# SRL: Combining SLIP Model and Reinforcement Learning for Agile Robotic Jumping

Xiaowen Hu[a,1], Linqi Ye[a,*,2], Yudi Zhu[b,3], Chenyue Shao[b,4], Rankun Li[a,5], Qingdu Li[b,6] and Yan Peng[a,7]

[a]*Institute of Artificial Intelligence, Shanghai University, Shangda Road 99, Baoshan District, 200444, Shanghai, China*
[b]*Institute of Machine Intelligence, University of Shanghai for Science and Technology, Jungong Road 516, Yangpu District, 200093, Shanghai, China*

ARTICLE INFO

*Keywords*:
Robotic Jumping
SLIP Model
Reinforcement Learning
Quadruped Robots
Biped Robots

ABSTRACT

Robotic jumping is pivotal in applications such as search and rescue and logistics, where crossing obstacles and enhancing mobility efficiency are critical. The Spring-Loaded Inverted Pendulum (SLIP) model leverages simplified spring–mass dynamics that naturally encode biologically plausible hopping motions, yet its performance degrades on irregular terrain due to idealized assumptions regarding contact and joint dynamics. Meanwhile, Reinforcement Learning (RL) can adapt to diverse and complex environments but often requires extensive data from unguided exploration. The complementary strengths of SLIP's physically grounded baseline and RL's adaptive capabilities motivate a hybrid framework that overcomes these individual limitations. We therefore propose Spring-loaded Reinforcement Learning (SRL), which integrates SLIP-based feedforward control signals with RL-driven real-time feedback, enabling continuous optimization of robotic jumping. Experimental results demonstrate that SRL can achieve more stable jumps with much less training time than the baseline method, maintaining an average position tracking error below 0.1 m and velocity tracking errors within ±3 % of the target values. Through bipedal and quadrupedal simulations of ground and stair jumping, as well as sim-to-sim and sim-to-real validations, SRL exhibits robust adaptability to various task requirements and environmental complexities, underscoring its potential for real-world deployment.

## 1. Introduction

With the growing adoption of robotics in diverse settings—ranging from high-risk rescue missions to warehouse logistics—the ability for robots to perform complex maneuvers, such as jumping[1, 2], has become increasingly critical. Jumping not only enables robots to overcome substantial obstacles, i.e., gullies and elevated platforms, but also significantly enhances mobility and obstacle avoidance in unstructured terrain[3–6].

Contemporary robotic jumping control strategies can be broadly categorized into model-driven methods, such as bio-inspired controllers[7–13], the SLIP model[14–16], and model predictive control (MPC)[17–22]; and data-driven methods, primarily RL[23, 24]. Model-driven methods, particularly SLIP-based methods, efficiently capture energy storage and release dynamics, enabling physically grounded jumping motions. However, their applicability is often constrained by oversimplified assumptions, such as neglecting ground friction and multi-joint leg dynamics, which limits adaptability to real-world, uneven terrains[25–27].

To address these limitations, data-driven methods, such as RL, offer an alternative by autonomously optimizing control policies through interaction with the environment. RL has demonstrated remarkable adaptability across various locomotion tasks. However, its reliance on trial-and-error learning often results in poor sample efficiency and unstable policy exploration due to the lack of physical priors. In robotic jumping, this leads to inefficient motion strategies and increased training costs, hindering real-world deployment.

To bridge this gap, we propose SRL, a hybrid control framework that integrates the SLIP model with RL to achieve both efficiency and adaptability. In the SRL framework, the SLIP model serves as the foundation of feedforward control, leveraging simplified spring–mass dynamics to generate efficient reference trajectories for jumping motions. Additionally, we design an RL-based feedback module, implemented using Proximal Policy Optimization (PPO)[28–30], to generate corrective actions that improve stability and robustness in unstructured terrains. The PPO agent

*Corresponding author
hxw2035143846@shu.edu.cn (X. Hu); yelinqi@shu.edu.cn (L. Ye); 221240082@st.usst.edu.cn (Y. Zhu); 13736413526@139.com (C. Shao); rankunli@shu.edu.cn (R. Li); liqd@usst.edu.cn (Q. Li); pengyan@shu.edu.cn (Y. Peng)

continuously processes environmental observations, including body posture, velocity, and foot placement, and outputs feedback corrections to the SLIP-based reference trajectory. The final control signal is obtained by blending the SLIP-generated feedforward commands with RL-derived feedback corrections through a weighted fusion mechanism, balancing predictive efficiency with adaptive flexibility. To ensure precise motion execution, the output control signal is further fine-tuned by a proportional-derivative (PD) controller, which regulates joint torques for smooth and stable jumping. Through this structured integration of model-driven and data-driven control, SRL achieves robust and efficient robotic jumping, effectively mitigating the limitations of both individual methods.

Our contributions are summarized as follows:

- We propose SRL, a hybrid control framework for robotic jumping that integrates a six-state Finite State Machine (FSM), structured observations, and phase-specific rewards to combine the physical priors of the SLIP model with the adaptability of RL;
- We validate SRL through extensive simulations on both biped and quadruped robots, demonstrating superior learning efficiency and stability over various baseline methods;
- We further validate SRL through sim-to-sim and sim-to-real experiments, achieving superior trajectory tracking accuracy and demonstrating a real-world biped jumping height of 15 cm and forward distance of 20 cm.

## 2. Related Works

The current mainstream methods for achieving robot jumping include several methods. First, bio-inspired control methods mimic animal jumping movements, combined with model predictive techniques to precisely track and optimize trajectories[7–9, 18, 31, 32]. Although this approach can realistically replicate natural jumping behaviors, it heavily relies on high-precision control and sensing technology, making it challenging to implement in complex actions[10–12]. Second, the SLIP model uses a simplified spring and point mass system to effectively capture the energy dynamics of jumping and walking[14–16, 33–36]. Its structure is simple and easy to implement, but due to the model's oversimplification, it struggles to handle precise jump control in various environments[25–27]. MPC optimizes control strategies by predicting future states, allowing robots to perform jumps in complex environments[17–22]. While MPC excels in precise control, it has high computational complexity and demands significant real-time performance. Deep Reinforcement Learning (DRL)[23, 37], through interaction with the environment, can learn optimal strategies in continuous action spaces, adapting to nonlinear and unknown challenges[38–42]. Although DRL has adaptive and self-learning capabilities, the learning process often requires substantial computational resources.

In the aforementioned methods, the SLIP model has garnered favor among researchers due to its simplicity and effectiveness in capturing the key characteristics of leg dynamics[43]. Geyer and Saranli[44] strengthened the application of the SLIP model in biped gait generation through their research. Xie et al.[36] proposed Variable Spring-Loaded Inverted Pendulum Model with Finite-sized Foot (VSLIP-FF) with adjustable leg stiffness and ankle joints to enable compliant bipedal walking. He et al.[26] validated the SLIP model's efficacy in quadruped robot running control, achieving stable high-speed locomotion through comprehensive dynamics analysis.

Despite the SLIP model's effectiveness in simulating basic jumping dynamics, its limitations in adapting to complex environments necessitate the integration of other advanced technologies to enhance the practicality and adaptability of robotic systems. In recent years, RL has emerged as a key technology for completing complex tasks in the field of robotics, particularly demonstrating unique advantages in scenarios where specific instructions are not predetermined. Through real-time interaction with the environment and a trial-and-error process, RL enables robots to explore and learn optimal or near-optimal decision-making strategies. This versatility has allowed RL to find broad applications in various fields such as robotic arm control and autonomous navigation[23, 24]. Additionally, through the implementation of the Deep Deterministic Policy Gradient (DDPG) algorithm, RL has demonstrated significant potential in continuous action spaces, effectively facilitating the generation of complex actions such as jumping[38, 39]. Moreover, RL has gradually revealed its potential in enabling robotic jumping capabilities[24, 37].

To the best of our knowledge, despite the significant progress made in various domains, the integration of the SLIP model with RL to achieve autonomous and efficient jumping in more complex environments remains an underexplored area.

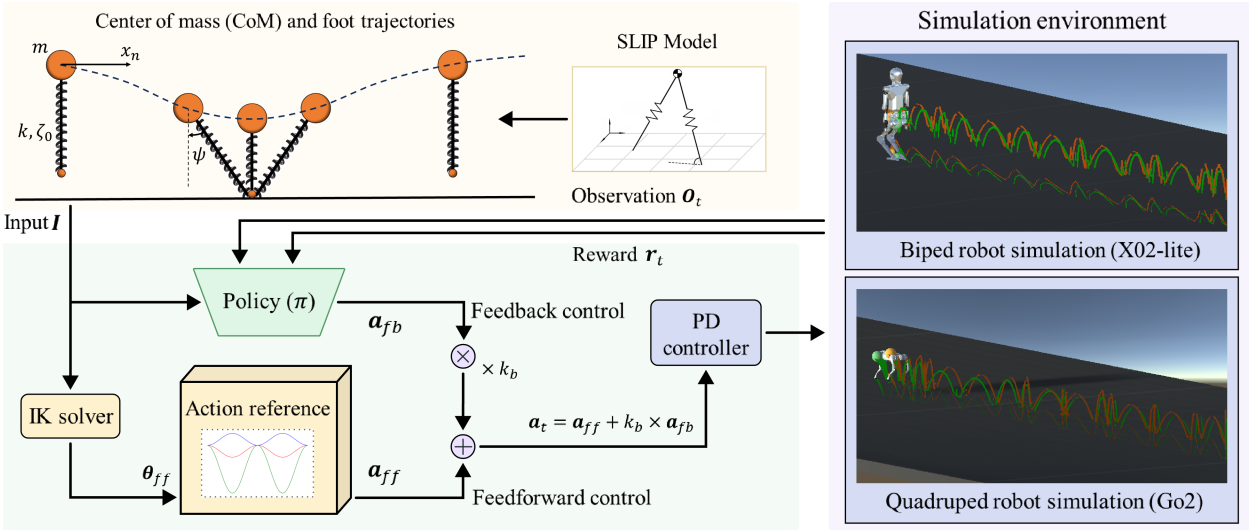



**Figure 1:** Overview of the SRL control framework for robot jumping tasks, integrating the SLIP model, RL, and simulation environment. SRL combines the physically grounded motion dynamics of the SLIP model with the adaptability of RL to optimize jumping performance in complex environments such as flat ground with varying disturbances, stairs, and boxes.

# 3. Methods

## 3.1. SRL Structure

This study proposes a novel robot jumping control framework, SRL, which integrates the SLIP model with RL, as shown in Fig. 1. SRL consists of three core components: 1) the SLIP model, 2) the RL module, and 3) the simulation environment. The SLIP model is responsible for generating feedforward control signals, which are then translated into specific motor commands through an Inverse Kinematics (IK) solver. The RL module continuously refines the control policy by processing environmental observations and reward signals, generating real-time feedback control. The final control signal is a weighted combination of the feedforward and feedback outputs, which is further fine-tuned by a PD controller to achieve precise motion. SRL has been validated in simulations for biped and quadruped robots as well as in selected real-world environments, demonstrating its adaptability and great performance in a variety of jumping tasks.

## 3.2. SLIP Model

The SLIP model, illustrated in the upper left corner of Fig. 1, consists of a point mass connected to a single spring, representing a simplified virtual leg. This model emulates legged locomotion through a dynamic mass-spring system. We developed a six-state FSM (Fig. 2) that achieves stable jumping patterns through state-dependent parameter modulation, including dynamic adjustment of spring stiffness, damping coefficients, and rest length across different locomotion phases. The forces on the CoM in each state can be expressed by the equation:

$$F_i = -k_i \Delta L - c_i v_m, \tag{1}$$

where $i \in \{-1, 0, 1, 2, 3, 4\}$ denotes the state machine mode, $\Delta L = \left(\|\boldsymbol{p}_m - \boldsymbol{p}_f\| - \|\boldsymbol{p}_{m0} - \boldsymbol{p}_{f0}\|\right)$ represents the change in the distance between the CoM and the foot relative to its initial value, $v_m$ is the velocity of the CoM, $k_i$ and $c_i$ are the state-dependent stiffness and damping coefficients, respectively.

### *3.2.1. Initialization Phase (State -1)*

In the initialization phase, the robot sets the initial positions of the Center of Mass ($\boldsymbol{p}_m$) and the foot ($\boldsymbol{p}_f$), and calculates the initial force $F_{-1}$ based on the displacement between $\boldsymbol{p}_m$ and $\boldsymbol{p}_f$. Once initialization is complete, the state transitions to the next phase.

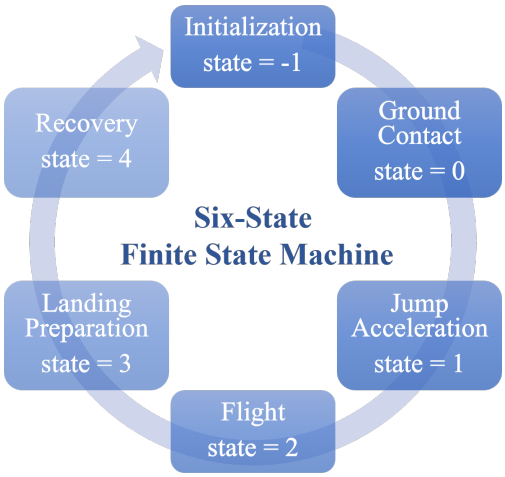



**Figure 2:** A six-state FSM constructed based on the SLIP model, where each state represents a key stage in the jump cycle.

#### *3.2.2. Ground Contact Phase (State 0)*

In the ground contact phase, the robot's foot makes contact with the ground, and the leg compresses like a spring, generating force $F_0$ to push the CoM upward. The magnitude of the force is determined by the displacement $\Delta L$ between the CoM and the foot, and the unit direction vector $\hat{\boldsymbol{n}}$. When the displacement $\Delta L$ becomes sufficiently small and the vertical component of the velocity of the CoM $(\boldsymbol{v}_m)_y > 0$, the state transitions to the jump acceleration phase.

#### *3.2.3. Jump Acceleration Phase (State 1)*

In this phase, the robot further increases the force $F_1$ to accelerate the CoM upward, ensuring the robot can leave the ground and enter the flight phase. This phase is primarily responsible for providing the robot with sufficient kinetic energy to complete the jump. When the displacement $\Delta L$ exceeds a threshold, the state transitions to the flight phase.

#### *3.2.4. Flight Phase (State 2)*

In the flight phase, the robot leaves the ground and enters free motion. No external forces are applied during this phase. The motion of the CoM $(\boldsymbol{p}_m)$ is dominated by inertia, while the foot position $(\boldsymbol{p}_f)$ is adjusted based on the previous relative displacement to ensure the correct landing position. When $(\boldsymbol{v}_m)_y < 0$ and the foot approaches the ground, the state transitions to the next phase. No force is applied during this phase.

#### *3.2.5. Landing Preparation Phase (State 3)*

In this phase, the robot begins to gradually contact the ground, applying a cushioning force $F_3$ to slow down the descent of the CoM. When the vertical velocity of the CoM $(\boldsymbol{v}_m)_y$ starts to rise, the robot transitions to the recovery phase. This phase applies a smaller force to cushion the landing.

#### *3.2.6. Recovery Phase (State 4)*

In the recovery phase, the robot re-establishes contact with the ground, and the force is calculated based on the initial displacement to restore the CoM and foot to their original relative positions. When the positions of the CoM $(\boldsymbol{p}_m)$ and the foot $(\boldsymbol{p}_f)$ approach the initial state, the state transitions back to State -1, initiating a new jumping cycle.

### 3.3. Robot IK

To apply the theoretical SLIP model to an actual robot, we employ an analytical IK method to determine the optimal angles for each joint. Taking a biped robot as an example, the target positions of the robot's CoM $(\boldsymbol{p}_m)$ and the foot $(\boldsymbol{p}_f)$ are obtained according to the SLIP model, from which the corresponding angles of its hip, knee, and ankle joints are calculated.

First, the distance $d_1$ between the CoM and the foot is calculated, and the hip joint angle is determined using trigonometric relationships:

$$dc_1 = \arccos\left(\frac{L_1^2 + d_1^2 - L_2^2}{2L_1 d_1}\right) + \arcsin\left(\frac{(\boldsymbol{p}_m - \boldsymbol{p}_f)_z}{d_1}\right), \tag{2}$$

where $L_1$ and $L_2$ are the lengths of the thigh and shank, respectively, and $(\boldsymbol{p}_m - \boldsymbol{p}_f)_z$ denotes the displacement between the center of mass (CoM) and the foot along the $z$-axis.

Next, the knee joint angle $dc_2$ is computed using the following equation:

$$dc_2 = -\arccos\left(\frac{L_1^2 + L_2^2 - d_1^2}{2L_1 L_2}\right) - dc_1. \tag{3}$$

Finally, the ankle joint angle $dc_3$ is determined by the sum of the hip and knee joint angles:

$$dc_3 = dc_2 + dc_1. \tag{4}$$

Through these equations, the IK method enables fast and accurate adjustments to the biped robot's joint angles, ensuring the correct positioning and posture of the foot during the jump.

### 3.4. RL

In our previous work[45], we proposed a learning framework named Instructional Learning, which integrates feedforward and feedback control mechanisms inspired by human learning processes. This method has proven to be efficient, flexible, and generalizable in robot motion learning. In the current research, we build upon this "Instructional Learning" method to design an extended control system architecture, integrating it with RL — specifically, PPO [28–30] — to optimize the robot's jumping tasks. PPO is a widely-used policy gradient algorithm particularly suited for continuous control tasks such as robotic locomotion. By employing a clipped surrogate objective function, it ensures stable policy updates, preventing overly aggressive changes that could destabilize learning. This enables improved convergence and robust performance in complex environments.

**Feedforward Signal Design:** The feedforward signal in our system is derived from the SLIP model, denoted as $\boldsymbol{a}_{ff}(t) = \boldsymbol{f}(SLIP, t)$, and serves as an initial reference trajectory for the controller. Specifically, the joint angles obtained from IK solver are used as feedforward inputs to the neural network. For the biped robot, this includes the pitch angles of the hip, knee, and ankle joints. In the case of the quadruped robot, only the hip and knee joint angles are used, as the design does not include ankle joints.

**Action Space:** The robot's actions are generated by combining the feedforward control signal with the feedback control signal produced by the RL algorithm. At each time step, the joint control signal is calculated. The weighted sum of the feedback signal $\boldsymbol{a}_{fb}$ and the feedforward signal $\boldsymbol{a}_{ff}$ forms the final target joint angle configuration: $\boldsymbol{a}_t = \boldsymbol{a}_{ff} + k_b \times \boldsymbol{a}_{fb}$. Finally, a PD controller is used to convert this target angle into joint torques, ensuring the precise execution of the jumping motion.

**State Space:** In our state space $\boldsymbol{O}_t$, the agent observes key dynamic parameters of the robot during the jumping process, enabling precise adjustment of its movements. The state space includes the following observations: the robot's body linear velocity $\boldsymbol{v}_{base}$ and angular velocity $\boldsymbol{\omega}_{base}$ in the local coordinate frame, as well as the pitch angle $\theta_{pitch}$ and roll angle $\theta_{roll}$. Additionally, the joint angle deviations $\Delta a_1, \Delta a_2, \ldots, \Delta a_n$, which represent the difference between the feedforward signal (the target angles from the SLIP model) and the actual joint angles, are also included. Joint velocities $v_1, v_2, \ldots, v_n$ are further observed to monitor the dynamic state of the joints. These observations provide the agent with a comprehensive understanding of the robot's posture and movement, allowing it to adjust actions dynamically to maintain stability and precision during complex jumping tasks.

**Reward:** In our robot jumping task, the reward function is crucial to ensure precise and stable movement control. The specific implementation of the reward function is described through the following components:

1) **Live Reward:** This reward ensures that the robot remains active during each time step, which is essential for completing long-term jumping tasks.

$$r_{live} = 1. \tag{5}$$

**Table 1**
Key Control and Reward Parameters of the Proposed SRL Framework

| Parameter | Biped | Quadruped | Reward Weight | Early Stage | Late Stage |
|---|---|---|---|---|---|
| $k_b$ | 30 | [10, 30, 60]† | $w_1$ | 2 | 1 |
| $k_p$ | 0.1 | 0.02 | $w_2$ | 0.5 | 1 |
| $k_y$ | 0.5 | 2 | $w_3$ | 0 | 1 |
| $k_r$ | 0.1 | 0.02 | $w_4$ | 1 | 1 |

† For the quadruped robot, per-leg feedback gains are 10, 30, and 60 for the hip, thigh, and knee joints, respectively.

2) **Orientation Reward:** This reward penalizes deviations in pitch, yaw, and roll angles to ensure the robot's stability in mid-air and upon landing.

$$r_{ori} = -k_p \cdot |\theta_{pitch}| - k_y \cdot |\theta_{yaw}| - k_r \cdot |\theta_{roll}|, \tag{6}$$

where $\theta_{pitch}$ is the pitch angle deviation, $\theta_{yaw}$ is the yaw angle deviation, $\theta_{roll}$ is the roll angle deviation, $k_p$, $k_y$, $k_r$ are coefficients for the pitch, yaw, and roll angles, respectively.

3) **Velocity Reward:** This reward ensures the robot maintains smooth motion during jumps by penalizing deviations between its actual and expected speed.

$$r_{vel} = -\|\boldsymbol{v}_{actual} - \boldsymbol{v}_{expected}\|, \tag{7}$$

where $\boldsymbol{v}_{actual}$ is the robot's current velocity and $\boldsymbol{v}_{expected}$ is the expected velocity for the task.

4) **Synchronization Reward:** This reward penalizes differences in joint angles between the legs, ensuring synchronized movement.

For biped robots:

$$r_{syn} = -\sum_{i=1}^{N} |a_i^{left} - a_i^{right}|. \tag{8}$$

For quadruped robots:

$$r_{syn} = -\sum_{i=1}^{N} \left( |a_i^{FL} - a_i^{FR}| + |a_i^{RL} - a_i^{RR}| \right). \tag{9}$$

5) **Foot Position Tracking Reward:** This reward minimizes the deviation of the robot's foot position from the target position, ensuring that each step follows the predetermined trajectory.

$$r_{pos} = -\sum_{j=1}^{M} \|\boldsymbol{p}_j^{actual} - \boldsymbol{p}_j^{target}\|, \tag{10}$$

where $\boldsymbol{p}_j^{actual}$ is the actual foot position of leg $j$, and $\boldsymbol{p}_j^{target}$ is the target foot position.

6) **Total Reward:** The total reward is a weighted combination of all the individual rewards.

$$r_{total} = r_{live} + w_1 r_{ori} + w_2 r_{vel} + w_3 r_{syn} + w_4 r_{pos}, \tag{11}$$

where $w_1$, $w_2$, $w_3$, and $w_4$ are the weight coefficients for the orientation, velocity, synchronization, and foot position rewards, respectively. The control parameters and reward weights are summarized in Table 1.

Termination Condition: We terminate the episode if the robot "falls", defined as either the pitch angle $\theta_{pitch}$ or the roll angle $\theta_{roll}$ exceeding 30 degrees. The episode also ends if the time step reaches 1000.

**Network and Hyperparameters:** The network architecture and training hyperparameters used by the PPO algorithm are shown in Table 2 and Table 3, respectively.

**Table 2**
Network Architecture for Biped and Quadruped Robots

| Hyperparameter | Value |
|---|---|
| Actor Hidden Layer | [512, 512, 512] |
| Critic Hidden Layer | [512, 512, 512] |
| Input Observation Dimension | 28 / 32 |
| Output Action Dimension | 10 / 12 |
| Activation Function | ReLU |

**Table 3**
PPO Hyperparameters for Biped and Quadruped Robots

| Hyperparameter | Value |
|---|---|
| Discount Factor | 0.995 |
| GAE Discount Factor | 0.95 |
| Minibatch Size | 2024 / 2048 |
| Timesteps per Rollout | 20240 / 20480 |
| Epochs per Rollout | 3 |
| Entropy Bonus Coefficient | 0.005 |
| Value Loss Coefficient | 1.0 |
| Clip Range | 0.2 |
| Learning Rate | 0.0003 |
| Environments | 32 |

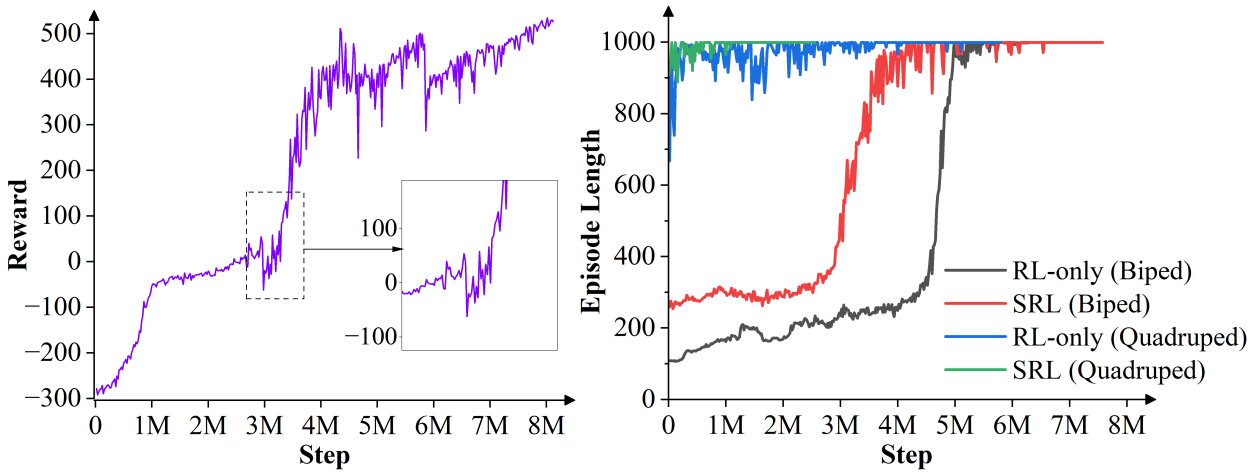



**Figure 3:** Left: Reward evolution for the biped robot's random-distance jump; Right: Learning efficiency comparison of RL-only and SRL in fixed-distance jumping.

## 4. Experiments

### 4.1. Simulation Setup

In this study, we use the Unity engine and ML-Agents toolkit for simulation and training experiments. The simulation runs with a time step of 0.01 seconds, corresponding to a control frequency of 100Hz. To accelerate the training process, we deploy 32 agent instances in parallel, ensuring efficient policy learning.

We selected two types of robots to validate our approach: the X02-lite biped robot, developed by Shanghai Droid Robotics, and the Unitree Go2 quadruped robot, developed by Hangzhou Unitree Technology. These robots represent typical biped and quadruped designs, making them well-suited to evaluate the generalization of SRL.

### 4.2. Training Process

#### *4.2.1. Phase-Based Training Strategy*

The training process is structured in phases to gradually shift the focus from stability to speed and efficiency. For instance, during the training process for the biped robot's random-distance jumping task, the reward function's weight parameters $w_1$, $w_2$, $w_3$, and $w_4$ are adjusted across different stages, as shown in the left part of Fig.3 and summarized in Table 1. Initially, the emphasis is placed on stability by assigning a higher weight to the orientation reward. As training progresses, the reward weights are gradually adjusted to prioritize velocity over orientation, aiming to improve the robot's speed and jumping efficiency while maintaining basic stability. This staged reward shaping strategy enables the policy to transition from stable locomotion to speed-optimized performance across diverse jumping distances.

**Table 4**
Baseline Comparison and Ablation Study

| Method | Training Steps ($10^6$) | | Success Rate | |
|---|---|---|---|---|
| | Biped | Quadruped | Biped | Quadruped |
| **(a) Baseline** | | | | |
| SLIP-based MPC | N/A | N/A | 0.680 | 0.752 |
| RL-only (PPO) | 7.2 | 3.8 | 0.913 | 0.957 |
| Ours (SRL) | **5.7** | **1.1** | **0.985** | **0.998** |
| **(b) Ablations** | | | | |
| Ours | 5.7 | 1.1 | 0.985 | 0.998 |
| w/o FSM Structure | 7.5 | 3.8 | 0.803 | 0.886 |
| w/o Velocity Reward | 6.2 | 1.6 | 0.895 | 0.912 |

#### *4.2.2. Jumping Tasks Design*

We conducted fixed-distance and random-distance jumping tasks with biped and quadruped robots. Fixed-distance tasks (e.g., 0.1 m, 0.2 m) steadily improved strategy performance with continuous goal training, while random-distance tasks (0-0.3 m for biped robots) may require sudden adaptation to long-distance jumps (e.g., 0.3 m) after consecutive short-distance jumps, thus rigorously testing the dynamic adaptability of the strategies. Biped training prioritises balance control during jumps because of their high CoM and the need to simultaneously maintain stability and adjust motion during distance changes, in contrast to quadruped robots that exhibit higher stability, which motivated us to perform more challenging tasks for quadruped robots. Therefore we conducted the task of jumping onto a box, which introduces vertical height variations alongside horizontal jumps, comprehensively validating the robustness of the SRL framework in multidimensional motion scenarios.

### 4.3. Performance Evaluation

#### *4.3.1. Baseline Comparison*

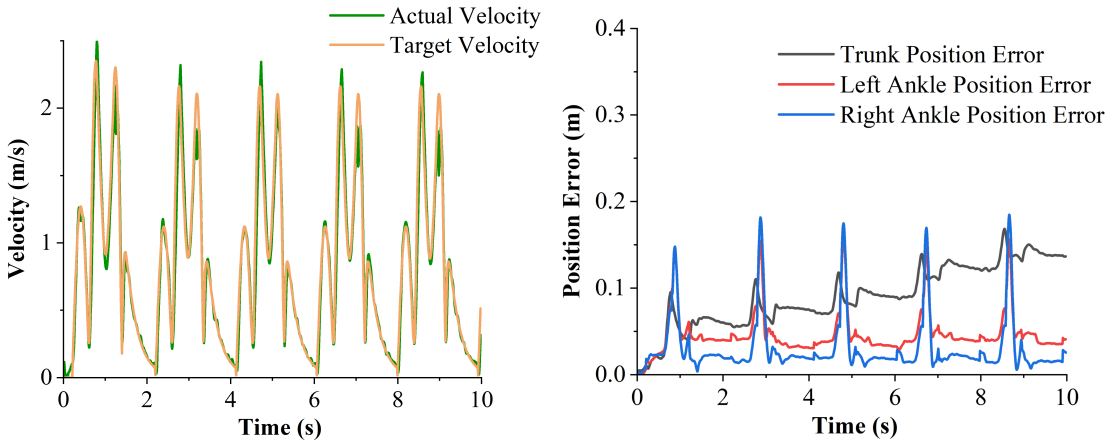
Table 4 presents a comprehensive comparison of the learning efficiency across three different control methods: SLIP-based MPC, RL-only(PPO), and Our SRL. Due to fundamental differences in reward structures, a direct comparison of reward values between SRL and RL-only methods is not meaningful. Therefore, we focus on higher-level performance metrics — namely, training steps and success rate — to enable a more valid and insightful evaluation of learning efficiency. Training steps refer to the total number of learning updates necessary to reach a converged policy, defined as consistently achieving episode lengths exceeding 950 and stable reward levels, under a maximum episode length of 1000. Success rate is calculated as the ratio of successful jumps without falling over 1000 test trials (e.g., a success rate of 0.998 corresponds to 998 successful trials).

The SLIP-based MPC method achieves moderate performance with success rates of 68.0% for the biped and 75.2% for the quadruped. While this demonstrates the effectiveness of incorporating biomechanical priors into control, the lack of online learning limits its robustness and adaptability. In contrast, the RL-only method demonstrates significantly improved performance, achieving high success rates of 91.3% for biped and 95.7% for quadruped, albeit at the cost of substantial training time (7.2M and 3.8M steps respectively). Our proposed SRL method further improves upon these results, attaining the highest success rates — 98.5% for the biped and 99.8% for the quadruped — while requiring notably fewer training steps (5.7M and 1.1M steps respectively), as shown in the right part of Fig. 3. This reduction in training complexity suggests that the integration of analytical priors with RL not only enhances performance but also accelerates learning convergence significantly.

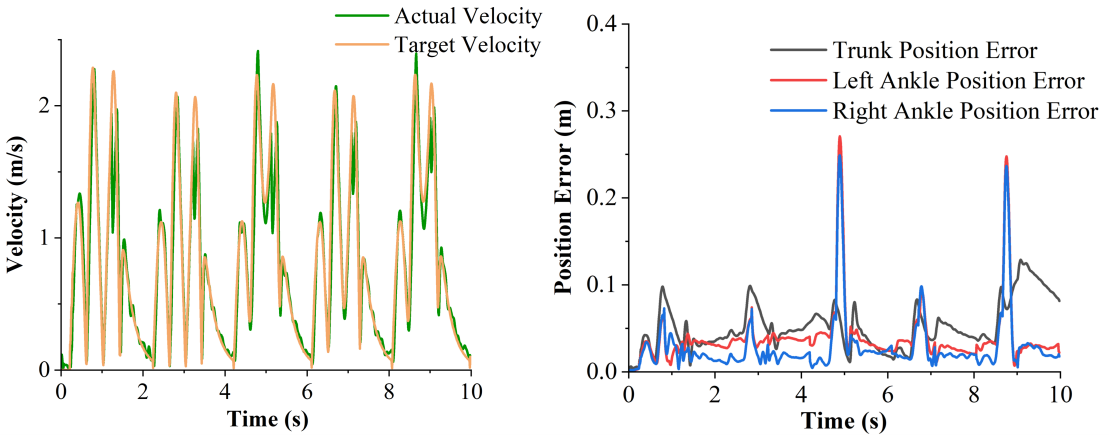
#### *4.3.2. Ablation Study*

To verify the contribution of each core component in the SRL architecture, we conduct ablation studies by removing the six-state FSM structure and the velocity reward. All variants are trained and evaluated under identical experimental settings. As shown in Table 4, removing either component leads to increased training steps and degraded success rates, highlighting the importance of both phase-aware control and velocity tracking for efficient and stable jumping.

In the "w/o FSM" setting, the discrete phase-transition mechanism is disabled. The jumping cycle is instead generated by a predefined cosine-based oscillator, producing a smooth periodic signal with fixed period and continuous




(a) Fixed-distance jumping.




(b) Random-distance jumping.

**Figure 4:** Performance of the biped robot during fixed-distance jumping, showing trunk velocity, absolute tracking errors of the trunk position, and relative tracking errors of the ankle.

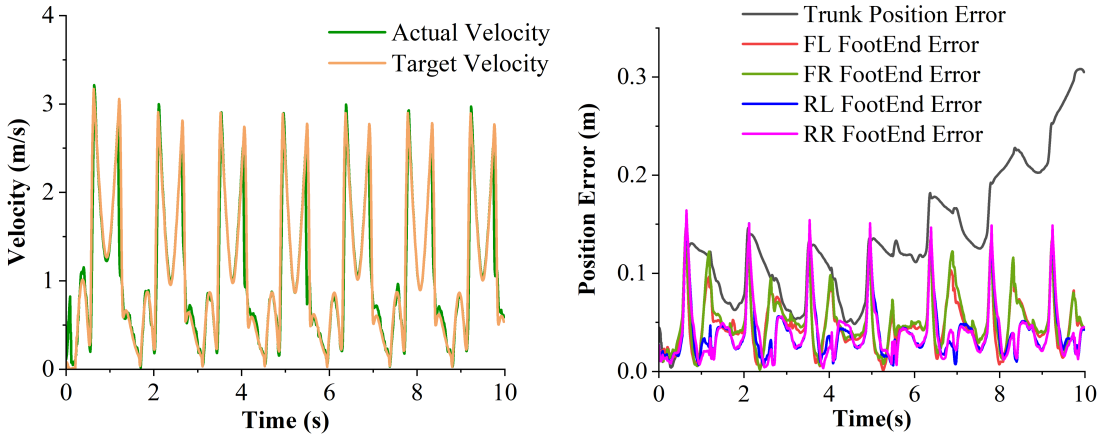
feedforward joint references for bounding locomotion. This time-driven formulation replaces state-dependent phase switching with a fixed-frequency rhythmic pattern. Without adaptive phase modulation, the controller relies on a fixed rhythmic signal, which degrades phase coordination and locomotion stability, resulting in inferior jumping performance.

Removing the velocity reward slows convergence and reduces stability, indicating that velocity tracking provides essential guidance for robust control.

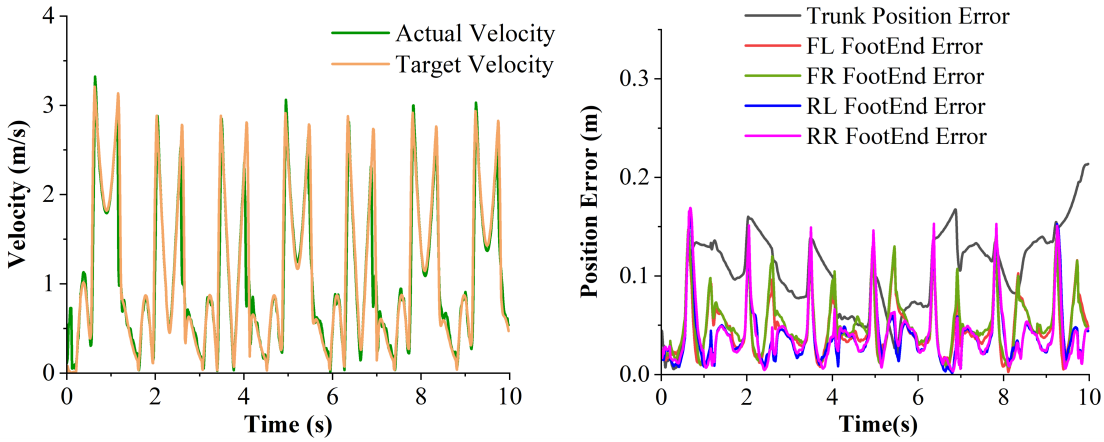
Quantitatively, removing the FSM structure increases training steps from 5.7M to 7.5M and reduces success rate from 98.5% to 80.3%, while removing the velocity reward leads to 6.2M training steps and 89.5% success rate. These results demonstrate that both phase-aware control and velocity tracking significantly improve sample efficiency and final performance, fully validating the proposed SRL framework.

## 4.4. Simulation Results

**Biped Robot :** As illustrated in Fig. 4, the performance of the biped robot during both fixed-distance and random-distance jumping tasks is presented. The figure shows the comparison between the robot's body velocity and the target velocity, as well as the tracking errors of the body and foot-effectors. In the fixed-distance jumping task, the robot effectively tracked the target velocity, reaching a peak speed of 2 m/s. However, over time, the position tracking errors began to accumulate. Conversely, during the random-distance jumping task, the variation in jump distances allowed the robot to correct its position-tracking errors more easily, though this resulted in a slight reduction in position-tracking




(a) Fixed-distance jumping.




(b) Random-distance jumping.

**Figure 5:** Performance of the quadruped robot during fixed-distance and random-distance jumping tasks, showing trunk velocity, absolute tracking errors of the trunk position, and relative tracking errors of the foot.

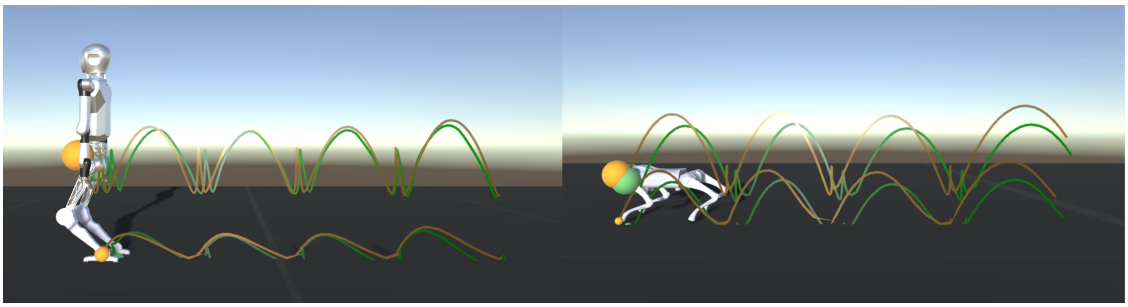


**Figure 6:** Fixed-distance jumping simulations of the biped and quadruped robots. Orange and green denote the target and actual states, respectively.

accuracy. Overall, across both tasks, the biped robot demonstrated strong capability in tracking the target velocity, keeping body tracking errors within 0.2 m over a 10-second period.

Although the foot-effectors occasionally exhibited periodic error spikes during jumping, analysis showed that these peaks primarily occurred at the moment of takeoff and landing, especially immediately after takeoff, when the robot's

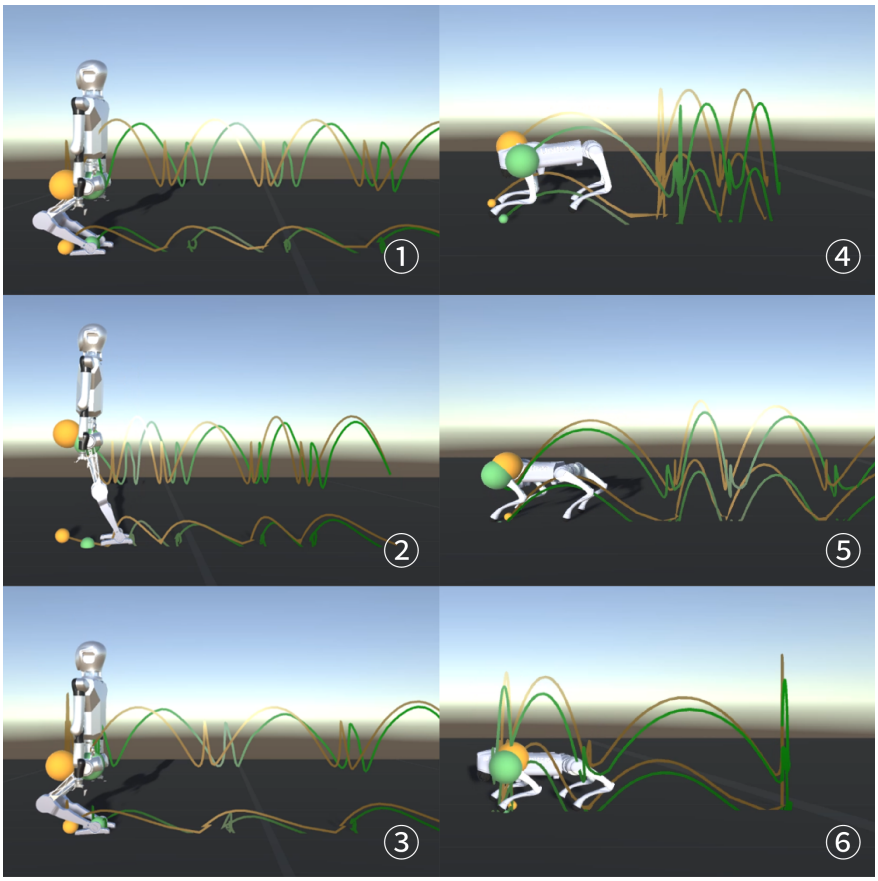



**Figure 7:** Simulation of the biped (left) and quadruped (right) robot in random-distance jumping tasks, illustrating the motion trajectories of the body and ankle/foot.

leg joints experienced maximum explosive force and its body velocity reached its peak. Due to the system's increased dynamic sensitivity, even small position errors were amplified. Despite the relatively large error spikes shown in the graph, these errors were transient in nature (lasting less than 100 milliseconds), occurring during the flight phase without ground contact and not leading to long-term instability. Rapid adjustments during other phases successfully compensated on average for these errors, maintaining errors within the range of 0 to 0.1 m. Similar phenomena were observed with quadruped robots performing jumping tasks (Fig. 5).

Fig. 6 illustrates the body and ankle trajectories of the biped robot during the fixed-distance jumping task, clearly presenting the robot's movement patterns throughout the task. Fig. 7 demonstrates the performance of the biped robot in random-distance jumping tasks. The figure provides a clear visualization of the robot's ability to flexibly adapt to varying jump distances, showcasing its remarkable agility and jumping capabilities. This dynamic adaptability to varied commands further demonstrates the effectiveness and precision of SRL in handling complex and variable conditions.

**Quadruped Robot :** Compared to the biped robot, the quadruped robot was able to complete a greater number of jumps within the same time frame and achieved higher body velocities, with a maximum speed of 3 m/s. Similar to the biped case, the quadruped robot demonstrated better velocity-tracking ability during the fixed-distance jumping task but also exhibited an accumulation of position-tracking errors over time. Detailed results can be seen in Fig. 5.

The simulation results of the quadruped robot in fixed-distance jumping tasks are shown in Fig.6. In the random-distance jumping tasks (as shown in Fig. 7), the robot similarly demonstrates the ability to adapt flexibly to varying jump distances, exhibiting excellent stability and jumping efficiency.

In the box-jumping experiment, the quadruped robot performed great (as shown in Fig. 8). Although the absolute position errors of the body increased over time due to the complexity of synchronizing the four legs to adapt to uneven

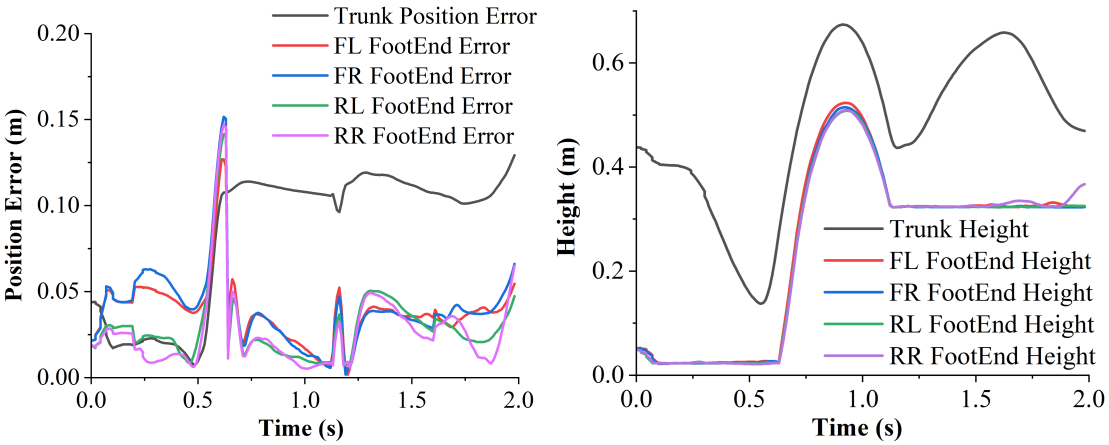



**Figure 8:** Performance of the quadruped robot during the box-jumping experiment. The left shows absolute body position errors and relative foot errors, while the right shows the height of the body and feet.

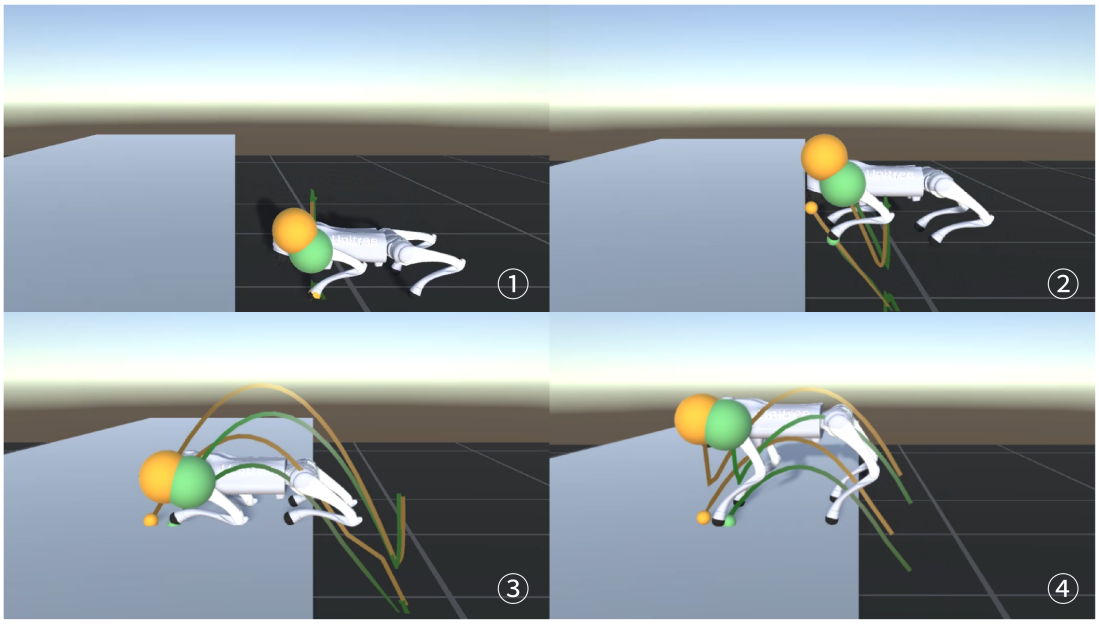



**Figure 9:** Simulation of the quadruped robot during the box-jumping experiment.

terrain, the robot was still able to maintain the relative foot errors mostly within 0.05 m. Moreover, the height (y-coordinate) of the body and feet during the jumping process indicates that the robot successfully adjusted its posture and extended its legs to manage the vertical displacement when stepping up. The trajectories of the four legs were largely synchronized, and the figure clearly shows that the robot was able to successfully jump onto a box with a height of 0.35 m. Fig. 9 shows the simulation with the body and foot trajectories during the box-jumping process, providing a clear visualization of the robot's coordination and movement patterns throughout the task.

The above results confirm the effectiveness of our SRL framework across various locomotion tasks and platforms. It is worth noting that the current policies are trained separately for each specific task, given the distinct control demands across different jumping scenarios. Exploring a unified policy that generalizes across diverse tasks through high-level commands would be an important direction for future work.

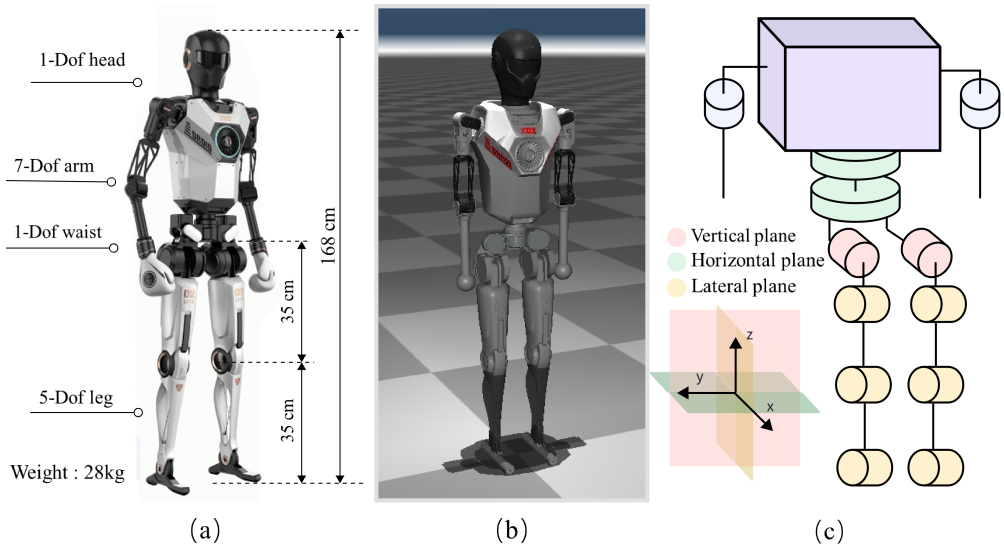



**Figure 10:** The X02-lite humanoid robot (Shanghai Droid Robotics) used in the experiment, featuring a height of 168 cm and a total mass of 28 kg. (a) Real robot. (b) Mujoco model. (c) Simplified link model (focusing on the 10 DOF legs).

**Table 5**
Domain Randomization Parameters

| Parameter | Range[Min,Max] |
|---|---|
| Base Mass | [-3, 3] kg |
| CoM | [-0.15, 0.15] m |
| Ground Friction | [0.5, 1.5] |
| Restitution | [0.2, 1.0] |
| Push Interval | 4 s |
| Push Velocity (XY) | [0, 0.5] m/s |
| Push Angular Velocity | [0, 0.4] rad/s |
| Motor Strength | [0.9, 1.2] × default Nm |
| Motor Offset | [-0.05, 0.05] rad × default rad |
| Joint Kp | [0.8, 1.3]× default |
| Joint Kd | [0.5, 1.5]× default |
| Gravity Interval | 7 s |
| Gravity Impulse Duration | 1.0 s |
| Gravity Range | [-0.5, 0.5] × default m/s² |

## 4.5. Real-World Experiments

To further validate the proposed SRL framework, we conducted real-world experiments using a biped robot platform and compared the results with our simulation outcomes. The physical robot used in our experiments is the X02-lite biped robot (Fig. 10).

X02-lite has a total height of 168 cm and an arm span of 167 cm, closely resembling average human proportions. It weighs 28 kg and features a lower body with two 35 cm leg segments — thigh and shank. The system includes 26 Degrees of Freedom (DOF), with each leg having 5 independently actuated joints, including a knee joint capable of delivering up to 180 Nm of peak torque. Joint positions are measured using a dual-encoder configuration, providing high-precision feedback for motion control. Low-level control runs at 1 kHz for fast and responsive joint dynamics. Environmental perception is achieved through a stereo vision system and a YIS320 IMU, which combines triaxial MEMS accelerometers and gyroscopes for real-time 3D orientation estimation. To better utilize these sensor measurements and estimate linear velocity, we implemented a Contact-assisted Invariant Extended Kalman Filter (CI-EKF) framework[46]. This is a symmetry-aware observer design based on Lie group theory and has been shown to

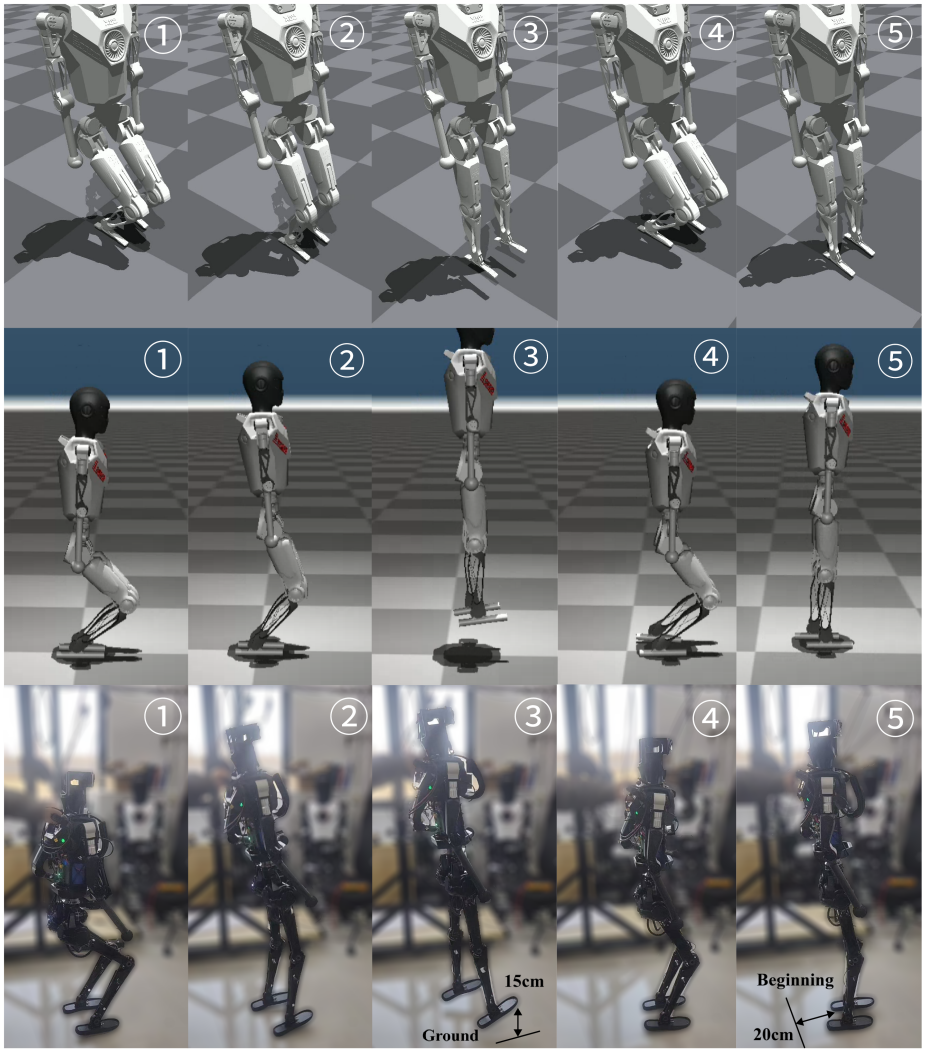



**Figure 11:** Performance comparison of the training policy in simulation (Isaac Gym and Mujoco) and real-world experiments, demonstrating successful transfer. The robot achieved a maximum jump height of 15 cm, distance of 20 cm, and peak velocity of 2 m/s.

provide superior estimation performance compared to standard quaternion-based EKFs. The estimated velocity serves as input to the RL policy during real robot deployment.

In the training and simulation phase, to improve the transferability of the learned policy to real-world scenarios, we utilized the Humanoid Gym framework[47] and adopted the Isaac Gym and Mujoco platforms. The training pipeline remained consistent with the Unity-based simulation, incorporating the same feedforward control, observation space, and reward function. Domain randomization was applied throughout the training process to various physical parameters to improve the policy's robustness to environmental variations and hardware imperfections. The specific domain randomization parameters are summarized in Table 5.

The training process began with training the policy in Isaac Gym. We further optimized the policy by adjusting the domain randomization parameters, transferred it to MuJoCo to evaluate its performance, and then deployed it on the X02-lite robot through a sim-to-real transition. Fig. 11 shows the results from Isaac Gym, MuJoCo, and the real world.

In real-world tests, the robot successfully executed the jumping task, closely replicating simulation performance. During the experiments, the robot achieved a maximum jump height of 15 cm, a single jump distance of 20 cm, and a peak velocity of 2 m/s.

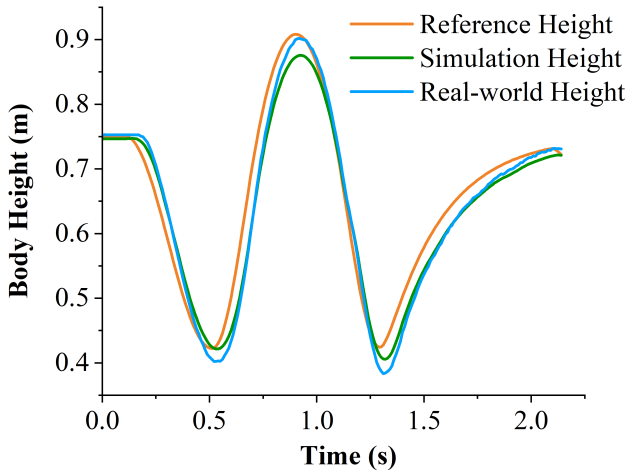



**Figure 12:** Average body height comparison over one gait cycle: reference vs. simulation vs. real-world.

**Table 6**
Average key performance metrics over one cycle under different setups: reference, simulation, and real-world.

| Setup | Peak Velocity (m/s) | Jump Height (m) | Flight Time (s) |
|---|---|---|---|
| Reference | 1.998 | 0.153 | 0.718 |
| Simulation | 2.011 | 0.127 | 0.723 |
| Real-world | 2.003 | 0.150 | 0.721 |

To further quantify the sim-to-real performance, we compare the robot's average base height over a single gait cycle across three setups: reference, simulation, and real-world — as shown in Fig. 12. Although the overall locomotion pattern was successfully transferred, minor deviations in amplitude and phase were observed during flight and contact transitions. These discrepancies may stem from a combination of hardware limitations, sensor noise, and estimation inaccuracies — particularly during the flight phase, the linear velocity estimates provided by the CI-EKF drift, which can affect the policy's perception of the robot's state and lead to slight tracking errors.

Table 6 summarizes the average key performance metrics of the robot's jumps over one gait cycle under three different settings. Peak velocity exhibited nearly identical values across all setups ($\approx 2.0$ m/s). The simulated jump height was approximately 15% lower than the real test (0.127 m vs. 0.150 m). This discrepancy may be due to limitations of the simulation environment, such as the simplified contact model. Flight time remained relatively consistent across all settings ($\approx 0.72$ s), indicating that the duration of the flight phase was well preserved, even in simulation.

The results suggest promising transfer performance to real-world scenarios, warranting further investigation into practical deployment.

## 5. Conclusion

This paper proposes SRL, a control framework that integrates the SLIP model with RL to optimize robot jumping performance. The SLIP model generates fundamental jumping dynamics, while RL enhances environmental adaptability. The framework's effectiveness is validated through fixed-distance and random-distance jumping tasks on both biped and quadruped robots. Simulations and real-world experiments demonstrate centimeter-level localization accuracy (average trajectory tracking error $< 10$ cm), velocity tracking errors within $\pm 3$ % of the target values, and the quadruped's capability to ascend 0.35 m high stairs. Due to safety and actuator constraints of the quadruped hardware platform, real-world validation on quadruped robots is not included in this work.

Current limitations include error accumulation in prolonged tasks and unstable hardware deployment. Furthermore, although SRL theoretically supports generating arbitrary 3D trajectories and then performing 3D jumping via RL, we currently limit our experiments to 2D due to hardware security concerns. Future work will focus on improving sustained

stability, developing efficient sim-to-real strategies, and extending the application to 3D jumping. The proposed SRL framework offers an efficient and adaptable solution for dynamic robot control research.

## Declarations

**Funding** This work was supported by the Science and Technology Commission of Shanghai Municipality (24511103304).

**Availability of Data and Material** All relevant data are included within this paper. The implementation code and supporting materials associated with the proposed SRL framework are publicly available at `https://github.com/Kr1shu01/SRL-Locomotion`.

**Conflict of Interest** The authors declare that they have no competing interests.

**Ethics Approval and Consent to Participate** Not applicable.

**Consent for Publication** Not applicable.

## CRediT authorship contribution statement

**Xiaowen Hu:** Conceptualization, Methodology, Software, Formal Analysis, Investigation, Validation, Writing – Original Draft. **Linqi Ye:** Conceptualization, Supervision, Funding Acquisition, Resources, Project Administration, Writing – Review & Editing. **Yudi Zhu:** Software, Validation, Data Curation. **Chenyue Shao:** Methodology, Software, Formal Analysis. **Rankun Li:** Investigation, Data Curation. **Qingdu Li:** Supervision, Resources, Validation. **Yan Peng:** Supervision, Resources, Writing – Review & Editing.

## References

[1] X. Mo, W. Ge, M. Miraglia, F. Inglese, D. Zhao, C. Stefanini, D. Romano, Jumping locomotion strategies: From animals to bioinspired robots, Applied Sciences 10 (23) (2020) 8607. doi:10.3390/app10238607.

[2] J.-S. Koh, E. Yang, G.-P. Jung, S.-P. Jung, J. H. Son, S.-I. Lee, P. G. Jablonski, R. J. Wood, H.-Y. Kim, K.-J. Cho, Jumping on water: Surface tension–dominated jumping of water striders and robotic insects, Science 349 (6247) (2015) 517–521. doi:10.1126/science.aab1637.

[3] C. Yi, X. Chen, Y. Zhang, Z. Yu, H. Qi, Y. Liu, Q. Huang, Simulating the grf of humanoid robot vertical jumping using a simplified model with a foot structure for foot design, Journal of Bionic Engineering 21 (1) (2024) 112–125. doi:10.1007/s42235-023-00429-8.

[4] X. Wang, W. Guo, Z. He, R. Li, F. Zha, L. Sun, Bionic jumping of humanoid robot via online centroid trajectory optimization and high dynamic motion controller, Journal of Bionic Engineering 21 (6) (2024) 2759–2778. doi:10.1007/s42235-024-00586-4.

[5] Z. Zhao, S. Sun, H. Huang, Q. Gao, W. Xu, Design and control of continuous jumping gaits for humanoid robots based on motion function and reinforcement learning, Procedia Computer Science 250 (2024) 51–57. doi:10.1016/j.procs.2024.11.008.

[6] Y. Liu, X. Chen, Z. Yu, H. Qi, C. Yi, Single sequential trajectory optimization with centroidal dynamics and whole-body kinematics for vertical jump of humanoid robot, Biomimetics 9 (5) (2024) 274. doi:10.3390/biomimetics9050274.

[7] G. Ribak, Insect-inspired jumping robots: challenges and solutions to jump stability, Current Opinion in Insect Science 42 (2020) 32–38. doi:10.1016/j.cois.2020.09.001.

[8] K. Y. Su, J. Z. Gul, K. H. Choi, A biomimetic jumping locomotion of functionally graded frog soft robot, in: 2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), IEEE, 2017, pp. 675–676. doi:10.1109/URAI.2017.7992792.

[9] M. Afschrift, E. Van Asseldonk, M. Van Mierlo, C. Bayon, A. Keemink, L. D'Hondt, H. Van Der Kooij, F. De Groote, Assisting walking balance using a bio-inspired exoskeleton controller, Journal of Neuroengineering and Rehabilitation 20 (1) (2023) 82. doi:10.1186/s12984-023-01205-9.

[10] D. Ezekiel, R. Samikannu, O. Matsebe, Bio-inspired jumping spider optimization for controller tuning/parameter estimation of an uncertain aerodynamic mimo system, Chaos Theory and Applications 6 (3) (2024) 205–217. doi:10.51537/chaos.1396823.

[11] H. Elliott, X. An, M. Wang, A bio-inspired jumping robot: Design, modelling and experimental tests, in: Annual Conference Towards Autonomous Robotic Systems, Springer, 2024, pp. 164–170. doi:10.1007/978-3-031-72062-8_15.

[12] M. Kabir, A. Anand, P. Sundaravadivel, Hop-bot: a bio-inspired approach to locomotion and stability in modular robotics, in: 2024 IEEE International Conference on Electro Information Technology (EIT), IEEE, 2024, pp. 285–290. doi:10.1109/eIT60633.2024.10609916.

[13] J. Hong, C. Yeo, S. Bae, J. Hong, S. Oh, Slip embodied robust quadruped robot control, in: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2024, pp. 14219–14224. doi:10.1109/IROS58592.2024.10802545.

[14] G. Piovan, K. Byl, Reachability-based control for the active slip model, The International Journal of Robotics Research 34 (3) (2015) 270–287. doi:10.1177/0278364914552112.

[15] G. Piovan, K. Byl, Approximation and control of the slip model dynamics via partial feedback linearization and two-element leg actuation strategy, IEEE Transactions on Robotics 32 (2) (2016) 399–412. doi:10.1109/TRO.2016.2529649.

[16] H. Hamzaçebi, I. Uyanik, Ö. Morgül, On the analysis and control of a bipedal legged locomotion model via partial feedback linearization, Bioinspiration & Biomimetics 19 (5) (2024) 056004. doi:10.1088/1748-3190/ad5cb6.

[17] H.-W. Park, P. M. Wensing, S. Kim, Jumping over obstacles with mit cheetah 2, Robotics and Autonomous Systems 136 (2021) 103703. doi:10.1016/j.robot.2020.103703.

[18] D. Ahn, B.-K. Cho, Online jumping motion generation via model predictive control, IEEE Transactions on Industrial Electronics 69 (5) (2021) 4957–4965. doi:10.1109/TIE.2021.3078396.
[19] G. Ji, J. Mun, H. Kim, J. Hwangbo, Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion, IEEE Robotics and Automation Letters 7 (2) (2022) 4630–4637. doi:10.1109/LRA.2022.3151396.
[20] Z. He, J. Wu, J. Zhang, S. Zhang, Y. Shi, H. Liu, L. Sun, Y. Su, X. Leng, Cdm-mpc: An integrated dynamic planning and control framework for bipedal robots jumping, IEEE Robotics and Automation Letters 9 (7) (2024) 6672–6679. doi:10.1109/LRA.2024.3408487.
[21] Z. Xu, J. Xie, K. Hashimoto, Human-inspired gait and jumping motion generation for bipedal robots using model predictive control, Biomimetics 10 (1) (2025) 17. doi:10.3390/biomimetics10010017.
[22] Z. Fu, Z. Yu, X. Chen, L. Han, P. Gergondet, J. Zhang, Q. Huang, Continuous bipedal jumping via sliding-mode regularized predictive control, IEEE/ASME Transactions on Mechatronics (2024). doi:10.1109/TMECH.2024.3515151.
[23] J. Kober, J. A. Bagnell, J. Peters, Reinforcement learning in robotics: A survey, The International Journal of Robotics Research 32 (11) (2013) 1238–1274. doi:10.1177/0278364913495721.
[24] C. Tao, M. Li, F. Cao, Z. Gao, Z. Zhang, A multiobjective collaborative deep reinforcement learning algorithm for jumping optimization of bipedal robot, Advanced Intelligent Systems 6 (1) (2024) 2300352. doi:10.1002/aisy.202300352.
[25] M. Hutter, C. D. Remy, M. A. Höpflinger, R. Siegwart, Slip running with an articulated robotic leg, in: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2010, pp. 4934–4939. doi:10.1109/IROS.2010.5651461.
[26] X. He, X. Li, X. Wang, F. Meng, X. Guan, Z. Jiang, L. Yuan, K. Ba, G. Ma, B. Yu, Running gait and control of quadruped robot based on slip model, Biomimetics 9 (1) (2024). doi:10.3390/biomimetics9010024.
[27] P. M. Wensing, D. E. Orin, Control of humanoid hopping based on a slip model, Advances in Mechanisms, Robotics and Design Education and Research (2013) 265–274doi:10.1007/978-3-319-00398-6_21.
[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, arXiv preprint arXiv:1707.06347 (2017). doi:10.48550/arXiv.1707.06347.
[29] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, Y. Wu, The surprising effectiveness of ppo in cooperative multi-agent games, Advances in Neural Information Processing Systems 35 (2022) 24611–24624.
[30] Y. Zhao, T. Wu, Y. Zhu, X. Lu, J. Wang, H. Bou-Ammar, X. Zhang, P. Du, Zsl-rppo: Zero-shot learning for quadrupedal locomotion in challenging terrains using recurrent proximal policy optimization, arXiv preprint arXiv:2403.01928 (2024). doi:/10.48550/arXiv.2403.01928.
[31] Z. Zhang, J. Zhao, H. Chen, D. Chen, A survey of bioinspired jumping robot: takeoff, air posture adjustment, and landing buffer, Applied Bionics and Biomechanics 2017 (1) (2017) 4780160. doi:10.1155/2017/4780160.
[32] C. Zhang, W. Zou, L. Ma, Z. Wang, Biologically inspired jumping robots: A comprehensive review, Robotics and Autonomous Systems 124 (2020) 103362. doi:10.1016/j.robot.2019.103362.
[33] G. Garofalo, C. Ott, A. Albu-Schäffer, Walking control of fully actuated robots based on the bipedal slip model, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 1456–1463. doi:10.1109/ICRA.2012.6225272.
[34] M. Shahbazi, R. Babuška, G. A. Lopes, Unified modeling and control of walking and running on the spring-loaded inverted pendulum, IEEE Transactions on Robotics 32 (5) (2016) 1178–1195. doi:10.1109/TRO.2016.2593483.
[35] J. Rummel, Y. Blum, A. Seyfarth, Robust and efficient walking with spring-like legs, Bioinspiration & Biomimetics 5 (4) (2010) 046004. doi:10.1088/1748-3182/5/4/046004.
[36] S. Xie, X. Li, H. Zhong, C. Hu, L. Gao, Compliant bipedal walking based on variable spring-loaded inverted pendulum model with finite-sized foot, in: 2021 6th IEEE International Conference on Advanced Robotics and Mechatronics (ICARM), IEEE, 2021, pp. 667–672. doi:10.1109/ICARM52023.2021.9536096.
[37] H. Sang, S. Wang, Lunar leap robot: 3m architecture–enhanced deep reinforcement learning method for quadruped robot jumping in low-gravity environment, Journal of Aerospace Engineering 37 (6) (2024) 04024076. doi:10.1061/JAEEEZ.ASENG-5619.
[38] G. Bellegarda, C. Nguyen, Q. Nguyen, Robust quadruped jumping via deep reinforcement learning, Robotics and Autonomous Systems 182 (2024) 104799. doi:10.1016/j.robot.2024.104799.
[39] G. Bellegarda, M. Shafiee, M. E. Özberk, A. Ijspeert, Quadruped-frog: Rapid online optimization of continuous quadruped jumping, in: 2024 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2024, pp. 1443–1450. doi:10.1109/ICRA57147.2024.10610141.
[40] G. Bellegarda, A. Ijspeert, Cpg-rl: Learning central pattern generators for quadruped locomotion, IEEE Robotics and Automation Letters 7 (4) (2022) 12547–12554. doi:10.1109/LRA.2022.3218167.
[41] X. B. Peng, M. Andrychowicz, W. Zaremba, P. Abbeel, Sim-to-real transfer of robotic control with dynamics randomization, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 3803–3810. doi:10.1109/ICRA.2018.8460528.
[42] Q. Zhou, G. Li, R. Tang, Y. Xu, H. Wen, Q. Shi, Stable jumping control based on deep reinforcement learning for a locust-inspired robot, Biomimetics 9 (9) (2024) 548. doi:10.3390/biomimetics9090548.
[43] R. J. Full, D. E. Koditschek, Templates and anchors: neuromechanical hypotheses of legged locomotion on land, Journal of Experimental Biology 202 (23) (1999) 3325–3332. doi:10.1242/jeb.202.23.3325.
[44] H. Geyer, U. Saranli, Gait based on the spring-loaded inverted pendulum, in: A. Goswami, P. Vadakkepat (Eds.), Humanoid Robotics: A Reference, Springer, Dordrecht, 2019, pp. 923–947. doi:10.1007/978-94-007-6046-2_43.
[45] L. Ye, Y. Cheng, J. Li, X. Wang, B. Liang, Y. Peng, From knowing to doing: learning diverse motor skills through instruction learning, Biomimetic Intelligence and Robotics (2026) 100286.
[46] R. Hartley, M. Ghaffari, R. M. Eustice, J. W. Grizzle, Contact-aided invariant extended kalman filtering for robot state estimation, The International Journal of Robotics Research 39 (4) (2020) 402–430. doi:10.1177/0278364919894385.
[47] X. Gu, Y.-J. Wang, J. Chen, Humanoid-gym: Reinforcement learning for humanoid robot with zero-shot sim2real transfer, arXiv preprint arXiv:2404.05695 (2024). doi:10.48550/arXiv.2404.05695.