. Zhang, A. Iscen, K. Goldberg, and P. Abbeel. Adversarial motion priors make good substitutes for complex reward functions. *arXiv preprint arXiv:2203.15103*, 2022.

[18] T. He, Z. Luo, W. Xiao, C. Zhang, K. Kitani, C. Liu, and G. Shi. Learning human-to-humanoid real-time whole-body teleoperation. *arXiv preprint arXiv:2403.07208*, 2024.

[19] B. Kang, X. Ma, C. Du, T. Pang, and S. Yan. Efficient diffusion policies for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[20] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[21] A. Z. Ren, J. Lidard, L. L. Ankile, A. Simeonov, P. Agrawal, A. Majumdar, B. Burchfiel, H. Dai, and M. Simchowitz. Diffusion policy policy optimization. *arXiv preprint arXiv:2409.00588*, 2024.

[22] G. V. S. Mothish, M. Tayal, and S. Kolathaya. Birodiff: Diffusion policies for bipedal robot locomotion on unseen terrains. *arXiv preprint arXiv:2407.05424*, 2024.

[23] 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)*, pages 23–30, 2017.

[24] A. M. James, A. Makoviychuk, T. W. Yu, J. Tan, and D. Fox. Glide: Generalizable quadrupedal locomotion in diverse environments with a centroidal model. *Robotics: Science and Systems*, 2022.

[25] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics*, 40:1 – 20, 2021.

[26] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. J. Black. Amass: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5442–5451, 2019.

[27] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia, May 2018.

[28] F. Al-Hafez, G. Zhao, J. Peters, and D. Tateo. LocoMuJoCo: A comprehensive imitation learning benchmark for locomotion. In *6th Robot Learning Workshop, NeurIPS*, New Orleans, Louisiana, United States, Dec. 2023.

[29] F. Al-Hafez, G. Zhao, J. Peters, and D. Tateo. Time-efficient reinforcement learning with stochastic stateful policies. In *ICLR*, 2024.