

Supplementary Material

Multi-Loco: Unifying Multi-Embodiment Legged Locomotion via Reinforcement Learning Augmented Diffusion

Anonymous Author(s)

Affiliation

Address

email

One policy for multiple robots Multi Loco

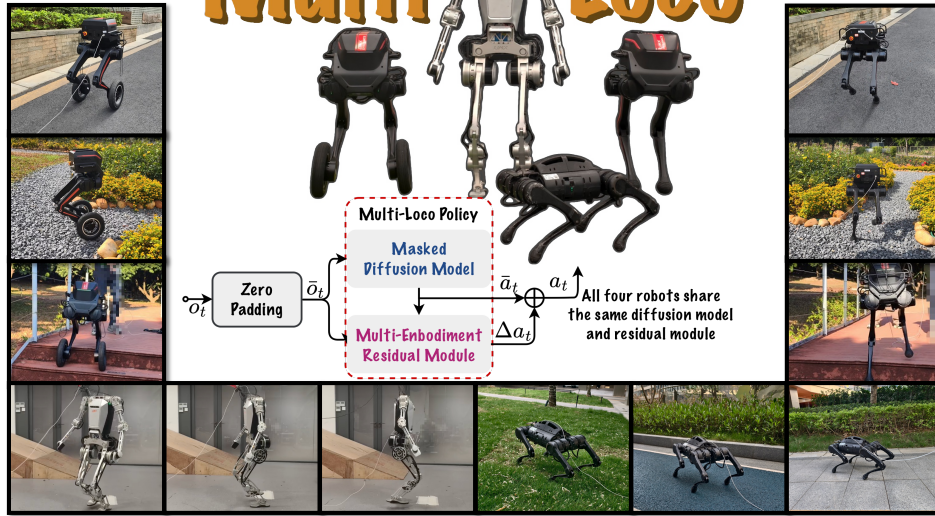


Figure 1: Deployment of the reinforcement learning augmented diffusion policy on four platforms (biped, wheeled biped, humanoid and quadruped). The experimental results demonstrate that the unified policy can effectively control the robots across various types of uneven terrain, including grass, slopes, stairs, and gravel paths. These results highlight the policy’s robustness and exceptional control capabilities.

Appendix

Experiment Videos

We conducted comprehensive real-world evaluations of our framework across four distinct legged robotic platforms. For detailed empirical validation, we encourage readers to view the supplementary video. As demonstrated in the experimental recordings, our framework demonstrates the capability of a unified control policy to govern four morphologically diverse legged robots with different actuator configurations and mass distributions. This cross-platform adaptability is achieved through our novel methodology, which enables robust policy generalization while maintaining dynamic locomotion performance.

A Details of Diffusion Model Training and Inference

DDPM, as one kind of diffusion models, has achieved remarkable results in tasks such as robotic manipulation and locomotion for legged robots due to its ability to represent multimodality and complex distributions. Due to the high number of denoising steps required by DDPM, the inference time can be quite long. For humanoid robots, the diffusion policy must operate at a frequency of at least 50 Hz, ideally reaching 100 Hz. Therefore, it is necessary to either accelerate its performance or replace it with alternatives such as DDIM, EDM, or consistency models. In this context, we have chosen the EDM model to ensure high sample quality while minimizing the inference time to enable high-frequency feedback control.

EDM (Elucidated Diffusion Model) [1] is based on the reverse-time ODE of variance-exploding SMLD [2]. In order to elucidate the design space of diffusion model, EDM proposes a more general form while considering time-dependent scaling and introduces preconditioning to cancel the effects caused by increasing noise variance in denoising score matching. EDM reparametrized the denoiser $D_\theta(\hat{\mathbf{x}}, \sigma)$ as the following form

$$D_\theta(\hat{\mathbf{x}}, \sigma) = c_{\text{skip}}(\sigma)\hat{\mathbf{x}} + c_{\text{out}}(\sigma)F_\theta(c_{\text{in}}(\sigma)\hat{\mathbf{x}}, c_{\text{noise}}(\sigma)). \quad (1)$$

and the corresponding objective of denoising score matching is changed to

$$\mathbb{E}_{\mathbf{x}, \mathbf{n}} \left[\frac{c_{\text{out}}(\sigma)^2}{\sigma} \|F_\theta(c_{\text{in}}(\sigma)(\mathbf{x} + \mathbf{n}), c_{\text{noise}}(\sigma)) - \mathbf{x}_d\|_2^2 \right] \quad (2)$$

where $\mathbf{x}_d = (\mathbf{x} - c_{\text{skip}}(\sigma)(\mathbf{x} + \mathbf{n}))/c_{\text{out}}(\sigma)$. With the requirements of unit variance of input and output while minimizing c_{out} , the value of parameters are chosen as below:

$$c_{\text{noise}}(\sigma) = \ln(\sigma)/4, \quad (3a)$$

$$c_{\text{in}}(\sigma) = 1/\sqrt{\sigma_{\mathbf{x}}^2 + \sigma^2}, \quad (3b)$$

$$c_{\text{skip}}(\sigma) = \sigma_{\mathbf{x}}^2/(\sigma^2 + \sigma_{\mathbf{x}}^2), \quad (3c)$$

$$c_{\text{out}}(\sigma) = \sigma \cdot \sigma_{\mathbf{x}}/(\sigma^2 + \sigma_{\mathbf{x}}^2), \quad (3d)$$

where $\sigma_{\mathbf{x}}$ is the variance of the data. [1] suggests that the optimal choice for this function is $\sigma(t) = t$, an approach we also adopt in our work. As a result, EDM solves the following probability flow ODE

$$\frac{d\mathbf{x}}{dt} = \frac{\mathbf{x} - D_\theta(\mathbf{x}, \sigma)}{t} \quad (4)$$

for sampling and starting from $\mathbf{x}(T) \sim \mathcal{N}(0, T^2 \mathbf{I})$ and stopping at $\mathbf{x}(0)$. In practice, we use Euler method to solve this ODE for sampling.

In terms of network architecture, diffusion models commonly utilize U-Net and Transformer-based structures. In our scenario, we chose DiT (Denoising Transformer) model [3] as backbone to fit $F_\theta(\cdot)$. The padded observations $\bar{\mathbf{o}}_t$ as condition variable was embed into a latent space by a multilayer perceptron (MLP) denoted by $g(\cdot)$. To incorporate $\bar{\mathbf{o}}_t$ as condition variable, $D_\theta(\hat{\mathbf{x}}, \sigma)$ is rewritten as $D_\theta(\hat{\mathbf{a}}_t, \bar{\mathbf{o}}_t, \sigma)$ and $F_\theta(c_{\text{in}}\hat{\mathbf{x}}, c_{\text{noise}}(\sigma))$ is changed to $F_\theta(c_{\text{in}}\hat{\mathbf{a}}_t, g(\bar{\mathbf{o}}_t), c_{\text{noise}}(\sigma))$. The details of the network structure can be found in Fig. 2.

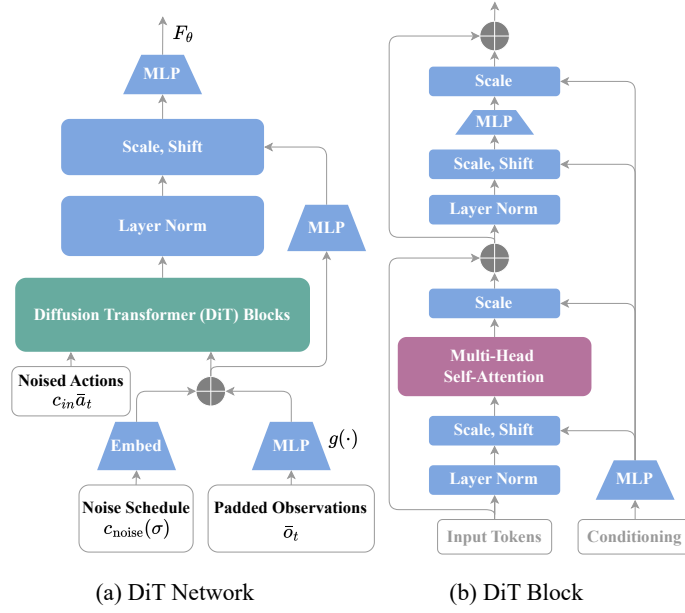


Figure 2: Neural network structure of DiT which is used to fit F_θ .

Parameter	Value / Description
Data Samples	
Prediction Horizon	4
Observation Horizon	10
Maximum Samples per Robot	2048000
Batch Size	8192
Diffusion Network (DiT)	
Input Dimension	20
Embedding Dimension (emb_dim)	128
Model Dimension (d_model)	256
Number of Attention Heads (n_heads)	8
Network Depth (depth)	3
Timestep Embedding Type	Fourier
MLP Condition Network $g(\cdot)$	
Input Dimension	683
Output Dimension	emb_dim
Hidden Dimensions (hidden_dim)	[512, 256]
Activation Function	ELU
Training Parameters	
Learning Rate	3e-4
Optimizer	Adam
Learning Rate Scheduler	Cosine Annealing
EMA Rate	0.999
Number of Epochs	500
Seed	3407 (inspired by [4])

Table 1: Configurations for Unified Diffusion Model

Algorithm 1 Masked EDM Training and Inference

```
1: Initialize:
2: - Networks: DiT  $F_\theta$  and Condition MLP  $g_\phi$ 
3: - EDM params:  $\sigma_{\text{data}} = 0.5$ ,  $\sigma_{\text{min}} = 0.002$ ,  $\sigma_{\text{max}} = 80.0$ ,  $\rho = 7.0$ 
4: - Noise sampling:  $P_{\text{mean}} = -1.2$ ,  $P_{\text{std}} = 1.2$ 
5: - Training: batch size = 512, learning rate =  $3 \times 10^{-4}$ , EMA rate = 0.999
6: procedure DATAPREPROCESSING
7:   Extract trajectory segments (observations  $\mathbf{o}$ , actions  $\mathbf{a}$ )
8:   Apply zero-padding to handle variable-length sequences
9:   Compute quantile-based MinMax normalization:
10:  - Calculate 5% and 95% quantiles for each feature dimension
11:  - Normalize to  $[-1, 1]$  range:  $\mathbf{x}_{\text{norm}} = 2 \cdot \frac{\mathbf{x} - \mathbf{x}_{\text{min}}}{\mathbf{x}_{\text{max}} - \mathbf{x}_{\text{min}}} - 1$ 
12:   Create condition vector by concatenating normalized obs and commands
13:   return Normalized dataset, normalization statistics
14: end procedure
15: procedure TRAIN(Dataset  $\mathcal{D}$ , Epochs = 400K)
16:   for each epoch do
17:     for batch  $(\mathbf{o}, \mathbf{a}_0, \mathbf{b}) \sim \mathcal{D}$  do
18:       Sample noise  $\sigma \sim e^{\mathcal{N}(-1.2, 1.2^2)}$ ,  $\epsilon \sim \mathcal{N}(0, I)$ 
19:       Compute noisy actions  $\mathbf{a}_t = \mathbf{a}_0 + \sigma \cdot \epsilon$ 
20:       Compute EDM preconditioning coefficients:
21:        $c_{\text{skip}} = \frac{0.5^2}{0.5^2 + \sigma^2}$ 
22:        $c_{\text{out}} = \frac{\sigma \cdot 0.5}{\sqrt{0.5^2 + \sigma^2}}$ 
23:        $c_{\text{in}} = \frac{1}{\sqrt{0.5^2 + \sigma^2}}$ 
24:        $c_{\text{noise}} = 0.25 \cdot \log \sigma$ 
25:       Compute network prediction:  $D_\theta(\mathbf{a}_t, \sigma, g_\phi(\mathbf{o}))$ 
26:       Compute masked loss:  $\mathcal{L} = ((1 - \mathbf{b}) \cdot (D_\theta(\mathbf{a}_t, \sigma, g_\phi(\mathbf{o})) - \mathbf{a}_0))^2$ 
27:       Apply EDM weighting:  $\mathcal{L}_{\text{final}} = \left( \mathcal{L} \cdot \frac{0.5^2 + \sigma^2}{(\sigma \cdot 0.5)^2} \right) . \text{mean}()$ 
28:       Update parameters using Adam ( $\eta = 3 \times 10^{-4}$ )
29:       Update EMA model parameters (rate = 0.999)
30:     end for
31:   end for
32: end procedure
33: procedure INFERENCE(Observation  $\mathbf{o}$ , Action mask  $\mathbf{b}$ , Sampling steps  $S = 5$ )
34:   Normalize observations using stored statistics
35:   Initialize  $\mathbf{a}_S \sim \mathcal{N}(0, 80.0^2 \cdot I)$ 
36:   Compute noise schedule:  $\sigma_i = (0.002^{1/7} + \frac{i}{S}(80.0^{1/7} - 0.002^{1/7}))^7$  for  $i \in \{0, 1, \dots, S\}$ 
37:   for  $i = S$  down to 1 do
38:     Apply action mask:  $\mathbf{a}'_i = (1 - \mathbf{b}) \cdot \mathbf{a}_i$ 
39:     Compute network prediction with preconditioning
40:     ODE update:  $\dot{\mathbf{a}} = \frac{\mathbf{a}_i - D_\theta(\mathbf{a}'_i, \sigma_i, g_\phi(\mathbf{o}))}{\sigma_i}$ 
41:     Euler step:  $\mathbf{a}_{i-1} = \mathbf{a}_i - \dot{\mathbf{a}} \cdot (\sigma_i - \sigma_{i-1})$ 
42:   end for
43:   Denormalize using stored statistics
44:   return  $\mathbf{a}_{\text{final}}$ 
45: end procedure
```

37 A.1 Hyperparameters and Training Hardware

38 The DiT model configuration and training parameters are listed in Table. 1. Our implementation
 39 builds on the open-source *CleanDiffuser* library [5], which provides a modular and extensible inter-
 40 face for diffusion models in decision-making tasks. We adopt its implementation of EDM sampling
 41 for both training and inference. If the model is trained on a single NVIDIA 3090 GPU, it costs total
 42 training time ranging from 3 to 12 hours depending on robot embodiment and dataset size.

43 A.2 Details of Training and Inference

44 Our diffusion-based policy handles diverse embodiments through a unified framework that accom-
 45 modates variable-dimensional observation and action spaces. During training, we normalize all
 46 inputs using quantile-based MinMax normalization with 5th and 95th percentiles to mitigate the ef-
 47 fects of outliers. This maps features to a common $[-1, 1]$ range while preserving the relative scaling
 48 of the majority of data points.

49 To support cross-embodiment generalization, we zero-pad observation and action vectors to match
 50 the largest dimension across all robot morphologies (observations: 68D, actions: 20D) and the
 51 dimension configurations of . During both training and inference, we apply morphology-specific
 52 binary masks \mathbf{b} to ensure only valid dimensions contribute to the loss computation and prediction.
 53 Specifically, during training, the loss is computed as: $\mathcal{L} = ((1 - \mathbf{b}) \cdot (D_\theta(\mathbf{a}_t, \sigma, g_\phi(\mathbf{o})) - \mathbf{a}_0))^2$

54 During inference, we employ the Euler method with 5 sampling steps to solve the ODE and generate
 55 actions. At each denoising step, we apply the action mask to ensure prediction consistency: $\mathbf{a}'_i =$
 56 $(1 - \mathbf{b}) \cdot \mathbf{a}_i$.

Robot	Observation	Action	Command	Components
Point-Foot Biped	$\mathcal{O}_1 \in \mathbb{R}^{26}$	$\mathcal{A}_1 \in \mathbb{R}^6$	$\mathcal{C}_1 \in \mathbb{R}^3$	Base pose (7D), velocity (6D), joint states (6D), gait phase (7D)
Wheeled Biped	$\mathcal{O}_2 \in \mathbb{R}^{28}$	$\mathcal{A}_2 \in \mathbb{R}^8$	$\mathcal{C}_2 \in \mathbb{R}^3$	Base pose (7D), velocity (6D), joint states (8D), wheel states (7D)
Humanoid	$\mathcal{O}_3 \in \mathbb{R}^{68}$	$\mathcal{A}_3 \in \mathbb{R}^{20}$	$\mathcal{C}_3 \in \mathbb{R}^3$	Base pose (7D), velocity (6D), joint states (48D), gait phase (7D)
Quadruped	$\mathcal{O}_4 \in \mathbb{R}^{44}$	$\mathcal{A}_4 \in \mathbb{R}^{12}$	$\mathcal{C}_4 \in \mathbb{R}^3$	Base pose (7D), velocity (6D), joint states (24D), gait phase (7D)

Table 2: Configuration-Specific Observation and Action Spaces

57 A.3 Ablation Study: Diffusion Sampling Steps

58 We evaluate how the number of diffusion sampling steps impacts control performance. Fewer steps
 59 reduce computation time but may degrade trajectory quality. Results (Fig 3) show that using [3,
 60 5,10,15] denoising steps balances control accuracy and sampling efficiency, with diminishing re-
 61 turns beyond 20 steps. According to this results, we choose 5 denoising steps in the deployment
 62 experiments.

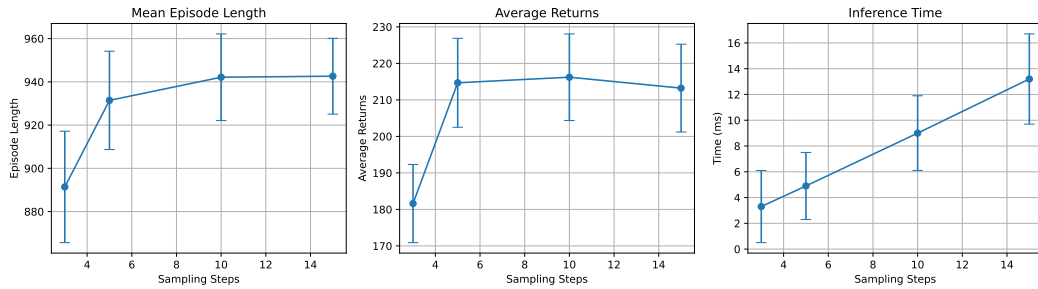


Figure 3: Comparison with Different Sampling Steps in EDM

63 A.4 Ablation Study: Dataset Size for Diffusion Training

64 To assess data efficiency, we train diffusion models on varying dataset sizes (1%, 10%, 25%, 50%,
65 100%) of biped robot while keep others unchanged. The performance scales sublinearly with data,
66 suggesting strong generalization even with partial data. However, extremely small datasets ($< 25\%$)
67 lead to unstable behaviors and poor terrain negotiation.

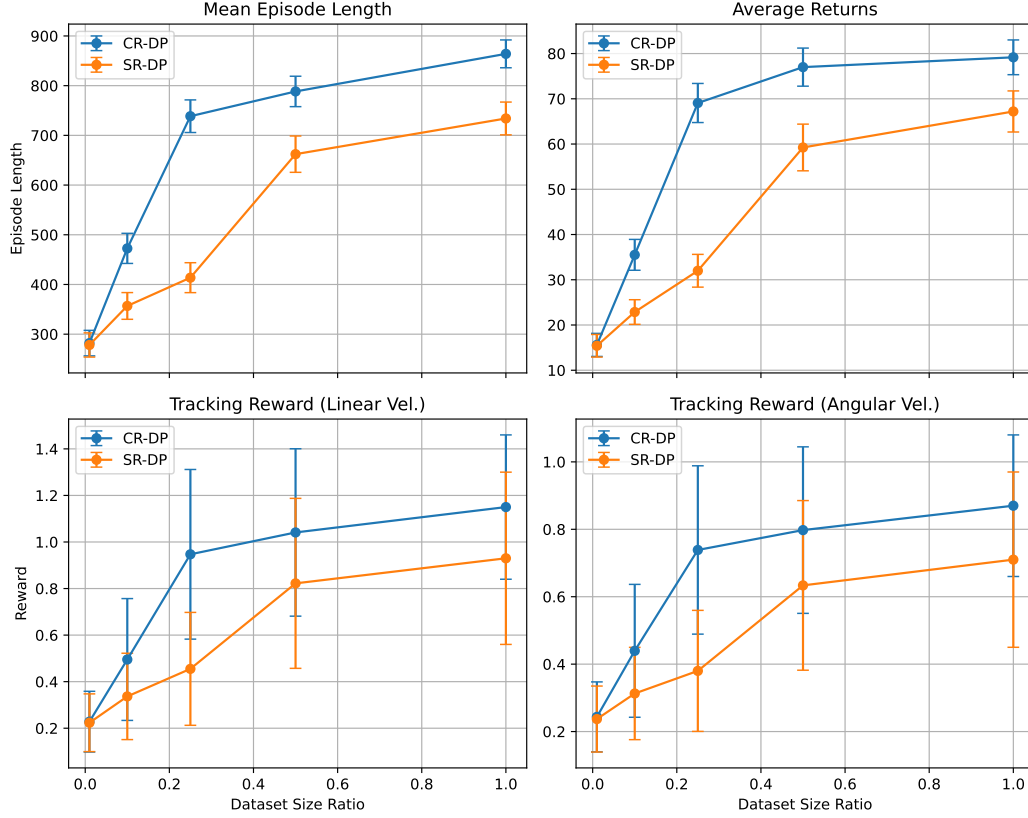


Figure 4: Biped Statistics: Comparison with EDMs Trained with Different Datasets Size. (The maximum number of samples of bipedal robots in the dataset is 2048000)

68 A.5 Ablation Study: Dataset Composition Analysis for Diffusion Training

69 We conducted a systematic investigation into how varying data distribution ratios across robot mor-
70 morphologies impact policy performance while keeping the total dataset size constant. Using a baseline
71 configuration with equal 25%-25%-25%-25% allocations for four robot types (pointed-foot biped,
72 wheeled biped, humanoid, and quadruped), we created four experimental conditions by sequentially
73 reducing one category to 10% while proportionally increasing others to 30%. The resulting Average
74 Return (AR) and Mean Episode Length (MEL) metrics were subsequently analyzed in Fig 5.

75 Reducing data allocation for pointed-foot bipeds and quadrupeds showed negligible performance
76 impacts, suggesting their relatively simple locomotion tasks require minimal training data. Con-
77 versely, decreasing wheeled biped data caused significant performance degradation in its corre-
78 sponding policy, likely due to the inherent complexity of wheeled locomotion combined with limited
79 cross-morphology knowledge transfer from more dissimilar robot types. Most notably, humanoid
80 data reduction demonstrated dual impacts - not only expected self-performance deterioration from
81 reduced training on its high-dimensional observation space, but also an unexpected AR decrease in
82 wheeled biped performance. This cross-domain dependency confirms the hypothesis presented in
83 our main text regarding humanoid data’s complementary role in enhancing wheeled locomotion ca-
84 pabilities, suggesting previously unrecognized synergies between morphologically distinct systems.

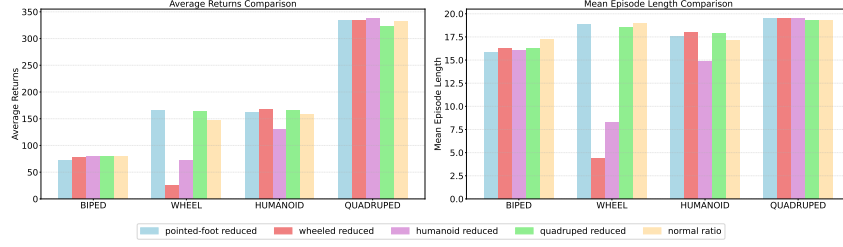


Figure 5: Impact of dataset composition ratios on diffusion policy training. The baseline normal ratio configuration (25%-25%-25%-25%) equally distributes data across four morphologies: pointed-foot biped, wheeled biped, humanoid, and quadruped. Four experimental variations were created by reducing one morphology’s share to 10% while proportionally increasing others to 30% each: Pointed-foot reduced (10%-30%-30%-30%), Wheeled reduced (30%-10%-30%-30%), Humanoid reduced (30%-30%-10%-30%), and Quadruped reduced (30%-30%-30%-10%).

A.6 Zero-Shot Transfer to an Unseen Platform: Unitree Go2

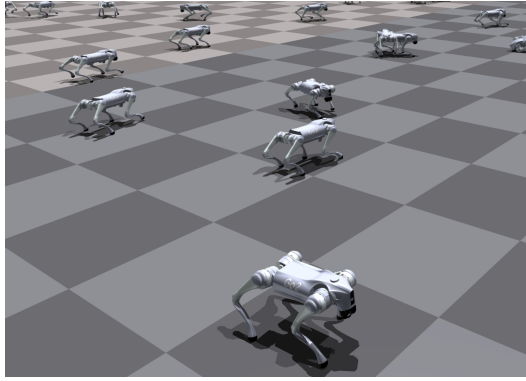


Figure 6: Zero-Shot Transfer to Unitree Go2

To evaluate generalization beyond training morphologies, we directly deploy the unified policy on Unitree Go2 without any finetuning. Despite being unseen during training, the policy achieves stable forward locomotion, demonstrating the model’s morphology-agnostic generalization. Detailed results and rollout videos are available in the supplementary video.

For locomotion task evaluation, the Mean Episode Length (MEL), Locomotion Velocity Tracking (LVT), and Actuator Variation Threshold (AVT) metrics quantify task completion quality. As Table 3 indicates, the statistically insignificant differences of CR-DP+RA in MEL (0.62 ~ 3.2%), LVT (0.22 ~ 4.1%), and AVT (0.23 ~ 5.3%) between configurations a1 and go2 confirm successful zero-shot policy transfer to the go2 morphology.

Also, the simulation outcomes illustrated in Figure 6 confirm successful zero-shot transfer to the go2 configuration, evidencing strong generalization capability of our policy framework. While Table 3 reveals a substantial disparity in Average Return (AR) between CR-DP+RA implementations for a1 (387.57) and go2 (295.96), our analysis identifies this discrepancy as primarily attributable to differing desired base height parameters in the base height tracking reward module. Due to morphological variations between configurations, the optimal base height settings should be configuration-specific. After normalizing for this parameter by subtracting the base height reward component, the adjusted AR values for CR-DP+RA become 306.69 (a1) and 294.28 (go2), indicating comparable performance when accounting for morphological differences. This adjustment validates that the observed AR gap stems from parameter specification rather than fundamental policy limitations.

Group	Method	Metrics			
		AR↑	MEL↑	LVT↑	AVT↑
a1	CR-DP+RA	387.57±6.15	19.60±0.19	5.42±0.41	4.32±0.41
	CR-DP	325.24±8.63	19.14±0.36	5.22±0.72	4.14±0.58
go2	CR-DP+RA	295.96±5.81	18.98±0.33	5.20±0.74	4.09±0.60
	CR-DP	256.05±5.72	18.40±0.37	5.02±0.91	3.93±0.74

Table 3: Performance Comparison of quadruped a1 and go2

B RL residual adaptation Detail

While diffusion models provide strong priors for action generation, they may lack task-awareness or fail to account for nuanced terrain interactions. To complement the generative prior, we introduce a residual policy trained via reinforcement learning, detailed below.

Our implementation leverages NVIDIA IsaacGym’s GPU-accelerated parallel simulation environment specifically designed for robotic learning. The system architecture integrates with the established from `humanoid_gym` [6] and `legged_gym` [7] open-source frameworks, implementing a scalable reinforcement learning pipeline based on Proximal Policy Optimization (PPO).

B.1 Environment Setup

The policy observations \mathbf{o} for the robot include velocity control commands and proprioceptive sensory data with gait cycle parameters (excluding wheeled locomotion) in the past 10 steps. Specifically:

- Proprioceptive Data:
 - Robot pose (orientation) and angular velocity measured by the IMU.
 - Joint angles and angular velocities of all robotic limbs.
- Velocity Control Commands:
 - Linear velocity commands in the XY-plane (Cartesian coordinates).
 - Angular velocity command around the Z-axis (yaw direction).
- Gait Cycle:

A time-dependent periodic signal defined by parametric sinusoidal curves:

$$\text{Gait}(t) = \begin{cases} \sin\left(\frac{2\pi t}{T}\right) \\ \cos\left(\frac{2\pi t}{T}\right) \end{cases}$$

where T is the gait period (time to complete one cycle), t represents the current time step.

The PPO hyperparameters are detailed in Table 4, with a notable configuration choice of disabling value prediction clipping. This design decision stems from our hybrid architecture that integrates a pretrained diffusion model as prior guidance. The pre-trained dynamics awareness enables more stable value function initialization compared with conventional scratch training paradigms.

PPO Parameter	Value
Desired KL	0.01
Learning Rate	4e-4
Discount Factor	0.99
Lambda(GAE)	0.95
Mini Batches	4
Learning Epochs	5
Entropy Loss Scale	0.001
PPO Clip Range	0.2
Values Predicted Clip	False
Residual Coeff	0.2
Max Iterations	10001
Rollouts	24

Table 4: Training Parameters for PPO Unified EDM

Table 5 presents our heterogeneous actor-critic architecture featuring a unified actor network parameters and four specialized critic network parameters.

Model - Critic Biped	
Activations	["elu", "elu", "elu", "linear"]
Hidden Dims	[512, 256, 128]
Model - Critic Biped Wheel	
Activations	["elu", "elu", "elu", "linear"]
Hidden Dims	[512, 256, 128]
Model - Critic Humanoid	
Activations	["elu", "elu", "elu", "linear"]
Hidden Dims	[512, 256, 128]
Model - Critic Quadruped	
Activations	["elu", "elu", "elu", "linear"]
Hidden Dims	[512, 256, 128]
Model - Actor	
Log Std Max	4.0
Log Std Min	-20.0
Std Init	1.0
Activations	["elu", "elu", "elu", "linear"]
Hidden Dims	[512, 256, 128]

Table 5: Actor-Critic Model Parameters

B.2 Reward Design

Our reward function comprises three coordinated components systematically designed for robust policy learning:

- Task-specific objectives governing locomotion performance, shown in Table 6
- Morphology-aware regularization terms addressing physical constraints, shown in Table 7
- A diffusion-guided residual penalty enforcing dynamic feasibility through pretrained motion priors:

$$r_d(\Delta \mathbf{a}_t) = \alpha \|\Delta \mathbf{a}_t\|_1$$

where α is the reward coefficient of residual penalty, $\Delta \mathbf{a}_t$ is the residual action.

Reward	Expression	Biped-Wheel	Pointed-Foot-Biped	Humanoid	Quadruped
Tracking Linear Velocity	$\exp(-\ v_b^{des} - v_b\ \times \sigma)$	4.0	2.0	2.0	6.0
Tracking Angular Velocity	$\exp(-\ \Omega_b^{des} - \Omega\ \times \sigma)$	2.0	1.5	1.5	5.0
Base Height	$\exp(-\ h_b^{des} - h_b\ \times 100)$	-	2.0	4.0	6.0
Orientation	$\exp(-\ \theta_b^{des} - \theta_b\ \times 10)$	5.0	5.0	-10.0	4.0

Table 6: Locomotion Task Reward

Reward	Expression	Biped-Wheel	Pointed-Foot-Biped	Humanoid	Quadruped
Joint Torque	$\ \boldsymbol{\tau}\ ^2$	-1.6e-4	-8e-5	-8e-5	-2e-4
Power	$ \boldsymbol{\tau} \cdot \dot{\mathbf{q}} $	-2e-5	-2e-5	-	-5e-4
Joint Vel	$\ \dot{\mathbf{q}}\ ^2$	-5e-4	-	-	-
Joint Acc	$\ \ddot{\mathbf{q}}\ ^2$	-1.5e-7	-2.5e-7	-2.5e-7	-2.5e-7
Linear Velocity Z	$\ v_b^z\ ^2$	-0.3	-0.5	-2.0	-2.0
Angular Velocity XY	$\ \Omega_{xy}\ ^2$	-0.3	-0.05	-0.05	-0.1
Action Smoothness	$\ a_{k+1} + a_{k-1} - 2a_k\ ^2$	-0.03	-0.01	-0.01	-0.02
Action Rate	$\ a_{k+1} - a_k\ ^2$	-0.03	-0.01	-0.01	-0.02
Collision	$\ \mathbf{F}_c\ > 0.$	-0.1	-0.02	-1.0	-10.0
Contact Force	$\text{clip}\{\ \mathbf{F}_l\ _2 + \ \mathbf{F}_r\ _2 - F_{max}\}_0^{400}$	-0.1	-0.1	-	-
Default Joint Position	$\ \mathbf{q} - \mathbf{q}_0\ $	-0.05	-	3.0	2.0
Foot Distance	$\text{clip}(\mathbf{d}_{foot}^{min} - \mathbf{d}_{foot})$	-100	-100	-100	-
Nominal Foot Height	$\exp(-\ h_{foot}^{des} - h_{foot}\ ^2 \times 200)$	4.0	-	-	3.0

Table 7: Regularization Reward

References

- [1] T. Karras, M. Aittala, T. Aila, and S. Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.
- [2] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.
- [3] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.
- [4] D. Picard. Torch. manual_seed (3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision. *arXiv preprint arXiv:2109.08203*, 2021.
- [5] Z. Dong, Y. Yuan, J. Hao, F. Ni, Y. Ma, P. Li, and Y. Zheng. Cleandiffuser: An easy-to-use modularized library for diffusion models in decision making. *arXiv preprint arXiv:2406.09509*, 2024. URL <https://arxiv.org/abs/2406.09509>.
- [6] X. Gu, Y.-J. Wang, X. Zhu, C. Shi, Y. Guo, Y. Liu, and J. Chen. Advancing humanoid locomotion: Mastering challenging terrains with denoising world model learning. *arXiv preprint arXiv:2408.14472*, 2024.
- [7] 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.