biomimetics

MDPI

Article

A Multi-Agent Reinforcement Learning Method for Omnidirectional Walking of Bipedal Robots

Haiming Mou [1,2], Jie Xue [1,2], Jian Liu [2], Zhen Feng [1,2], Qingdu Li [1,2,*] and Jianwei Zhang [3]

[1] School of Optoelectronic Information and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China; 201440059@st.usst.edu.cn (H.M.); 221240078@st.usst.edu.cn (J.X.); 213330674@st.usst.edu.cn (Z.F.)
[2] Institute of Machine Intelligence, University of Shanghai for Science and Technology, Shanghai 200093, China; liujianwgx@163.com
[3] Department of Informatics, University of Hamburg, 20146 Hamburg, Germany; jianwei.zhang@uni-hamburg.de
* Correspondence: liqd@usst.edu.cn

Abstract: Achieving omnidirectional walking for bipedal robots is considered one of the most challenging tasks in robotics technology. Reinforcement learning (RL) methods have proved effective in bipedal walking tasks. However, most existing methods use state machines to switch between multiple policies and achieve omnidirectional gait, which results in shaking during the policy switching process for bipedal robots. To achieve a seamless transition between omnidirectional gait and transient motion for full-size bipedal robots, we propose a novel multi-agent RL method. Firstly, a multi-agent RL algorithm based on the actor–critic framework is designed, and policy entropy is introduced to improve exploration efficiency. By learning agents with parallel initial state distributions, we minimize reliance on gait planner effectiveness in the Robot Operating System (ROS). Additionally, we design a novel heterogeneous policy experience replay mechanism based on Euclidean distance. Secondly, considering the periodicity of bipedal robot walking, we develop a new periodic gait function. Including periodic objectives in the policy can accelerate the convergence speed of training periodic gait functions. Finally, to enhance the robustness of the policy, we construct a novel curriculum learning method by discretizing Gaussian distribution and incorporate it into the robot's training task. Our method is validated in a simulation environment, and the results show that our method can achieve multiple gaits through a policy network and achieve smooth transitions between different gaits.

Keywords: omnidirectional walking; bipedal robot; multi-agent reinforcement learning; experience replay mechanism; curriculum learning

Citation: Mou, H.; Xue, J.; Liu, J.; Feng, Z.; Li, Q.; Zhang, J. A Multi-Agent Reinforcement Learning Method for Omnidirectional Walking of Bipedal Robots. *Biomimetics* **2023**, *8*, 616. https://doi.org/10.3390/biomimetics8080616

Academic Editors: Xuechao Chen and Gan Ma

Received: 5 September 2023
Revised: 10 November 2023
Accepted: 14 December 2023
Published: 16 December 2023

1. Introduction

As fundamental research in robotics continually advances, service robots are increasingly permeating our daily lives [1]. In specific application scenarios, bipedal robots demonstrate greater flexibility and efficiency than their wheeled counterparts, making them a significant subject within the field of robotics research. Gait optimization is foundational to the normal operation of bipedal robots, chiefly referring to the robots' ability to achieve rapid, stable locomotion through self-balancing [2,3]. To enable balanced motion in bipedal robots, gait optimization needs to circumvent leg–foot collisions during robot movement. Traditional gait control methods, such as human walking parameters [4,5], zero moment point (ZMP) [6,7], passive walking [8], fuzzy logic control [9], and optimization algorithms [10,11], have been consistently employed in bipedal robot gait control for many years. However, these traditional methods present problems. Specifically, human walking parameter methods require time-intensive manual parameter tuning, often resulting in less than optimal values. ZMP methods, on the other hand, present several drawbacks,

such as insufficient energy, restricted walking speed, and limited resistance to external disturbances [6]. These traditional techniques exhibit high computational complexity, low robustness, and poor generality. While Model Predictive Control (MPC) is not classified as a traditional gait control method, its wide application in bipedal robot gait control is noticeable [12]. Reference [13] proposes a holistic MPC scheme based on differential dynamic programming to tackle the challenges of physical constraints and model discrepancies in bipedal robots.

RL methods are widely utilized for bipedal robot control due to their aptitude for model-free learning [14,15]. RL techniques offer benefits such as reduced hardware requirements for robots and significant time savings in the debugging process [16]. However, choosing an appropriate learning rate for RL-based methods poses a practical challenge. A small learning rate may result in protracted training progress, whereas a large one could trigger oscillations or even algorithm divergence, compromising training performance [17]. Integrating RL with neural networks has provided some solutions for this issue [18,19]. The robot interacts with the environment and learns through continuous trial and error to obtain a reward function to judge whether the skill is good or bad, and eventually learns the skill. This learning approach circumvents inaccuracies induced by mathematical models and bolsters the robustness of the training process [20]. The Proximal Policy Optimization (PPO) algorithm, a popular actor–critic algorithm, is frequently used for training bipedal walking [21,22]. It facilitates direct control through end-to-end learning, broadening the applicability of RL. PPO-based model-free learning for bipedal walking gravitates around multi-action policy learning algorithms based on Markov Decision Processes (MDPs) [18,23]. Presently, the PPO algorithm has emerged as the standard method for training bipedal robot gaits. However, it faces challenges concerning slow training speed and low sample utilization in gait optimization. To resolve these issues, an experience replay mechanism has been introduced, which increases past data replay frequency and curbs resource wastage, thereby enhancing the convergence speed and efficiency of RL algorithms [24]. Nonetheless, the experience replay mechanism also has its limitations, including high storage space requirements, substantial computational load, and demanding hardware prerequisites. Consequently, employing a heterogeneous strategy can mitigate the computational overhead associated with RL algorithms. Although leveraging heterogeneous RL algorithms significantly accelerates the training process through multi-threading techniques, a considerable amount of training time is still required [25]. Moreover, existing off-policy RL algorithms do not fully take advantage of good and general experiences since they depend on replaying individual experiences. Model-based RL methods can improve exploration efficiency by leveraging partial knowledge of the environment, thereby minimizing ineffective exploration. The amalgamation of MPC and RL can effectively carry out safe exploration [26]. However, these methods might not entirely utilize the robot's capabilities, potentially resulting in suboptimal policies.

Unfortunately, the majority of current gait research tends to focus predominantly on straight-line gaits or turning gaits. Turning gaits are often achieved by integrating the PPO algorithm with curriculum learning methods. Existing research suggests that training strategies incorporating curriculum learning can present advantages in generating complex actions. A common approach for training bipedal robots for omnidirectional walking is to commence with straight-line movement training, and then progressively introduce larger turning angles. This gradual progression method has been demonstrated to effectively bolster the training success rate of complex strategies. However, curriculum learning methods are prone to catastrophic forgetting, wherein the difficulty of early training stages is lost, posing challenges in achieving multiple gait patterns using a single policy network. Unlike previously mentioned RL gait control methods that require switching between multiple policy networks, our approach allows for omnidirectional gait using a single policy network. This mitigates the constraints of controller switching and facilitates more fluid transitions between different gaits. Specifically, this paper offers the following advancements:

(1) A novel approach to multi-agent RL is constructed which is based on the actor–critic framework. A combinatorial optimization approach is employed to maximize the cumulative reward and policy entropy. Additionally, a new experience replay mechanism is designed specifically for multi-agent RL methods.
(2) A new periodic gait function is designed and incorporated as an objective into our RL policy. Our periodic gait function enables bipedal gaits to exhibit symmetry and periodicity.
(3) A curriculum learning method based on Gaussian distribution discretization is proposed. During the training process, the turning angle is dynamically adjusted, and a single strategy is used to achieve omnidirectional walking with different turning angles.

2. Related Work

Traditional gait optimization typically involves planning joint trajectories based on walking requirements, followed by the calculation of joint angles using an inverse kinematic model. The goal is to ensure stability, avert falls, and enhance aesthetic appeal. To tackle the complexities associated with multi-link structured bipedal robots, researchers have conceived simplified models, such as the three-mass inverted pendulum model. This model takes into account the influence of the support leg on motion stability, reducing the robot to three crucial components: the center of mass (COM) of the torso, the COM of the support leg, and the COM of the swing leg during the single-leg support phase [27]. This streamlined framework aids in studying motion stability and control strategies in bipedal locomotion.

In [28], a contemporary online optimization technique is presented, enabling humanoid robots to execute vertical jumps effectively by controlling the centroidal angular momentum and mitigating the impact upon landing. Nevertheless, manual parameter adjustment is not only time-consuming but also does not guarantee the generation of optimal values. Consequently, numerous scholars have suggested optimization algorithms for robot training. At present, bio-inspired optimization algorithms are extensively applied to bipedal robot gait optimization. However, owing to the abundance of robot parameters, bio-inspired optimization algorithms are prone to becoming ensnared in local optimization during the optimization process, thus making it challenging to achieve the robot's maximum walking speed.

RL methods have demonstrated significant potential and the capacity to supplant traditional model-based controllers in various studies. Remarkably, significant progress has been achieved using RL methods for tasks such as walking and jumping on the Cassie robot [29,30]. In a study of Cassie walking gaits by Jonah Siekmann et al. [31], a clock signal-based reward function was employed to attain periodic walking gaits with swing and stance phases for each leg. The study in reference [32] utilizes domain randomization techniques to facilitate policy learning adaptation to variations in system dynamics, thereby achieving robust behaviors. In [33], a formulation to counteract the limitations of RL controllers for bipedal robots is proposed by integrating footstep constraints. This formulation allows the learned controllers to maintain robust and dynamic gaits while adhering to external environmental constraints. The study in reference [23] leverages model-free RL techniques to fine-tune the base policy, adapting it to an imperfect extrinsics estimator, and demonstrates successful transference to a physical robot. RL has also been used to learn diverse motion skills and achieve seamless transitions between them. Yu et al. utilize precomputed trajectory data and terminal rewards to learn specific turning gaits for bipedal robots, enabling seamless transitions in the turning process [34]. In reference [35], a policy learning method based on footprint planning is designed to accomplish omnidirectional walking on 2D and 3D terrains by following a learning strategy for a given sequence of steps in the ROS framework. Rodriguez et al. propose a novel approach for achieving omnidirectional gaits for bipedal robots [36]. They amalgamate RL with a curriculum learning

approach, progressively increasing the difficulty of gait training tasks. This integration enables the learning of robust and versatile gaits in various directions.

3. Preliminaries

3.1. Actor–Critic Algorithm

Central to the concept of RL is an intelligent agent engaging with an environment while executing a specific task. This interaction involves taking actions and receiving reward signals from the environment, which indicate the success or failure of the agent's actions. As the agent continually interacts with the environment, it refines its action selection strategy based on the received feedback, aiming to maximize cumulative rewards. Through numerous iterations, the agent eventually unearths the optimal strategy to complete the assigned task. During the actual interaction between the agent and the environment, constraining the number of interactions could result in a significant discrepancy between the presumed reward sequence and the actual values, thereby exacerbating the variance of the reward signals. Moreover, traditional policy gradient algorithms often exhibit a protracted convergence speed. One possible resolution to these issues is the utilization of an actor–critic algorithm, merging the roles of an actor and a critic and leveraging joint training to bolster stability and learning efficiency. In the actor–critic algorithm, the actor is tasked with action selection, while the critic evaluates the value of these actions. The actor generates actions a_t based on the current state and policy $\pi(a_t|s_t, \theta)$, where θ represents the parameters of the policy network. The critic estimates the action value function $Q(s_t, a_t, \theta_q)$ or the advantage function $A(s_t, a_t, \theta_a)$ based on the current state and action, where θ_q and θ_a represent the parameters of the critic network. The goal of the actor is to maximize the total return $J(\theta) = \mathbb{E}_\pi[R]$, where R represents the return value. To achieve this goal, the actor utilizes policy gradient methods to update the parameters θ by maximizing the expected value of the policy value function $V(s_t, \theta)$:

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{t=1}^{N} \nabla_\theta \log \pi(a_t|s_t, \theta) A(s_t, a_t, \theta_a), \tag{1}$$

where N denotes the number of samples. By iteratively updating the parameters θ, the actor optimizes the policy to choose actions that yield higher returns.

The critic estimates the action value function or advantage function based on the chosen actions and the true returns. Specifically, the critic aims to minimize the difference between the action value function or advantage function and the true return using the following loss functions:

$$\mathcal{L}(\theta_q) = \frac{1}{2} \mathbb{E}_\pi \Big[\big(Q(s_t, a_t, \theta_q) - R_t \big)^2 \Big], \tag{2}$$

$$\mathcal{L}(\theta_a) = \frac{1}{2} \mathbb{E}_\pi \Big[\big(A(s_t, a_t, \theta_a) - R_t \big)^2 \Big], \tag{3}$$

where R_t represents the true return. Through gradient descent, the parameters θ_q and θ_a of the critic network are updated to improve the accuracy of the estimation for the action value function or advantage function.

By training and updating the actor and critic together, the actor–critic algorithm can learn more stably and find better policies in complex environments. The key feature of this algorithm is the simultaneous training of both the policy and the value function through the estimation of action values.

3.2. Robotics Platform

To conduct bipedal locomotion studies, we construct our bipedal robot in a simulated environment. The robot stands at a height of 133 cm and weighs 30 kg. It comprises a total of 20 degrees of freedom, evenly split with 10 degrees of freedom allocated to the upper body and an equivalent number to the lower body. To focus on the qualitative aspects of the walking gait, the 10 degrees of freedom corresponding to the upper body, arms, and hands

are immobilized, activating only the hip joints and the remaining 10 degrees of freedom related to the legs. The joint parameters are outlined in Table 1.

Table 1. The joint information of the robot.

Joint Name	Range of Motion	Peak Torque	Joint Velocity	Stiffness	Damping
Hip Yaw	−90°–90°	133 Nm	240°/s	0.1	1000
Hip Pitch	−60°–90°	45.9 Nm	576°/s	0.1	1000
Hip Roll	−10°–10°	5000 Nm	120°/s	0.1	1500
Knee Pitch	−140°–0°	48.6 Nm	540°/s	1	1500
Ankle Pitch	−80°–80°	28.8 Nm	918°/s	0.01	100

We only introduce information about the joints used in the lower body.

Figure 1 presents the distribution of the lower body joints and actuators, encompassing hip yaw, hip roll, hip pitch, knee pitch, and ankle pitch. To monitor the robot's body posture, an inertial measurement unit (IMU), the 3DM-GX5-25, is installed 5 cm above the hip yaw degree of freedom. This IMU allows for the measurement of the robot's center of mass roll, pitch, and yaw angles, along with the estimation of the center of mass's horizontal velocity, angular velocity, and angular acceleration.

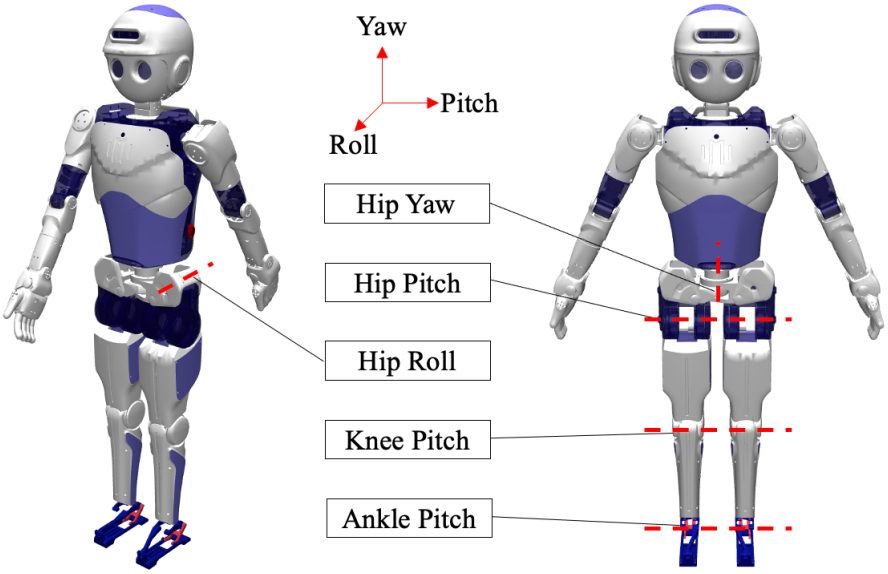


Figure 1. Appearance and structure diagram of humanoid robot. In the simulation, we locked the joints of the upper body and only had 10 joints belonging to the lower body.

4. Method

4.1. Robot Operating System Footstep Planner

Developing an omnidirectional gait for bipedal robots is a highly challenging task due to several inherent complexities. These complexities include the high-dimensional dynamics of the robot, limitations in sensing and actuation capabilities, as well as computational constraints that require real-time processing. As a result, achieving an effective and efficient omnidirectional gait involves overcoming numerous technical hurdles. We propose a method of imitating target foot landing points to solve this challenging problem. Firstly, the paper describes the footstep state information using the global position and orientation of the bipedal robot's supporting feet. Specifically, this information consists of a two-dimensional coordinate point and a heading vector θ provided by a sensor on the hip joint of one side. Next, the footstep planner in the ROS package is used to randomly generate a curve and export a set of 2D trajectory points (x, y, θ) with heading information, so that the bipedal robot's foot landing points imitate those points, thereby achieving omnidirectional gait for the robot. To use the footstep planner to generate curve trajectories,

a 2D grid map and initial and target poses (x, y, θ) are required as inputs. The paper sets the initial pose to the origin and generates 800 target foot landing points by randomly sampling from the target range $(0, -1, -\pi/2)$ to $(0, 1, \pi/2)$ on an empty map.

4.2. Bipedal Periodic Objective Function

Bipedal gait typically exhibits a symmetrical character. During walking, leg movement can be segmented into swing and stance phases, with the stance phase further subcategorized into single-leg stance and double-leg stance periods. Thus, a gait cycle can be divided into two single-leg stance phases and two double-leg stance phases. To implement symmetrical gait in bipedal robot motion, the concept of periodic reward synthesis is employed. During the single-leg stance phase, one foot retains static contact with the ground while the other swings through the air. In the succeeding phase, the feet roles are interchanged, with the previously grounded foot now swinging and the previously swinging foot establishing supportive contact with the ground. As such, employing symmetry as a learning structure can improve learning efficiency. Symmetrical motion facilitates quicker convergence to more effective solutions and results in a more visually appealing bipedal gait. The policy objective needs to master walking ability in a simulated environment while also satisfying the requirement of gait periodic symmetry.

To manifest periodic locomotion for the bipedal robot and endow it with various gait styles, we designed a periodic symmetrical function based on expert experience. This function enables the robot's left and right feet to track the lifted foot height outputted by the symmetrical function in real time. This function serves a dual purpose; it not only accomplishes periodic gait movement but also accelerates the convergence speed of the training process. Specifically, the mathematical expression of this function is shown in Equations (4) and (5).

$$h_t^r = \begin{cases} 0, & 0 < t < T_{std} \quad or \quad T_h < t < T \\ k_s h_{max}^{foot}(\frac{t}{T_s} - k_t)^2 + (h_{foot}^{leg} - h_{max}^{foot}), & T_{sd} \le t \le T_h \end{cases}, \tag{4}$$

$$h_t^l = \begin{cases} 0, & 0 < t < T_{std} + T_h \\ k_s h_{max}^{foot}(\frac{t}{T_s} - k_t)^2 + (h_{foot}^{leg} - h_{max}^{foot}), & T_{sd} + T_h \le t, \end{cases}. \tag{5}$$

where h_{max}^{foot} represents the maximum height of the lifted foot, h_{foot}^{leg} represents the height between the leg and ground at the initial pose, T is the period, T_h is half of the period T, T_{swing} is the period of the swing phase, and T_{std} is the period of the stance phase. Moreover, the expression needs to satisfy $k_s * k_t^2 = 1$. We introduce a periodic symmetric objective g_t to the policy. Finally, by mapping the given state s_t and target g_t to actions, the policy is modeled as $\pi(a_t|s_t, g_t)$.

4.3. Multi-Agent Systems and Policy Entropy

Reference [37] has shown that $p(s_0)$ has a strong effect on the training outcomes of RL. An excellent distribution of $p(s_0)$ can effectively reduce unnecessary exploration during the process of RL. Sampling the initial state from the data can enable the robot to execute turning maneuvers smoothly when an imitation robot simulates the walking data of the ROS footprint planner. Due to the differences in modeling between footstep data and robots, directly sampling $p(s_0)$ from footstep data may not be appropriate for reproducing omnidirectional walking of imitation robots. The dependence on the quality of footstep data is significant for this trained strategy.

To mitigate the influence of data quality in ROS during the training phase, we propose learning the initial state distribution. This approach transforms the problem into a multi-agent RL problem, where all agents cooperate in fully cooperative tasks and share a common reward function. The training framework of our multi-agent RL method is illustrated in Figure 2.

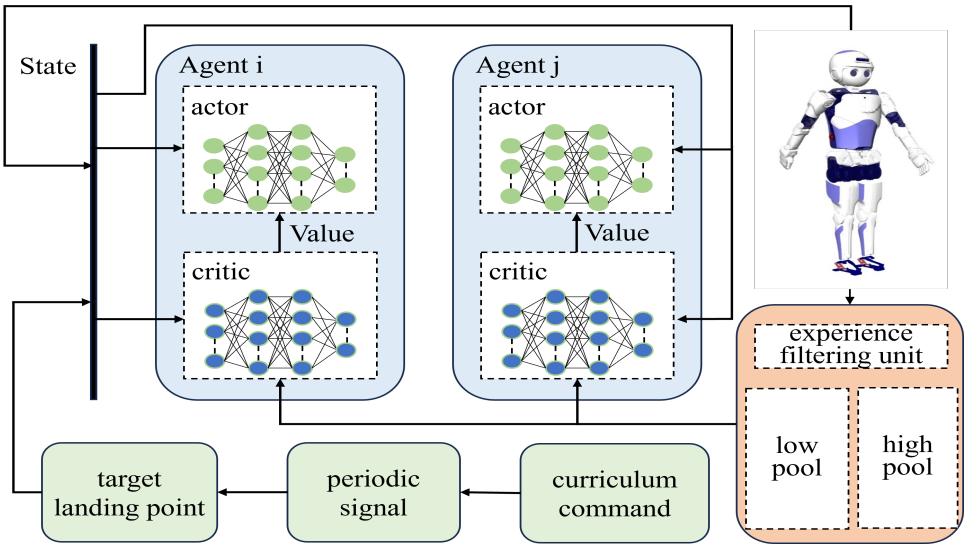


Figure 2. Training framework. Multi-agent collaboration solves the problem of initial state distribution differences. Policy entropy and experience replay mechanisms to improve training speed and quality. Course learning methods to implement multiple expressions of a single policy.

The joint strategy of all agents is denoted as $\pi = \{\pi^1, \pi^2, \cdots, \pi^N\}$. The primary objective of the cooperative task is for all agents to collaborate and find an optimal joint strategy π^* that satisfies

$$\pi^* = \arg\max_{\pi} \zeta(\pi), \tag{6}$$

where, $\zeta(\pi) = \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[\sum_{t=0}^{+\infty} r(s_t, a_t)]$ is the expected discount reward.

In a multi-agent environment, the state transition is jointly determined by the actions of all agents, and the reward obtained by each agent is interrelated with other agents. Therefore, altering the policy of a single agent directly impacts the selection of optimal decisions and the accuracy of value function estimation for other agents. To ensure convergence of the multi-agent system algorithm, we employ a centralized training-distributed execution architecture for training. Within this framework of centralized training and distributed decision making, we adopt an optimization approach that simultaneously maximizes cumulative reward and policy entropy to enhance exploration efficiency for each agent in an unknown environment. A policy entropy term can be added as

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[\sum_{t=0}^{\infty} r(s_t, a_t) + \vartheta\varepsilon(\pi(\cdot|s_t))], \tag{7}$$

where $\varepsilon(\pi(\cdot|s_t)) = -\log(\pi(a_t|s_t))$; ϑ is the entropy coefficient.

The term "policy entropy" refers to the entropy of the policy distribution, which quantifies the level of randomness or exploration ability in an agent's decision-making process. A higher policy entropy indicates a stronger inclination towards exploring unknown environments. This level of exploration is crucial to acquiring a comprehensive understanding of the environment and circumventing the risk of being trapped in suboptimal solutions. To strike a balance between exploration and exploitation, we introduce an optimization objective function for the entropy coefficient ϱ. The value of ϱ is updated using gradient descent, aiming to optimize the overall performance of the system. The objective function for optimizing the entropy coefficient is

$$J(\vartheta) = \mathbb{E}_{a_t\sim\pi}[-\vartheta \log \pi(a_t|s_t) - \vartheta\varrho], \tag{8}$$

where ϱ is the action dimension of the agent.

The policy gradient of $p_\omega(s_0)$ is designed as

$$J(\theta, \omega) = \mathbb{E}_{\tau \sim p_{\theta,\omega}}[\nabla_\omega \log(p_\omega)(s_0) \sum_{t=0}^{T} \gamma^t r_t], \tag{9}$$

where $p_\omega(s_0)$ is the initial state distribution.

The motion control of the bipedal robot is governed by the actions of the first agent, which are determined by the policy $\pi(a_t|s_t, g_t)$. On the other hand, the second agent is responsible for proposing the initial state at the start of each round of RL training, as defined by the likelihood function $p_\omega(s_0)$. There is a cooperative relationship between the two agents. To maximize the multi-agent objective, the cooperative interaction between the two agents is expressed as

$$J(\theta, \omega) = \mathbb{E}_{\tau \sim p_{\theta,\omega}}[\Sigma_{t=0}^{T} \gamma^t r_t]. \tag{10}$$

4.4. Heterogeneous Policy Experience Replay Mechanism Based on Euclidean Distance

Traditional experience replay mechanisms suffer from an imbalance in the distribution of experience data due to their reliance on a single experience buffer. This limitation often leads to prolonged training times required to achieve convergence. In many existing studies, the experience buffer is partitioned based on the quality of experiences, with a predominant focus on sampling from the pool of excellent experiences. However, this approach can lead to the model gradient overlooking the significance of ordinary experiences during the updating process, thereby narrowing the algorithm's exploration range. Additionally, this division of experience may result in the disregard of valuable experience samples, thereby limiting the overall learning potential of the algorithm.

To address the aforementioned issues, we propose a novel heterogeneous policy experience replay mechanism based on Euclidean distance. This method tackles the problems by employing an experience filtering unit to store the generated experiences and conducting experience replay through two distinct experience pools: the low pool and the high pool. The former contains samples whose similarity is less than a preset threshold ϕ, while the latter contains those whose similarity is greater than ϕ. Initially, experience generated by training is stored in the experience filtering unit $F = (\varphi_1, \varphi_2, \cdots, \varphi_n)$, where $\varphi_i = (s_i, a_i, r_i, s_{i+1})$ consists of states and actions resulting from biped robot training. We establish an initial experience sample in F as a baseline experience sample. The similarity between the new experience sample and the baseline experience sample is then calculated using the Euclidean distance by:

$$sim_i = \min(\|s_i - s_o\|_2 + \|a_i - a_o\|_2). \tag{11}$$

We also design a sample similarity distance threshold ϕ to differentiate between similar samples, which is calculated as the average distance between new experience samples and baseline experience samples. In order to distinguish similar samples, a method is devised as:

$$\phi = (sim_1 + sim_2 + \cdots + sim_n)/n. \tag{12}$$

The low pool stores experience samples with similarity less than ϕ, while the high pool is used to store experience samples with similarity greater than ϕ. In each round, experience samples are added to set F. At the end of each round, the experience filtering unit selects and categorizes the stored experience samples into their respective pools. Simultaneously, the benchmark experience samples are also updated. The sizes of the low pool and high pool are fixed. The probability of sampling and updating parameters from the high pool is α, while the probability from the low pool is $1 - \alpha$. Compared to existing research, this heterogeneous strategy method not only maintains the role of the experience replay mechanism and expands the exploration range, but also enhances the influence of excellent samples on network gradients. Compared to [38], our dynamic baseline experience samples and similarity measure can to some extent filter out low-quality samples, thereby improving the model's ability to learn from and generate high-quality samples. Additionally, the distribution of samples may change over time during the training stages of bipedal robots.

The policy network may encounter different types of samples with varying features and importance. By continuously updating the baseline experience samples, we can better reflect the characteristics of samples in the current training stage, enabling the model to adapt to new data distributions and improve its generalization performance. Our experience replay mechanism is shown in Figure 3.

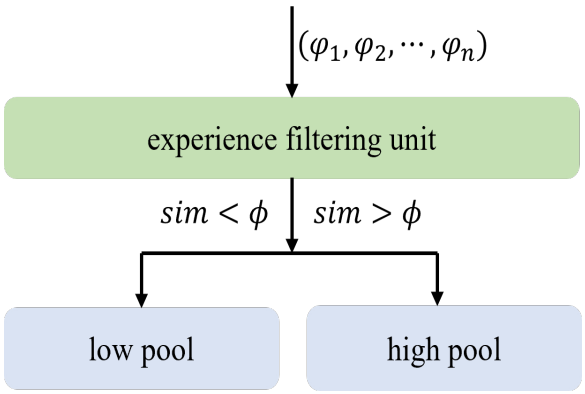


Figure 3. Heterogeneous policy experience replay mechanism. We establish an experience cache unit, baseline samples, and similarity thresholds.

4.5. Design of Curriculum Learning Framework for Omnidirectional Gait

In this paper, we divide the difficulty of the task into various levels, which are determined by the magnitude of the turning angle. We propose an approach for selecting the appropriate difficulty level within the curriculum.

To generate new training tasks at the target difficulty level, we define the curriculum parameter set Λ as a sequence of gait training tasks arranged in ascending order of difficulty.

$$\Lambda = \{l | 0 \leq l \leq l_{max}\}, \tag{13}$$

where l_{max} is the maximum level of difficulty in the curriculum.

In course strategy design, completion rate is used as a metric to gauge the current progress of the course. Additionally, it is also an important measure in evaluating whether one can advance to the next level of difficulty. We define the indicator function S_l, which uses course parameters to flag the task completion status.

$$s_l = \begin{cases} 1, & Task \quad completed \\ 0, & otherwise \end{cases}. \tag{14}$$

The completion rate of a course under the course parameter l is defined as

$$Pr(l) = \frac{\sum_{i=0}^{N} S_l(i)}{N}. \tag{15}$$

If a specific set of course parameter l satisfies $Pr(l) > k$, where k is a preset threshold for triggering, then the robot can complete the training task smoothly under this difficulty level. Afterwards, we elevate the task difficulty by increasing the angle of turning, and set the starting difficulty level for the next training session when resetting the environment.

With a course difficulty of c and a variance of σ^2, the density function of course l is

$$\rho_{\sigma,c}(l) = \exp(-\pi \| l - c \| / \sigma^2), \tag{16}$$

where c determines the location of the distribution and σ^2 determines the magnitude of the distribution. Under $l \in \Lambda$, all $\rho_{\sigma,c}$ discrete integrals are

$$\rho_{\sigma,c}(\Lambda) = \Sigma_{x \in \Lambda} \rho_{\sigma,c}(l). \tag{17}$$

With a course difficulty of c and a variance of σ^2, the distribution function of course l is

$$D_{\Lambda,\sigma,c}(l) = \frac{\rho_{\sigma,c}(l)}{\rho_{\sigma,c}(\Lambda)}. \tag{18}$$

Each thread selects an appropriate task difficulty level at each environment reset based on Equation (18). Our course learning steps are summarized in Algorithm 1. This method ensures that robots are not all put at the same difficulty level under each course difficulty level, but rather different environments with various difficulty levels are selected by discretizing according to a Gaussian distribution.

Algorithm 1 Curriculum learning algorithm

1: Initial curriculum difficulty
2: The robot collects data and calculates whether the curriculum is completed or not by using Equation (13)
3: Calculate the completion rate $Pr(l)$ of the curriculum by using Equation (14)
4: **If:** $Pr(l) > k$:
5: **If:** the highest difficulty level is reached:
6: Randomize the difficulty of the curriculum
7: **Else:**
8: Increase the difficulty of the curriculum
9: **Else:**
10: Maintain the difficulty of the curriculum
11: Return to step 2

4.6. Markov Decision Process Modeling for Omnidirectional Gait

(1) Statespace

The input to the controller in this paper is a 41-dimensional state space $s_t \in R^{41}$ including the pelvis orientation $(R_{roll}, R_{pit}, R_{yaw}) \in R^3$, the linear velocity $v_p = (v_x, v_y, v_z) \in R^3$ of the pelvis, the angular velocity ω_p of the pelvis, the target position of the pelvis $(p_{tarx}, p_{tary}, p_{tarz}) \in R^3$, the joint position $q \in R^{10}$ and joint velocity $\dot{q} \in R^{10}$ of each drive joint, and the target control velocity $v_{cmd} = (v_x^{cmd}, v_y^{cmd}, \omega_z^{cmd}) \in R^3$ of the robot. In addition, we include the next two target steps of the robot and its heading information $(x, y, \theta) \in R^6$. The phase vector is constructed as $[sin((2 * pi * t)/T), cos((2 * pi * t)/T)]$.

(2) Action space

We use the PD control target as the action space and select the position of the movable joint as the action. Due to the underdrive of the robot, directly using PD control to apply the magnitude of the torque to track the reference angle will cause the robot to fall down easily. Therefore, we change the method of PD control and use position incremental control to solve the problem of robot underdrive. This method keeps the robot balanced by calculating the position increment of the robot and adjusting its torque. Using this method not only improves the stability of the robot, but also enables the robot to perform its tasks more flexibly and finely.

Therefore, our RL strategy outputs the incremental joint position σq_t, which is added to the target joint position q_{t-1} at the previous moment. Then the target joint position at the current moment is defined as

$$q_t = q_{t-1} + \sigma q_t. \tag{19}$$

To track these joint angles, the torque applied to the joint is calculated using a low-level PD controller. The torque is calculated as

$$\tau = k_p(q_{tar} - q) + k_d(\dot{q}_{tar} - \dot{q}), \tag{20}$$

where k_p, k_d are PD gains, which have been corrected in the actual simulation. q_{tar} and q_{tar-1} are the target position and target velocity of the joint, respectively. The linear velocity of the target joint is set to 0 during the training process. q and $\dot{q}$ are the current position and linear velocity of the active joint, respectively.

(3) Reward Functions

The purpose of the reward function is to motivate the robot to follow specified commands, keep its body smooth, and achieve stable motion. The reward function is designed as

$$R = r_{vel} + r_o + r_{ad} + r_\tau + r_{alive} + r_{per} + r_{per} + r_{for}, \quad (21)$$

Table 2 shows the details of the reward function of Formula 21. h_{tar}^{foot} is the target height output by the cycle symmetric foot lift height curve imitated by the left and right feet; h_{rel}^{foot} is the actual height of the left and right feet. p_{tar}^{foot} is the target foothold; p_{rel}^{foot} is the actual foothold. p_{tar} is the target foothold; p_{root} is the projection point of the actual position of the floating base. r_{imi} encourages humanoid robots to place their feet on the target point. When the distance between one or both feet and the target point is within the target radius, it will be considered as stepping on the target point. r_{for} keeps the distance between the projection of the root node and the target foothold constantly close, thereby preventing the robot from standing still, so that the floating base of the robot can continuously move to the next target point.

Table 2. Reward functions.

Effect	Expression
command tracking	$r_{vel} = exp((v - v_{cmd})/max(\delta(\Sigma(v_{tar})^2), 0.01))$
keep balance	$r_o = exp(1 - < q, \hat{q} >^2)$
smooth action	$r_{ad} = exp(\sum_{i=0}^{n}(la_i - a_i)^2)$
	$r_\tau = exp(\sum_{i=0}^{n}(l\tau_i - \tau_i)^2)$
Sport termination	$r_{alive} = \begin{cases} -1, & if stop conition \\ 0, & else \end{cases}$
Periodic Gait	$r_{per} = exp(\sum_{i=0}^{2}(h_{tar}^{foot_i} - h_{rel}^{foot_i})^2)$
Omnidirectional gait	$r_{imi} = exp(\sum_{i=0}^{2}(p_{tar}^{foot_i} - p_{rel}^{foot_i})^2)$
	$r_{for} = exp(-\sum_{i=0}^{2}(p_{tar} - p_{root})^2)$

5. Results and Discussion

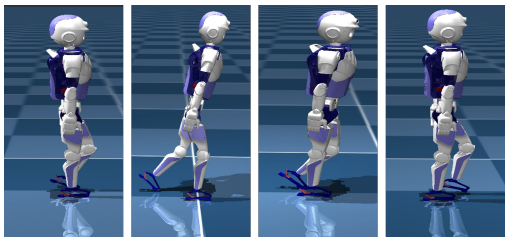
5.1. Experimental Environment and Settings

This experiment leverages the Ubuntu 20.04LTS operating system and the Pytorch deep learning framework. Our model is built upon the AMD Ryzen ThreadRipper 3990X CPU and the Nvidia GeForce RTX 3090 (24 GB) GPU. The MuJoCo platform serves as the foundation for our physical simulations, offering support for parallel learning, closed-chain or flexible robot simulation, and exceptional performance in various robot training tasks. During the experiment, we set the MuJoCo simulator to operate at a frequency of 500 Hz, the PD controller at 2000 Hz, and the neural network at 50 Hz. To expedite the training process, we simultaneously employed 16 parallel environments, allowing us to complete the training in approximately 6 h and collect a total of 40 million samples. Our novel approach incorporates curriculum learning, enabling the generation of different velocity commands with a single policy, without the need for policy retraining. Furthermore, we compared our method against both the direct adoption of the PPO method and the integration of traditional curriculum learning into the PPO method (CLPPO) in subsequent experiments. In order to guarantee consistent imitation behavior among the three sets of commands, we set the parameters T, T_{std}, and T_{swi} to 30, 4, and 14, respectively. The k_p and

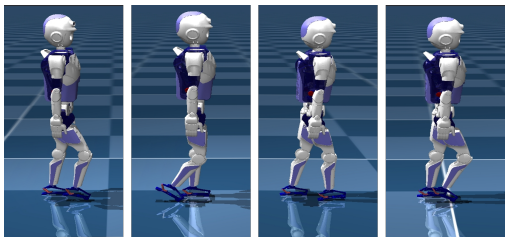
k_d are set to 80 and 5, respectively. Our actor and critic networks both consist of two hidden layers, with 256 neurons in each layer. We parallelized the environment across 64 instances. The policy network is optimized every 512 steps, which corresponds to optimizing the policy network every 32,768 samples. We use a learning rate of 0.0003 and a discount factor of 0.98. Each round of training has 600 steps.

5.2. Straight Gait

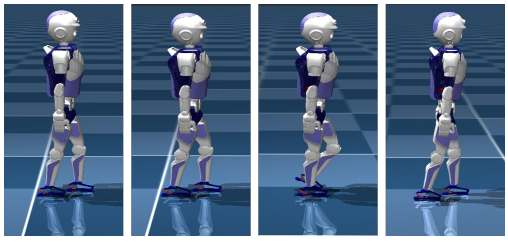
Figure 4 presents the state diagrams of the bipedal robot's straight walking achieved by training its policy network with three different methods. In the performance comparison of the forward gait, the PPO method employs velocity commands $v_{cmd} = [0.5, 0, 0]$. Curriculum learning is incorporated into both CLPPO and our method based on these instructions. Both CLPPO and our method have a velocity command range of $v_x^{cmd} \in [-0.5, 0.5]$, $v_y^{cmd} \in [-0.5, 0.5]$, and $\omega_z^{cmd} \in [-0.5, 0.5]$. It is evident from Figure 4 that the bipedal robot walking with the PPO method experiences noticeable sway. However, both our method and the CLPPO method do not demonstrate noticeable sway. We also compare the convergence of the three algorithms during the training process, as shown in Figure 5. We also include the training convergence of the multi-agent approach (MPPO) that treats the robot as an agent. The total reward of the MPPO method is the lowest among the four methods. Furthermore, only our method has a total reward exceeding 400. Therefore, only the two baseline algorithms PPO and CLPPO are compared below.



(**a**) PPO



(**b**) CLPPO



(**c**) Our method

Figure 4. Walking state diagram of bipedal robot under three algorithms. We used the same straight-line speed command to conduct a comparison test of the robot's walking using three algorithms.

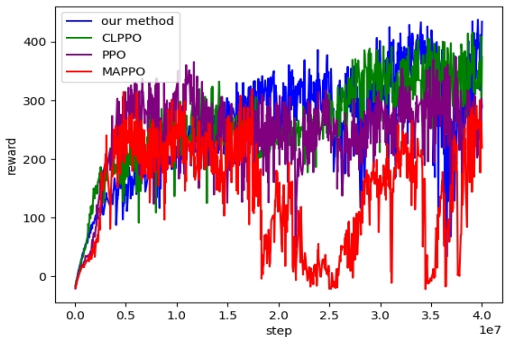
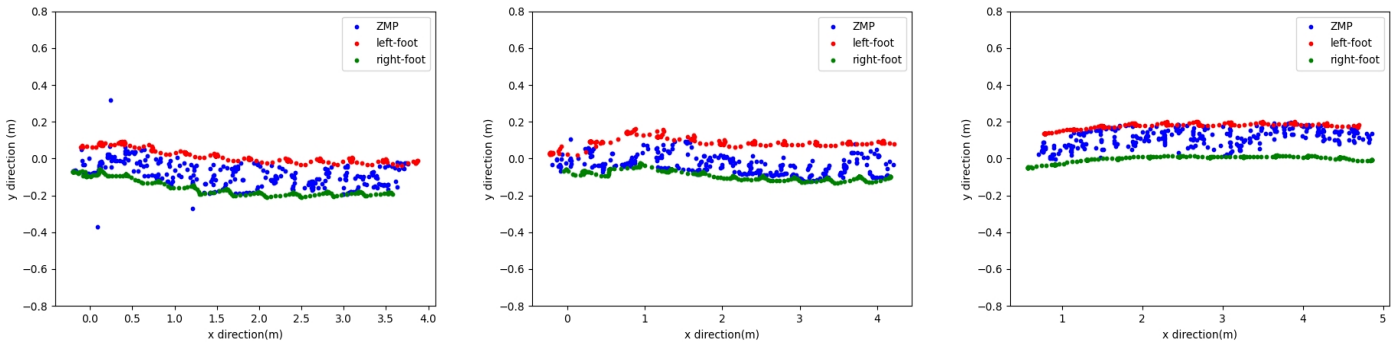


Figure 5. Training Process: A total of 40 million samples were collected to compare our algorithm with three other algorithms. The reward metric represents the average total reward value per iteration.

Figure 6 illustrates the ZMP position and the positions of the left and right feet for the three methods. Our proposed method maintains a consistent ZMP position between the touchdown points of the left and right feet, ensuring stability during bipedal robot locomotion. In contrast, the ZMP position under the PPO method displays significant jumps, while the CLPPO method effectively keeps the ZMP positioned between the two feet. The ZMP position under the PPO method is the least stable among the three methods, which is consistent with the significant body sway observed in Figure 4 due to the PPO policy control.

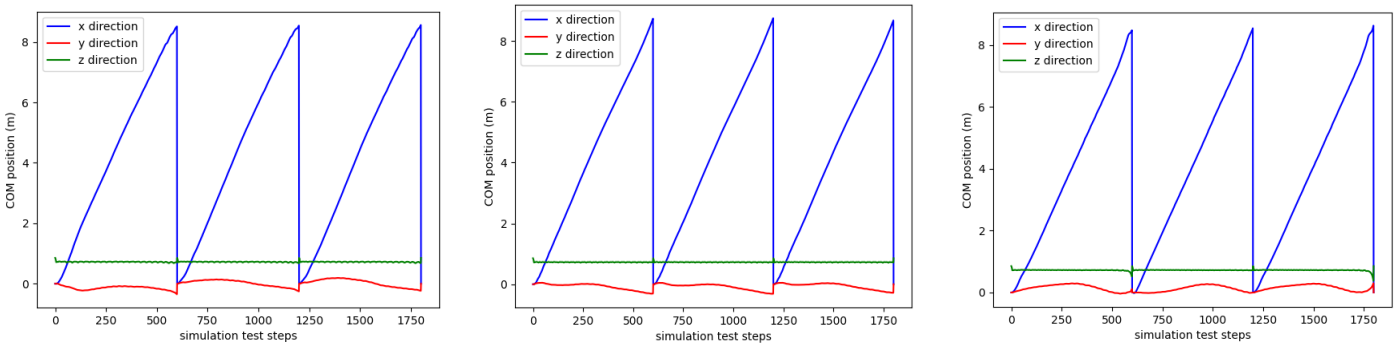


(**a**) PPO (**b**) CLPPO (**c**) Our method

Figure 6. Diagrams of ZMP and positions of left and right feet. Testing robot straight motion using three algorithm-based training policies.

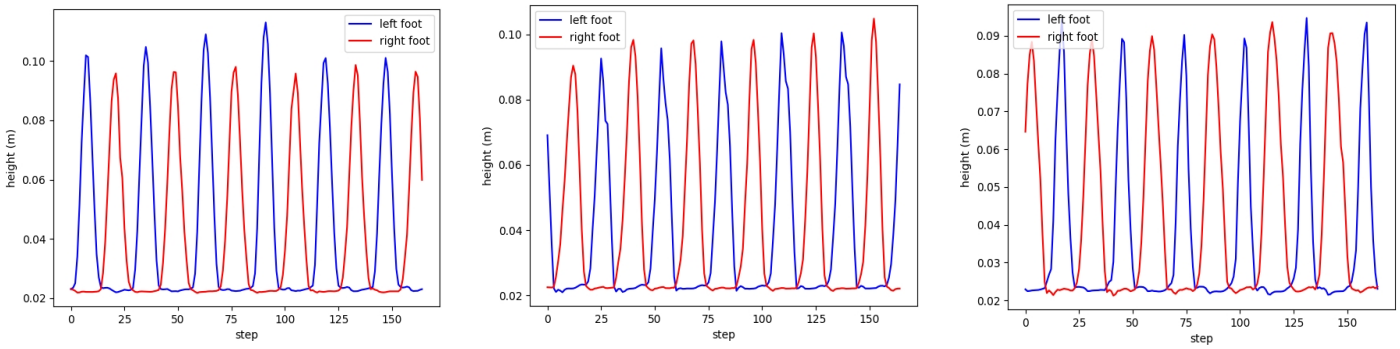
We conduct tests on the positions of the center of mass (COM) of the bipedal robot along the x, y, and z axes during the walking process. Each algorithm is reset after 600 steps during each execution, and a total of three rounds are tested, amounting to a total of 1800 steps. The results are presented in Figure 7. The change in the x-axis position under our method is smooth, indicating a consistent walking speed with no variations for the bipedal robot. Both the PPO and CLPPO methods show slight variations in the x-axis velocity, which result in body swaying during the robot's walking process.

From Figure 8a, it can be observed that PPO, which does not utilize the periodic reward designed by us, exhibits significant variations in foot height and fails to meet the periodic characteristics of bipedal walking. Our approach and CLPPO demonstrate consistent periodic variations in left foot height, as illustrated in Figure 8b,c. Failure to meet the periodic characteristics can lead to falls during the bipedal robot's walking process. We also conduct tests using the cyclic gait method described in reference [33], as shown in Figure 9. However, due to the inability to specify foot lifting height, the periodic characteristics of foot elevation in bipedal robots cannot be satisfied.



(**a**) PPO (**b**) CLPPO (**c**) Our method

Figure 7. COM position diagram for the three methods. The position of COM in the three directions (x, y, z) during the walking process of the bipedal robot.



(**a**) PPO (**b**) CLPPO (**c**) Our method

Figure 8. Height diagram of left and right feet for the three methods. Our method is basically the same height.

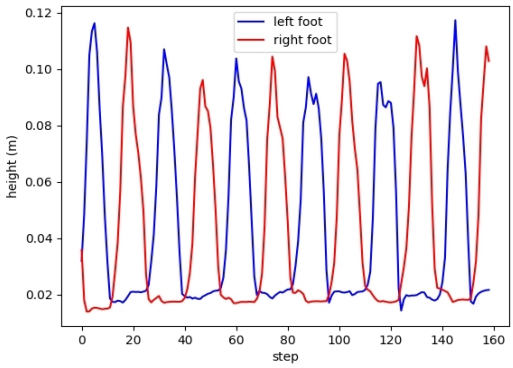


Figure 9. Height of left and right feet using the periodic gait method in reference [33]. The height of the foot lift is different.

5.3. Steering Gait

Next, we evaluate the steering gait performance of the three methods. We administer a steering velocity command $v_{cmd} = [0.5, 0.1, 0.1]$ to retrain the steering gait for the PPO method. CLPPO and our method continue to employ the previously trained policies. The PPO method displays convergence already in the early stages of training, as shown in Figure 10. Since the PPO method fails to achieve a steering gait for the bipedal robot, we exclusively compare the steering performance between CLPPO and our method.

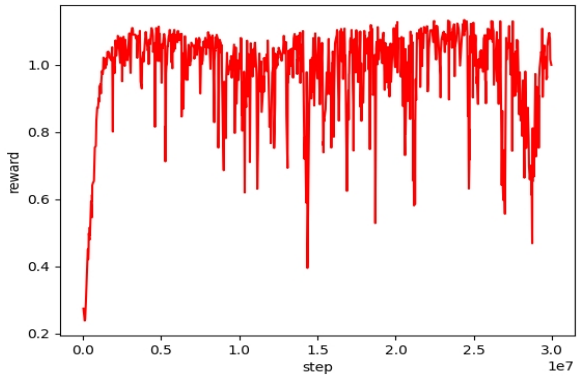


Figure 10. Training reward diagram of PPO method. Test our strategy by running in circles using $v_{cmd} = [0.5, 0.1, 0.1]$.

The walking situation of our method in the simulation environment is shown in Figure 11. Figure 12 shows the ZMP positions of two methods and the positions of the left and right feet. Both methods can keep the ZMP position between the left and right feet, indicating that the robots can maintain stable walking. The oscillation in Figure 12a is caused by the inability of the robot to track the velocity command throughout the motion. This results in bipedal robots being unable to maintain smooth and continuous turning while walking, and the body of the robot also shakes when oscillating. However, our method controls the robot in turning and walking on a smoother trajectory without the oscillation observed in Figure 12b.

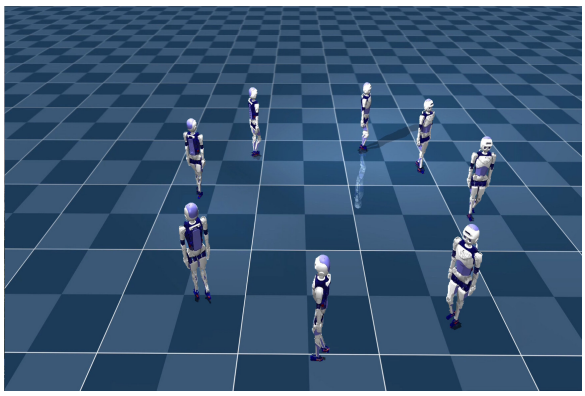
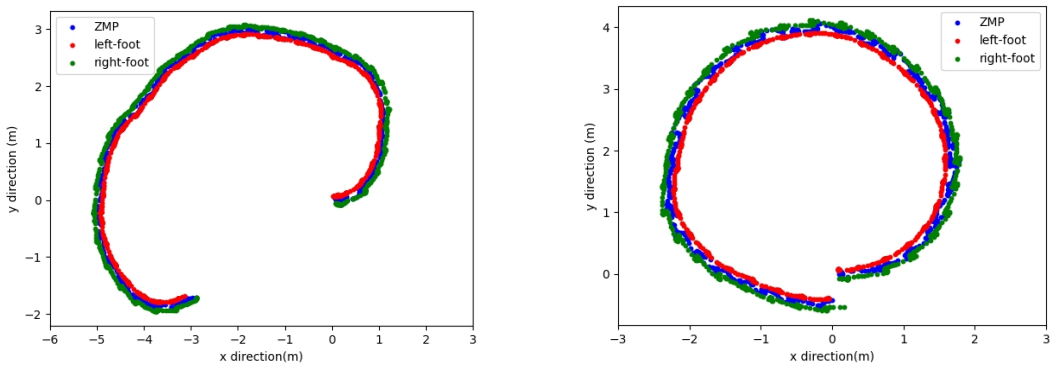


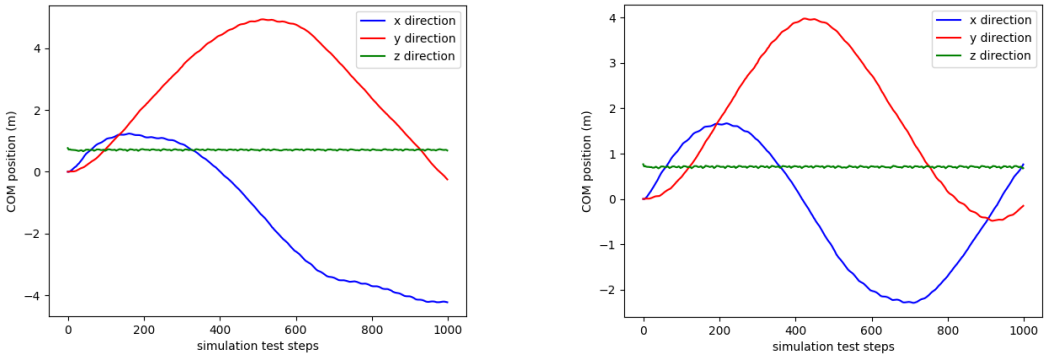
Figure 11. Walking state diagram of bipedal robot. The bipedal robot turns in circles at the same turning speed and finally returns to near the original place.



(**a**) CLPPO (**b**) Our method

Figure 12. Diagrams of ZMP and positions of left and right feet.

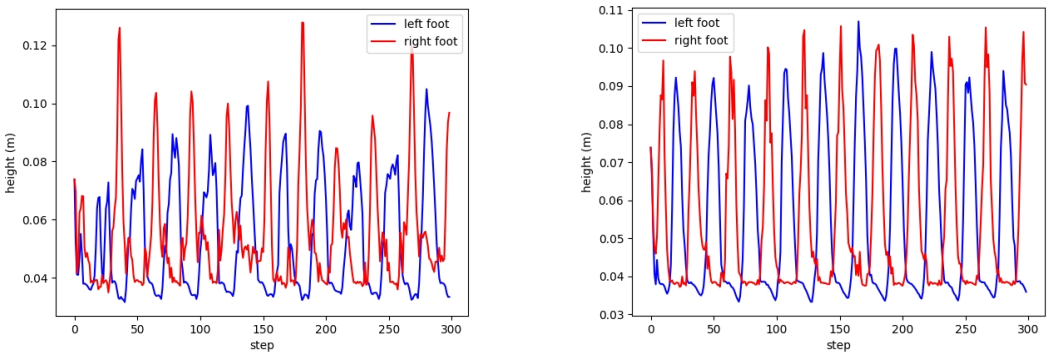
Figure 13 shows the COM position of two methods. Our method ensures smoother variations in the robot's position along the x-axis. This means that the robot can maintain a stable walking speed to ensure stability.



(**a**) CLPPO (**b**) Our method

Figure 13. COM position diagram for the two methods.

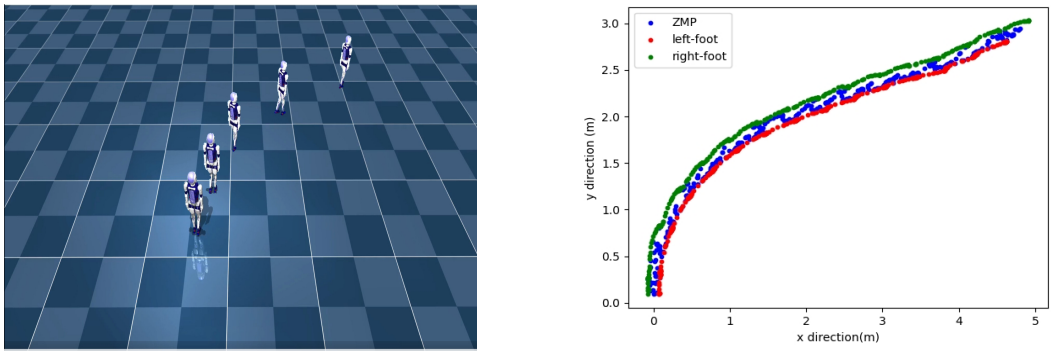
From Figure 14a, it is evident that there are pronounced discontinuities in the foot height of the bipedal robot when employing the CLPPO method. This does not guarantee the stability of the robot during the walking process. The foot heights during the steering process of our method are illustrated in Figure 14b. The foot heights of both feet continue to display periodic characteristics without experiencing substantial discontinuities, thus enabling a stable steering process.



(**a**) CLPPO (**b**) Our method

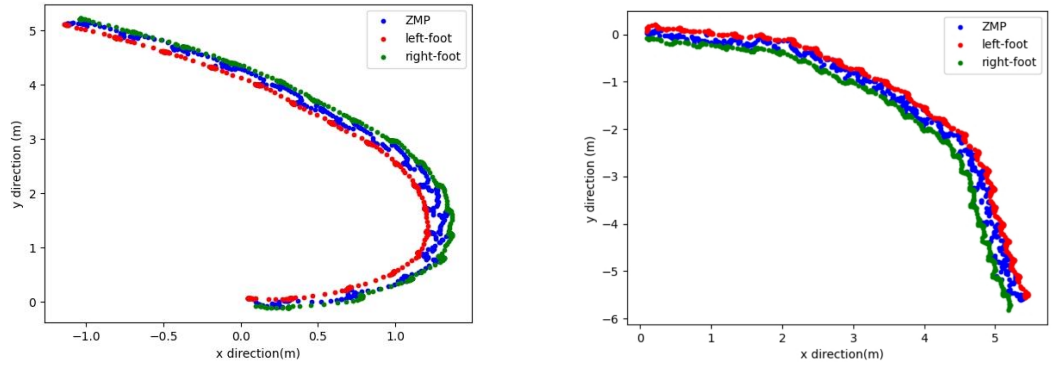
Figure 14. Height diagram of left and right feet for the two methods. Both methods executed the same speed command.

Lastly, we test the performance of two methods in terms of omnidirectional gait. Firstly, we initiate straight walking; then, a rotational velocity command is given to the robot, transitioning it into straight walking after a certain duration. The CLPPO method experiences instances of backward movement and falling during the transition between the two gaits. In contrast, our method successfully executes the given velocity command independently, as shown in Figure 15a. The ZMP position and the positions of both feet in our method are illustrated in Figure 15b. Our method achieves smooth gait transitions through a single policy. We further validate the effectiveness of our method through additional velocity commands, as shown in Figure 16a,b. From the foot landing points and ZMP locations, it can be observed that our method ensures stability and smoothness during gait transitions.



(**a**) (**b**)

Figure 15. Robot walking gait switching diagram. The robot first goes straight, then turns, and finally continues going straight. (**a**) Scenario diagram of the robot in the simulation scene. (**b**) Diagrams of ZMP and positions of left and right feet.



(**a**) CLPPO (**b**) Our method

Figure 16. Our method performs gait switching tests between straight walking and turning in two steering modes.

6. Conclusions

In this work, we have proposed a novel RL framework for generating periodic symmetric bipedal omnidirectional gaits. The design of our bipedal periodic symmetric functions is crucial for achieving excellent gaits during the straight walking phase. Our approach leverages learning from the ROS footprint planner and curriculum learning techniques, enabling seamless transitions between omnidirectional gaits using a single policy network. The introduction of multi-agent RL mitigates the impact of differences between the footprint planner and our robot, allowing for parallel learning of initial state distributions for the agents. We have also incorporated policy entropy and heterogeneous experience replay mechanisms to expedite gait training for bipedal robots. The results from comparative experiments demonstrate that our method enables seamless transitions between different gaits for full-sized bipedal robots, not just smooth turning motions. Our approach achieves the smooth execution of omnidirectional gaits using a single policy, which is a significant accomplishment. Future research can explore more effective and robust methods for bipedal gait training. Robots need to walk on different terrains, such as stairs, slopes, and uneven surfaces. These terrains have different characteristics and difficulties, so it is necessary to design corresponding gait control strategies for different terrains. We will explore how to use techniques such as deep learning to automatically learn and optimize gait control strategies that adapt to different terrains. However, it is worth noting that although this study achieved omnidirectional gait of a bipedal robot in a reinforcement learning simulation environment, the ultimate goal of reinforcement learning is to simulate and transfer to the real world. We will try to solve this problem from two directions, of which one is to improve the robustness of the learned policies by changing the model

parameters within a certain range in each iteration of the learning algorithm, and another is to combine learning with a model-based control.

Author Contributions: Conceptualization, H.M. and Q.L.; Methodology: H.M. and J.X.; Formal analysis and investigation: H.M. and J.X.; Writing - original draft preparation: H.M., J.X. and J.L.; Writing—review and editing: H.M., J.X. and J.L.; Software: H.M., J.X. and Z.F.; Funding acquisition: Q.L.; Supervision: J.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This study is supported by the Pujiang Talents Plan of Shanghai (Grant No. 2019PJD035) and the Artificial Intelligence Innovation and Development Special Fund of Shanghai (Grant No. 2019RGZN01041).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Do, H.M.; Welch, K.C.; Sheng, W. Soham: A sound-based human activity monitoring framework for home service robots. *IEEE Trans. Autom. Sci. Eng.* **2021**, *19*, 2369–2383. [CrossRef]
2. Huang, J.K.; Grizzle, J.W. Efficient anytime clf reactive planning system for a bipedal robot on undulating terrain. *IEEE Trans. Robot.* **2023**, *39*, 2093–2110. [CrossRef]
3. Wang, Z.; Kou, L.; Ke, W.; Chen, Y.; Bai, Y.; Li, Q.; Lu, D. A Spring Compensation Method for a Low-Cost Biped Robot Based on Whole Body Control. *Biomimetics* **2023**, *8*, 126. [CrossRef] [PubMed]
4. Singh, B.; Vijayvargiya, A.; Kumar, R. Mapping model for genesis of joint trajectory using human gait dataset. In Proceedings of the 2021 Smart Technologies, Communication and Robotics (STCR), Sathyamangalam, India, 9–10 October 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–5.
5. Zhao, Y.; Gao, Y.; Sun, Q.; Tian, Y.; Mao, L.; Gao, F. A real-time low-computation cost human-following framework in outdoor environment for legged robots. *Robot. Auton. Syst.* **2021**, *146*, 103899. [CrossRef]
6. Park, H.Y.; Kim, J.H.; Yamamoto, K. A new stability framework for trajectory tracking control of biped walking robots. *IEEE Trans. Ind. Inform.* **2022**, *18*, 6767–6777. [CrossRef]
7. Sugihara, T.; Imanishi, K.; Yamamoto, T.; Caron, S. 3D biped locomotion control including seamless transition between walking and running via 3D ZMP manipulation. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 6258–6263.
8. Huang, Q.; Dong, C.; Yu, Z.; Chen, X.; Li, Q.; Chen, H.; Liu, H. Resistant compliance control for biped robot inspired by humanlike behavior. *IEEE/ASME Trans. Mechatronics* **2022**, *27*, 3463–3473. [CrossRef]
9. Dong, C.; Yu, Z.; Chen, X.; Chen, H.; Huang, Y.; Huang, Q. Adaptability control towards complex ground based on fuzzy logic for humanoid robots. *IEEE Trans. Fuzzy Syst.* **2022**, *30*, 1574–1584. [CrossRef]
10. Hong, Y.D.; Lee, B. Real-time feasible footstep planning for bipedal robots in three-dimensional environments using particle swarm optimization. *IEEE/ASME Trans. Mechatronics* **2019**, *25*, 429–437. [CrossRef]
11. Wang, Y.; Li, J.P.; Xue, X.; Wang, B.C. Utilizing the correlation between constraints and objective function for constrained evolutionary optimization. *IEEE Trans. Evol. Comput.* **2019**, *24*, 29–43. [CrossRef]
12. Yang, Y.; Shi, J.; Huang, S.; Ge, Y.; Cai, W.; Li, Q.; Chen, X.; Li, X.; Zhao, M. Balanced standing on one foot of biped robot based on three-particle model predictive control. *Biomimetics* **2022**, *7*, 244. [CrossRef]
13. Dantec, E.; Naveau, M.; Fernbach, P.; Villa, N.; Saurel, G.; Stasse, O.; Taix, M.; Mansard, N. Whole-Body Model Predictive Control for Biped Locomotion on a Torque-Controlled Humanoid Robot. In Proceedings of the 2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids), Ginowan, Japan, 28–30 November 2022; pp. 638–644.
14. Beranek, R.; Karimi, M.; Ahmadi, M. A behavior-based reinforcement learning approach to control walking bipedal robots under unknown disturbances. *IEEE/ASME Trans. Mechatronics* **2021**, *27*, 2710–2720. [CrossRef]
15. Yuan, Y.; Li, Z.; Zhao, T.; Gan, D. DMP-based motion generation for a walking exoskeleton robot using reinforcement learning. *IEEE Trans. Ind. Electron.* **2019**, *67*, 3830–3839. [CrossRef]
16. Chang, H.H.; Song, Y.; Doan, T.T.; Liu, L. Federated Multi-Agent Deep Reinforcement Learning (Fed-MADRL) for Dynamic Spectrum Access. *IEEE Trans. Wirel. Commun.* **2023**, *22*, 5337–5348. [CrossRef]
17. Kumar, H.; Koppel, A.; Ribeiro, A. On the sample complexity of actor-critic method for reinforcement learning with function approximation. *Mach. Learn.* **2023**, *112*, 2433–2467. [CrossRef]
18. Henderson, P.; Islam, R.; Bachman, P.; Pineau, J.; Precup, D.; Meger, D. Deep reinforcement learning that matters. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LO, USA, 2–7 February 2018; Volume 32.

19. Lu, F.; Zhou, G.; Zhang, C.; Liu, Y.; Chang, F.; Xiao, Z. Energy-efficient multi-pass cutting parameters optimisation for aviation parts in flank milling with deep reinforcement learning. *Robot. Comput. Integr. Manuf.* **2023**, *81*, 102488. [CrossRef]
20. Yang, S.; Song, S.; Chu, S.; Song, R.; Cheng, J.; Li, Y.; Zhang, W. Heuristics Integrated Deep Reinforcement Learning for Online 3D Bin Packing. *IEEE Trans. Autom. Sci. Eng.* **2023**, 1–12. [CrossRef]
21. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347.
22. Xie, Z.; Berseth, G.; Clary, P.; Hurst, J.; van de Panne, M. Feedback control for cassie with deep reinforcement learning. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1241–1246.
23. Kumar, A.; Li, Z.; Zeng, J.; Pathak, D.; Sreenath, K.; Malik, J. Adapting rapid motor adaptation for bipedal robots. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1161–1168.
24. Cui, J.; Yuan, L.; He, L.; Xiao, W.; Ran, T.; Zhang, J. Multi-input autonomous driving based on deep reinforcement learning with double bias experience replay. *IEEE Sensors J.* **2023**, *23*, 11253–11261. [CrossRef]
25. Ho, S.; Liu, M.; Du, L.; Gao, L.; Xiang, Y. Prototype-Guided Memory Replay for Continual Learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, 1–11. [CrossRef]
26. Yang, Y.; Caluwaerts, K.; Iscen, A.; Zhang, T.; Tan, J.; Sindhwani, V. Data Efficient Reinforcement Learning for Legged Robots. In Proceedings of the Conference on Robot Learning, PMLR, Cambridge, MA, USA, 30 October–1 November 2020; Volume 100, pp. 1–10.
27. Hwang, K.S.; Lin, J.L.; Yeh, K.H. Learning to adjust and refine gait patterns for a biped robot. *IEEE Trans. Syst. Man Cybern. Syst.* **2015**, *45*, 1481–1490. [CrossRef]
28. Qi, H.; Chen, X.; Yu, Z.; Huang, G.; Liu, Y.; Meng, L.; Huang, Q. Vertical Jump of a Humanoid Robot with CoP-Guided Angular Momentum Control and Impact Absorption. *IEEE Trans. Robot.* **2023**, *39*, 3154–3166. [CrossRef]
29. Xie, Z.; Clary, P.; Dao, J.; Morais, P.; Hurst, J.; Panne, M. Learning locomotion skills for cassie: Iterative design and sim-to-real. In Proceedings of the Conference on Robot Learning, PMLR, Cambridge, MA, USA, 16–18 November 2020; pp. 317–329.
30. Wu, Q.; Zhang, C.; Liu, Y. Custom Sine Waves Are Enough for Imitation Learning of Bipedal Gaits with Different Styles. In Proceedings of the 2022 IEEE International Conference on Mechatronics and Automation (ICMA), Guilin, China, 7–10 August 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 499–505.
31. Siekmann, J.; Godse, Y.; Fern, A.; Hurst, J. Sim-to-real learning of all common bipedal gaits via periodic reward composition. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 7309–7315.
32. Li, Z.; Cheng, X.; Peng, X.B.; Abbeel, P.; Levine, S.; Berseth, G.; Sreenath, K. Reinforcement learning for robust parameterized locomotion control of bipedal robots. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 2811–2817.
33. Duan, H.; Malik, A.; Dao, J.; Saxena, A.; Green, K.; Siekmann, J.; Fern, A.; Hurst, J. Sim-to-real learning of footstep-constrained bipedal dynamic walking. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 10428–10434.
34. Yu, F.; Batke, R.; Dao, J.; Hurst, J.; Green, K.; Fern, A. Dynamic Bipedal Maneuvers through Sim-to-Real Reinforcement Learning. *arXiv* **2022**, arXiv:2207.07835.
35. Singh, R.P.; Benallegue, M.; Morisawa, M.; Cisneros, R.; Kanehiro, F. Learning Bipedal Walking On Planned Footsteps For Humanoid Robots. In Proceedings of the 2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids), Ginowan, Japan, 28–30 November 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 686–693.
36. Rodriguez, D.; Behnke, S. DeepWalk: Omnidirectional bipedal gait by deep reinforcement learning. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 3033–3039.
37. Peng, X.B.; Ma, Z.; Abbeel, P.; Levine, S.; Kanazawa, A. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Trans. Graph. (TOG)* **2021**, *40*, 144. [CrossRef]
38. Li, C.; Li, M.; Tao, C. A parallel heterogeneous policy deep reinforcement learning algorithm for bipedal walking motion design. *Front. Neurorobotics* **2023**, *17*, 1205775. [CrossRef]