From Sim-to-Real to Learn-in-Real: Real-world Online Learning for Humanoid Robots

Rankun Li[1], Yuhang Xie[1], Linqing Zhu[1], Linqi Ye[1(✉)], Qingdu Li[2], and Yan Peng[1]

[1] Shanghai University, Shanghai 200444, China,
yelinqi@shu.edu.cn
[2] University of Shanghai for Science and Technology, Shanghai 200093, China

Abstract. In recent years, reinforcement learning has significantly accelerated the development of legged robot control systems. The prevalent paradigm involves conducting reinforcement learning training in simulated environments initially, followed by a transition to real-world applications, a process known as sim-to-real transfer. However, this paradigm still cannot fully bridge the gap between between simulation and reality. To further narrow the gap between simulation and reality, this paper proposes an innovative online learning strategy that aims to conduct training directly on the physical robot. To achieve this, we harness the power of pre-training and instruction learning to enhance learning efficiency. Additionally, we have designed an autonomous resetting system that enables the robot to automatically reconfigure and seamlessly resume learning after a fall, ensuring continuous progress. Our findings indicate that the performance of the robot after online learning has been enhanced to a certain extent compared to direct deployment using sim-to-real. The research results demonstrate the effectiveness of the Learn-in-Real paradigm in enhancing the locomotion capabilities of legged robots and provide a promising pathway for improving the performance of other legged robots.

Keywords: Online Learning, Instruction Learning, Reinforcement Learning, Sim-to-Real, Humanoid Robots.

1 Introduction

The rapid development of model-free Reinforcement Learning (RL) has led to significant breakthroughs in the field of robotic motion control. RL enables agents to learn optimal policies through trial and error in interaction with the environment, making it an ideal tool for training robotic control systems in simulation environments[1]. In the domain of locomotion control, the application of RL is particularly widespread, as it allows robots to autonomously acquire complex locomotion skills through interaction with simulated environments, without relying on precise dynamic models. However, despite the fact that simulation training provides a safe and cost-effective platform for robotic control, the deployment

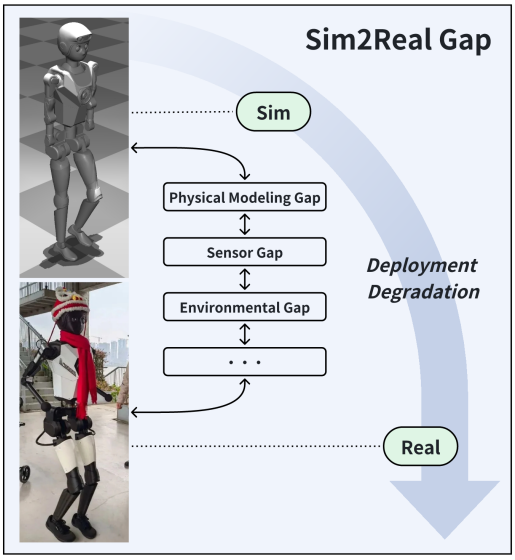

Fig. 1. Sim-to-Real Gap.

of physical robots using this approach has long faced challenges due to the Sim-to-Real gap (Fig.1) between simulation and the real world.

To address this issue, existing methods have largely attempted to narrow the Sim-to-Real gap by introducing perturbations into simulation software, such as adding noise[2][3], domain randomization[4][5], and meta-learning[6][7]. Although these methods have enabled robots to transfer from simulation to reality, they still fail to completely eliminate the differences between simulation and reality, resulting in a decline in robotic performance during the transfer process.

We believe that embodied intelligent robots should perform anthropomorphic tasks through active perception of the environment, autonomous learning, and autonomous decision making, with their brains being agents based on RL. Therefore, constructing a new Learn-in-Real learning paradigm is a key method for the large-scale deployment of robots in the future, that is, robots can directly perform online RL in the real environment.Our goal is to conduct online reinforcement learning based on Sim-to-Real, thus compensating for the performance degradation of robot deployment and even achieving a leap in robot capabilities.

Our main contributions can be summarized in the following three points:

1) Online learning further bridges the significant gap between physical entities and simulation environments. It empowers physical robots to engage in ongoing learning processes, leading to continuous improvement and performance enhancement as the duration of learning increases.

2) The adoption of pre-training and instruction learning solves the problem of low learning efficiency and slow learning speed. Prevents the robot from learning from scratch and greatly accelerates the learning process.

3) The automatic reset of the sky track addresses the issue of physical objects being prone to falling and difficult to reset. The innovative design of the sky track system can detect the robot's fall and reset it, ensuring safety and efficiency.

2 Related work

2.1 Robot Reinforcement Learning

To bridge the gap between simulation and reality, domain randomization techniques have been widely adopted. In 2019, Hwangbo et al. [8] of ETH Zurich first combined domain randomization with reinforcement learning in simulation environments. The trained models were successfully deployed on the real quadruped robot ANYmal, significantly improving robot performance in dynamic and agile movements. Haarnoja et al. [9] from DeepMind combined high frequency control, target dynamics randomization, and perturbations for reinforcement learning training, achieving sample-free transfer from simulation to reality. Gu et al. [10] open-sourced the Humanoid-Gym framework for end-to-end reinforcement learning training of humanoid robots. This framework significantly simplified the training process and the difficulty of Sim-to-Real transfer through its carefully designed reward functions and domain randomization techniques, lowering the development threshold for humanoid robot algorithms.

Introducing realistic noise is another technique to narrow the gap between simulation and reality. Kaufmann et al. [2] from the University of Zurich enhanced the realism of simulation environments by pre-training real-world perception systems and empirical noise models. This approach helped them achieve the first autonomous drone control at a human-champion level, featured on the cover of Nature. Gu et al. [11] proposed the Denoising World Model Learning (DWL) framework, which uses an encoder-decoder structure to process noisy data in simulation environments and learn effective state representations. This method reduced the difficulty of transferring from simulation to reality and improved the walking capabilities of humanoid robots in complex and challenging real-world environments.

Meta-learning is also a strategy to address the Sim-to-Real problem. Arndt et al. [12] used meta-learning techniques to train policies that can adapt to various dynamic conditions. By combining these policies with task-specific trajectory generation models, they provided an action space for rapid exploration, effectively tackling the Sim-to-Real problem.

2.2 Robot Online Learning

To address the challenge of maintaining consistency between highly realistic simulation environments and the real world, researchers have begun to integrate online learning into the field of reinforcement learning, that is, training directly in real world environments. For example, in fixed-base robots such as robotic arms, online learning has demonstrated its effectiveness through massive data

training [13]. Luo et al. [14][15] have consistently employed the integration of reinforcement learning with physical robotic systems, underscoring that while simulation facilitates rapid data generation, the intrinsic value of real-world experimental data remains indispensable.

However, for floating-base physical systems, such as legged robots, the cost of online policy adjustment is extremely high because they are prone to damage during repeated trial and error. Therefore, improving sample efficiency and ensuring operational safety have become two key considerations in the research of legged robots. Haarnoja et al. [16] proposed a sample-efficient deep reinforcement learning algorithm based on maximum entropy, which can learn quadruped locomotion controllers from scratch in an end-to-end manner on real-world robots, automatically forming walking gaits in a short time.

Other researchers have adopted a hybrid approach that combines simulation and real-world environments to narrow the gap between simulation and reality. Jonnarth et al. [17] trained robots using motion imitation and semi-virtual environments, achieving environmental diversity and automatic scene resetting through simulated sensors and randomized obstacles, reducing the differences between simulation and real-world applications. They also found that higher inference frequencies allow Markovian policies to be directly transferred from simulation to the real world, while more complex higher-order policies can further close the gap through fine-tuning.

In terms of reducing human intervention, Bloesch et al. [18] trained small humanoid robots to walk and interact using onboard sensors and limited hardware prior knowledge, helping the robot to stand up again through preprogrammed feedforward controllers when it falls. Gupta et al. [19] proposed a method that does not require resetting, achieving mutual resetting by learning multiple tasks simultaneously, reducing the need for human intervention.

In this research context, the work of Smith et al. [20] is particularly noteworthy. They combined imitation learning and online learning to fine-tune locomotion policies in the real world, demonstrating that a small amount of real-world training can significantly improve deployment performance, allowing the A1 quadruped robot to autonomously fine-tune locomotion skills in various environments. Subsequently, they used model-based non-policy reinforcement learning to learn from scratch in the real world, quickly learning quadruped locomotion through autonomous data collection [21] . Building on this, Ye et al. [22] proposed a guided learning paradigm combined with online learning, which allows the A1 robot to quickly learn various gaits, turns, and forward walking without data collection, preventing falls by increasing step frequency and avoiding the need for resetting, thus achieving higher training efficiency.

3 Methods

In this study, we used Unity, Isaac Gym, and MuJoCo as the primary software platforms, in conjunction with the Droid humanoid robot Walker II X02 Lite A

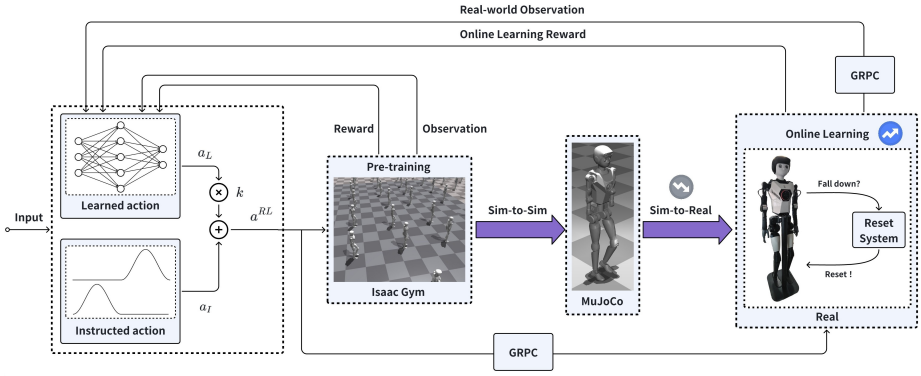

Fig. 2. The framework for online learning in the real world for humanoid robots.

hardware platform, to validate the Learn-in-Real paradigm. The framework for online learning in the real world is depicted in Fig.2.

We initially employ instruction learning within Isaac Gym to train the robot's basic walking gaits. Subsequently, we migrate to MuJoCo to validate the feasibility and ensure the safety of the pre-trained model. Following this, we deploy the pre-trained model onto the physical robot. At this stage, the robot has essentially learned to maintain balance. Building upon this pre-trained network, we conduct online reinforcement learning through real-time interaction with the robot via gRPC (Google Remote Procedure Call), ultimately refining the policy network further.

In this section, we will introduce the main components of the Learn-in-Real framework: Instruction Learning, Pre-training, Online Learning, and the Reset System.

3.1 Instruction Learning

Instruction Learning, as an advanced learning method that integrates traditional reinforcement learning with direct feedforward control, optimizes the learning path for robots to handle complex action sequences. It enables robots to use basic stepping actions as feedforward signals. Starting from these signals, combined with a reward mechanism, robots can quickly learn and master a variety of gaits.

The input is the reference trajectory for the position for each joint. Since walking motion can be regarded as the left and right legs alternately performing a cosine waveform, we use a periodic trajectory, which means that the joints cycle through motion with a fixed period. In this case, the reference angle is designed as a sinusoidal signal

$$\theta_t^{ref} = \theta_0 + \Delta\theta \frac{1 - \cos\left(\frac{2\pi t}{T}\right)}{2} \tag{1}$$

The feedforward action is obtained by mapping the reference angle to $[-1, 1]$:

$$a_I = 2\frac{\theta_t^{ref} - \theta_{min}}{\theta_{max} - \theta_{min}} - 1 \tag{2}$$

where θ_t^{ref} represents the reference angle of the joint and θ_{min} and θ_{max} denote the minimum and maximum limits of the joint range of motion, respectively. Then, the feedback action a_L from the neural network is weighted by the proportional coefficient k and added to the feedforward guiding action a_I to obtain the final action output.

$$a^{RL} = ka_L + a_I \tag{3}$$

This approach significantly improves data utilization efficiency during the learning process and reduces the amount of exploration required when starting from random policies[22].

3.2 Pre-training

To enhance the efficiency of the pre-training phase, we leveraged the capabilities of the Isaac Gym simulation environment, enabling the parallel training of thousands of robots. This approach significantly accelerated the learning process for the robots.

During the pre-training phase, we employed a reinforcement learning-based model $M = \langle S, A, T, O, R, \gamma \rangle$. In this model, S and A define the state and action spaces, respectively. $T(s'|s, a)$ describes the probability of transitioning to state s' given the current state s and action a, $R(s, a)$ is the reward function, $\gamma \in [0, 1]$ is the discount factor, and O represents the observation space. This model is designed to facilitate a smooth transition from full observability in simulation environments ($s \in S$) to partial observability in the real world ($o \in O$). In the training process, we utilized the loss function of the Proximal Policy Optimization (PPO) algorithm [23], combined with an asymmetric actor-critic approach [24]. Furthermore, we incorporated privileged information during the training phase and switched to non-privileged observations during deployment.

Table 1. Overview of Domain Randomization.

Parameter	Unit	Range	Operator	Type
Joint Position	rad	[-0.02, 0.02]	additive	Gaussian (1σ)
Joint Velocity	rad/s	[-0.5, 0.5]	additive	Gaussian (1σ)
Angular Velocity	rad/s	[-0.2, 0.2]	additive	Gaussian (1σ)
Euler Angle	rad	[-0.1, 0.1]	additive	Gaussian (1σ)
Body Mass	kg	[-6, 6]	additive	Gaussian (1σ)
System Delay	ms	[0, 10]	-	Uniform
Friction	-	[0.1, 1.5]	-	Uniform
Motor Strength	%	[90, 110]	scaling	Gaussian (1σ)
Kp	%	[50, 150]	scaling	Gaussian (1σ)
Kd	%	[50, 150]	scaling	Gaussian (1σ)

This design allows us to fully leverage complete information in the simulation environment while ensuring the model's adaptability and robustness in the real world.

The design of the reward function is crucial for reinforcement learning as it directly influences the agent's behavior and performance. Our reward function consists of three key components: velocity tracking reward, gait reward and regularization term.

To enable successful transfer of the pre-trained model to physical robots, we carefully designed a set of domain randomization parameters, as detailed in Table 1.

3.3 Online Learning

Online learning utilizes data that is continuously updated over time. When the model is presented with new data, it undergoes partial training or adjustment to adapt to these updates, creating a cyclical process. Consequently, compared to offline learning, models that employ online learning are better equipped to handle scenarios where data evolves over time. Due to its high learning efficiency, instruction learning is particularly well-suited for online learning with real robots. Online learning collects data directly from actual robots and updates control strategies in real-time, thereby eliminating the sim-to-real problem[22]. By placing the agent in a variety of potentially changing environments, it enhances the agent's adaptability to unfamiliar settings, ensuring stability in the real world.

We selected the Unity ML-Agents framework for online learning. This framework features an off-the-shelf reinforcement learning library and superior visualization capabilities. Communication with the robot is facilitated through gRPC, with neural networks receiving observations in real-time to generate actions designed to maximize rewards. The observations used for online reinforcement learning align with the non-privileged observations. The reward function for online learning can be tailored to address specific outcomes following the sim-to-real transfer.

3.4 Reset System

Traditionally, if a robot falls during the learning process, it not only risks damaging its mechanical structure but also greatly limits the continuity and efficiency of learning. This is because manual resetting is not only time-consuming and labor-intensive, but also increases the risk of maintenance costs and operational interruptions.

In light of this, we have innovatively introduced an automatic celestial track reset system(Fig.3), which can monitor the robot's status in real-time. This system mainly consists of a wire hanger equipped with a reducer. The robot's ascent and descent are controlled by a Xiaomi CyberGear servo motor through the wire. We use the ESP32 C3 Super Mini Bluetooth communication module to communicate with the computer, which automatically sends reset commands to the system during training. Once it detects signs of the robot losing balance

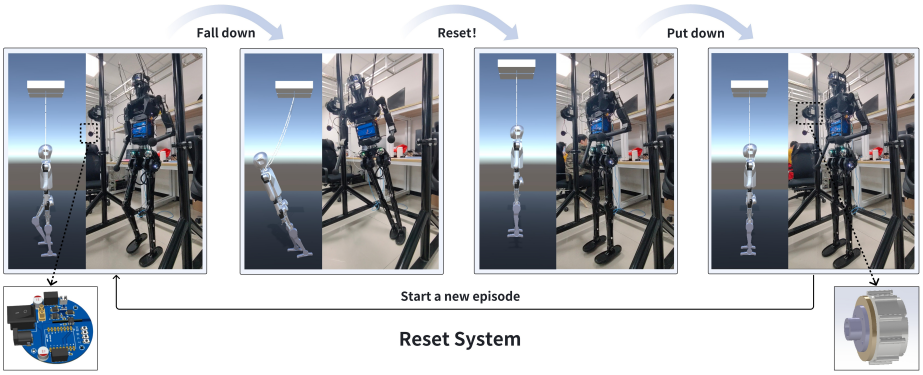

Fig. 3. Reset system with servo-driven lift wire.

or falling, the system immediately initiates the reset procedure. By precisely controlling the motor to pull the rope, the robot is quickly and smoothly pulled back to a preset safe position or the starting point for continued learning, all without the need for human intervention, achieving a seamless transition from detection to reset.

4 Experimental Results

The humanoid robot we used, the X02LiteA bipedal robot, is a tendon-driven robot with a height of 1.6 meters, a weight of 30 kilograms, and a total of 18 degrees of freedom. In this scenario, only 10 legs joints were used, including hip yaw, hip roll, hip pitch, knee pitch, and ankle pitch.

In this section, we will provide a detailed description of the content of our experimental validation.

4.1 Learn in Sim

After training in Isaac Gym, the pre-trained model can be successfully transferred to MuJoCo, where it performs well during walking tasks. Subsequently, deploying it directly onto the physical robot is also successful. The reward curve obtained during training in Isaac Gym is shown in Fig.4.

However, after transferring the policy to the physical robot, we observed that despite the incorporation of domain randomization during training, there are still noticeable differences between the performance of the policy network on the physical robot and that in the MuJoCo simulation environment. One particularly evident phenomenon is the significant body vibrations during the robot's walking process on the physical platform, as illustrated in Fig.4. This figure indicates that while the differences in body angular velocity in the x-direction and base roll between MuJoCo and the real robot are relatively small, there are substantial discrepancies in body angular velocity in the y-direction and base pitch. These

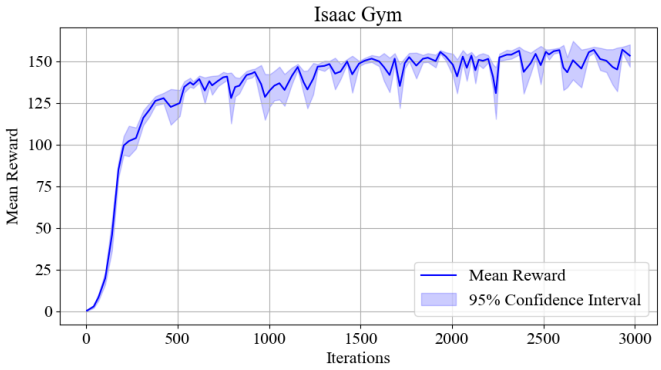

Fig. 4. Mean reward during pre-training in Isaac Gym.

findings suggest that the robot's performance deteriorates significantly after the sim-to-real transfer compared to the simulation environment, highlighting the persistent gap between simulation and reality.

To further close the Sim-to-Real gap, one approach is to continue tuning the parameters during training. This method involves multiple training and debugging iterations and may still fail to achieve a very small Sim-to-Real gap. Another approach is to conduct online reinforcement learning on the physical robot, leveraging real-time interaction data from the real world to bridge the gap between simulation and reality.

4.2 Learn in Real

Given that we use Unity for inference through the ONNX policy network, the pre-trained policy network is also saved in ONNX format. The training setup for online learning includes a fixed time step of 0.01 seconds, a time scale of 1, and a maximum of 1000 steps per episode.

The observations and actions used in online learning are consistent with those in the Sim-to-Real setup. In terms of reward function design, we base it on the performance from simulation to reality. For example, after deploying the policy to the physical robot, we observed significant vibrations. To address this issue, we designed a reward function aimed at reducing the robot's body oscillation.

$$r_{\text{online}} = r_{\text{live}} + r_{\text{euler}} + r_{\omega} \tag{4}$$

Here, r_{live} represents the survival reward, r_{euler} is the penalty term for Euler angles, which imposes a penalty when the Euler angle exceeds a set threshold (0.02 in this experiment), r_{ω} is the penalty term for angular velocity, which imposes a penalty when the angular velocity exceeds a set threshold (0.5 in this experiment).

At the beginning of the training, the policy network is fine-tuned based on the pre-trained network, and the reward value starts to gradually increase, as shown in Fig. 6.

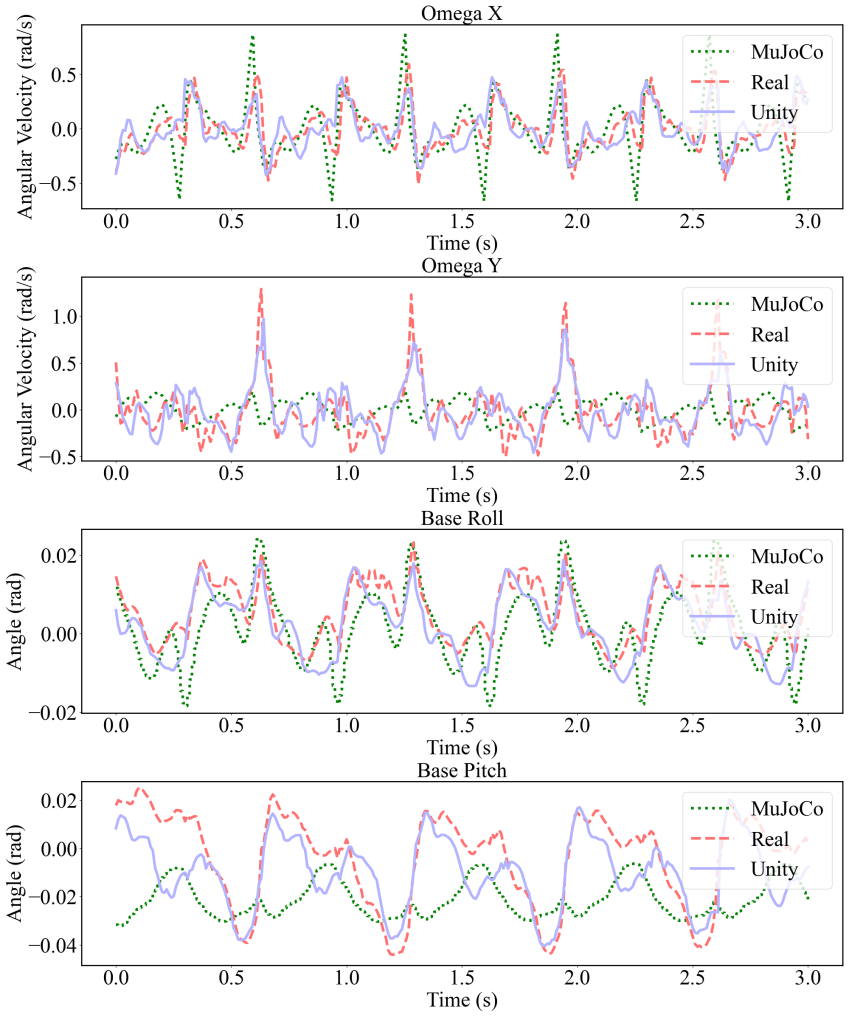

Fig. 5. Comparison chart of the robot's body angular velocity (in the x and y directions) and Euler angles (roll and pitch) after MuJoCo, Sim2Real, and Learn-in-Real.

After 40 minutes of online learning, by comparing and analyzing the changes in the robot's base angular velocity and Euler angles before and after online learning, as shown in Fig. 5. We found that the peak values of the robot's angular velocity in the x-direction and the Euler angle around the *roll*-axis were lower after online learning than in the MuJoCo simulation environment. Moreover, compared with the state immediately after deploying the policy using Sim-to-Real, the peak values of all parameters were reduced after online learning.

This indicates that online learning successfully reduced the robot's body oscillation during walking. Compared with the deployment strategy directly from

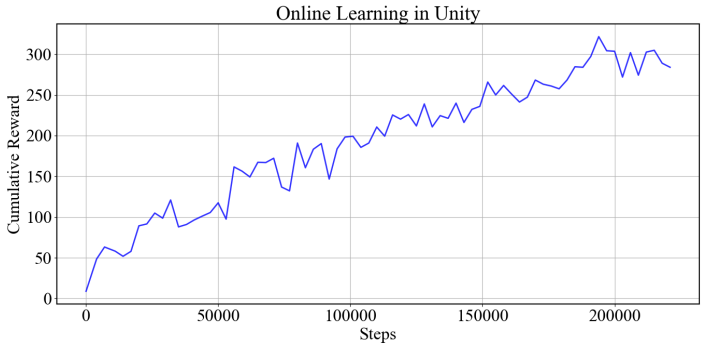

Fig. 6. Cumulative reward during online learning in Unity.

simulation to reality, the performance is significantly improved. The experimental results fully demonstrate the effectiveness of online learning on the physical robot. It not only further narrows the performance gap of Sim-to-Real but also performs better than in the simulation environment in some aspects.

5 Conclusion

In conclusion, the Learn-in-Real paradigm offers a novel and effective approach for humanoid robots to tackle the Sim-to-Real challenge. By conducting online learning in real environments, it adapts to environmental changes in real-time, effectively reducing the gap between simulated and real-world scenarios. The use of feedforward guidance learning combined with pre-trained networks significantly speeds up the learning process. Moreover, the automatic reset system addresses the issue of robot resetting during online learning, ensuring that the robot can automatically recover after a fall, thus maintaining the continuity and safety of training. We anticipate that this concept will advance the development of robotic technology to a higher level and enable broader applications.

References

1. Zhao, W., Queralta, J.P., Westerlund, T.: Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey. IEEE Symposium Series on Computational Intelligence (SSCI), Canberra, ACT, Australia, 2020, 737–744 (2020)
2. Kaufmann, E., et al.: Champion-level drone racing using deep reinforcement learning. Nature 620, 982–987 (2023)
3. Zhao, W., et al.: Towards closing the sim-to-real gap in collaborative multi-robot deep reinforcement learning. 2020 5th International Conference on Robotics and Automation Engineering (ICRAE), 7–12 (2020)
4. Muratore, F., et al.: Data-efficient domain randomization with Bayesian optimization. IEEE Robot. Autom. Lett. 6(2), 911–918 (2021)

5. Tobin, J., et al.: Domain randomization for transferring deep neural networks from simulation to the real world. Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vancouver, BC, Canada, 23–30 (2017)
6. Finn, C., Abbeel, P., Levine, S.: Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. Proc. 34th Int. Conf. Mach. Learn., 1126–1135 (2017)
7. Nagabandi, A., et al.: Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. arXiv:cs.LG/2008.07875 (2019)
8. Hwangbo, J., et al.: Learning agile and dynamic motor skills for legged robots. Sci. Robot. 4, eaau5872 (2019)
9. Haarnoja, T., et al.: Learning agile soccer skills for a bipedal robot with deep reinforcement learning. Sci. Robot. 9(89), eadi8022 (2024)
10. Gu, X., et al.: Humanoid-Gym: Reinforcement Learning for Humanoid Robot with Zero-Shot Sim2Real Transfer. arXiv:2404.05695 (2024)
11. Gu, X., et al.: Advancing Humanoid Locomotion: Mastering Challenging Terrains with Denoising World Model Learning. arXiv:2408.14472 (2024)
12. Arndt, K., et al.: Meta reinforcement learning for sim-to-real domain adaptation. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), Paris, France, 2725–2731 (2020)
13. Levine, S., et al.: Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection. arXiv:1603.02199 (2016)
14. Luo, J., et al.: SERL: A Software Suite for Sample-Efficient Robotic Reinforcement Learning. IEEE International Conference on Robotics and Automation (ICRA), Yokohama, Japan, May 13-17, 2024, 16961–16969 (2024)
15. Luo, J., et al.: Precise and Dexterous Robotic Manipulation via Human-in-the-Loop Reinforcement Learning. arXiv:cs.RO/2410.21845 (2024)
16. Haarnoja, T., et al.: Learning to Walk via Deep Reinforcement Learning. arXiv:1812.11103 (2019)
17. Jonnarth, A., et al.: Sim-to-Real Transfer of Deep Reinforcement Learning Agents for Online Coverage Path Planning. arXiv:2406.04920 (2024)
18. Bloesch, M., et al.: Towards Real Robot Learning in the Wild: A Case Study in Bipedal Locomotion. Proc. Conf. Robot. Learn. (CoRL), vol. 164, 1502–1511 (2022)
19. Gupta, A., et al.: Reset-Free Reinforcement Learning via Multi-Task Learning: Learning Dexterous Manipulation Behaviors without Human Intervention. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), Xi'an, China, 6664–6671 (2021)
20. Smith, L., et al.: Legged Robots that Keep on Learning: Fine-Tuning Locomotion Policies in the Real World. Proc. Conf. Robot. Learn. (CoRL) (2021)
21. Smith, L., et al.: A Walk in the Park: Learning to Walk in 20 Minutes With Model-Free Reinforcement Learning. arXiv:2208.07860 (2022)
22. Ye, L., et al.: From Knowing to Doing: Learning Diverse Motor Skills through Instruction Learning. arXiv:abs/2309.09167 (2023)
23. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv:1707.06347 (2017)
24. Pinto, L., Andrychowicz, M., Welinder, P., Zaremba, W., Abbeel, P.: Asymmetric actor critic for image-based robot learning. arXiv:1710.06542 (2017)