심층 강화학습 기반 이족 보행 로봇의 민첩한 장애물 넘기 알고리즘

Learning Agile Jumping Over the Obstacle for A Bipedal Robot Using Deep Reinforcement Learning

○Dat Thanh Truong[1], 이 중 환[1], 김 현 주[1], 이 재 홍[1], 강 한 솔[2], Minh Thang Bui[2],Hoang Anh Nguyen Duc[1], 최 혁 렬[2*]
[1)] 성균관대학교 지능형로봇학과 (TEL: 010-2197-2602; E-mail: thanh.truongdat9@gmail.com)
[2)] 성균관대학교 기계공학부 (TEL: 010-9921-3845; E-mail: kanghs0822@skku.edu)

Abstract Humanoid robotics has progressed rapidly, targeting tasks in challenging environments like rescue missions. However, fully autonomous navigation over complex terrains remains difficult. This paper introduces a reinforcement learning framework using position-based commands, eliminating penetration depth checks for efficient jumping. We release our framework on Isaac Lab and Isaac Sim as an open-source tool to advance research in robot agility.

Keywords Deep Reinforcement Learning, Humanoid Agile Locomotion

1. Introduction

Humanoid robotics has gained significant attention, with rapid advancements driven by research institutions and specialized companies. Developing fully autonomous robots with these capabilities remains a challenge.

The development of Isaac Gym—a GPU-accelerated simulation platform—has facilitated learning-based approaches for quadrupeds to tackle rough terrains. However, the methods in [1] rely on velocity commands and encourage jumping by creating virtual hurdles and manually checking penetration depth at multiple points around the robot. This duplicates the simulator's physics engine, increasing data collection time.

In this paper, we address the mentioned limitation in bipedal robot learning for obstacle jumping with the following contributions:

- A reinforcement learning framework for tracking position commands without requiring penetration depth checks, enabling effective jumping behavior.
- An open-source release of this framework, built on the state-of-the-art robot learning platforms Isaac Lab and Isaac Sim, which are successors to Isaac Gym.

2. Method

In this section, we introduce the position-based command approaches. The training process is shown in [Fig. 2]. First, the jumping policy trains the robot to walk using proprioceptive data and a height scan. Next, it is trained in an environment with hurdles of various heights, outputting motor joint commands at 100 Hz tracked by a PD controller running at 1000 Hz.

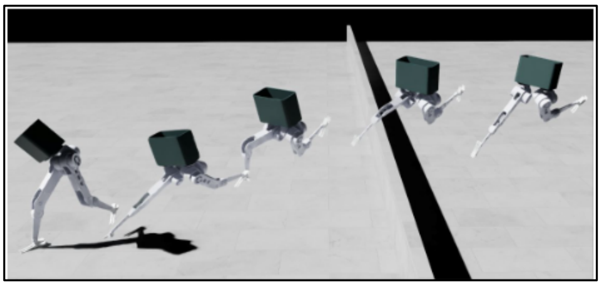

Figure 1. A sequence of images showing a bipedal robot jumping over an obstacle. The obstacle is 0.5 meters in height and 0.1 meters in width.

2.1 Baseline Policy Observations

In the simulation environment, the observations s_t fed into the policy are divided into two parts: baseline observations, s_t^b , and command, c_t . The baseline observations $s_t^b \in \mathbb{R}^{240}$, listed in [Table 1]. These inputs accelerate training and enhance robustness against disturbances.

2.2 Policy Representation

The policy architecture, shown in [Fig. 2], includes two MLPs with ELU activation functions and a GRU. The first MLP acts as an encoder for the height scan. This encoded vector is fed into the GRU, producing a hidden state. Finally, the hidden state serves as input to the second, larger MLP, which processes it to generate the actions.

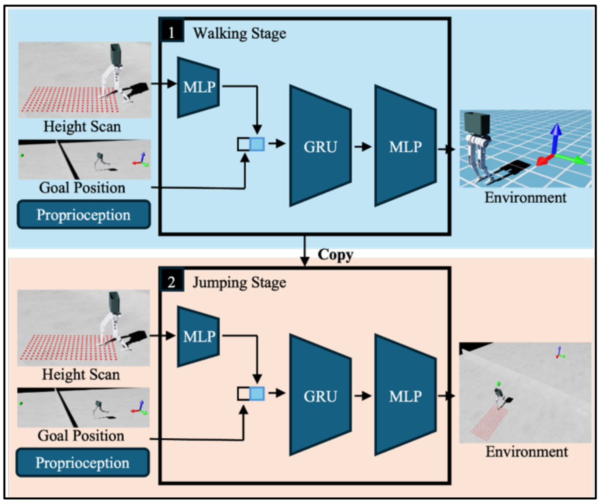

Figure 2. Training pipeline for position-based command approach.

2.3 Baseline Rewards

As illustrated in [Fig. 2], the first stage of training involves teaching the robot to walk. To achieve this, a set of baseline rewards is defined (see Table 1). These rewards serve as regularization terms, guiding the agent to generate appropriate joint commands while avoiding hardware limit violations.

2.4 Position-based Command Approach

To develop the jumping policy with a position-based command approach, we introduce an additional command, $c_p = [x, y, \psi]^T$which represents the relative position from the robot's base to the desired position in the base frame.

We define two reward terms that do not require penetration depth checks, as follows:

$$r_{postion} = \overline{1}_{\Delta p_d<0.15} \cdot v_b \cdot u_d + 1_{\Delta p_d<0.15} \cdot 1.5$$
$$r_{heading} = \varphi(\psi_d - \psi) \qquad (1)$$

Where Δp_d represent the relative distance between the robot's base and the desired position, u_d be the unit direction vector from the robot's base to the desired position (expressed in the base frame), ψ the current heading angle of the robot and ψ_d the desired heading angle.

Table 1. Baseline Policy Observations

Symbol	Description	Dim
$\boldsymbol{v_b}$	Velocity of the based in base frame	3
$\boldsymbol{\omega_b}$	Angular velocity of the based in base frame	3
$\boldsymbol{q}$	Joint positions of the motors	10
$\dot{\boldsymbol{q}}$	Joint velocities of the motors	10
$\boldsymbol{g}$	Projected gravity in base frame	3
$\boldsymbol{b_c}$	Binary contact state	3
h	Height scan	209

Table 2. Baseline Rewards

Description	Weight	Function
Base height	5.0	$\boldsymbol{\varphi(h - h_d)}$
Flat orientation	5.0	$\boldsymbol{\varphi(g_{xy})}$
Joint regularization	7.0	$\boldsymbol{\varphi(q_{yaw}) + \varphi(q_{roll})}$
Joint velocity	-1e-3	$\boldsymbol{\|\dot{q}\|^2}$
Joint acceleration	-1e-7	$\boldsymbol{\|\ddot{q}\|^2}$
Action rate	-1e-4	$\boldsymbol{\|a_t - a_{t-1}\|^2}$
Action acceleration	-1e-5	$\boldsymbol{\|a_t - 2a_{t-1} + a_{t-2}\|^2}$

3. Results

3.1 Simulation Results

The jumping policy was tested through simulations with various initial conditions, obstacle sizes, and joint frictions to evaluate its effectiveness. Obstacles were 0.5 m high and 0.1 m wide, representing a high difficulty level. We sampled 5000 robots with randomized starting velocities of [-0.5, 0.5], joint angles of [-0.2, 0.2], and joint frictions of [0.6, 0.9]. The x-y position commands were set at [7.5, 1.5] m, requiring the robots to jump over the obstacle to reach the target.

The results show that the robots navigated and jumped over obstacles, as seen in [Fig. 1]. After jumping, they stabilize at average coordinates of x = 6.98 ± 2.11 meters and y = 1.46 ± 0.45 meters, as shown in [Fig. 3]. This variability highlights the significant influence of joint conditions on their jumping precision and base position maintenance.

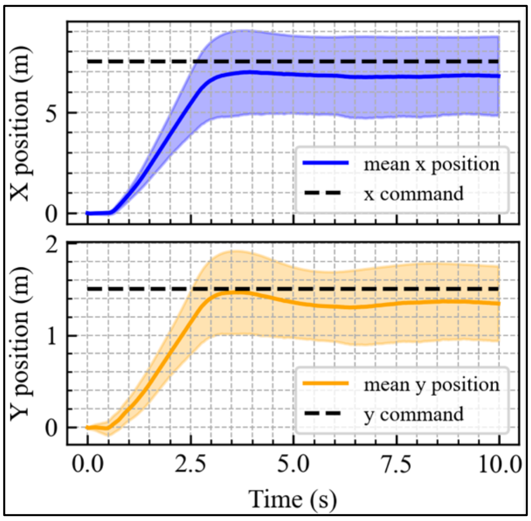

Figure 3. Position tracking performance of 5000 robots with randomized conditions.

REFERENCES

[1] Zhuang, Z., Yao, S., & Zhao, H., "Humanoid Parkour Learning", Conference on Robot Learning (CoRL), 2024.