Humanoid Whole-Body Manipulation

Bhaswanth Ayapilla[1,2] **Kshitij Madhav Bhat**[1,2]
[1]Robotics Institute, [2]School of Computer Science
Carnegie Mellon University
{bayapill, kmadhavb}@andrew.cmu.edu

Abstract: Humanoid robots with dexterous hands hold promise for versatile manipulation in human environments, yet learning to control their high-dimensional dynamics (61+ DoF) remains challenging. Recent benchmarks show that state-of-the-art reinforcement learning algorithms require millions of samples and struggle on complex manipulation tasks. We investigate FastTD3 [1], a sample-efficient off-policy algorithm with fast value bootstrapping, on the challenging window cleaning task from HumanoidBench [2]. We propose two modifications: (1) direct application of FastTD3, and (2) PPO [3] augmented with FastTD3-learned value functions for reward shaping. Our experiments show that SAC achieves returns of 60-75 within 2M steps but suffers from systematic policy instability and collapse, while PPO learns slowly but stably, reaching returns of 19-21 over 20M steps. Neither method achieves task success, motivating our proposed modifications. Please refer to the Project Website for more details.

Keywords: Humanoid, Robots, Reinforcement Learning

1 Introduction

The recently introduced HumanoidBench [2] provides comprehensive evaluation of RL algorithms on 27 humanoid tasks. Their benchmarking revealed that state-of-the-art methods such as DreamerV3 [4], TD-MPC2 [5], SAC [6], and PPO [3] struggle significantly on manipulation tasks. Particularly challenging are tasks requiring tool use with coordinated locomotion, such as window cleaning, which combines tool grasping, maintaining tool-surface contact, following prescribed trajectories, and lateral body movement, all simultaneously.

FastTD3 [1] is a recently proposed off-policy algorithm that achieves superior sample efficiency through fast value bootstrapping. By maintaining multiple value estimates at different temporal scales and using them for accelerated policy updates, FastTD3 demonstrates 2-3$\times$ faster learning than TD3 on continuous control benchmarks. However, its effectiveness on high-dimensional humanoid manipulation tasks is still largely unexplored.

In this work, we investigate: (1) Can FastTD3's accelerated learning improve sample efficiency on the window cleaning task? (2) Can FastTD3's learned value functions provide effective reward shaping for on-policy methods like PPO? We focus specifically on the window cleaning task as it represents a realistic household scenario requiring sophisticated coordination between manipulation (tool control) and locomotion (lateral movement).

2 Environment

We use HumanoidBench, a MuJoCo-based simulation environment featuring a *Unitree G1* humanoid robot with two dexterous Shadow Hands. The robot has:

1. Observation space: 151D (51D body proprioception + 50D per hand)
2. Action space: 61D position control (19D body + 21D per hand) at 50 Hz

10th Conference on Robot Learning (CoRL 2026), Austin TX, USA.

3. Physics: MuJoCo MJX with realistic contact dynamics, runs at ~ 1000 FPS

3 Reward Function

We use *Unitree G1* humanoid robot for the `g1-window-v0` task, a whole-body manipulation scenario where the robot must grasp a window wiping tool and keep its tip parallel to a window by following a prescribed vertical velocity. Success requires precise whole-body coordination to maintain tool-surface contact while simultaneously adjusting posture to execute the wiping motion. The reward function drives this behavior through two primary components: a manipulation term that rewards the robot for maintaining a stable upright stance, keeping its hands close to the tool, and tracking a target vertical tool velocity of 0.5 m/s; and a strict contact term that is only active when the tool touches the glass, incentivizing the robot to keep five distinct contact points on the wiper flush against the window pane. Please refer to Table 1 for the details of the reward function.

Table 1: Reward Function Specification for `g1-window-v0`

Component	Weight	Equation / Logic	Parameters (G1)
Total Reward	1.0	$r_{\text{total}} = 0.5 \cdot r_{\text{manip}} + 0.5 \cdot r_{\text{contact}}$	
1. Manipulation Term (r_{manip})			
Sub-weight sum		$r_{\text{manip}} = 0.2 \cdot r_{\text{stable}} + 0.4 \cdot r_{\text{move}} + 0.4 \cdot r_{\text{prox}}$	
a. Stability Composite (r_{stable})	0.10	$r_{\text{stable}} = r_{\text{stand}} \cdot r_{\text{ctrl}} \cdot r_{\text{head}}$	
i. Standing Height	-	$r_{\text{stand\_h}} = \text{tol}(h_{\text{head}})$	bounds $= (1.28, \infty)$, margin $= 0.32$
ii. Upright Torso	-	$r_{\text{upright}} = \text{tol}(u_{\text{torso}})$	bounds $= (0.9, \infty)$, margin $= 1.9$
iii. Actuator Control	-	$r_{\text{ctrl}} = 0.8 + 0.2 \cdot \text{mean}(\text{tol}(F_{\text{act}}))$	bounds $= (0, 0)$, margin $= 10$, sigmoid=quad
iv. Head Dist. Constraint	-	$r_{\text{head}} = \text{tol}(\|p_{\text{head}} - p_{\text{init}}\|)$	bounds $= (0.4, 0.4)$, margin $= 0.1$
b. Tool Velocity (r_{move})	0.20	$r_{\text{move}} = \text{tol}(\|v_z^{\text{tool}}\|)$	bounds $= (0.5, 0.5)$, margin $= 0.5$
c. Hand-Tool Proximity (r_{prox})	0.20	$r_{\text{prox}} = \min(\text{tol}(d_{\text{left}}), \text{tol}(d_{\text{right}}))$	bounds $= (0, 0.2)$, margin $= 0.5$
2. Window Contact Term (r_{contact})			
Sub-weight sum		$r_{\text{contact}} = \mathbb{I}_{\text{collision}} \cdot r_{\text{surface}}$	
a. Contact Filter ($\mathbb{I}_{\text{collision}}$)	Mask	$\mathbb{I} = 1$ if $\text{geom}_{\text{pane}} \cap \text{geom}_{\text{wipe}} \neq \emptyset$	Binary mask (0 or 1)
b. Surface Alignment (r_{surface})	0.50	$r_{\text{surface}} = \min_{i \in \{a..e\}} \text{tol}(x_{\text{site}_i})$ *(alignment of 5 wiper sites to glass x-plane)*	bounds $= (0.92, 0.92)$, margin $= 0.4$

Note: $\text{tol}(v)$ refers to the `dm_control` tolerance function. Weights in the second column represent the effective contribution to the total reward. For G1, the stand height is set to 1.28m.

4 Method

We evaluate two distinct reinforcement learning strategies: a highly-optimized off-policy approach (FastTD3) and an on-policy approach (PPO) augmented with dense rewards derived from the off-policy (FastTD3) agent's value estimates, i.e., the modifications are as follows:

- **Modification 1:** Implement TD3 for the `g1-window-v0` task and compare it against other methods
- **Modification 2:** Change the rewards of PPO for the `g1-window-v0` task to the dense rewards from FastTD3 and compare it against other methods
- **Other Analysis:** We also investigate the hyperparameter dependence of FastTD3, by varying the update-to-data (UTD) ratio, and the distributional critic's range parameters (v_{min} and v_{max}) for FastTD3.

4.1 FastTD3

FastTD3 is a high-performance variant of the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm, specifically optimized for complex robotics tasks by utilizing parallel simulation, large batch sizes, and a distributional critic. Our implementation employs a multi-layer perceptron (MLP) architecture with layers of (1024, 512, 256) units for the critic and (512, 256, 128) for the

actor, trained with a massive batch size of 32,768 to ensure stable learning signals and high data diversity in each update. We also utilize Clipped Double Q-learning (CDQ) to manage function approximation error, which remains critical for stability in the absence of layer normalization. We want to investigate the hyperparameter dependence of FastTD3 by analyzing the update-to-data (UTD) ratio, and the distributional critic's range parameters (v_{min} and v_{max}). Because CDQ and the distributional limits are often task-specific, we aim to quantify how these choices impact the speed and stability of learning for the high-dimensional window cleaning task.

4.2 PPO with FastTD3 Reward

We evaluate the performance of PPO, a popular on-policy algorithm for robotic control, when augmented with reward functions specifically tuned for FastTD3. This approach involves applying the dense reward structures developed for FastTD3, which is characterized by stronger penalty terms designed to produce smooth, visually appealing, and stable gaits, to the PPO training process. We intend to investigate the hyperparameter dependence of this method by examining how PPO responds to the intensity of these penalty-heavy reward structures compared to standard, PPO-tuned rewards. Previous research suggests that PPO can be highly sensitive to these reward definitions; specifically, using rewards with heavy penalties can significantly slow down training and result in overly conservative, slowly-walking gaits. Our analysis will determine if the *FastTD3 recipe* for rewards is algorithm-agnostic or if PPO requires a distinct balance of reward shaping to achieve optimal performance.

5 Experimental Results

The random policy baseline achieved a mean return of 2.87 ± 0.66 over 100 episodes on the `g1-window-v0` task. As expected for random actions, the performance remains consistent with no learning trend.

Figure 1: Episode returns for `g1-window-v0` task with random policy

5.1 Learning Curves (PPO, SAC, Random Baseline)

We evaluate two reinforcement learning approaches on the `g1-window-v0` task: PPO as the on-policy policy-gradient baseline and SAC as the off-policy Q-function-based baseline. We also in-

clude a random policy baseline (mean return of 2.87 ± 0.66 over 100 episodes) as a lower bound reference. All results report mean episode return versus environment steps. The PPO and SAC results are evaluated over three random seeds.

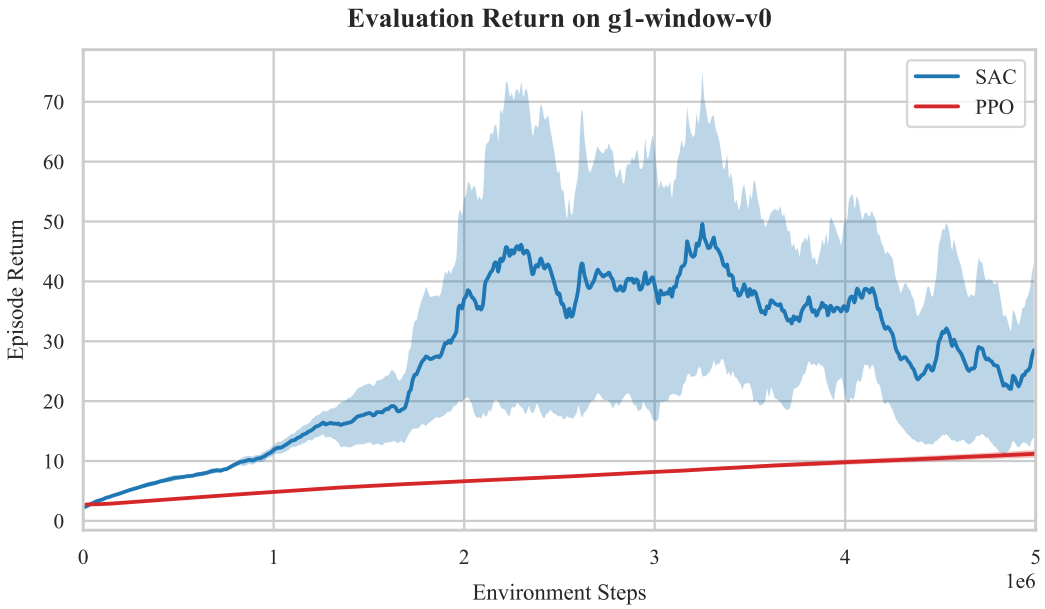

Figure 2: Evaluation returns for `g1-window-v0` task with PPO and SAC (mean $\pm$ std over 3 seeds)

5.2 Analysis of Learning Curves

Figure 2 presents the episode returns for PPO, SAC, and the random policy baseline across three independent seeds each. The two methods tell fundamentally different stories about how and whether an agent can make progress on the `g1-window-v0` task.

5.2.1 PPO

PPO exhibits slow but remarkably consistent learning throughout the entire 20M step training budget. All three seeds start near the random baseline and climb steadily, converging to returns of 19-21 by the end of training, with inter-seed variance never exceeding 2-3 return units at any point (Figure 3). This reproducibility is a direct consequence of PPO's conservative on-policy updates, which constrain the policy from making large jumps between iterations.

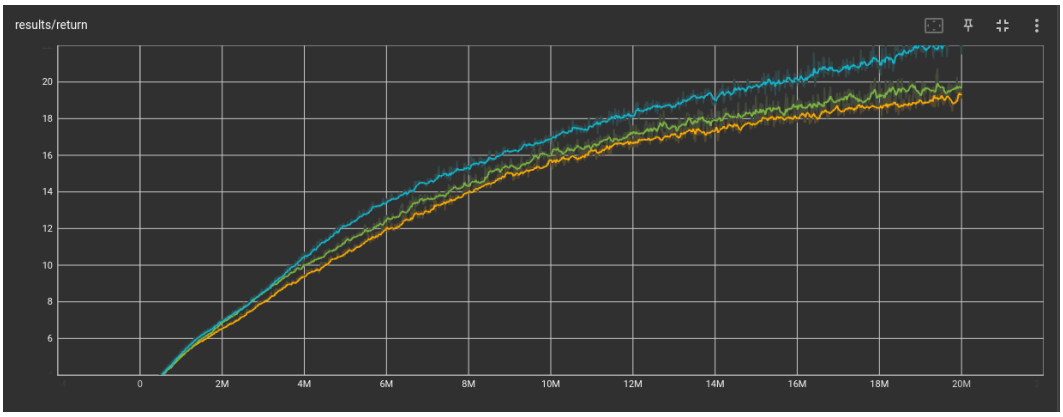

Figure 3: Episode returns for `g1-window-v0` task with PPO on 3 seeds

The episode length curve (Figure 4) tells the same story. Episodes grow from ∼20 steps to ∼65 steps at roughly the same rate as returns, and the three seeds remain nearly indistinguishable throughout. The parallel growth of return and episode length strongly suggests that PPO's reward accumulation is driven primarily by the agent learning to maintain balance and avoid early termination, rather than by discovering any meaningful window-cleaning behavior.

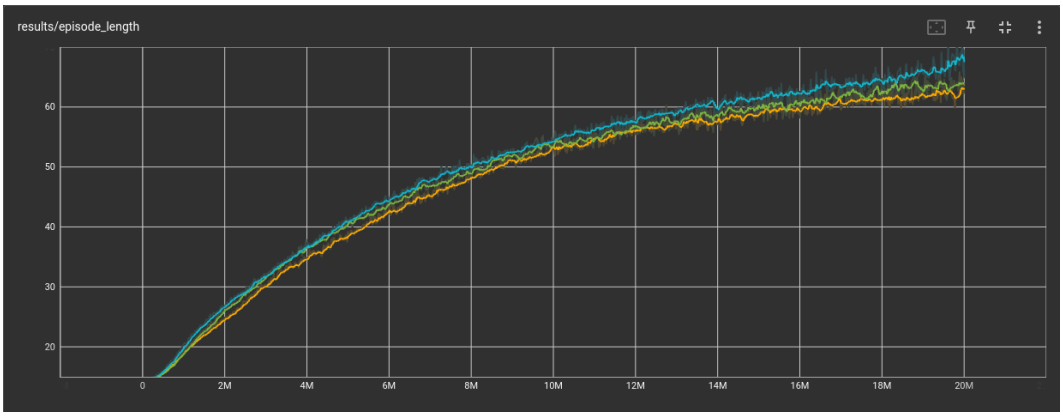

Figure 4: Episode lengths for `g1-window-v0` task with PPO on 3 seeds

The learning curve shows no clear sign of plateauing at 20M steps, though the rate of improvement slows noticeably after 14M steps. Gains of less than 2 return units over the final 6M steps suggest that PPO is approaching the limits of what its current policy parameterization can achieve without further architectural or algorithmic changes.

5.2.2 SAC

As shown in Figure 5, all three seeds rise sharply in the first 500k steps, already surpassing PPO's 5M-step performance before SAC has even used a quarter of PPO's sample budget. Two seeds reach peak returns of 60-75 around 2-2.5M steps, which is 3-4× higher than PPO's final performance. This reflects SAC's key advantage: the replay buffer enables aggressive reuse of past experience, and entropy regularization drives the broad exploration needed to discover the coordinated whole-body behaviors this task requires. However, none of the three seeds sustain these peak returns. The pink seed oscillates violently between 20 and 75 from 2M steps onward. The grey seed collapses almost entirely after 4.5M steps, falling to ∼10, barely above the random baseline, and never recovers. The purple seed is the most stable, maintaining returns in the 30-45 range through 5M steps, but with high variance throughout.

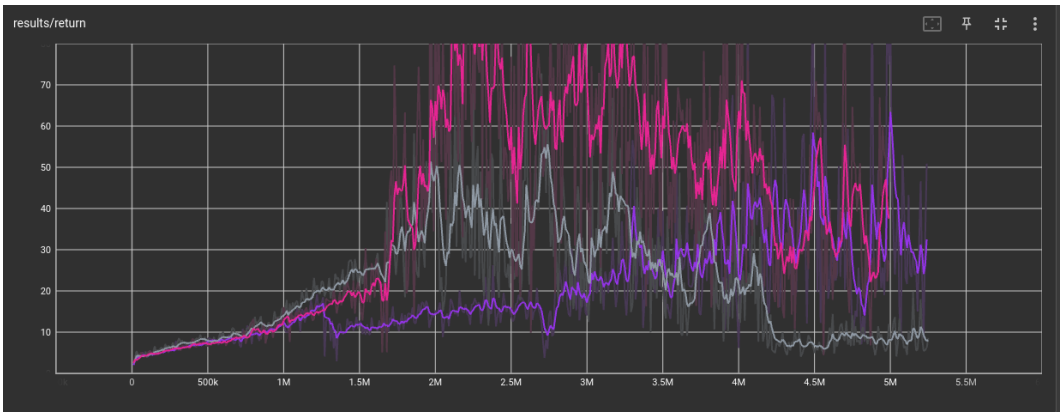

Figure 5: Episode returns for `g1-window-v0` task with SAC on 3 seeds

The episode length curves for SAC (Figure 6) mirror the return curves: every spike and collapse in return corresponds directly to a spike and collapse in episode length. This correspondence shows that SAC's performance drops are not caused by the agent adopting a worse within-episode strategy, but by episodes terminating earlier, most likely due to the robot losing balance or dropping the tool. The `per_timestep_reward` metric (Figure 7) supports this interpretation: even during return collapses, per-step reward remains relatively stable between 0.10 and 0.25, indicating that the agent's behavior quality when alive has not degraded, but it is simply staying alive for far fewer steps. This pattern of sharp rise followed by oscillatory collapse is consistent with Q-value overestimation in off-policy methods, where the critic learns to overestimate returns for certain state-action pairs, the policy exploits these overestimates to adopt physically aggressive behaviors that initially score well but are dynamically unstable, and performance degrades as a result. The consistency of this collapse across all three seeds at roughly the same training stage (∼2M steps) points to a systematic issue rather than seed-specific bad luck.

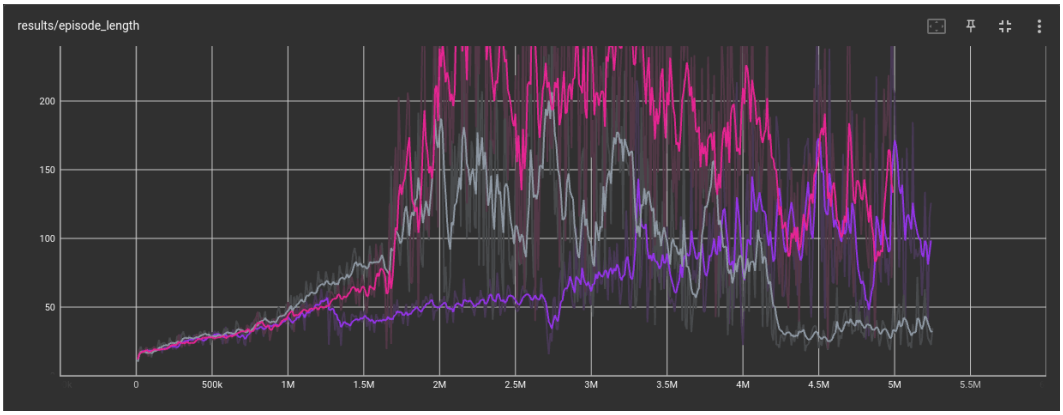

Figure 6: Episode lengths for `g1-window-v0` task with SAC on 3 seeds

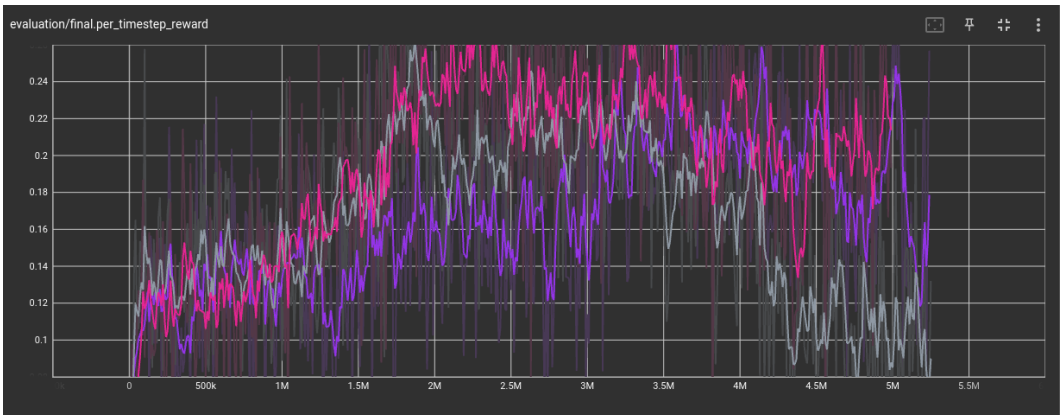

Figure 7: Per-timestep reward for `g1-window-v0` task with SAC on 3 seeds

Taken together, SAC is far more sample-efficient but fails to consolidate its gains into a stable policy, while PPO is slow but monotonic and reproducible.

5.3 Success Rates

Neither PPO nor SAC achieves any measurable task success throughout training. Both `results/success` and `results/success_subtasks` remain flatlined at zero for the entire training duration across all seeds for both methods (Figure 8). However, the two methods fail at fundamentally different stages of the task pipeline.

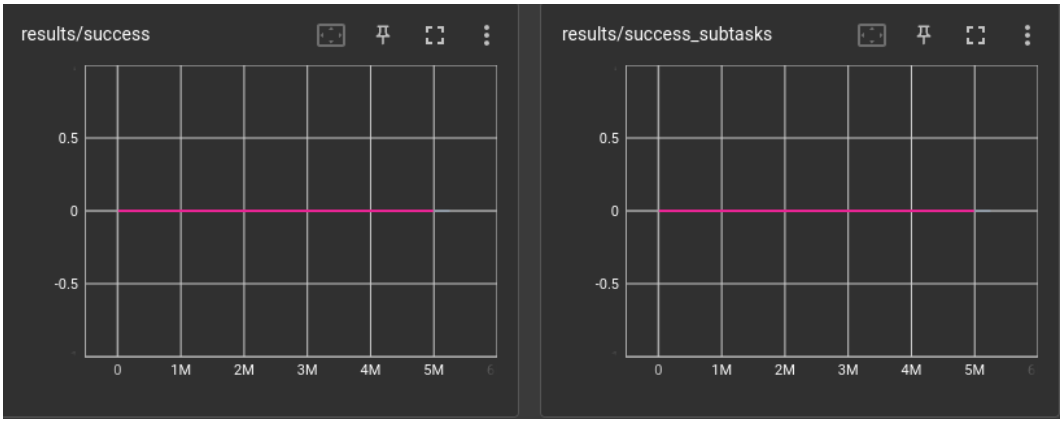

Figure 8: Task success and success subtasks for `g1-window-v0` task with PPO and SAC on 3 seeds each

5.3.1 PPO

For PPO, we do not see any plots for `hand_tool_proximity_reward`, `moving_wipe_reward`, and `stand_reward`. These metrics are never triggered because the robot does not meaningfully interact with the task at any point. It never gets near the tool, never makes wiper contact, and never achieves stable upright posture in a manipulation context. Over 20M steps, PPO learns to survive longer episodes, but it never once reaches the tool, let alone attempts to wipe the window. The task, from PPO's perspective, remains essentially unexplored.

5.3.2 SAC

The `hand_tool_proximity_reward` (Figure 9) rises from ∼0.4 to 0.7-0.85 by 2M steps, indicating that SAC does learn to bring its hands close to the tool. However, progress stalls there. The `moving_wipe_reward` (Figure 10) shows that it fluctuates noisily between 0.05 and 0.30 throughout training with no clear upward trend, meaning SAC makes intermittent wiper contact but never learns to execute the sustained, controlled wiping motion the task requires.

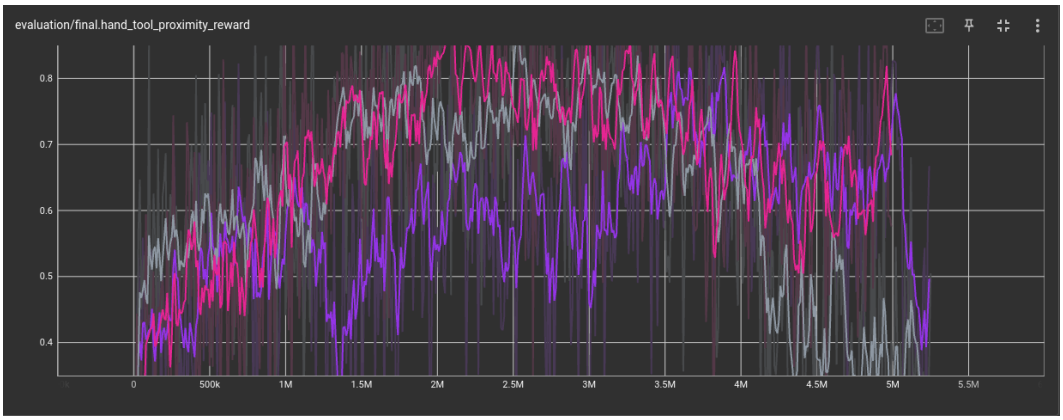

Figure 9: Hand-tool proximity reward for `g1-window-v0` task with SAC on 3 seeds

The `stand_reward` metric (Figure 11) remains low and highly variable throughout training, ranging between 0.04 and 0.19 with no improvement trend. For a task that fundamentally requires the robot to maintain stable upright posture while executing a coordinated arm motion, the inability to reliably satisfy the standing constraint is a critical failure. This directly explains the episode termination pattern discussed in Section 5.2.2: SAC is not collapsing because it forgets how to wipe, it is collapsing because it never reliably learns to stand while wiping in the first place.

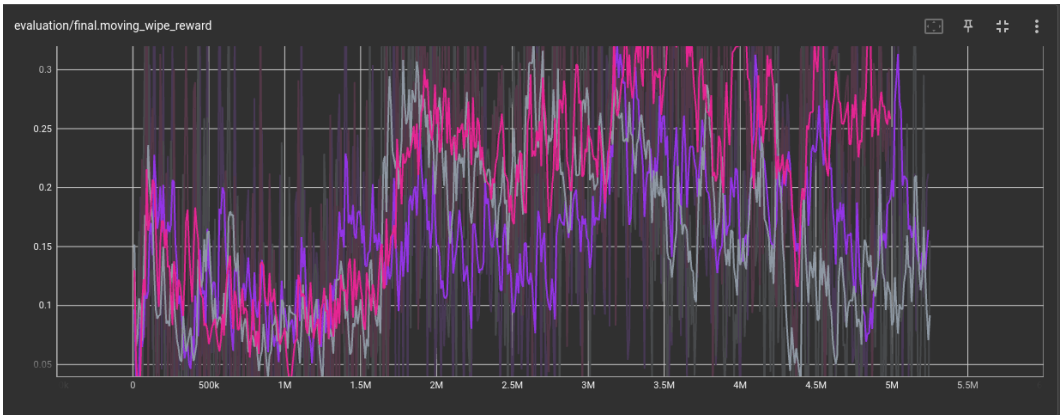

Figure 10: Moving wipe reward for `g1-window-v0` task with SAC on 3 seeds

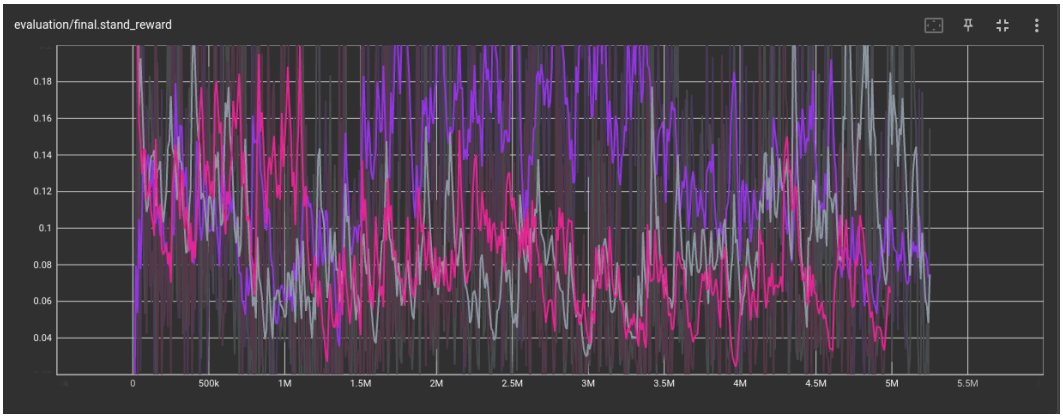

Figure 11: Stand reward for `g1-window-v0` task with SAC on 3 seeds

5.4 Failure Cases

PPO's failure is purely one of exploration. The agent never progresses beyond basic postural stability. None of the task-relevant metrics are ever triggered, and the reward signal is dominated entirely by the stability component. In a 61-DoF action space, PPO's conservative on-policy updates simply never discover the behavioral sequence needed to reach the tool, let alone execute the wiping motion.

SAC's failure is more structural. Despite learning approximate hand-tool proximity, it cannot simultaneously satisfy postural stability and wiper contact precision. The `stand_reward` (Figure 11) never improves meaningfully, and every attempt to interact with the tool destabilizes the robot's posture, causing early termination. This creates a damaging cycle: manipulation attempts destabilize posture, posture failures cause early termination, and early termination prevents the agent from accumulating the experience needed to learn stable manipulation. This cycle, combined by the Q-value overestimation discussed in Section 5.2.2, explains why SAC's performance oscillates rather than converges.

5.5 Next Steps

The results point to two clear next steps. First, we will run FastTD3 on the `g1-window-v0` task. SAC shows that off-policy methods can discover high-return behaviors quickly, but policy instability prevents these gains from consolidating. FastTD3's distributional critic and fast value bootstrapping directly address this, and we will additionally ablate the UTD ratio to characterize the tradeoff between sample efficiency and stability. Second, we will augment PPO with reward shaping derived

from FastTD3's learned value functions. PPO's failure is fundamentally one of exploration: without a dense guiding signal, it never discovers task-relevant behaviors. FastTD3's value estimates provide that signal while preserving PPO's stable learning dynamics. We will also revisit SAC's hyperparameters, particularly the critic learning rate and replay buffer size, to assess whether the oscillatory collapse can be mitigated independently of the algorithmic modifications.

6 Conclusion

We evaluated PPO and SAC on the g1-window-v0 whole-body manipulation task and found that both methods fail to achieve task success, albeit for different reasons. PPO is stable and reproducible but never progresses beyond basic postural stability, failing to discover any task-relevant behavior within a 20M step budget. SAC is far more sample-efficient and reaches returns 3-4$\times$ higher than PPO within 2M steps, but suffers from systematic policy instability driven by Q-value overestimation and an inability to simultaneously satisfy postural and contact precision constraints. These results motivate our proposed modifications: FastTD3 for improved off-policy stability, and FastTD3-guided reward shaping for PPO, which we will evaluate in the final report.

References

[1] Y. Seo, C. Sferrazza, H. Geng, M. Nauman, Z.-H. Yin, and P. Abbeel. Fasttd3: Simple, fast, and capable reinforcement learning for humanoid control, 2025. URL https://arxiv.org/abs/2505.22642.

[2] C. Sferrazza, D.-M. Huang, X. Lin, Y. Lee, and P. Abbeel. Humanoidbench: Simulated humanoid benchmark for whole-body locomotion and manipulation, 2024. URL https://arxiv.org/abs/2403.10506.

[3] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

[4] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models, 2024. URL https://arxiv.org/abs/2301.04104.

[5] N. Hansen, H. Su, and X. Wang. Td-mpc2: Scalable, robust world models for continuous control. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun, editors, *International Conference on Learning Representations*, volume 2024, pages 47376–47405, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/cf73d57b6dcda32b293df7c2d5341f49-Paper-Conference.pdf.

[6] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://arxiv.org/abs/1801.01290.