

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

Linqi YE^{1*}, Boyang XING², Bin LIANG³, Lei JIANG², and Yan PENG¹

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.

Simulation platforms are pivotal to robotics research and have greatly advanced the development of the robotics field. Traditional robotic simulation tools such as Webots [1], Gazebo [2], and V-REP (now CoppeliaSim) [3] have notably contributed to the field by supporting kinematic and dynamic model-based research. However, as artificial intelligence and robotics converge ever more closely, robotics research has entered the era of embodied intelligence, rendering traditional robotic simulation platforms inadequate for current research needs. The advent of learning-based methods, particularly reinforcement learning (RL), has raised new demands for simulation platforms, which now require more than just physical simulation capabilities—they also need seamless integration of RL algorithms, efficient interactive training, and reliable sim-to-real transfer functionality. To this end, a new generation of robotic simulation platforms has been developed, with Isaac Lab [4], MuJoCo Playground [5] and Genesis as the classic representatives. These platforms are all well-suited for embodied intelligence research, while each features its own unique emphasis. Isaac Lab prioritizes GPU-accelerated large-scale multimodal robot learning, MuJoCo Playground focuses on efficient physical simulation and fast sim-to-real transfer, and Genesis features a universal physics engine alongside high-fidelity dynamic environment generation driven by generative data.

Unity, primarily known as a game engine, has also emerged as a powerful tool for embodied intelligence research. For instance, AI2-Thor [6] is a well-known Unity-based framework that features realistic interactive household scenes, specifically tailored to train agents on navigation and object interaction tasks. The Unity ML-Agents Toolkit [7], by contrast, is an open-source Unity-based framework designed for building custom simulation environments and training intelligent agents through RL. Beyond these dedicated frameworks, Unity itself offers core capabilities ideal for embodied intelligence research: near-photorealistic rendering, multi-modal perceptual inputs, advanced physics simulation, and cross-platform compatibility (Windows, Linux, macOS). It also enables the design of hierarchical tasks and multi-agent collaboration via its intuitive C# scripting system. This stands in clear contrast to other state-of-the-art simulation platforms: MuJoCo Playground prioritizes lightweight physics simulation but lacks robust support for multi-modal perception and multi-agent systems, while Isaac Lab excels in GPU-accelerated robotic motion control yet provides limited flexibility in sensory simulation and dynamic environment design. Collectively, these strengths render Unity exceptionally well-suited for embodied intelligence research, particularly for investigations into real-world-aligned unstructured scenarios.

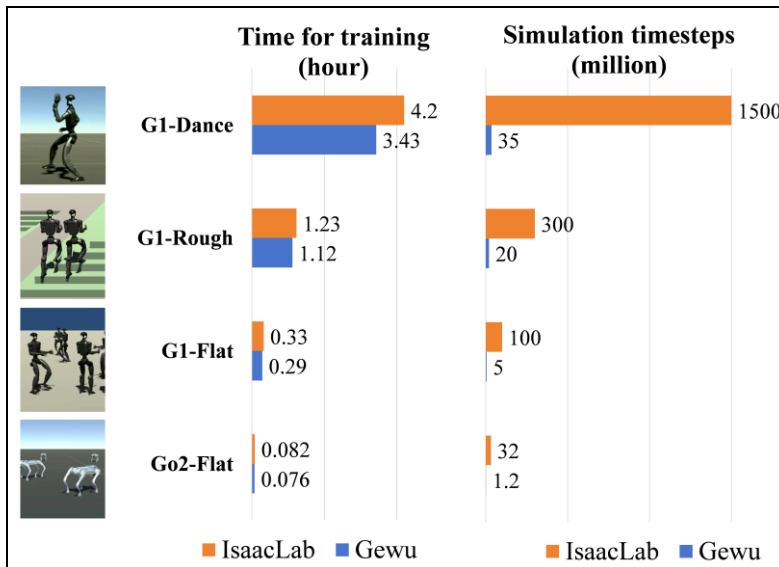
*Corresponding author (email: yelinqi@shu.edu.cn)

Built on Unity/Tuanjie, 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. Extending the capabilities of Unity RL Playground [8], Gewu Playground provides enhanced features for locomotion, manipulation, and navigation tasks, facilitating rapid prototyping, accommodating diverse robot types, and enabling seamless sim2real transfer via ROS2 integration.

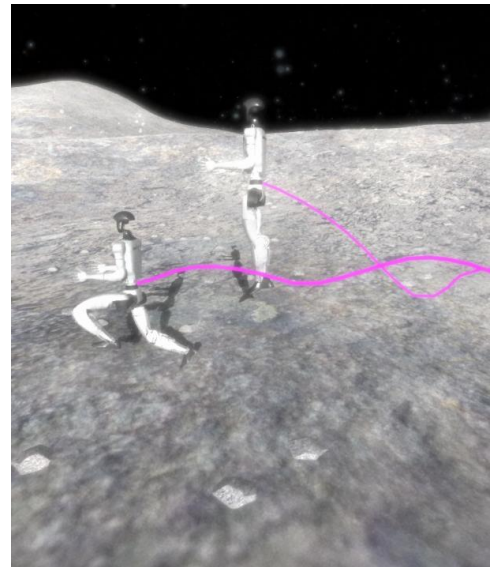
Core to Gewu Playground is a framework with three key innovations for embodied intelligence: an efficient instruction learning-based RL framework; low-cost hardware requirements that broadens researchers' accessibility to efficient CPU-based training; and deep integration with Unity's ecosystem. The latter innovation enables Gewu Playground to leverage the high-quality simulation capabilities of Unity, including photorealistic multi-modal rendering, flexible physics (rigid/soft-body,



(a) Gewu Playground Main Function Modules (Gewu 2.0)



(b) Performance Benchmark: Gewu Playground vs. IsaacLab



(c) Lunar Locomotion Simulation on Gewu

Figure 1 Overview and Performance of Gewu Playground.

fluid dynamics), and modular scripting.

Gewu Playground (Gewu 2.0) 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. Further details can be found in the Supplementary File PDF and the video attachment (<https://linqi-ye.github.io/video/Gewu/>).

Leveraging Unity's modular ecosystem, the platform seamlessly converts Unified Robot Description Format (URDF) models into Unity's ArticulationBody components using the URDF Importer package, enabling high-fidelity physics simulations. RL in Gewu Playground employs the ML-Agents toolkit, which supports proximal policy optimization (PPO), soft actor-critic (SAC), and multi-agent policy optimization with credit assignment (MA-POCA) algorithms, ensuring robust policy optimization across various robotic tasks.

Intended as a general-purpose, universally accessible embodied intelligence platform, Gewu Playground uniquely achieves high learning efficiency via instruction learning [9]. Whereas Isaac Lab requires high-end GPUs, Gewu can be efficiently trained on CPUs and achieves comparable performance with far fewer timesteps than Isaac Lab (Fig. 1b, Intel i9-14900HX + NVIDIA RTX 4080 laptop). That is, Gewu offers a high "learning efficiency per step." Nevertheless, recognizing the value of GPUs in large-scale tasks, we are collaborating with Unity China to upgrade the physics engine in Gewu, enabling GPU-accelerated parallel simulations that will boost the step throughput by 10–20 times while still supporting CPU-only use.

The high-quality simulation features and terrain construction tools of Gewu Playground provide an ideal environment for training extraterrestrial-exploration robots deployed in lunar or Martian missions. We have demonstrated the capabilities of Gewu Playground in locomotion strategies for humanoid robots under lunar low-gravity conditions (Fig. 1c). The reduced gravity in the lunar environment (1/6 that of Earth's) considerably challenges the movement and balance of the robot. Training under simulated lunar conditions is essential for the development of adaptive locomotion strategies by the robot.

Within Gewu Playground, we trained the Unitree G1 robot on 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 enables efficient learning and the development of diverse behaviors.

The lunar surface terrain was constructed using Unity Terrain Editor, which precisely replicates the cratered landscape and undulating terrain of the Moon. The robot

was trained over 10 million steps on each task in less than two hours, achieving stable and efficient locomotion policies. Performance evaluation consistently revealed larger travel distances and higher stability under the running policy than under the jumping policy. These findings provide valuable insights for future robotic lunar exploration missions, demonstrating that Gewu Playground enables the training of humanoid robots for adaptive movement strategies in extraterrestrial environments.

As a unified, user-friendly framework for embodied intelligence research, Gewu Playground represents a noteworthy advancement in robot simulation platforms. Endowed with a universally efficient RL framework, ease of use with low hardware requirements, and seamless integration with Unity's rich ecosystem, Gewu stands as a vital and competitive complement to other embodied intelligence simulation platforms. Further, it supports a wide range of locomotion, manipulation, and navigation tasks, making it well-suited for both traditional and learning-based robotic research.

Most importantly, Gewu Playground lowers the technical barriers of robotic research, broadening the range of researchers that can contribute to next-generation intelligent robotic systems. Future plans include enabling GPU-accelerated training, adapting to more physical robot models, and expanding the embodied intelligence applications. With its versatile learning architecture, Gewu Playground is poised to advance the education and research of embodied intelligence, emerging as a foundational infrastructure for the era of embodied intelligence.

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. Cyberbotics Ltd. Webots™: professional mobile robot simulation. *Int J Adv Robotic Syst*, 2004, 1: 39–42
- 2 Koenig N, Howard A. Design and use paradigms for gazebo, an open-source multirobot simulator. In: *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2004. 2149–2154
- 3 Rohmer E, Singh S P, Freese M. V-REP: A versatile and scalable robot simulation framework. In: *Proceedings of 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2013. 1321–1326
- 4 Mittal M, Yu C, Yu Q, et al. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics Autom Lett*, 2023, 8: 3740–3747
- 5 Zakka K, Tabanpour B, Liao Q, et al. Mujoco playground. *arXiv preprint arXiv:2502.08844*, 2025
- 6 Kolve E, Mottaghi R, Han W, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017
- 7 Juliani A, Berges V P, Teng E, et al. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*, 2018
- 8 Ye L, Li R, Hu X, et al. Unity RL playground: A versatile reinforcement learning framework for mobile robots. In: *Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2025
- 9 Ye L, Li J, Cheng Y, et al. From knowing to doing: learning diverse motor skills through instruction learning. *Biomimetic Intell Robot*, 2026

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

Linqi YE^{1*}, Boyang XING², Bin LIANG³, Lei JIANG², and Yan PENG¹

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 physics simulations to efficiently train locomotion policies

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

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 S1 Comparative Summary: Unity vs. MuJoCo/IsaacSim

Feature	Unity	MuJoCo	IsaacSim
Sensory Simulation	Multi-modal, photorealistic rendering	Low-dimension state inputs	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 S1. 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.

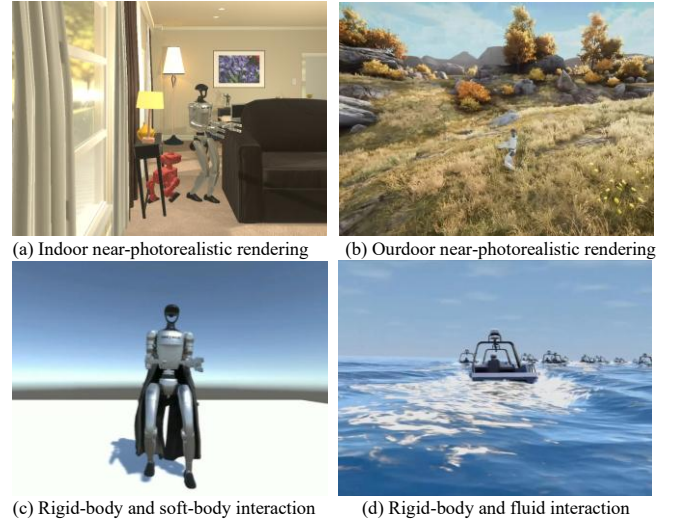


Figure S1 Unity high-quality simulation capabilities.

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 S1), 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 S2. 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.

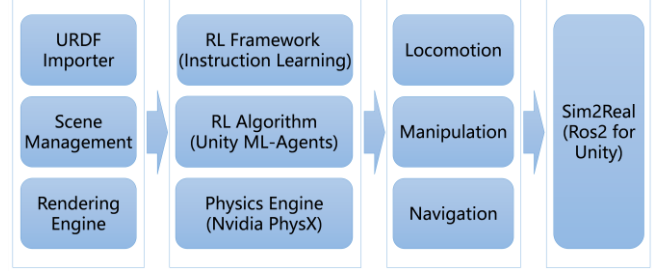


Figure S2 Gewu Playground framework.

Gewu Playground (Gewu 2.0) 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 S3, we have tested over 80 different robots using Gewu Playground.



Figure S3 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 S4, 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 multiple (usually around several dozen) parallel robot instances during training, achieving simulation acceleration while maintaining physics fidelity through Unity’s PhysX integration. It should be noted that GPU-based parallel training is currently not well 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.

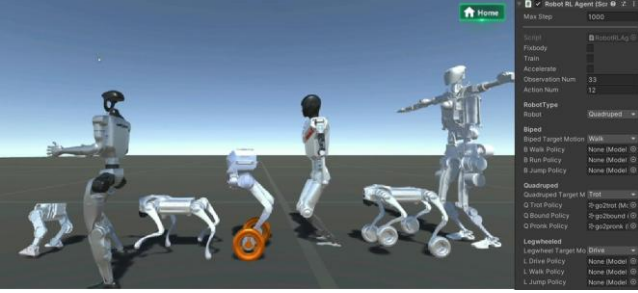


Figure S4 Universal locomotion diagram.

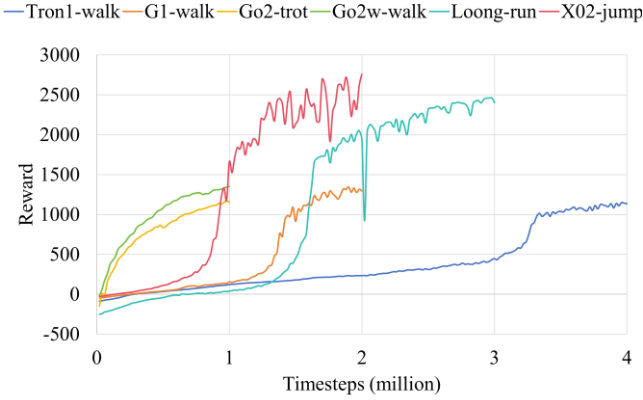


Figure S5 Reward curves for universal locomotion task.

Table S2 Training Details for Universal Locomotion Task

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

The reward curve is depicted in Figure S5, and the training details are listed in Table S2. It can be observed that all six locomotion tasks can be trained within a very short time (1 to 4 million timesteps, spanning 0.1 to 0.3 hours).

Remark: All results in this paper were trained on a laptop with Intel i9-14900HX + NVIDIA RTX 4080.

(2) Rough Terrain Locomotion

To train complex terrain locomotion capabilities, we developed a pyramid staircase environment incorporating both upward and downward stairs (Figure S6). The neural network architecture remained identical to prior implementations, with the key modification 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.

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 S7, and the training details are listed in Table S3.

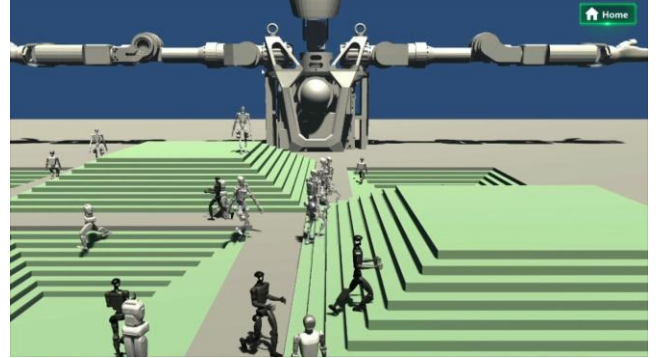


Figure S6 Rough terrain locomotion diagram.

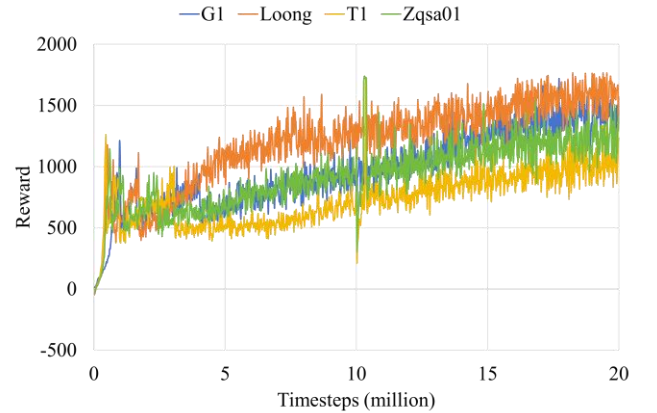


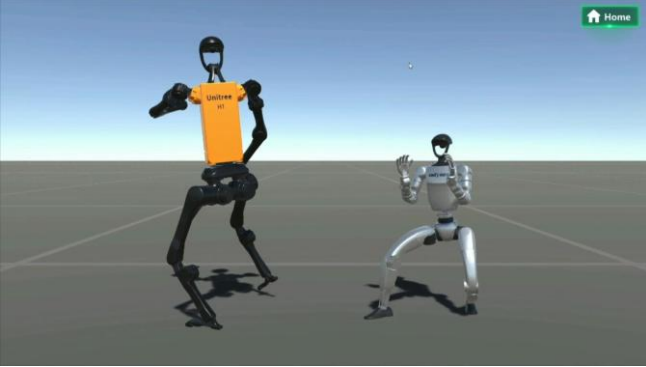
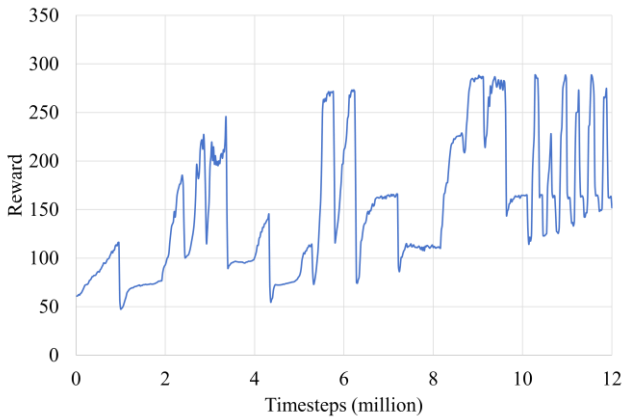
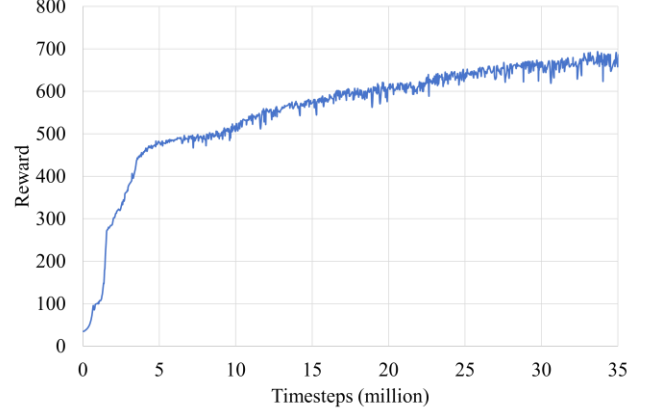
Figure S7 Reward curves for rough terrain locomotion task.

Table S3 Training Details for Rough Terrain Locomotion Task

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 (Figure S8) 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 and train, select targeted motion id via the inspector window, and use the Replay option to visualize motion retargeting animations.

**Figure S8** Motion imitation diagram.**Figure S9** Reward curve for H1 imitation task.**Figure S10** Reward curve for G1 imitation task.**Table S4** Training Details for Imitation Learning

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

The reward curves are depicted in Figure S9 and Figure S10, and the training details are listed in Table S4. 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 movement; 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 training.

(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 Unity engine with the ROS2 ecosystem. To enable real-time robot communication with Unity, 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 simulation and real (see Figure S11). 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 S11 Sim2Real diagram.

(5) Robot Soccer

In the robot soccer module (Figure S12), we implemented a dual-robot 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 initial positions. This configuration produces rich adversarial interactions between the robots. Additionally, the high-level strategy can also be trained via reinforcement learning using MA-POCA algorithm.



Figure S12 Robot soccer diagram.

(6) Mobile Manipulation

The mobile manipulation module is designed for

humanoid robot operation tasks. Currently, it provides a foundational keyboard/VR-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 or Pico VR devices (Figure S13).

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 sent to the corresponding joints of the controlled robot, enabling precise end-effector translation and rotation. This approach successfully validated cube grasping and flower arrangement tasks, demonstrating its effectiveness.

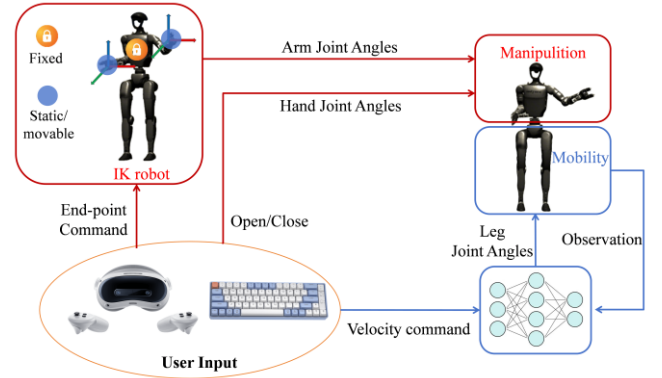


Figure S13 Mobile manipulation diagram.

(7) Autonomous Navigation

The autonomous navigation module (Figure S14) leverages Unity’s AI Navigation package—a 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 S14 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.

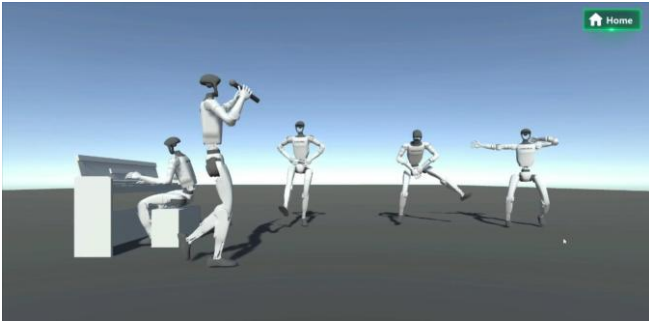


Figure S15 Robot animation diagram.

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 (Figure S15).

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 broadly accessible embodied intelligence platform, Gewu enables efficient training without relying on a GPU, which is attributed to the Instruction Learning framework we adopted.

To quantitatively evaluate the learning efficiency of Gewu, we conducted a comparative study with Isaac Lab across four reinforcement learning tasks: flat-ground locomotion with the Go2 quadruped robot (Go2-Flat), and three tasks with the G1 humanoid robot—flat-ground locomotion (G1-Flat), stair locomotion (G1-Rough), and imitation learning (G1-Dance). All training was conducted on the same laptop (Intel i9-14900HX CPU, NVIDIA RTX 4080 GPU; note: Gewu primarily ran on the CPU, while Isaac Lab utilized GPU acceleration). Corresponding simulation results are detailed below and can additionally be viewed 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 S16 and Figure S17, respectively.

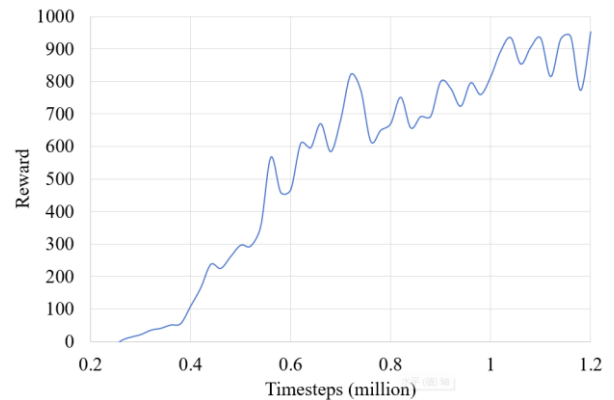


Figure S16 Reward curve for Go2-Flat-Gewu task.

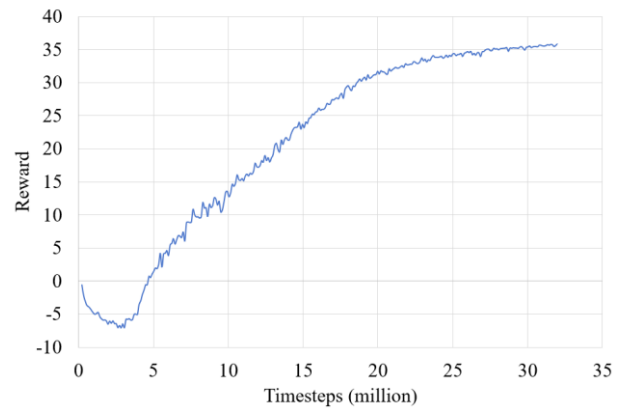


Figure S17 Reward curve for Go2-Flat-IsaacLab task.

(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 S18 and Figure S19, respectively.

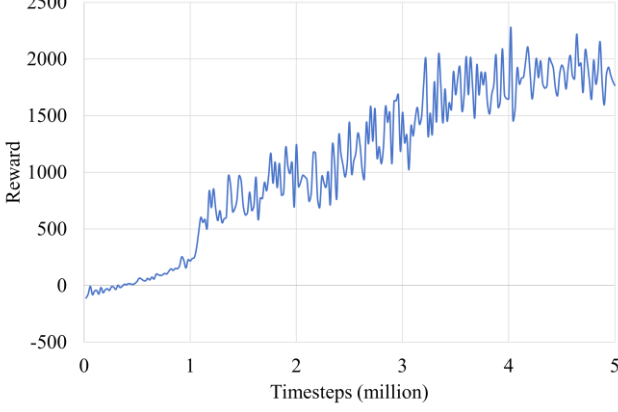


Figure S18 Reward curve for G1-Flat-Gewu task.

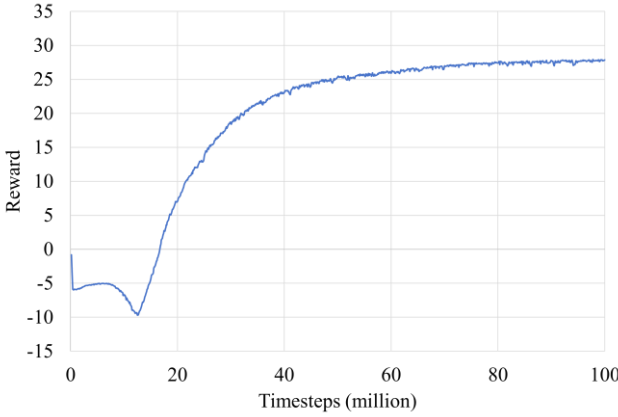


Figure S19 Reward curve for G1-Flat-IsaacLab task.

(3) G1-Rough Task

The G1-Rough task trains the robot to walk on a pyramidal staircase (15 cm high, 30 cm wide per step); training proceeds until the robot reaches a nearly 100% success rate in climbing up and down the stairs. The reward curves for Gewu and IsaacLab are depicted in Figure S20 and Figure S21, respectively. (Note: The drop in intermediate rewards for the G1-Rough-Gewu task is caused by a sudden increase in stair height during curriculum training.)

(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. In Gewu, instruction learning is

employed, with dance mocap data directly used as feedforward action; while in Isaac Lab, the native AMP algorithm is adopted, where dance mocap data is applied to construct an adversarial reward. The reward curves for Gewu and IsaacLab are depicted in Figure S22 and Figure S23, respectively.

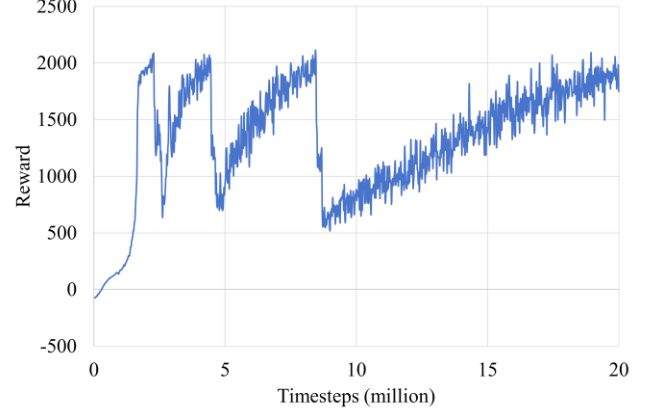


Figure S20 Reward curve for G1-Rough-Gewu task.

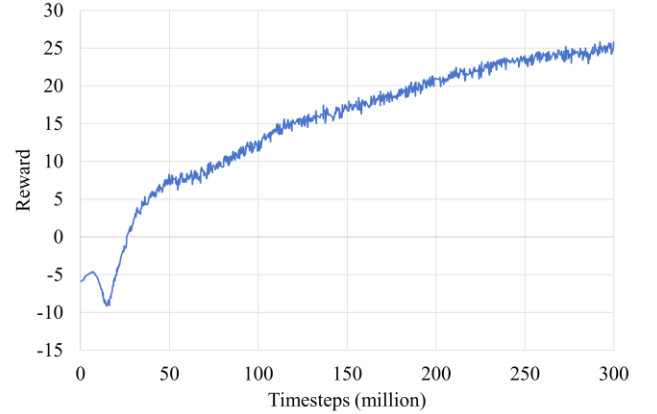


Figure S21 Reward curve for G1-Rough-IsaacLab task.

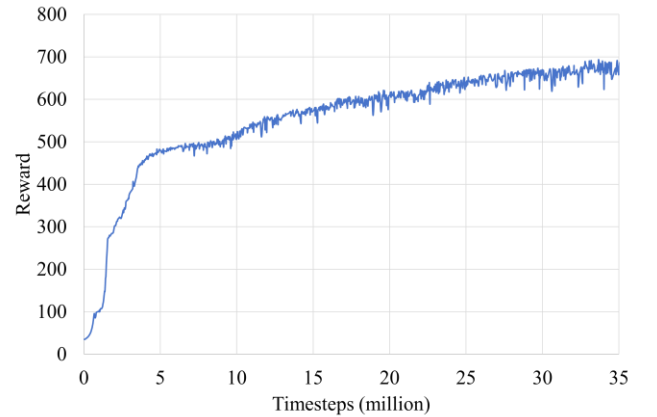


Figure S22 Reward curve for G1-Dance-Gewu task.

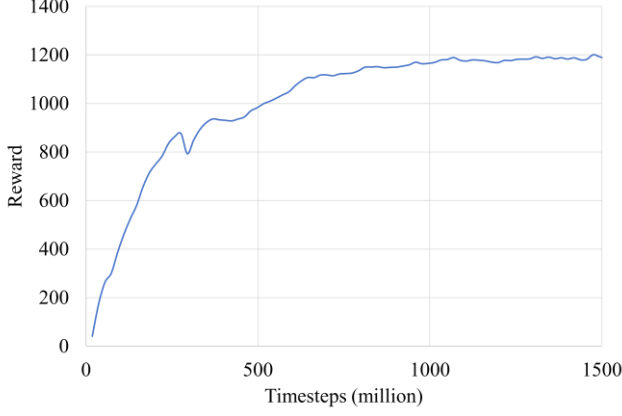


Figure S23 Reward curve for G1-Dance-IsaacLab task.

Table S5 summarizes the quantitative comparison between Gewu and IsaacLab across the four reinforcement learning tasks. It can be observed that they take nearly the same training time, with Gewu being slightly shorter. Although Gewu has a lower timesteps per hour compared to IsaacLab, it achieves comparable performance (see attached video) with significantly fewer simulation timesteps. For instance, in the Go2-Flat task, Gewu only needs 1.2 million simulation timesteps while IsaacLab requires 32 million; 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”, a far more meaningful metric in scenarios with limited computing power.

Table S5 Training Performance Comparison Between Gewu and Isaac Lab

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

To further accelerate training in Gewu, we investigated the impact of increasing the number of parallel training robots on the training process using the G1-Flat-Gewu task. 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).

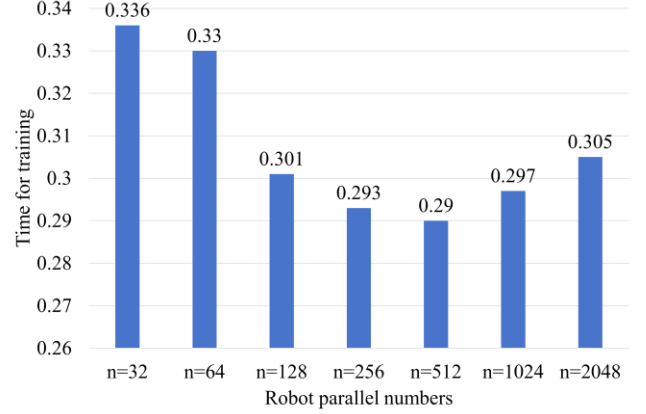


Figure S24 Gewu training time for different robot parallel numbers.

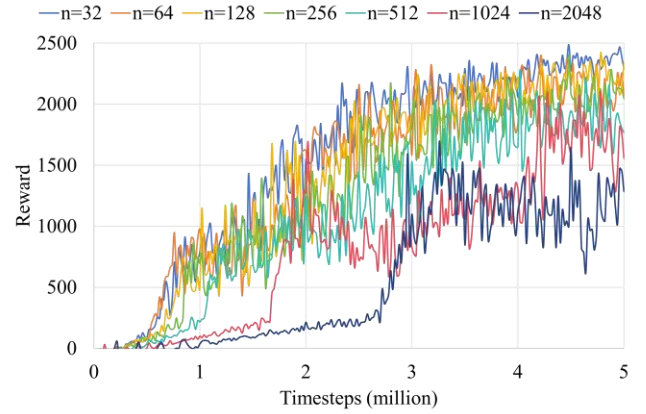


Figure S25 Gewu reward curves for different robot parallel numbers.

Training time is presented in Figure S24 and reward curves in Figure S25. As shown in Figure S24, training speed does not increase monotonically with the rise in parallel robot number n : speed peaks at $n=256$, yet this only represents a 13.7% improvement over $n=32$. Furthermore, increasing n leads to a degradation in reward curve performance—as shown in Figure S25, reward rises significantly more slowly and converges to a much lower value at $n=2048$. Thus, we recommend using a parallel robot number of less than 512 for training, such as $n=32$ or 64.

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 physics engine, aiming to enable GPU-accelerated parallel environment simulation for Gewu. This upgrade will enable Gewu to leverage GPU resources (when available) to support unified simulation and training, which is projected to boost training speed by 10–20 times while maintaining CPU-only compatibility for broad 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 Gewu Playground. 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 S6 lists the details of domain randomization and observation noise. The training reward curves are shown in Figure S26 and Figure S27, and the quantitative indicators of the training process are presented in Table S7.

As can be seen from Figure S26 and Figure S27, 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 S7, 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 S7 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 Gewu Playground.

Table S6 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

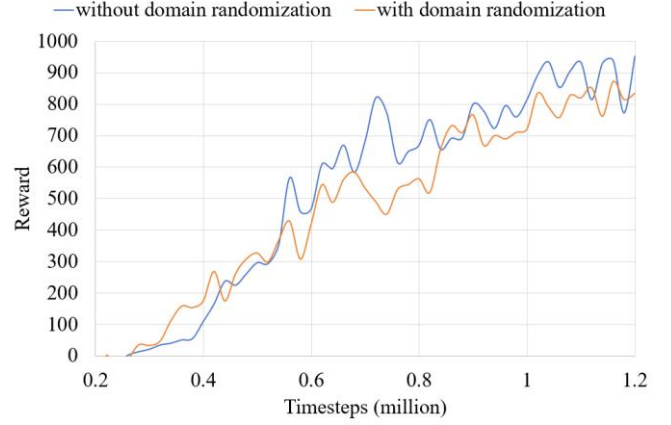


Figure S26 Go2-Flat reward curves: w/ vs. w/o domain randomization.

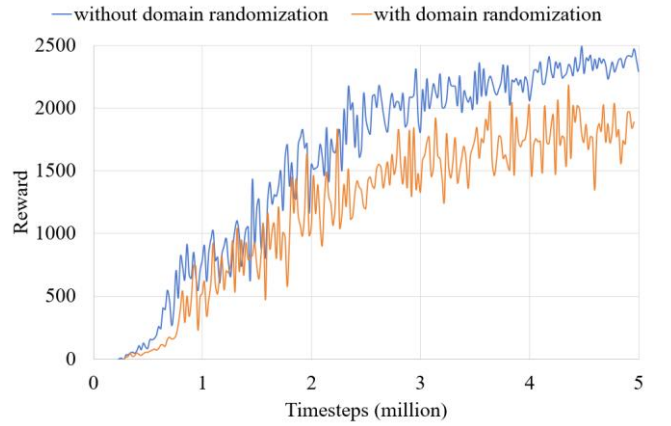


Figure S27 G1-Flat reward curves: w/ vs. w/o domain randomization.

Table S7 Performance Comparison: w/ vs. w/o Domain Randomization

Task	Simulation timesteps (million)	Time for training (min)	Failure case
Go2-Flat (w/o domain randomization)	1.2	4.5	6
Go2-Flat (w/ domain randomization)	1.2	5.7	0
G1-Flat (w/o domain randomization)	5	20	91
G1-Flat (w/ 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 in training is essential for robots to develop adaptive lunar surface locomotion strategies. To address this, we investigated lunar locomotion of the G1 robot in Gewu Playground, yielding preliminary insights into effective locomotion strategies.

(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, defining it as a hybrid neural network, as illustrated in Figure S28. 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. This module acts as a time-exclusive preset function, feeding open-loop actions directly into the robot, which remain frozen and unupdated 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. Each component of the reward is defined as follows:

- 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.

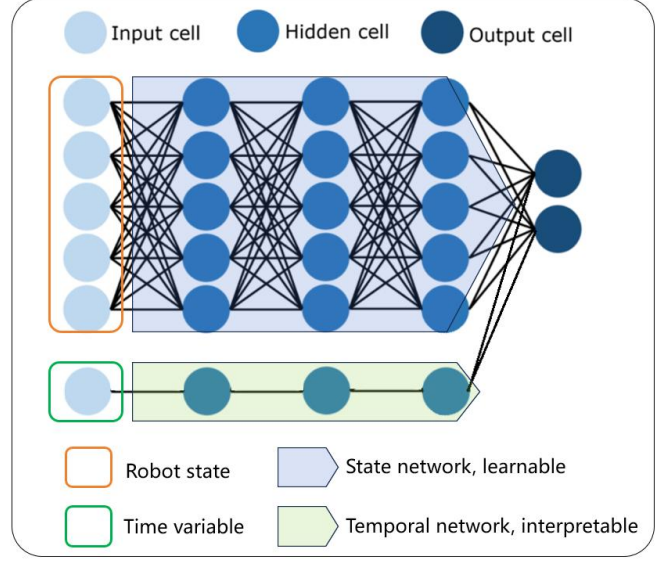


Figure S28 The concept of hybrid neural network.

(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 initial position for the robot. For training, 24 robot replicas were generated, each assigned a randomly sampled yaw angle (movement direction) as illustrated in Figure S29—this setup ensures the robots encounter a diverse range of terrain conditions. Adopting a 0.01-second simulation time step, the robots were reset to their initial position upon falling or at the expiration of every 10-second interval. We conducted 10-million-step training for both running and jumping tasks, each task corresponded to a simulated duration of 27.78 hours, with the actual training time taking only 1.67 hours. Training adopted the PPO algorithm, with the same hyperparameters applied for the two tasks. Reward curves for the two tasks are presented in Figure S30, where the running task exhibits a faster reward rise and a higher final reward value. Running appears to be a more adaptive and efficient locomotion mode for robots operating in lunar low-gravity conditions than jumping.



Figure S29 Locomotion training on simulated lunar surface.

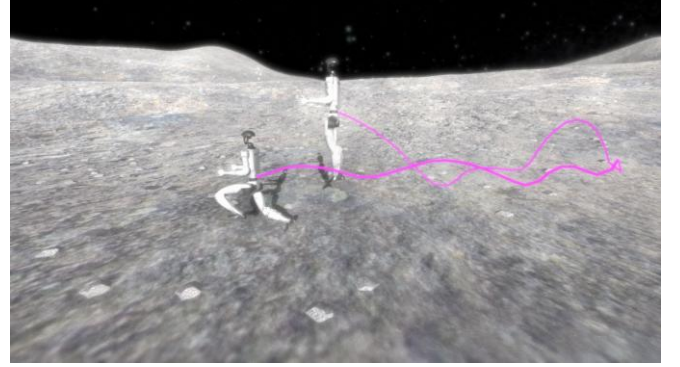


Figure S31 Lunar locomotion trajectory visualization.

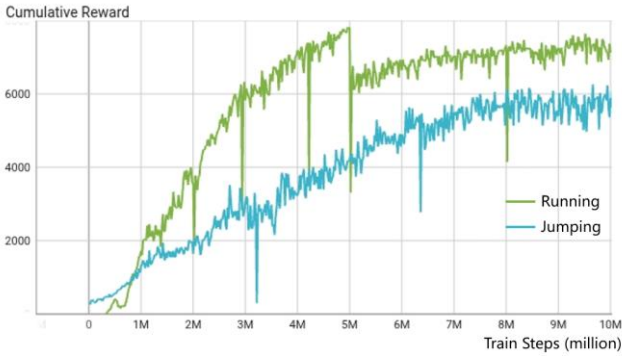


Figure S30 The reward curves for lunar locomotion.

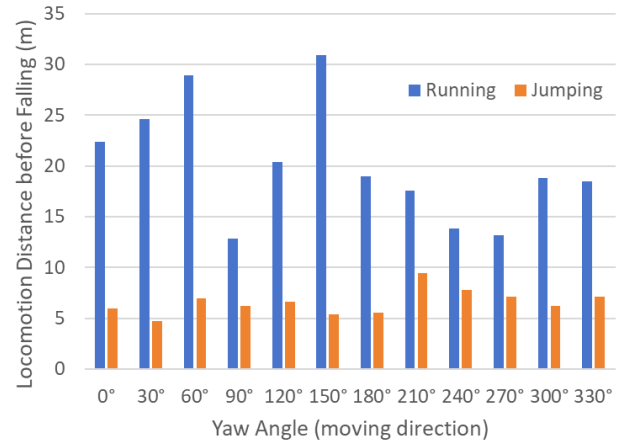


Figure S32 Lunar locomotion performance comparison.

(3) Performance Evaluation

To evaluate the performance of the derived locomotion policy, we had two robots perform running and jumping respectively, and recorded their body trajectories as shown in Figure S31. It can be observed that the robot can jump to a far greater height on the lunar surface than on Earth, and its running gait also differs from that on our planet—appearing light and floating, and seemingly more strenuous and inefficient (this is more evident in the supplementary video).

For quantitative assessment of locomotion performance, we commanded the robot to move continuously in a straight direction until it fell down, 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 S32. As shown, the running policy consistently outperformed the jumping policy in terms of travel distance across all directions, indicating that running may be a more efficient and stable mode of locomotion on the lunar surface. This is especially true in steep terrain, where jumping risks losing balance on landing, whereas running allows easier balance adjustments via more frequent ground contact. While this finding appears to contradict the lunar exploration experiences of astronauts, it nonetheless offers valuable insights for robotic lunar locomotion research.

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 a universally efficient RL framework, ease of use with low hardware requirements, and seamless integration with Unity’s rich ecosystem, enabling rapid prototyping and experimentation across diverse robot morphologies. A lunar locomotion case study demonstrates its efficacy in training adaptive low-gravity locomotion strategies, 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. Looking ahead, plans include enabling GPU-accelerated training, adapting to more physical robot models, and expanding the embodied intelligence applications. 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. Cyberbotics Ltd. Webots™: professional mobile robot simulation. *Int J Adv Robotic Syst*, 2004, 1(1): 39-42
- 2 Koenig N, Howard A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In: *Proceedings of IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2004, 3: 2149-2154
- 3 Rohmer E, Singh S P, Freese M. V-REP: A versatile and scalable robot simulation framework. In: *Proceedings of IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2013: 1321-1326
- 4 Collins J, Chand S, Vanderkop A, et al. A review of physics simulators for robotic applications. *IEEE Access*, 2021, 9: 51416-51431
- 5 Pitonakova L, Giuliani M, Pipe A, et al. Feature and performance comparison of the V-REP, Gazebo and ARGoS robot simulators. In: *Annual Conference Towards Autonomous Robotic Systems*. Cham: Springer Int Publ, 2018: 357-368
- 6 Miki T, Lee J, Hwangbo J, et al. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Sci Robotics*, 2022, 7(62): eabk2822
- 7 Vollenweider E, Bjelonic M, Klemm V, et al. Advanced skills through multiple adversarial motion priors in reinforcement learning. *arXiv preprint arXiv:2203.14912*, 2022
- 8 Hoeller D, Rudin N, Sako D, et al. Anymal parkour: Learning agile navigation for quadrupedal robots. *Sci Robotics*, 2024, 9(88): eadi7566
- 9 Liu Y, Chen W, Bai Y, et al. Aligning cyber space with physical world: A comprehensive survey on embodied AI. *arXiv preprint arXiv:2407.06886*, 2024
- 10 Brockman G, Cheung V, Pettersson L, et al. OpenAI gym. *arXiv preprint arXiv:1606.01540*, 2016
- 11 James S, Freese M, Davison A J. Pyrep: Bringing v-rep to deep robot learning. *arXiv preprint arXiv:1906.11176*, 2019
- 12 Rudin N, Hoeller D, Reist P, et al. Learning to walk in minutes using massively parallel deep reinforcement learning. In: *Conference on robot learning*. PMLR, 2022: 91-100
- 13 Gu X, Wang Y J, Chen J. Humanoid-gym: Reinforcement learning for humanoid robot with zero-shot sim2real transfer. *arXiv preprint arXiv:2404.05695*, 2024
- 14 Mittal M, Yu C, Yu Q, et al. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics Autom Lett*, 2023, 8(6): 3740-3747
- 15 Zakka K, Tabanpour B, Liao Q, et al. Mujoco playground. *arXiv preprint arXiv:2502.08844*, 2025
- 16 Todorov E, Erez T, Tassa Y. Mujoco: A physics engine for model-based control. In: *Proceedings of IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012: 5026-5033
- 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/>, [Accessed: 2026]
- 18 Kolve E, Mottaghi R, Han W, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017
- 19 Juliani A, Berges V P, Teng E, et al. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*, 2018
- 20 Ye L, Li R, Hu X, et al. Unity RL Playground: A Versatile Reinforcement Learning Framework for Mobile Robots. In: *Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2025
- 21 Kolvenbach H, Hampp E, Barton P, et al. Towards jumping locomotion for quadruped robots on the moon. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019: 5459-5466
- 22 Kolvenbach H, Bellicoso D, Jenelten F, et al. 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)*. ESA Conf Bureau, 2018
- 23 Ye L, Li J, Cheng Y, et al. From knowing to doing: learning diverse motor skills through instruction learning. *Biomimetic Intell Robot*, 2026
- 24 Mahmood N, Ghorbani N, Troje N F, et al. AMASS: Archive of motion capture as surface shapes. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019: 5442-5451
- 25 Lv H D. LAFAN1 Retargeting Dataset. https://huggingface.co/datasets/lvhaidong/LAFAN1_Retargeting_Dataset, 2025