SCIENCE CHINA
Technological Sciences

• Letter •

*** 2025 Vol.** No.*: ***–***
doi: 10.1007/s11431-****

Gewu Playground: An Open-Source Robot Simulation Platform for Embodied Intelligence Research

Linqi Ye[1*], Boyang Xing[2], Bin Liang[3], Lei Jiang[2], Yan Peng[1]

[1]*School of Future Technology, Shanghai University, 200444 Shanghai, China;*
[2]*National and Local Co-Built Humanoid Robotics Innovation Center, 201203 Shanghai, China;*
[3]*Navigation and Control Research Center, Department of Automation, Tsinghua University, 100084 Beijing, China*

Citation: Ye L Q, et al. GeWu Playground: An Open-Source Robot Simulation Platform for Embodied Intelligence Research. Sci China Tech Sci, 2025.

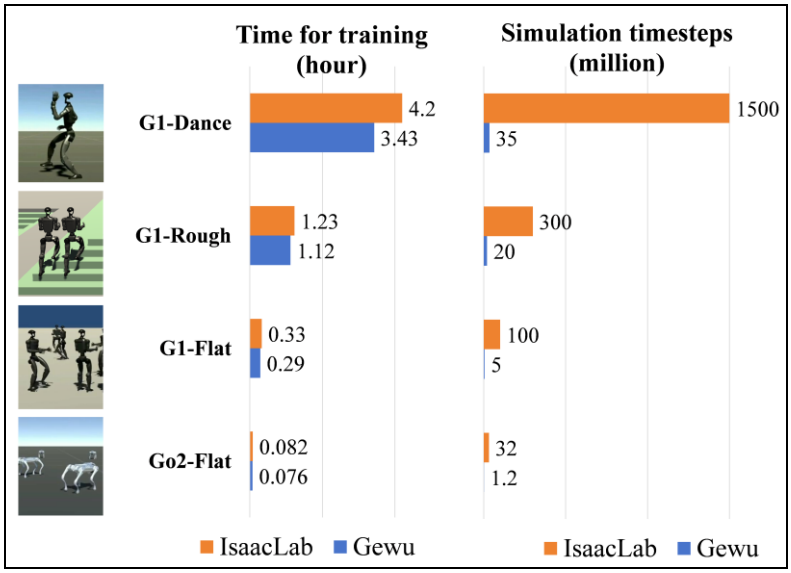
Embodied intelligence, which integrates robotics and artificial intelligence, relies heavily on simulation platforms to develop adaptive behaviors through perception, cognition, and action. Traditional robotic simulation tools such as Webots [1], Gazebo [2], and V-REP [3] have significantly contributed to the field by supporting kinematic and dynamic model-based research. However, the advent of learning-based methods, particularly reinforcement learning (RL), has created a demand for platforms specifically designed to facilitate these approaches, with IsaacLab [4], MuJoCo Playground [5], and Genesis emerging as key solutions. IsaacLab provides a high-fidelity NVIDIA simulation toolkit with photorealistic rendering, supporting multi-modal robotic platforms via a unified API for seamless reinforcement learning algorithm integration; MuJoCo Playground leverages the MuJoCo physics engine for rapid sim2real policy iteration, featuring on-device rendering, domain randomization, and pre-built benchmarks across quadrupeds, humanoids, and dexterous manipulators; while Genesis employs data-driven generative physics modeling to create dynamic simulation environments for manipulation and locomotion tasks, enabling on-the-fly scenario generation for scalable experimentation.

Unity, primarily known as a game engine, has emerged as a powerful tool for embodied AI research, offering frameworks like AI2-Thor [6] and Unity ML-Agents Toolkit [7] that provide intuitive environments for training intelligent agents. Unity offers unique advantages for embodied intelligence research by providing near-photorealistic rendering and multi-modal inputs for perception tasks, integrating advanced physics engines for dynamic interactions like deformable objects, and enabling hierarchical tasks and multi-agent collaboration through C# scripting—features that surpass MuJoCo's limited state inputs and lack of multi-agent support, as well as IsaacSim's focus on robotic fleets over sensory and dynamic environment flexibility.

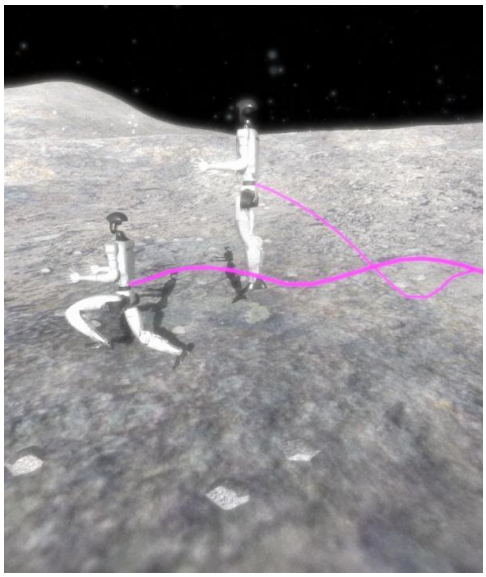
*Corresponding author (email: yelinqi@shu.edu.cn)

Built on Unity, we introduce Gewu Playground (https://github.com/loongOpen/Unity-RL-Playground/), a comprehensive, open-source robot simulation platform that supports a wide range of embodied intelligence tasks. Gewu Playground extends the capabilities of Unity RL Playground [8], offering enhanced features for locomotion, manipulation, and navigation tasks. It is designed to facilitate rapid prototyping, accommodate diverse robot types, and enable seamless sim2real transfer via ROS2 integration. Core to Gewu Playground are three innovations addressing key embodied intelligence challenges: an efficient instruction learning-based RL framework; low-cost hardware requirements enabling efficient CPU-based training accessible to broader researchers; and deep integration with Unity's ecosystem to leverage its high-quality simulation capabilities, including photorealistic multi-modal rendering, flexible physics (rigid/soft-body, fluid dynamics), and modular scripting.

(a) Gewu Playground Main Menu



(b) Comparison of Gewu and IsaacLab



(c) Lunar Locomotion with Gewu

Figure 1 Gewu Playground Introduction.

Gewu Playground integrates eight specialized modules (Fig. 1a): universal locomotion, uneven terrain locomotion, motion imitation, Sim2Real transfer, robot soccer, mobile manipulation, autonomous navigation, and robot animation. Each module provides guided examples and supports the import of custom robot models with minimal configuration overhead, ensuring accessibility across diverse research applications.

The platform leverages Unity's modular ecosystem, utilizing the URDF Importer package to seamlessly convert Unified Robot Description Format (URDF) models into Unity's ArticulationBody components for high-fidelity physics simulation. For reinforcement learning, Gewu Playground employs the ML-Agents toolkit, which supports Proximal Policy Optimization (PPO), Soft Actor-Critic (SAC), as well as Multi-Agent Policy Optimization with Credit Assignment (MA-POCA) algorithms, ensuring robust policy optimization across various robotic tasks.

Gewu aims to be an inclusive embodied intelligence platform accessible to all. A key feature of Gewu Playground is its high learning efficiency achieved through Instruction Learning [9]. Unlike IsaacLab, which needs high-end GPUs like GeForce RTX 4080, Gewu can train efficiently on CPUs, achieving equal-level results with far fewer simulation timesteps (Fig. 1b). This shows its high "learning efficiency per step". While recognizing GPUs' value for large-scale tasks, Gewu is collaborating with Unity China to upgrade the PhysX engine for GPU-accelerated parallel simulation, which will boost step throughput 10–20x while still supporting CPU-only use.

The platform' high-quality simulation features and terrain construction tools make it an ideal environment for training robots for extraterrestrial exploration, such as lunar or Martian missions. To demonstrate Gewu Playground's capabilities, we investigated locomotion strategies for humanoid robots under lunar low-gravity conditions (Fig. 1c). The lunar environment's unique physical characteristics, particularly its reduced gravity (1/6 of Earth's), pose significant challenges for robotic movement and balance. Simulating these conditions during training is essential for enabling robots to develop adaptive locomotion strategies.

Using the Unitree G1 robot within Gewu Playground, we trained two efficient locomotion patterns: running and jumping. The control architecture employed a hybrid neural network, combining a conventional learnable neural network with a temporal network that injects open-loop actions based on time variables. This architecture allowed for efficient learning and the development of stable locomotion policies.

The lunar surface terrain was constructed using Unity Terrain Editor, which enabled precise replication of the Moon's cratered landscape and undulating terrain. Training involved creating multiple replicas of the robot, each with a randomly chosen yaw angle, to experience diverse ground conditions. The robots were trained for 10 million steps each on running and jumping tasks, achieving stable and efficient locomotion policies.

Performance evaluation revealed that the running policy consistently outperformed the jumping policy in terms of travel distance and stability. These findings provide valuable insights for future robotic lunar exploration missions, demonstrating Gewu Playground's ability to train adaptive movement strategies for humanoid robots in extraterrestrial environments.

Gewu Playground represents a significant advancement in robot simulation platforms, offering a unified, user-friendly framework for embodied intelligence research. Its key innovations include universally efficient reinforcement learning framework, ease of use with low hardware requirements, and seamless integration with Unity's rich ecosystem. The platform supports a wide range of tasks, including locomotion, manipulation, and navigation, making it suitable for both traditional and learning-based robotic research.

By lowering technical barriers, Gewu Playground empowers a broader range of researchers to contribute to the next generation of intelligent robotic systems. Its future plans include enabling GPU-accelerated training, adapting to more physical robot models, and expanding embodied intelligence applications. With its versatile learning architecture, Gewu Playground is poised to accelerate the transition from laboratory prototypes to interplanetary robotic systems, aligning with global space initiatives.

This work was supported by the Shanghai "Science and Technology Innovation Action Plan" Next-Generation Information Technology Domain Key Technology Breakthrough Program (Grant No. 24511103304).

1 Michel, O. (2004). Cyberbotics ltd. Webots™: professional mobile robot simulation. *International Journal of Advanced Robotic Systems*, 1(1), 5.

2 Koenig, N., & Howard, A. (2004, September). Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Vol. 3, pp. 2149-2154). IEEE.

3 Rohmer, E., Singh, S. P., & Freese, M. (2013, November). V-REP: A versatile and scalable robot simulation framework. In 2013 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 1321-1326). IEEE.

4 Mittal, M., Yu, C., Yu, Q., Liu, J., Rudin, N., Hoeller, D., ... & Garg, A. (2023). Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6), 3740-3747.

5 Zakka, K., Tabanpour, B., Liao, Q., Haiderbhai, M., Holt, S., Luo, J. Y., ... & Abbeel, P. (2025). Mujoco playground. *arXiv preprint arXiv:2502.08844.*

6 Kolve, E., Mottaghi, R., Han, W., VanderBilt, E., Weihs, L., Herrasti, A., ... & Farhadi, A. (2017). Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474.*

7 Juliani, A., Berges, V. P., Teng, E., Cohen, A., Harper, J., Elion, C., ... & Lange, D. (2018). Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627.*

8 Ye, L., Li, R., Hu, X., Li, J., Xing, B., Peng, Y., & Liang, B. (2025). Unity RL Playground: A Versatile Reinforcement Learning Framework for Mobile Robots. *arXiv preprint arXiv:2503.05146.*

9 Ye, L., Li, J., Cheng, Y., Wang, X., Liang, B., & Peng, Y. (2023). From knowing to doing: learning diverse motor skills through instruction learning. *arXiv preprint arXiv:2309.09167.*

SCIENCE CHINA
Technological Sciences

• Supplementary File•

*** 2025 Vol.** No.*: ***–***
doi: 10.1007/s11431-****

Gewu Playground: An Open-Source Robot Simulation Platform for Embodied Intelligence Research

Linqi Ye[1], Boyang Xing[2], Bin Liang[3], Lei Jiang[2], Yan Peng[1]

[1]*School of Future Technology, Shanghai University, 200444 Shanghai, China;*
[2]*National and Local Co-Built Humanoid Robotics Innovation Center, 201203 Shanghai, China;*
[3]*Navigation and Control Research Center, Department of Automation, Tsinghua University, 100084 Beijing, China*

Embodied intelligence, robot simulation, reinforcement learning, lunar locomotion, humanoid robot.

S1 Introduction

Embodied intelligence represents the combination of robotics and artificial intelligence, where simulation platforms provide the essential infrastructure for robots to develop adaptive behaviors through integrated perception, cognition, and action—bridging the gap between virtual training and real-world deployment.

The evolution of robotics has been inextricably linked to simulation platforms, with a series of influential and widely adopted tools emerging since the 1990s. Notable examples include: Webots (1998) [1]: Developed by the Cyberbotics company, renowned for its physics-accurate rendering and cross-platform compatibility; Gazebo (2004) [2]: Rapidly gained prominence through ROS integration; V-REP (2013, now CoppeliaSim) [3]: Featuring a modular architecture that supports multiple physics engines concurrently. Those robotic simulation platforms reveals distinct functional paradigms: Webots prioritizes perceptual fidelity with photorealistic rendering and standardized robot models, facilitating perception-driven navigation research; Gazebo dominates open-source academic research via its scalable plugin ecosystem, ROS-native integration, and support for large-scale multi-agent simulations, as evidenced by its widespread adoption in DARPA Robotics Challenge; while V-REP (CoppeliaSim) excels in modular versatility, supporting multiple physics engines and rendering modes through its distributed architecture, making it ideal for industrial manipulation prototyping. These platforms collectively address the fidelity-performance tradeoff and usability-flexibility balance, reflecting evolving efforts to bridge simulation-to-reality gaps through enhanced physics modeling and sensor simulation, as underscored by recent comparative studies [4,5].

While the aforementioned simulation platforms have achieved remarkable success in traditional robotic control field by primarily supporting algorithm development based on kinematic and dynamic models, recent years have witnessed a paradigm shift: model-based approaches are being increasingly superseded by learning-based methods, with reinforcement learning [6-8] emerging as the dominant framework for robotic control. To better accommodate the demands of robotic reinforcement learning, a proliferation of novel simulation platforms has emerged in recent years [9].

OpenAI Gym [10], a foundational framework for reinforcement learning algorithm development, provides standardized environments for benchmarking control policies, including classical robotic tasks like CartPole and MountainCar, and has been extended to robotics via the Roboschool and PyBullet integrations. PyRep [11], built atop V-REP, bridges the gap between traditional robotic simulation and deep learning by offering a Python API for rapid scene construction, domain randomization, and real-time sensor simulation, enabling end-to-end training of vision-based manipulation policies. Legged Gym [12] specializes in legged robotics, offering GPU-accelerated

*Corresponding author (email: ****)

physics simulations to efficiently train locomotion policies across diverse robot morphologies and terrains, with emphasis on rapid experimentation and sim2real deployment. Humanoid-Gym [13], built on NVIDIA Isaac Gym, focuses on humanoid robot locomotion through zero-shot sim2real transfer, incorporating domain randomization and advanced reward shaping for policy robustness. IsaacLab [14] provides a high-fidelity NVIDIA simulation toolkit with photorealistic rendering, supporting multi-modal robotic platforms via a unified API for seamless reinforcement learning algorithm integration. MuJoCo Playground [15] leverages the MuJoCo physics engine [16] for rapid sim2real policy iteration, featuring on-device rendering, domain randomization, and pre-built benchmarks across quadrupeds, humanoids, and dexterous manipulators. Finally, Genesis [17] employs data-driven generative physics modeling to create dynamic simulation environments for manipulation and locomotion tasks, enabling on-the-fly scenario generation for scalable experimentation.

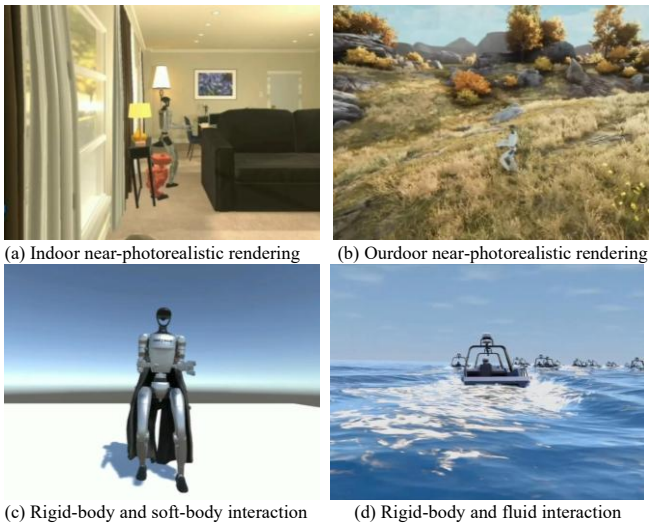
Besides, Unity is primarily known as a game engine, has emerged as a powerful platform for Embodied AI research, enabling the development of interactive 3D environments for robot learning. Notable frameworks built on Unity include AI2-Thor [18], which provides photorealistic indoor scenes for visual navigation tasks, and Unity ML-Agents Toolkit [19], which provides a user-friendly and versatile framework for the training of intelligent agents through reinforcement learning. Unity ML-Agents is designed to be intuitive and easy to use, with a focus on rapid development of games.

Table 1 Comparative Summary: Unity vs. MuJoCo/IsaacSim

Feature	Unity	MuJoCo	IsaacSim
Sensory Simulation	Multi-modal, photorealistic rendering	Low-dimension state inputs only	GPU-rendered but physics-prioritized
Physical Dynamics	Rigid/soft-body + real-time object spawning	Excellent rigid-body (fixed models only)	GPU-accelerated (fleet-focused)
Task/Multi-agent	Hierarchical tasks + native networking	Single-task (no multi-agent)	Robotic control (limited customization)

Unlike pure robotic control, embodied RL requires multi-dimensional environmental complexity (sensory, physical, task-logic, social) to bridge the reality gap—an area where Unity has particular advantages, as shown in Table 1. First, Unity supports near-photorealistic rendering (dynamic lighting, custom shaders) and multi-modal inputs (depth maps, LiDAR emulation), critical for pixel-level/multi-modal perception tasks. In contrast, MuJoCo focuses on low-dimensional state inputs, while IsaacSim prioritizes physics over sensory flexibility. Second, Unity integrates industry-leading physics engines (NVIDIA PhysX, Havok), enabling rigid-body, soft-body, and fluid dynamics—supporting dynamic interactions (e.g., deformable objects, real-time obstacle spawning). MuJoCo lacks flexible object instantiation, and IsaacSim is optimized for large robotic fleets but not dynamic environment design. Third, Unity's C# scripting enables hierarchical tasks and multi-agent collaboration via built-in networking. MuJoCo has no multi-agent support, and IsaacSim limits interactions to agent-robot scenarios.

Based upon Unity, we developed Unity RL Playground [20], a dedicated reinforcement learning framework for mobile robots. It is designed to be operated with minimal programming expertise, allowing users to easily import their custom robot models for comprehensive multi-modal motion training. However, Unity RL Playground was originally designed for locomotion tasks of mobile robots, restricting its applicability to broader embodied intelligence tasks. To address this, we developed the Gewu Playground as an enhanced extension of Unity RL Playground, transforming it into a comprehensive research platform for embodied intelligence with multi-domain task support.



(a) Indoor near-photorealistic rendering (b) Ourdoor near-photorealistic rendering

(c) Rigid-body and soft-body interaction (d) Rigid-body and fluid interaction

Figure 1 Gewu high-quality simulation capabilities.

Gewu Playground integrates three core innovations to address key challenges in embodied intelligence research: first, an efficient reinforcement learning framework built on the instruction learning paradigm, which achieves high "learning efficiency per step" by synergizing feedforward action primitives with RL-based stabilization, enabling rapid policy convergence even for complex tasks; second, low-cost hardware requirements that eliminate dependency on high-end GPUs—training can be completed efficiently on standard CPUs, making embodied intelligence research accessible to a broader audience; third, deep integration with Unity's ecosystem and access to its high-quality

simulation capabilities (see Figure 1), leveraging its photorealistic multi-modal rendering, flexible physics simulation (rigid-body, soft-body, fluid dynamics), and modular scripting to support diverse task scenarios from terrestrial manipulation to extraterrestrial locomotion.

In the coming era, embodied intelligence will transcend terrestrial boundaries—not only reshaping our daily life through intelligent agents but also spearheading humanity's expansion into space, as evidenced by China's 2035 vision for the International Lunar Research Station and SpaceX's Mars Base Alpha initiative. This cosmic frontier poses unprecedented challenges for robotic exploration: the lunar and Martian gravities—merely 1/6 and 3/8 of Earth's—combined with unique terrains demand radical rethinking of locomotion control. Previous research has primarily focused on quadrupedal robot locomotion on the Moon and Mars [21,22], while studies involving humanoid robots remain scarce. Addressing these extraterrestrial constraints, Gewu Playground emerges as a universal simulation infrastructure that enables cross-planetary dynamics modeling via adjustable gravity fields and terrain construction supports. By bridging the sim2real gap for both terrestrial and extraterrestrial environments, Gewu Playground establishes itself as a critical enabler for space-ready embodied intelligence, accelerating the transition from laboratory prototypes to interplanetary robotic systems.

The contributions of this paper are as follows. First, we develop the framework of Gewu Playground. Compared to Unity RL Playground, Gewu Playground has undergone a comprehensive upgrade, not only expanding locomotion tasks to include complex terrain adaptation and whole-body imitation learning but also introducing new modules including imitation learning, manipulation, and navigation. Furthermore, it integrates ROS2 to enable higher-performance and more universal sim2real transfer capabilities. Second, leveraging the Gewu Playground framework, we investigated locomotion strategies for humanoid robots under lunar low-gravity conditions, successfully training two efficient locomotion patterns—running and jumping—which were validated through simulation on virtual lunar terrain. These findings provide critical technical support for future robotic lunar exploration missions.

S2 Gewu Playground Framework

Gewu Playground framework is shown in Figure 2. Leveraging Unity's modular ecosystem, we streamline robotic system integration through the URDF Importer package, which enables seamless conversion of Unified Robot Description Format (URDF) models into Unity's ArticulationBody components—optimized for high-fidelity physics simulation via the PhysX engine. For reinforcement learning implementation, we utilize the ML-Agents toolkit, which provides PyTorch-backed training pipelines supporting Proximal Policy Optimization (PPO), Soft Actor-Critic (SAC) algorithms, and Multi-Agent Policy Optimization with Credit Assignment (MA-POCA), ensuring policy optimization across diverse robotic tasks.

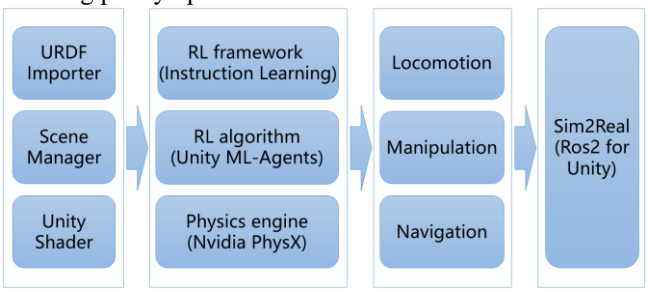


Figure 2 Gewu Playground framework.

Gewu Playground currently integrates eight specialized modules—universal locomotion, uneven terrain locomotion, motion imitation, Sim2Real transfer, robot soccer, mobile manipulation, autonomous navigation, and robot animation—where the first three modules form the foundational reinforcement learning training infrastructure for developing core robotic motor skills, enabling subsequent adaptation to complex tasks; each module includes guided examples demonstrating its functionality, and recognizing the need for custom hardware integration, we provide an intuitive template for importing and training user-defined robotic models with minimal configuration overhead, ensuring seamless accessibility across diverse research applications.

(1) Universal Locomotion

This module facilitates foundational locomotion training for diverse robotic morphologies, including bipedal, quadrupedal, biped-wheeled hybrid, and quadruped-wheeled hybrid systems. As presented in Figure 3, we have tested over 80 different robots using Gewu Playground.

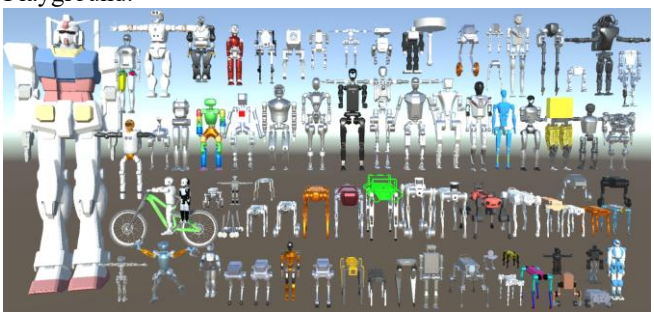


Figure 3 Diverse robot support of Gewu playground.

For each configuration, we provide three distinct motion control modes implemented via an instruction learning [23] paradigm that synergizes feedforward action primitives with reinforcement learning-based stabilization. To streamline

user interaction, we developed a dedicated setup interface within Unity's Inspector panel (Figure 4, right), enabling intuitive specification of robot type and target motion patterns during URDF model import. The training workflow is further optimized through a "Fixbody" validation toggle, allowing users to pre-verify feedforward action correctness before training. For computational efficiency, the system automatically spawns 32 parallel robot instances during training, achieving 20× real-time simulation acceleration while maintaining physics fidelity through Unity's deterministic PhysX integration. It should be noted that GPU-based parallel training is currently not supported; however, the training speed remains highly efficient, depending primarily on CPU performance. Even with a standard laptop, training can typically be completed within tens of minutes to a few hours.

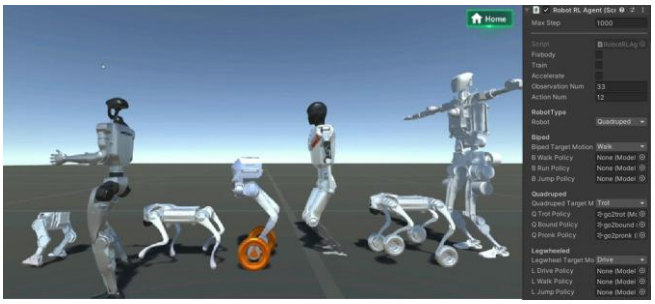


Figure 4 Universal locomotion diagram.

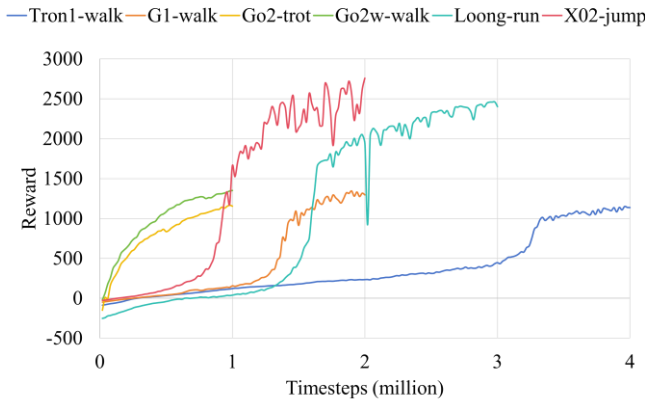


Figure 5 Reward curves for universal locomotion.

Table 2 Training Details for Universal Locomotion

Robot	Tron1-walk	G1-walk	Go2-trot	Go2w-walk	Loong-run	X02-jump
Simulation timesteps (million)	4	2	1	1	3	2
Time for training (hour)	0.31	0.19	0.10	0.10	0.33	0.17

We employed a laptop (Intel(R) Core(TM) i9-14900HX CPU and an NVIDIA GeForce RTX 4080 GPU) for training. The reward curve is depicted in Figure 5, and the training details are listed in Table 2. It can be observed that all six locomotion tasks can be trained and completed within a very short time (1 to 4 million timesteps, spanning 0.1 to 0.3 hours).

(2) Rough Terrain Locomotion

To train complex terrain locomotion capabilities, we developed a pyramid staircase environment incorporating both convex (upward) and concave (downward) stairs. The neural network architecture remained identical to prior implementations, with the key innovation being the adoption of a terrain-adaptive curriculum learning strategy. During training, we progressively increased step heights from 5 cm to 10 cm, and subsequently to 15 cm, to systematically enhance environmental complexity. Four humanoid robots with distinct structural designs and physical dimensions were selected for training, all of which ultimately demonstrated proficient stair climbing and descending abilities.

We employed a laptop (Intel(R) Core(TM) i9-14900HX CPU and an NVIDIA GeForce RTX 4080 GPU) for training. For the four robots, we trained each for 20 million timesteps, and the time taken ranged from 1.28 to 1.57 hours. The reward curve is depicted in Figure 7, and the training details are listed in Table 3.

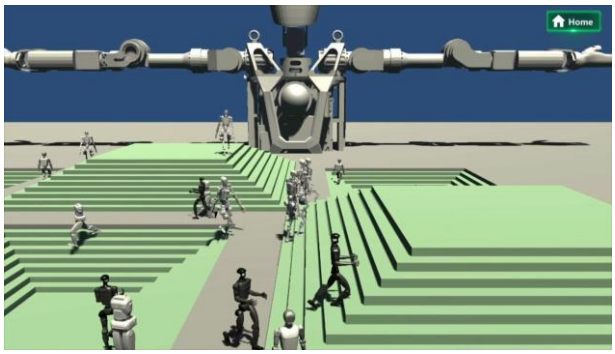


Figure 6 Rough terrain locomotion diagram.

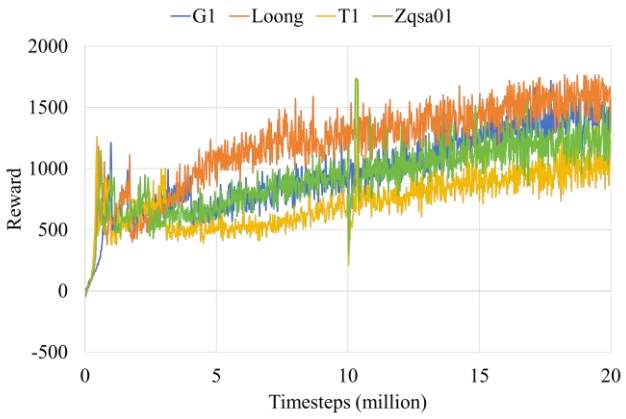


Figure 7 Reward curves for rough terrain locomotion.

Table 3 Training Details for Rough Terrain Locomotion

Robot	G1	Loong	T1	Zqsa01
Simulation timesteps (million)	20	20	20	20
Time for training (hour)	1.51	1.57	1.30	1.28

(3) Motion Imitation

The motion imitation module facilitates whole-body motion imitation learning for humanoid robots by retargeting human motion capture data to robotic joints and conducting training through instruction learning methods, enabling the robots to acquire human-like motion patterns. We have provided some retargeted data for Unitree H1 and G1 robots, sourced from the AMASS [24] and LEFAN1 [25] datasets respectively, and pre-trained several motion sequences including guitar playing, golf swinging, violin playing, and waving for H1 (using a shared neural network), as well as Charleston dancing for G1. To use this module, users can import new motion data, modify targeted motion to imitate via the inspector window, and enable the Replay option to visualize motion retargeting animations.

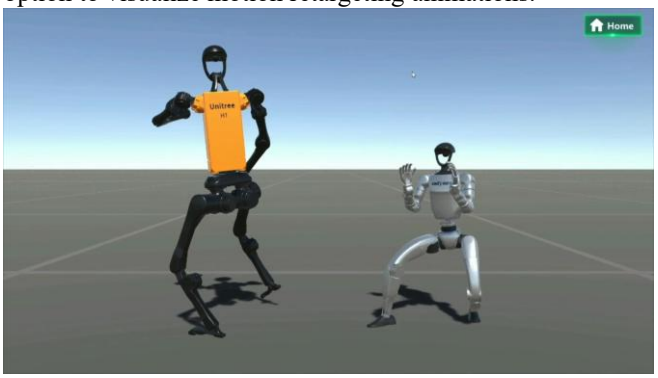


Figure 8 Motion imitation diagram.

We employed a laptop (Intel(R) Core(TM) i9-14900HX CPU and an NVIDIA GeForce RTX 4080 GPU) for training. The reward curves are depicted in Figure 9 and Figure 10, and the training details are listed in Table 4. For H1 robot, we trained a single neural network for 7 movements (golf, guitar, tennis, violin, wave both, wave left, wave right) over 12 million steps using curriculum learning. During the first 10 million steps, we trained each movement for 300 seconds before switching to the next to ensure full practice of every action; in the last 2 million steps, we reduced each movement's training time to 30 seconds (i.e., increased switching frequency) to effectively mitigate forgetting. For G1 robot, we trained a long-cycle task—the Charleston dance—which took 3.43 hours to complete.

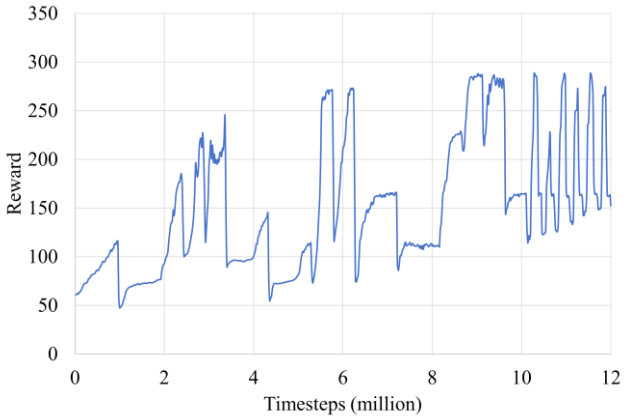


Figure 9 Reward curve for H1 imitation task.

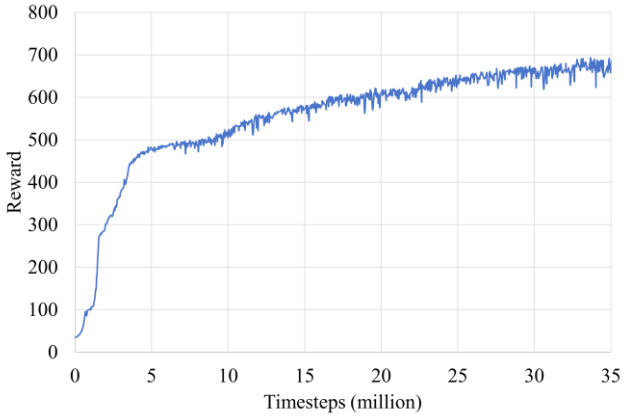


Figure 10 Reward curve for G1 imitation task.

Table 4 Training Details for Imitation Learning

Robot	H1	G1
Simulation timesteps (million)	12	35
Time for training (hour)	1.07	3.43

(4) Sim2Real

Sim2Real serves as a bridge connecting simulation and physical robots. Leveraging ROS2 For Unity, we developed an integrated Sim2Real framework and applied it to the Unitree Go2 robot. ROS2 For Unity provides a high-performance communication solution that natively connects the Unity3D engine with the ROS2 ecosystem. To enable real-time robot-Unity communication, we compiled ROS2 core functions and Unitree's ROS packages into dynamic link libraries (DLLs) callable within Unity, allowing the creation of ROS nodes in Unity for subscribing to sensor data from and publishing control commands to the Go2 robot. We established ROS communication via Ethernet connection between the robot and computer,

achieving state and sensor data alignment between the simulated and physical robots. This enables identical control of both simulated and physical robots through Unity, as if they were the same model. By deploying two identical pre-trained neural networks—one for the simulated robot and one for the physical counterpart—we simultaneously visualized control effects in both environments (see Figure 11). A dedicated control interface was developed: after program initiation, users click "Stand Up" to raise the robot, enable feedforward control via the "FF Enable" checkbox (triggering stepping motion), then activate neural network control with "NN Enable" for keyboard-based forward/turning movement, and finally use "Lie Down" to deactivate the robot. This functionality enables seamless transfer and rapid validation of trained policies. Moreover, owing to ROS2's platform-agnostic design, our approach can be easily extended to other robotic platforms.

Figure 11 Sim2Real diagram.

(5) Robot Soccer

In the robot soccer module, we implemented a dual-robot (OpenLoong) combat and soccer system. Adopting a hierarchical control architecture, the low-level motion control employs reinforcement learning to enable basic locomotion (forward movement and steering), while the high-level strategy utilizes rule-based decision-making: one robot tracks and kicks the ball, while the other pursues the kicking robot to initiate combat, delivering punches to knock it down when within striking distance. Fallen robots automatically reset to upright positions. This configuration produces rich adversarial interactions between the robots. Additionally, the high-level strategy can also be trained via reinforcement learning using ML-Agents' MA-POCA algorithm.

Figure 12 Robot soccer diagram.

(6) Mobile Manipulation

The mobile manipulation module is designed for humanoid robot operation tasks. Currently, it provides a foundational keyboard-based control interface, enabling users to manipulate robot locomotion (walking, stopping, forward/backward movement, and left/right turning) and dual-arm end-effector poses with gripper actuation via computer keystrokes. Locomotion is achieved through reinforcement learning, while manipulation employs inverse kinematics (IK). Since Unity lacks built-in IK algorithms, we developed an innovative solution: we duplicated an identical robot model for IK computation (referred to as the IK robot). This replica operates in a low-gain PD control mode (follower mode), allowing effortless end-effector positioning akin to manually guiding a robotic arm. By fixing the IK robot's torso and connecting both end-effectors to static objects via fixed joints, we manipulate the static objects to desired positions, causing the robotic arm joints to follow accordingly. The resulting joint angles of this IK robot represent the IK solutions, which are then applied to the corresponding joints of the controlled robot, enabling precise end-effector translation and rotation. This approach successfully validated cube grasping tasks, demonstrating its effectiveness.

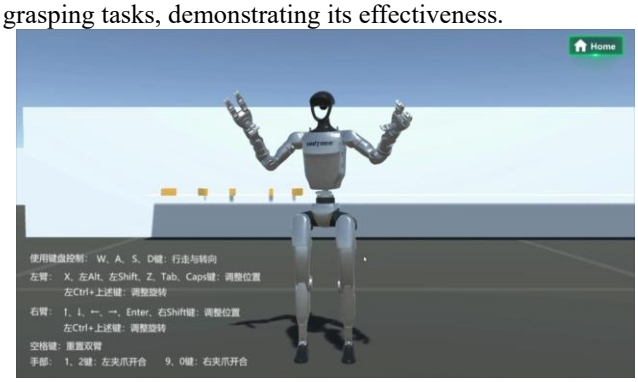


Figure 13 Mobile manipulation diagram.

(7) Autonomous Navigation

The autonomous navigation module leverages Unity's AI Navigation package—a game development-oriented navigation system that enables intelligent path planning and movement control for in-game characters. This system allows characters to automatically compute optimal paths based on environmental obstacles, terrain features, and other scene information, then navigate to target positions along these trajectories. When adapted for robotic applications, we employ AI Navigation to control the motion of a massless virtual object, with the physical robot following its movement. We developed a park scenario for testing and implemented it on a Unitree Go2 robot. Through mouse-click target selection on the screen, the robot successfully navigates to designated positions using AI Navigation-planned routes while avoiding obstacles.

Figure 14 Autonomous navigation diagram.

(8) Robot Animation

The robot animation module enables the creation of realistic robot animations for scenarios where dynamic simulation is not required. Most video games utilize animation systems rather than physics-based simulation, resulting in mature, user-friendly, and highly versatile animation frameworks.

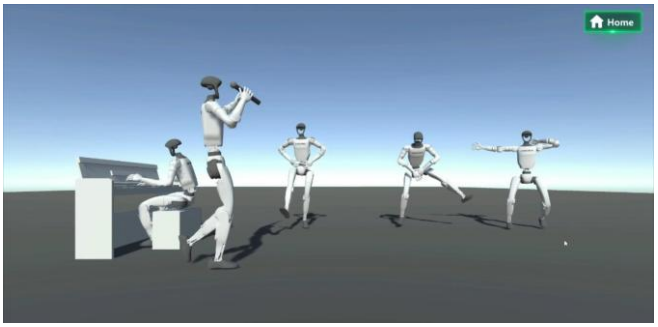


Figure 15 Robot animation diagram.

To control robots via animation, the critical step is converting them into skeletal structures. This process is straightforward in Unity: simply import the robot's URDF model, convert it to an FBX file, perform proper skeletal rigging, and then leverage Unity's animation tools. By dragging animation sequences onto the robot's skeletal hierarchy, rapid animation implementation becomes possible. We applied this workflow to the G1 robot, completing skeletal rigging and producing a concert performance animation that demonstrates exceptionally smooth motion effects.

S3 Learning Efficiency of Gewu

Gewu's core design goal is to build an inclusive embodied intelligence platform accessible to the general public. While GPU-based large-scale parallel training significantly accelerates training speed, it also imposes high hardware requirements on computers. For instance, IsaacLab mandates a minimum configuration of GeForce RTX 4080 with 16 GB VRAM, making it hardly accessible to the general public. In contrast, to develop a universal and inclusive embodied intelligence platform, Gewu enables efficient training without relying on a GPU—this advantage is attributed to the Instruction Learning framework we adopted.

To evaluate the learning efficiency of Gewu quantitatively, we compared it with IsaacLab on one reinforcement learning tasks using the Go2 quadruped robot—locomotion on flat ground (Go2-Flat), as well as three reinforcement learning tasks using the G1 humanoid robot—locomotion on flat ground (G1-Flat), locomotion on stairs (G1-Rough), and imitation learning (G1-Dance)—with training conducted on the same laptop (Intel(R) Core(TM) i9-14900HX CPU, NVIDIA GeForce RTX 4080 GPU; note: Gewu ran on CPU only, while IsaacLab used GPU acceleration). The obtained simulation results can be found in the attached video.

(1) Go2-Flat Task

The Go2-Flat task involves quadruped locomotion with omnidirectional movement on flat ground. We trained until the robot could complete the tasks stably. The reward curves for Gewu and IsaacLab are depicted in Figure 16 and Figure 17, respectively.

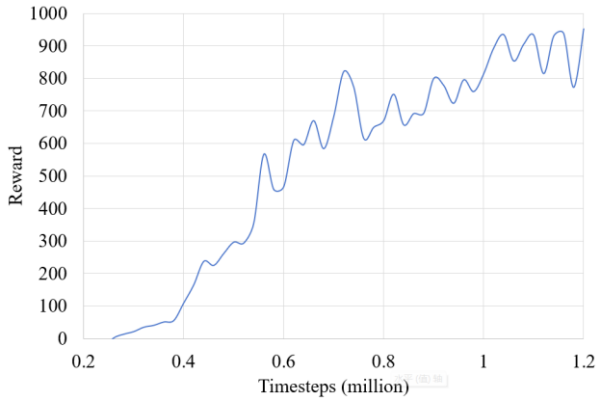


Figure 16 Reward curve for Go2-Flat-Gewu task.

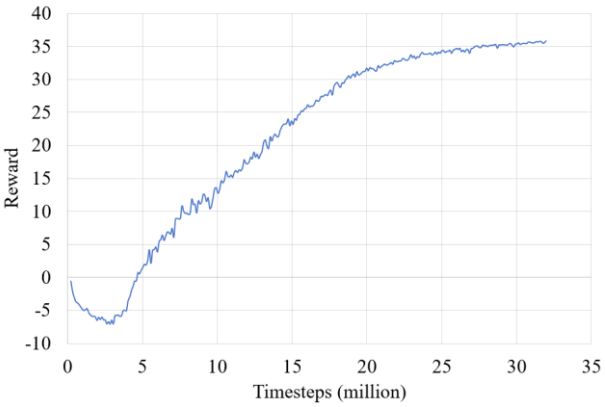


Figure 17 Reward curve for Go-Flat-IsaacLab task.

(2) G1-Flat Task

The G1-Flat task involves standing and omnidirectional walking on flat ground. We trained until the robot could complete the tasks stably. The reward curves for Gewu and IsaacLab are depicted in Figure 18 and Figure 19, respectively.

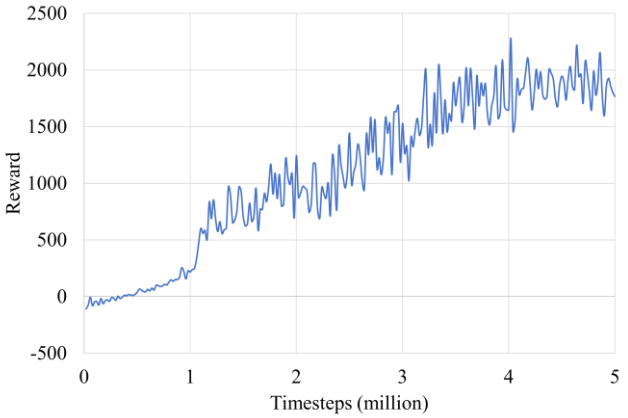


Figure 18 Reward curve for G1-Flat-Gewu task.

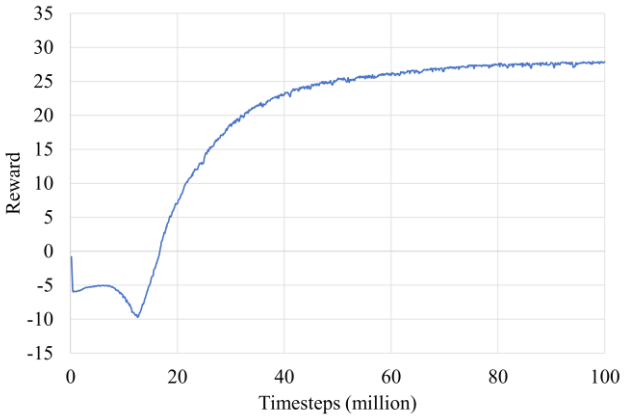


Figure 19 Reward curve for G1-Flat-IsaacLab task.

(3) G1-Rough Task

The G1-Rough task involves training the robot to walk on a rectangular-loop staircase (each step is 15 cm high and 30 cm wide), and we continued the training until the robot achieved an almost 100% success rate in climbing up and down the stairs. The reward curves for Gewu and IsaacLab are depicted in Figure 20 and Figure 21, respectively.

(4) G1-Dance Task

The G1-Dance task involves training the robot to perform the Charleston dance, and we continued the training until the robot could stably complete the first 10 seconds of the dance. The reward curves for Gewu and IsaacLab (using AMP algorithm) are depicted in Figure 22 and Figure 23, respectively.

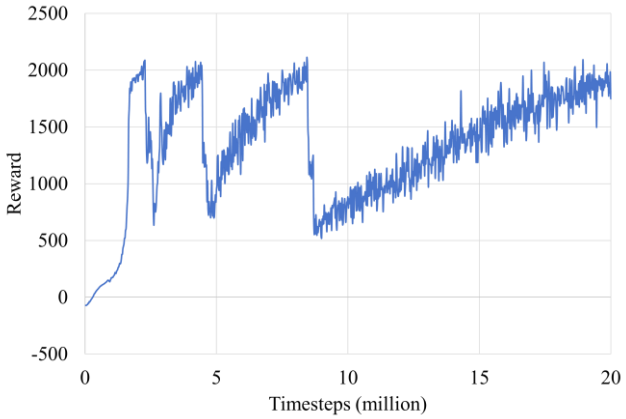


Figure 20 Reward curve for G1-Rough-Gewu task.

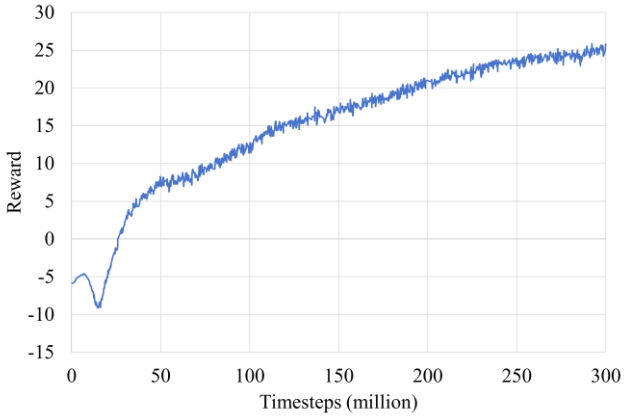


Figure 21 Reward curve for G1-Rough-IsaacLab task.

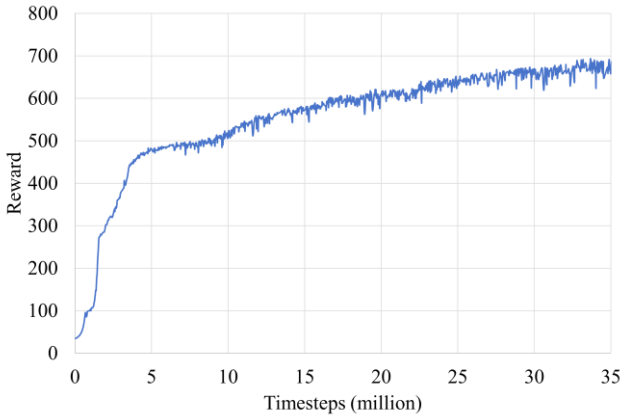


Figure 22 Reward curve for G1-Dance-Gewu task.

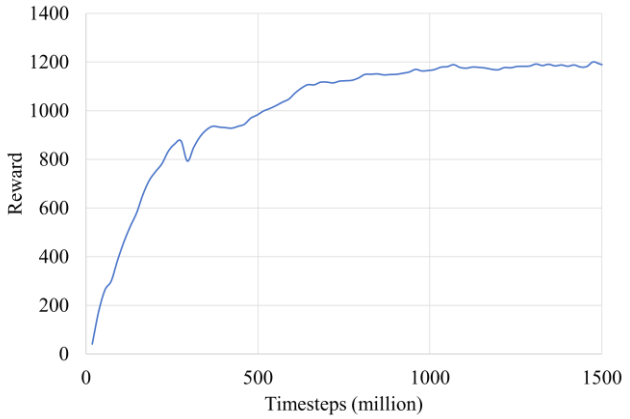


Figure 23 Reward curve for G1-Dance-IsaacLab task.

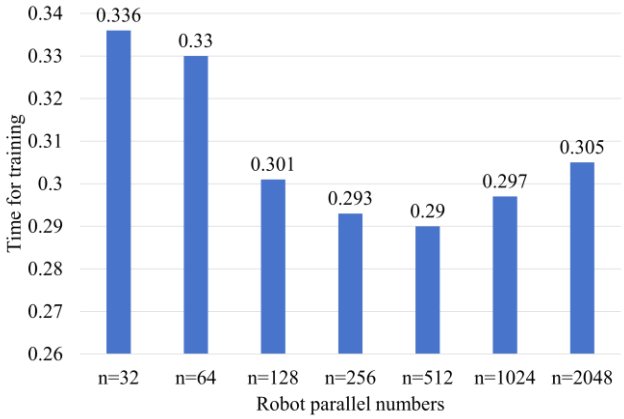


Figure 24 Gewu training time for different robot parallel numbers.

Table 5 summarizes the quantitative comparison between Gewu and IsaacLab across the three reinforcement learning tasks. It can be observed that although Gewu has a lower timesteps per hour compared to IsaacLab, it achieves the same performance goals (see attached video) with significantly fewer simulation timesteps. For instance, in the G1-Flat task, Gewu only needs 5 million simulation timesteps while IsaacLab requires 100 million; in the G1-Rough task, Gewu needs 20 million versus IsaacLab's 300 million; and in the G1-Dance task, Gewu needs 35 million compared to IsaacLab's 1500 million. This confirms Gewu's high "learning efficiency per step" which is a more meaningful metric for inclusive platforms than raw step throughput.

Table 5 Training Comparison between Gewu and IsaacLab

RL Task	Simulation timesteps (million)	Time for training (hour)	Training speed (million/hour)
Go2-Flat-Gewu	1.2	0.076	16
Go2-Flat-IaacLab	32	0.082	390
G1-Flat-Gewu	5	0.29	17
G1-Flat-IsaacLab	100	0.33	303
G1-Rough-Gewu	20	300	18
G1-Rough-IsaacLab	1.12	1.23	244
G1-Dance-Gewu	35	1500	10
G1-Dance-IsaacLab	3.43	4.20	357

To explore how to make Gewu train faster, we investigated the impact of increasing the number of parallel training robots on the training process based on G1-Flat-Gewu. Seven cases with parallel robot numbers n = 32, 64, 128, 256, 512, 1024, 2048 were considered, each trained for 5 million steps (using no-graphics mode without rendering).

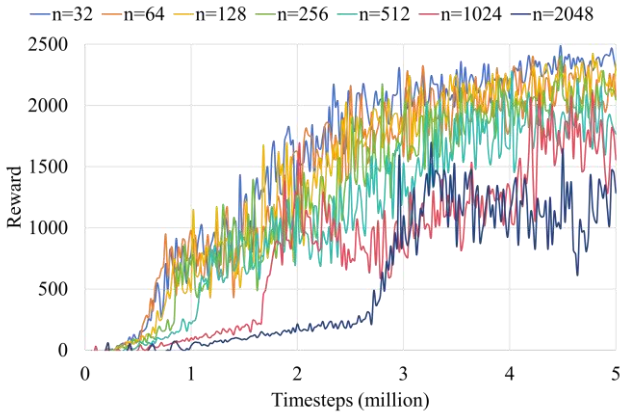


Figure 25 Gewu reward curves for different robot parallel numbers.

The training time is shown in Figure 24, and the reward curves are shown in Figure 25. It can be seen from Figure 24 that the training speed is not faster as n increases; the training speed is the fastest when n=256, but it is only 13.7% faster than when n=32. In addition, when n increases, the reward curve tends to deteriorate. For example, the reward rises much more slowly when n=2048. Therefore, it is recommended to use a parallel number below 512 for training.

Regarding GPU support, we acknowledge its importance for large-scale RL workloads and are actively addressing this: we have established collaboration with Unity China to upgrade Unity's PhysX engine, aiming to enable GPU-accelerated parallel environment simulation for Gewu. This upgrade will allow Gewu to leverage GPU resources (when available) for physics calculation offloading, projected to increase step throughput by 10–20x while retaining CPU-only compatibility for accessibility.

S4 Domain Randomization

To enhance the robustness of the reinforcement learning policy and reduce the sim-to-real gap, we have integrated domain randomization and observation noise into the Gewu platform. In this section, we will investigate the performance of Gewu under the conditions of domain randomization and observation noise.

Taking the Go2-Flat and G1-Flat tasks as examples, Table 6 lists the details of domain randomization and observation noise. The training reward curves are shown in Figs. 26 and 27, and the quantitative indicators of the training process are presented in Table 7.

As can be seen from Figs. 26 and 27, the reward decreases slightly after adding domain randomization. However, this does not indicate a decline in performance. To evaluate the performance of the trained policy, we tested the failure case under domain randomization conditions (conducted 20 runs, 10 seconds per run, with 16 robots, and recorded the number of fallen robots). As shown in Table 7, the failure rate of the policy trained with domain randomization is significantly reduced, demonstrating the effectiveness of domain randomization in improving robustness. Additionally, it can be observed from Table 7 that domain randomization has little impact on training time. For the same number of simulation timesteps, the training time only increases by around 25%, which verifies the efficiency of training with domain randomization on the Gewu platform.

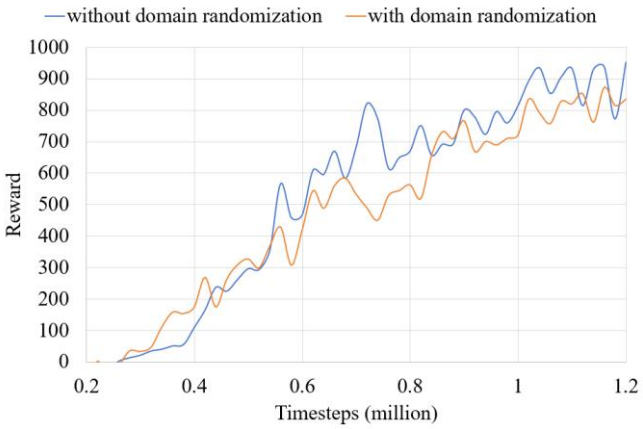


Figure 26 Go2 reward curve with and without domain randomization.

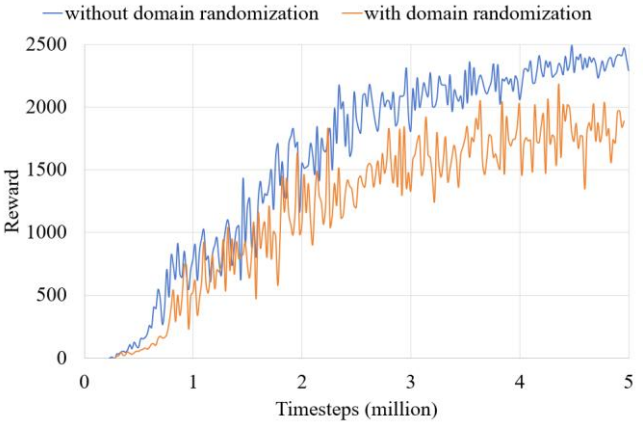


Figure 27 G1 reward curve with and without domain randomization.

Table 6 Domain Randomization and Observation Noise

Randomized Factor	Parameter Variation	Noise	Noise Level
Friction	[0.1, 1.25]	**Angular velocity noise**	0.05
Robot Mass	[-3.0, 8.0] kg	**Gravity projection noise**	0.05
Push velocity	G1: [1.0,1.5]m/s Go2: [1.5,3.0]m/s	**Joint position noise**	0.01
Push frequency	Every 3 seconds	**Joint velocity noise**	0.075

Table 7 Training Comparison for w/ and w/o Domain Randomization

Task	Simulation timesteps (million)	Time for training (min)	Failure case
Go2-Flat (trained without domain randomization)	1.2	4.5	6
Go2-Flat (trained with domain randomization)	1.2	5.7	0
G1-Flat (trained without domain randomization)	5	20	91
G1-Flat (trained with domain randomization)	5	25	3

S5 Application to Lunar Locomotion Task

When Apollo astronauts first set foot on the Moon, they discovered that movement and balance became extraordinarily difficult due to the lunar environment's unique physical conditions. This challenge is not exclusive

to humans—humanoid robots face equally daunting obstacles in lunar locomotion. The most critical factor affecting robotic movement in this environment is the reduced gravity, which is merely 1/6 of Earth's, rendering terrestrial gait patterns ineffective. Consequently, simulating low-gravity conditions during training is essential for enabling robots to develop adaptive locomotion strategies. To address this, we investigated lunar locomotion using the G1 robot within the Gewu Playground simulation framework, yielding preliminary insights into effective movement approaches.

(1) Control Architecture

We employ the control architecture provided by Gewu, which originates from the instruction learning method we proposed earlier. Here, we examine it from an alternative perspective and can regard it as a hybrid neural network, as illustrated in Figure 28. The upper part of this architecture is a conventional learnable neural network, whose input is the robot's state, and its weights are updated by the reinforcement learning algorithm during training. The lower part is a temporal network, with time variables as its input. It operates as a preset function solely dependent on time, directly injecting open-loop actions into the robot. These actions remain frozen and are not updated during training. The functions of these two networks are complementary, with the temporal network being interpretable and the state network being learnable. By integrating these two networks, efficient learning can be achieved.

For running and jumping, we only employed different temporal networks, while all other components, including the reward function, remained exactly the same. For running, the temporal network is designed to make the robot's legs alternate in a periodic stepping motion. For the jumping task, the temporal network is designed to make the robot's legs bend and straighten synchronously in a periodic manner.

We use the same reward $r = r_a + r_b + r_v$ for both running and jumping gaits (only difference is the feedforward signal). Each component of the reward is

1) $r_a = 1$ is the alive reward. The robot receives this reward per time step for not falling.

2) $r_b = -0.1|\theta_{pitch}| - 0.1|\theta_{roll}|$ is the balance reward (unit: degree), which encourages keeping the body upright.

3) $r_v = v_{forward} + |v_{vertical}| - |v_{lateral}| - 2|v_{yaw}|$ is the velocity reward, which encourages the robot to move forward.

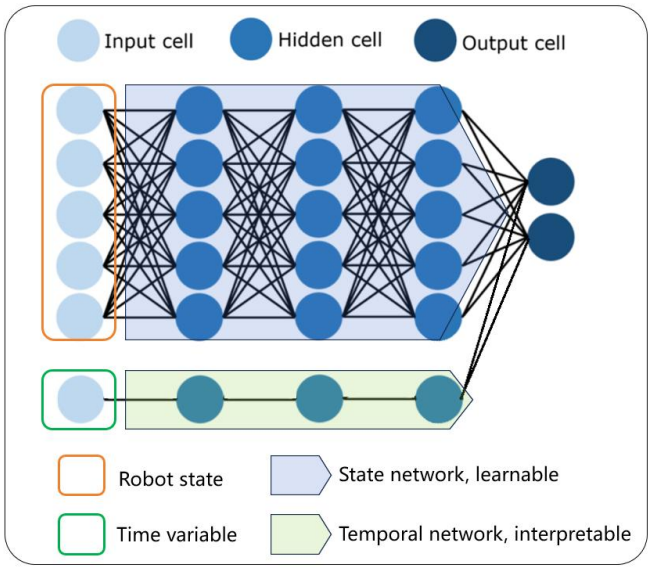


Figure 28 The concept of hybrid neural network.

(2) Locomotion Training

We utilize Unity Terrain Editor to construct the lunar surface terrain, leveraging its powerful functionality that allows for the intuitive creation and modification of large-scale terrains through tools like height mapping, texture painting, and rock placement. Given the unique characteristics of the lunar surface—such as its cratered landscape, dramatic undulating terrain with steep slopes and deep depressions, and uniform regolith texture—the Terrain Editor enables us to precisely replicate these features by adjusting elevation details, applying realistic lunar textures, and simulating lighting conditions to achieve an authentic lunar environment.

We selected a specific point on the constructed lunar surface as the starting position for the robot. During training, we created 24 replicas of the robot, each with a randomly chosen yaw angle (moving direction) as shown in Figure 29, which allows the robots to experience as many diverse ground conditions as possible. Using a simulation time step of 0.01 seconds, we reset the robots to the starting position every 10 seconds. We trained the robots for 10 million steps each on both the running and jumping tasks, with each task requiring a simulation duration of 27.78 hours, while the actual training time consumed was only 1.67 hours. The reward curves are depicted in Figure 30. It can be observed that the reward for running rises more rapidly and reaches a higher final value. This may suggest that on the lunar surface, running enables the robot to move faster and with greater stability.

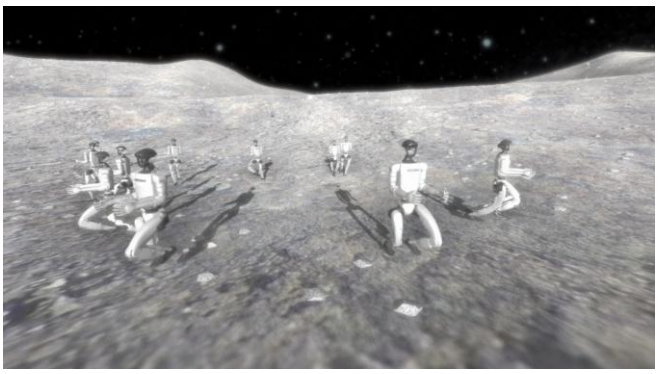


Figure 29 Locomotion training on simulated lunar surface.

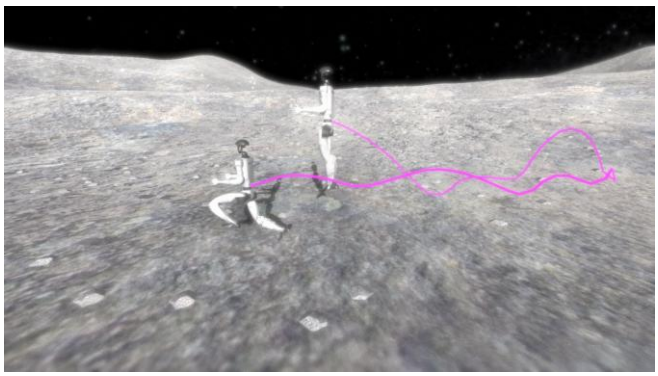


Figure 31 Moving trajectory visualization.

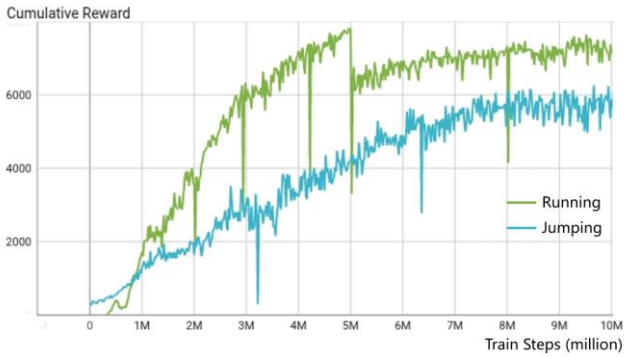


Figure 30 The reward curves.

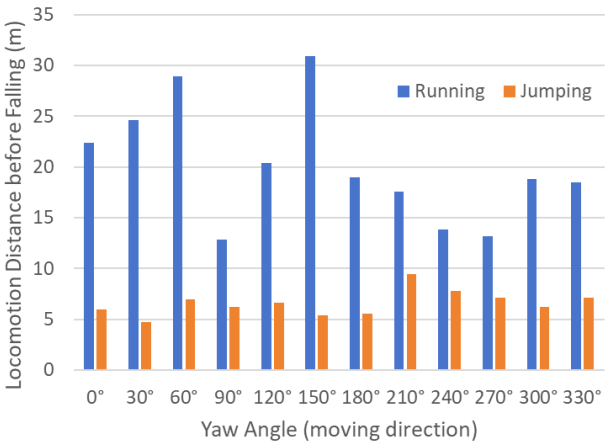


Figure 32 Locomotion performance comparison.

(3) Performance Evaluation

To evaluate the performance of the derived locomotion policy, we had two robots perform running and jumping respectively, and recorded their motion trajectories as shown in Figure 31. It can be observed that the robot is capable of jumping to significant heights on the lunar surface, and its running gait also differs from that on Earth (which is more evident in the supplementary video). For quantitative assessment of locomotion performance, we instructed the robot to move continuously in a straight direction until it lost balance and fell, recording the horizontal distance traveled. We selected forward directions at 30-degree intervals, conducting three trials per direction and averaging the results. The findings are presented in Figure 32. As shown, the running policy consistently outperformed the jumping policy in terms of travel distance across all directions, indicating that running remains a more efficient and stable mode of locomotion on the lunar surface.

S6 Conclusions

This paper introduces Gewu Playground, a versatile, open-source robot simulation platform built on Unity that supports a wide range of embodied intelligence research, including locomotion, manipulation, and autonomous navigation. Its key innovations include enhanced Sim2Real capabilities through ROS2 integration, adjustable gravity fields, and terrain tools for modeling extraterrestrial environments like lunar and Martian surfaces, enabling rapid prototyping and experimentation across diverse robot morphologies. A case study on lunar locomotion demonstrated its ability to train adaptive movement strategies for humanoid robots, revealing that running policies outperformed jumping in stability, providing valuable insights for future lunar exploration missions.

Gewu Playground represents a significant advancement by offering a unified, user-friendly framework for both traditional and learning-based robotic research across terrestrial and extraterrestrial scenarios, aligning with global space initiatives. Looking ahead, plans include enhancing physics modeling, expanding multi-agent simulations, and improving cross-platform compatibility, while fostering a

vibrant research community through comprehensive documentation and resources. By lowering technical barriers, Gewu Playground aims to empower a broader range of researchers to contribute to the next generation of intelligent robotic systems for Earth and beyond.

This work was supported by the Shanghai "Science and Technology Innovation Action Plan" Next-Generation Information Technology Domain Key Technology Breakthrough Program (Grant No. 24511103304).

1 Michel, O. (2004). Cyberbotics ltd. Webots™: professional mobile robot simulation. International Journal of Advanced Robotic Systems, 1(1), 5.
2 Koenig, N., & Howard, A. (2004, September). Design and use paradigms for gazebo, an open-source multi-robot simulator. In 2004 IEEE/RSJ international conference on intelligent robots and systems (IROS) (Vol. 3, pp. 2149-2154). IEEE.
3 Rohmer, E., Singh, S. P., & Freese, M. (2013, November). V-REP: A versatile and scalable robot simulation framework. In 2013 IEEE/RSJ international conference on intelligent robots and systems (pp. 1321-1326). IEEE.
4 Collins, J., Chand, S., Vanderkop, A., & Howard, D. (2021). A review of physics simulators for robotic applications. IEEE Access, 9, 51416-51431.
5 Pitonakova, L., Giuliani, M., Pipe, A., & Winfield, A. (2018, July). Feature and performance comparison of the V-REP, Gazebo and ARGoS robot simulators. In Annual Conference Towards Autonomous Robotic Systems (pp. 357-368). Cham: Springer International Publishing.
6 Miki, T., Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., & Hutter, M. (2022). Learning robust perceptive locomotion for quadrupedal robots in the wild. Science robotics, 7(62), eabk2822.
7 Vollenweider, E., Bjelonic, M., Klemm, V., Rudin, N., Lee, J., & Hutter, M. (2022). Advanced skills through multiple adversarial motion priors in reinforcement learning. arXiv preprint arXiv:2203.14912.
8 Hoeller, D., Rudin, N., Sako, D., & Hutter, M. (2024). Anymal parkour: Learning agile navigation for quadrupedal robots. Science Robotics, 9(88), eadi7566.
9 Liu, Y., Chen, W., Bai, Y., Liang, X., Li, G., Gao, W., & Lin, L. (2024). Aligning cyber space with physical world: A comprehensive survey on embodied AI. arXiv preprint arXiv:2407.06886.
10 Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). OpenAI gym. arXiv preprint arXiv:1606.01540.
11 James, S., Freese, M., & Davison, A. J. (2019). Pyrep: Bringing v-rep to deep robot learning. arXiv preprint arXiv:1906.11176.
12 Rudin, N., Hoeller, D., Reist, P., & Hutter, M. (2022, January). Learning to walk in minutes using massively parallel deep reinforcement learning. In Conference on robot learning (pp. 91-100). PMLR.
13 Gu, X., Wang, Y. J., & Chen, J. (2024). Humanoid-gym: Reinforcement learning for humanoid robot with zero-shot sim2real transfer. arXiv preprint arXiv:2404.05695.
14 Mittal, M., Yu, C., Yu, Q., Liu, J., Rudin, N., Hoeller, D., ... & Garg, A. (2023). Orbit: A unified simulation framework for interactive robot learning environments. IEEE Robotics and Automation Letters, 8(6), 3740-3747.
15 Zakka, K., Tabanpour, B., Liao, Q., Haiderbhai, M., Holt, S., Luo, J. Y., ... & Abbeel, P. (2025). Mujoco playground. arXiv preprint arXiv:2502.08844.
16 Todorov, E., Erez, T., & Tassa, Y. (2012, October). Mujoco: A physics engine for model-based control. In 2012 IEEE/RSJ international conference on intelligent robots and systems (pp. 5026-5033). IEEE.
17 Zhou, X., Qiao, Y., Xu, Z., et al., Genesis: A Generative and Universal Physics Engine for Robotics and Beyond, https://genesis-embodied-ai.github.io/.
18 Kolve, E., Mottaghi, R., Han, W., VanderBilt, E., Weihs, L., Herrasti, A., ... & Farhadi, A. (2017). Ai2-thor: An interactive 3d environment for visual ai. arXiv preprint arXiv:1712.05474.
19 Juliani, A., Berges, V. P., Teng, E., Cohen, A., Harper, J., Elion, C., ... & Lange, D. (2018). Unity: A general platform for intelligent agents. arXiv preprint arXiv:1809.02627.
20 Ye, L., Li, R., Hu, X., Li, J., Xing, B., Peng, Y., & Liang, B. (2025). Unity RL Playground: A Versatile Reinforcement Learning Framework for Mobile Robots. arXiv preprint arXiv:2503.05146.
21 Kolvenbach, H., Hampp, E., Barton, P., Zenkl, R., & Hutter, M. (2019, November). Towards jumping locomotion for quadruped robots on the moon. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 5459-5466). IEEE.
22 Kolvenbach, H., Bellicoso, D., Jenelten, F., Wellhausen, L., & Hutter, M. (2018, June). Efficient gait selection for quadrupedal robots on the moon and mars. In 14th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2018). ESA Conference Bureau.
23 Ye, L., Li, J., Cheng, Y., Wang, X., Liang, B., & Peng, Y. (2023). From knowing to doing: learning diverse motor skills through instruction learning. arXiv preprint arXiv:2309.09167.
24 Mahmood, N., Ghorbani, N., Troje, N. F., Pons-Moll, G., & Black, M. J. (2019). AMASS: Archive of motion capture as surface shapes. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 5442-5451).
25 https://huggingface.co/datasets/lvhaidong/LAFAN1_Retargeting_Dataset