

PD controller that actuates the robot’s degrees of freedom. To optimize the policy, we use the proximal policy optimization (PPO) [80], aiming to maximize the cumulative discounted reward $\mathbb{E} \left[\sum_{t=1}^T \gamma^{t-1} r_t \right]$. We identify several design choices that are crucial for achieving stable policy training:

a) Asymmetric Actor-Critic Training: Real-world humanoid control is inherently a partially observable Markov decision process (POMDP), where certain task-relevant properties that are readily available in simulation become unobservable in real-world scenarios. However, these missing properties can significantly facilitate policy training in simulation. To bridge this gap, we employ an asymmetric actor-critic framework, where the critic network has access to privileged information such as the global positions of the reference motion and the root linear velocity, while the actor network relies solely on proprioceptive inputs and a time-phase variable. This design not only enhances phase-based motion tracking during training but also enables a simple, phase-driven motion goal for sim-to-real transfer. Crucially, because the actor does not depend on position-based motion targets, our approach eliminates the need for odometry during real-world deployment—overcoming a well-documented challenge in prior work on humanoid robots [25, 24].

b) Termination Curriculum of Tracking Tolerance: Training a policy to track agile motions in simulation is challenging, as certain motions can be too difficult for the policy to learn effectively. For instance, when imitating a jumping motion, the policy often fails early in training and learns to remain on the ground to avoid landing penalties. To mitigate this issue, we introduce a termination curriculum that progressively refines the motion error tolerance throughout training, guiding the policy toward improved tracking performance. Initially, we set a generous termination threshold of 1.5m, meaning the episode terminates if the robot deviates from the reference motion by this margin. As training progresses, we gradually tighten this threshold to 0.3m, incrementally increasing the tracking demand on the policy. This curriculum allows the policy to first develop basic balancing skills before progressively enforcing stricter motion tracking, ultimately enabling successful execution of high-dynamic behaviors.

c) Reference State Initialization: Task initialization plays a crucial role in RL training. We find that naively initializing episodes at the start of the reference motion leads to policy failure. For example, in Cristiano Ronaldo’s jumping training, starting the episode from the beginning forces the policy to learn sequentially. However, a successful backflip requires mastering the landing first—if the policy cannot land correctly, it will struggle to complete the full motion from takeoff. To address this, we adopt the Reference State Initialization (RSI) framework [67]. Specifically, we randomly sample time-phase variables between 0 and 1, which effectively randomizes the starting point of the reference motion for the policy to track. We then initialize the robot’s state based on the corresponding reference motion at that phase, including root position and orientation, root linear and angular velocities

and joint positions and velocities. This initialization strategy significantly improves motion tracking training, particularly for agile whole-body motions, by allowing the policy to learn different motion phases in parallel rather than being constrained to a strictly sequential learning process.

d) Reward Terms: We define the reward function r_t with the sum of three terms: 1) penalty, 2) regularization, and 3) task rewards. A detailed summary of these components is provided in Table I.

TABLE I
REWARD TERMS FOR PRETRAINING

Term	Weight	Term	Weight
Penalty			
DoF position limits	−10.0	DoF velocity limits	−5.0
Torque limits	−5.0	Termination	−200.0
Regularization			
Torques	-1×10^{-6}	Action rate	−0.5
Feet orientation	−2.0	Feet heading	−0.1
Slippage	−1.0		
Task Reward			
Body position	1.0	VR 3-point	1.6
Body position (feet)	2.1	Body rotation	0.5
Body angular velocity	0.5	Body velocity	0.5
DoF position	0.75	DoF velocity	0.5

e) Domain Randomizations: To improve the robustness of the pre-trained policy in Figure 2 (a), we utilized basic domain randomization techniques listed in Table VI.

III. POST-TRAINING: TRAINING DELTA ACTION MODEL AND FINE-TUNING MOTION TRACKING POLICY

The policy trained in the first stage can track the reference motion in the real-world but does not achieve high motion quality. Thus, during the second stage, as shown in Figure 2 (b) and (c), we leverage real-world data rolled out by the pre-trained policy to train a delta action model, followed by policy refinement through dynamics compensation using this learned delta action model.

A. Data Collection

We deploy the pretrained policy in the real world to perform whole-body motion tracking tasks (as depicted in Figure 9) and record the resulting trajectories, denoted as $\mathcal{D}^r = \{s_0^r, a_0^r, \dots, s_T^r, a_T^r\}$, as illustrated in Figure 2 (a). At each timestep t , we use a motion capture device and onboard sensors to record the state: $s_t = [p_t^{\text{base}}, v_t^{\text{base}}, \alpha_t^{\text{base}}, \omega_t^{\text{base}}, q_t, \dot{q}_t]$, where $p_t^{\text{base}} \in \mathbb{R}^3$ represents the robot base 3D position, $v_t^{\text{base}} \in \mathbb{R}^3$ is base linear velocity, $\alpha_t^{\text{base}} \in \mathbb{R}^4$ is the robot base orientation represented as a quaternion, $\omega_t^{\text{base}} \in \mathbb{R}^3$ is the base angular velocity, $q_t \in \mathbb{R}^{23}$ is the vector of joint positions, and $\dot{q}_t \in \mathbb{R}^{23}$ represents joint velocities.

B. Training Delta Action Model

Due to the sim-to-real gap, when we replay the real-world trajectories in simulation, the resulting simulated trajectory will likely deviate significantly from real-world recorded trajectories. This discrepancy is a valuable learning signal for learning the mismatch between simulation and real-world