

Arm-Constrained Curriculum Learning for Loco-Manipulation of the Wheel-Legged Robot

Zifan Wang^{*1}, Yufei Jia^{*3}, Lu Shi^{*2}, Haoyu Wang^{2,4}, Haizhou Zhao^{2,5}, Xueyang Li⁶,
Jinni Zhou^{1†}, Jun Ma^{1†}, and Guyue Zhou^{2†}

Abstract—Incorporating a robotic manipulator into a wheel-legged robot enhances its agility and expands its potential for practical applications. However, the presence of potential instability and uncertainties presents additional challenges for control objectives. In this paper, we introduce an arm-constrained curriculum learning architecture to tackle the issues introduced by adding the manipulator. Firstly, we develop an arm-constrained reinforcement learning algorithm to ensure safety and stability in control performance. Additionally, to address discrepancies in reward settings between the arm and the base, we propose a reward-aware curriculum learning method. The policy is first trained in Isaac gym and transferred to the physical robot to do dynamic grasping tasks, including the door-opening task, fan-twitching task and the relay-baton-picking and following task. The results demonstrate that our proposed approach effectively controls the arm-equipped wheel-legged robot to master dynamic grasping skills, allowing it to chase and catch a moving object while in motion. Please refer to our website (<https://acodedog.github.io/wheel-legged-loco-manipulation/>) for the code and supplemental videos.

I. INTRODUCTION

Human beings are capable of easily completing a variety of complex tasks, such as navigating through areas with various obstacles and interacting with different objects. However, these seemingly common tasks introduce many challenges for robots to master these skills: robots need to perform dynamic movements at high speeds and interact with different objects while coordinating their various parts to ensure safety. The emergence of legged robot platforms has provided a feasible foundation for executing complex tasks in various uncertain environments [1], [2].

By augmenting robotic arms to the mobile robotic platform, e.g., quadrupedal robots [3], bipedal legged robots [4], and wheel-legged robots [5], etc, more complex interactions can be achieved. Among them, wheel-legged robots leverage the benefits of both wheeled and legged robots, harnessing the high energy efficiency of wheels along with the superior adaptability to surmount uneven terrain and obstacles using legs [6]. Research on robot whole-body control and object manipulation using Deep Reinforcement Learning (DRL) is increasingly growing which provides a more sample-efficient, flexible, and robust strategy for learning, and en-

ables the robot to more easily understand how to accomplish various complex tasks [7], [8].

Integrating a robot manipulator with a wheel-legged robot enhances agility and unlocks greater potential in practical applications. However, despite extensive studies on legged robots, a critical gap exists in the manipulation capabilities of wheel-legged robots. Coordinating the movement of both the arms and the wheels simultaneously requires managing multiple control modes, each with its own dynamics and constraints, that present additional challenges for the control. On the other hand, balancing the robot while moving its arms and wheels introduces extra dynamic stability requirements. In this work, we propose an arm-constrained curriculum RL framework for loco-manipulation of the wheel-legged robot. The game-inspired curriculum learning procedure enables the simultaneous control of both the arm and wheels. An arm-constrained network is introduced in the framework to ensure the safety and stability of the robot. Our novel framework enables dynamic grasping, leveraging the inherent stability, efficiency, and speed advantages of wheel-legged platforms, thereby enhancing their object manipulation capabilities.

In essence, the paper contributes to the following aspects:

- We proposed an arm-constrained curriculum reinforcement learning framework specifically designed for loco-manipulation of wheel-legged robots. The framework allows for simultaneous control of both the arm and wheels, addressing the stability, safety, and efficiency challenges of coordinating hybrid locomotion and manipulation tasks.
- We introduced a reward-aware curriculum learning process aimed at fostering balanced progress across all components of the agent, regardless of whether they have sparse or dense rewards. By implementing this approach, the risk of the system becoming stuck in a local minimum is mitigated.

II. RELATED WORK

A. RL-based Control of Legged Robots

Recent studies have shown impressive control performance when utilizing RL for legged robotic systems [9], [10], [11], [12], [13]. In [14], distributional RL is employed to train a risk-aware algorithm for the quadrupedal-legged robot, allowing it to dynamically adapt its behavior to different types of terrain. Meanwhile, researchers in [15] integrate proprioceptive data with noisy exteroceptive information to enable fast dynamic walking

^{*}Equal contribution. ¹The Hong Kong University of Science and Technology (Guangzhou), ²Institute for AI Industry Research (AIR), Tsinghua University, ³Department of Electronic Engineering, Tsinghua University, ⁴Harbin Institute of Technology, ⁵Xi'an Jiaotong-Liverpool University, ⁶DISCOVER Robotics

[†]Corresponding author. eejinni@hkust-gz.edu.cn, jun.ma@ust.hk, zhouguyue@air.tsinghua.edu.cn



Fig. 1. Tasks accomplished by the proposed architecture. Top-Left: door-opening-and-pulling task; Top-Right: fan-knob-twisting task; Bottom-Left: relay-baton-chasing task; Bottom-Right: door-opening-and-pushing task.

on various terrains. However, achieving such exceptional controllers with natural motion styles and high task performance often requires meticulous reward shaping to attain the desired behavior [16]. The researcher addresses this issue by employing motion imitation techniques, training control policies for simulated characters to replicate pre-recorded movements observed in animals or humans [17]. During the training process, the use of rewards encourages the policy to imitate movements from a motion clip, eliminating the need for extensive reward adjustments.

B. Loco-Manipulation for Legged Robots

Utilizing a versatile robotic platform equipped with arms enables dynamic object manipulation which is a challenging task [18] [19] [20]. In [21], an innovative approach is proposed that simultaneously identifies comprehensive whole-body trajectories and contact sequences to tackle diverse loco-manipulation scenarios within predefined environments. This method integrates trajectory optimization, informed graph search, and sampling-based planning, resulting in emergent behaviors for a quadrupedal mobile manipulator capable of both prehensile and nonprehensile interactions, enabling it to perform real-world tasks. In another study by [22], a novel approach is developed that combines learning-based locomotion policies with model-based manipulation. This enables legged mobile manipulators to adapt to challenging terrains and achieve robust locomotion. Additionally, [3], proposes a novel method to enhance the versatility of legged robots by learning a single unified policy. This policy facilitates seamless coordination between manipulation tasks (using an attached arm) and locomotion, overcoming the limitations of traditional hierarchical control pipelines. The method leverages reinforcement learning, Regularized Online Adaptation, and Advantage Mixing to bridge the Sim2Real gap and achieve dynamic and agile

behaviors across various task setups.

III. PRELIMINARY

In this section, we will introduce the technical preliminaries of the work.

A. Constrained Markov Decision Process (CMDP)

The environment of an RL problem is typically stated in the form of a Markov Decision Process, which is a mathematical framework used to model decision-making problems. It is described by a tuple $(S, A, R, P, \rho, \gamma)$, where S denotes the state space, A denotes the action space, $R \in \mathbb{R} : S \times A \rightarrow \mathbb{R}$ is the immediate reward function, $P : S \times A \times S \rightarrow \mathbb{R}$ is the transition model, and γ represents the discount factor. A Constrained Markov Decision Process (CMDP) [23], is a variant of the traditional MDP framework where additional constraints are imposed on the state-action pairs or policies to satisfy specific requirements or limitations during decision-making [24], [25]. It is usually described by a tuple $(S, A, R, P, C_{1,\dots,K}, \rho, \gamma)$, with the additional parameters $C_k : S \times A \rightarrow \mathbb{R}$ representing the cost function for $\forall k \in \{1, \dots, K\}$, and ρ being the initial state distribution. To solve a CMDP, we aim to find a policy π that maximizes:

$$J_R(\pi) = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t, s_{t+1}) \right] \quad (1)$$

$$\text{s.t. } J_{C_k}(\pi) \leq d_k \quad \forall k \in \{1, \dots, K\},$$

where J_{C_k} is the constraint, d_k is the constraint threshold, and the expectation $\mathbb{E}[\dots]$ represents discounted expected return.

B. Constrained Proximal Policy Optimization

Proximal Policy Optimization (PPO) is a model-free reinforcement learning algorithm that aims to find an optimal policy by iteratively updating a surrogate objective function

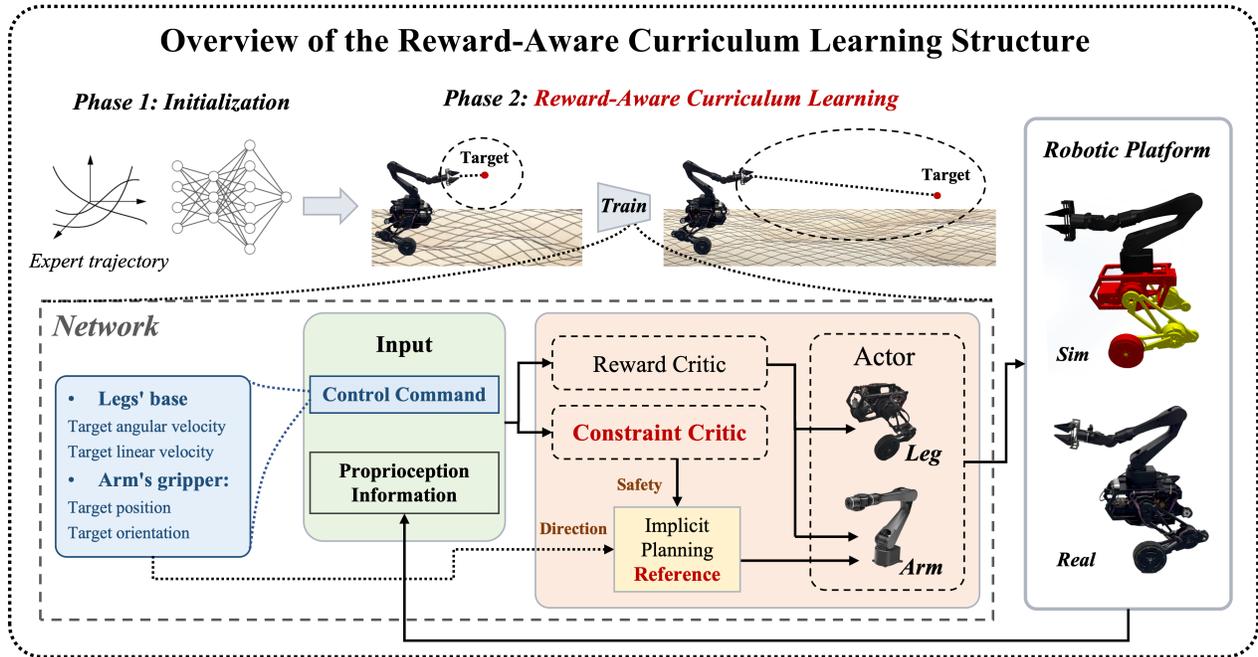


Fig. 2. The overall illustration of the proposed framework. Top: two-phase learning procedure; Bottom: the detailed representation of the network.

while enforcing a constraint on the size of policy updates, leading to stable and efficient learning. The objective function is described as

$$L^{\text{PPO}}(\theta) = \mathbb{E}_{\substack{s \sim d^{\pi_i} \\ a \sim \pi}} \left[\frac{\pi_{\theta}(a|s)}{\pi_i(a|s)} A^{\pi_i}(s, a) \right] - \beta D_{KL}[\pi_{\theta}(\cdot|s), \pi_i(\cdot|s)],$$

where D_{KL} is Kullback-Leibler Divergence and β represents weight of the strategy.

Penalized PPO (P3O) [26] is a variant of the PPO algorithm where additional constraints are incorporated into the optimization process to enforce specific criteria or limitations on the learned policies, ensuring compliance with desired behavior or safety constraints during training. The optimization problem becomes

$$\begin{aligned} \underset{\pi \in \Pi_{\theta}}{\text{maximize}} \quad & L^{\text{P3O}} = L^{\text{PPO}}(\theta) + \sum_{k=1}^K \log(d_k - J_{C_k}(\pi_{\theta})) / t \\ J_{C_k}(\pi_{\theta}) = & J_{C_k}(\pi_i) + \frac{1}{1 - \gamma} \mathbb{E}_{\substack{s \sim d^{\pi_i} \\ a \sim \pi}} \left[\frac{\pi_{\theta}(a|s)}{\pi_i(a|s)} A_{C_k}^{\pi_i}(s, a) \right], \end{aligned} \quad (2)$$

where J_{C_k} represents the constraint function introduced by the additional constraints $C_k, k \in 1, \dots, K$.

IV. METHOD

A. Overview of the structure

The overview of the proposed architecture is depicted in Fig. 2. The entire structure follows a two-phase learning procedure. Initially, a behavioral cloning process is employed to initialize the actor network. Subsequently, a reward-aware curriculum learning process is executed to iteratively improve the policy for task completion. Specifically, the network is trained to follow user commands, which comprise two components: the desired target pose for the arm and the desired velocities for the body. These user commands, along

with proprioception information from the robot, serve as the input (observation) to the Arm-Constrained Proximal Policy Optimization (AC-PPO) policy networks. The AC-PPO framework consists of three networks: the actor network, the reward critic network, and the additional constraint critic network. Safety information generated by the constraint critic network is incorporated into the implicit planning reference used to guide the robot arm. This framework enables coordinated hybrid locomotion and manipulation tasks. The details of each component are introduced as follows.

B. Arm-Constrained Proximal Policy Optimization

One of the most challenging aspects of loco-manipulation for wheel-legged robots involves effectively coordinating locomotion and manipulation while adapting to dynamic objects. To ensure that the manipulation task does not compromise locomotion safety, we formulate the velocity-tracking and loco-manipulation tasks as a CMDP. Addressing the aforementioned challenge, we incorporate designed constraints of the arm in (2) and train the P3O algorithm to generate stable and safe actions for *both the arm and the leg*. Specifically, the constraints are designed as follows:

- 1) **Arm joint constraints:** Although the action space may impose some constraints on the values of the action, these limits cannot always be reached in a continuous action space. Additionally, we aim to avoid situations where the arm's joint positions consistently reach their limits, which could do harm to the motors of the robot. To achieve this, we impose joint position constraints to encourage actions that stay as far as possible from the limits. This helps reduce the occurrence of unsafe

events and unstable movements. The expression is:

$$C_{\text{arm}} = \sum_{i \in \text{arm joints}} \|\max(q_{i,t} - q_i^{\text{upper}}, 0)\|_1 + \sum_{i \in \text{arm joints}} \|\min(q_{i,t} - q_i^{\text{lower}}, 0)\|_1 .$$

- 2) **Gripper position constraints:** This constraint pertains to the separation distance between the arm centroid and its base centroid. Adjusting the gripper’s position can alter the overall center of gravity of the robot, potentially leading to instability. To mitigate the risk of tipping over during manipulation tasks, this constraint is imposed to safeguard the robot’s stability and ensure its safety. The expression is described as:

$$C_{\text{Gripper}} = \|(P_{\text{centroid}}^{\text{arm}} - P_{\text{centroid}}^{\text{base}})\|_2 .$$

- 3) **Collision constraints:** To prevent self-collision and collisions between the robot arm and the environment, we impose a collision constraint on the force f_i exerted on each arm link. Ideally, this force should remain close to 0 when no collision is detected, indicating a balanced state. Enforcing this constraint is essential to ensure the robot operates safely within its environment. We present the constraint as:

$$C_{\text{force}} = \sum_{i \in \text{arm links}} \|(f_i^{\text{arm}})\|_2 .$$

C. Reward-Aware Curriculum Learning

Previous research has highlighted the advantages of employing a curriculum of task difficulty to train complex policies [27], [28]. The fundamental concept involves initially training the policy on simpler tasks before gradually increasing the complexity. In our framework, we introduce a reward-aware curriculum learning approach. Instead of adjusting the complexity of the tasks to achieve the curriculum, we initialize the agent closer to the target pose and progressively expand the range. This is especially beneficial for addressing the challenge of sparse reward settings, as the agent can quickly achieve a high reward in the early stages of the training process. In complex robotic platforms where the workspace and reward settings vary significantly across different components of the agent, we can encourage parts of the agent with sparse rewards to progress more equally alongside parts with dense rewards by employing this reward-aware curriculum learning process.

In this specific scenario, the robot arm has a broader workspace but receives sparser rewards compared to the wheeled-legs. Consequently, the robot may become trapped in a local minimum where the legs receive high rewards, leading to conservative actions of the arm to avoid affecting the stability of the legs. To overcome this challenge and enhance training efficiency, this reward-based curriculum learning structure is deployed to encourage the arm to learn to move. We initialize the end-effector of the arm close to the goal position, enabling relatively easy movements to achieve high rewards at the beginning of the training. Consequently, it

has less possibility to resist movement as we further increase the distance to the goal. An example of the procedure is:

$$P_{\text{ee}}^{\text{init}} \sim N(\mu, \Sigma) \quad \text{s.t. } \mu = P^{\text{goal}}, \quad \Sigma = 1 + \frac{t}{T} * D^2 , \quad (3)$$

where $P_{\text{ee}}^{\text{init}}$ is the gripper position where the robot is reset each time. μ and Σ are the mean and variance of the Gaussian distribution. P^{goal} is the target location of the gripper. T is the maximum episode length and D is the maximum desired target distance. Note that the expression of the process can be easily extended to other types of distributions. This approach enables both the legs and the manipulator to learn to track the references effectively.

D. Two-phase Learning using Behavior Cloning

As depicted in Fig. 2, a two-phase learning process was implemented to ensure the safety and efficiency of the architecture. In the first phase, data is collected from a Whole-Body Controller [29] operating within the target environment. Utilizing the baseline controller ensures the safe generation of data without endangering the robot. Subsequently, this dataset is employed to initialize the parameters of the actor-network in the subsequent phase, utilizing behavioral cloning techniques to replicate the decisions made by the baseline controller.

V. EXPERIMENTS

A. Experimental Setup

Several experiments are conducted to test the performance and stability of an arm-equipped wheel-legged robot as shown in Fig. 3, which consists of a wheel-legged chassis and a robot arm. For each leg, there are a total of 3 motors: a hip motor, a knee actuator, and a driving wheel. Additionally, it’s equipped with a 6-DoF serial robotic arm AIRBOT-Play and a gripper as the end-effector. Two cameras are mounted respectively on the robot’s base and the end effector of the robot arm. Power is provided by onboard battery and computation is also done onboard.

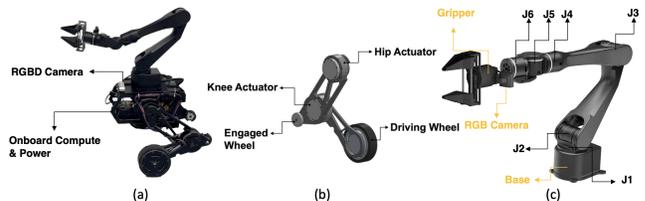


Fig. 3. Illustration of the arm-equipped wheel-legged robotic platform.

To train the policy according to the architecture outlined in Sec. IV, we defined the observation space, action space, and reward as follows:

1) **Observation Space:** The observation space is constructed by the state vector $S_t = \{S_t^{\text{base}}, S_t^{\text{arm}}, S_t^{\text{cmd}}\} \in \mathbb{R}^{46}$. The information of the base is represented in $S_t^{\text{base}} = [h, v, \omega, R, q_{\text{leg}}, \dot{q}_{\text{leg}}, \dot{q}_{\text{wheel}}] \in \mathbb{R}^{20}$. The observation of arm is stored in $S_t^{\text{arm}} = \{q_{\text{arm}}, \dot{q}_{\text{arm}}, p_{\text{ee}}, R_{\text{ee}}\} \in \mathbb{R}^{18}$, and

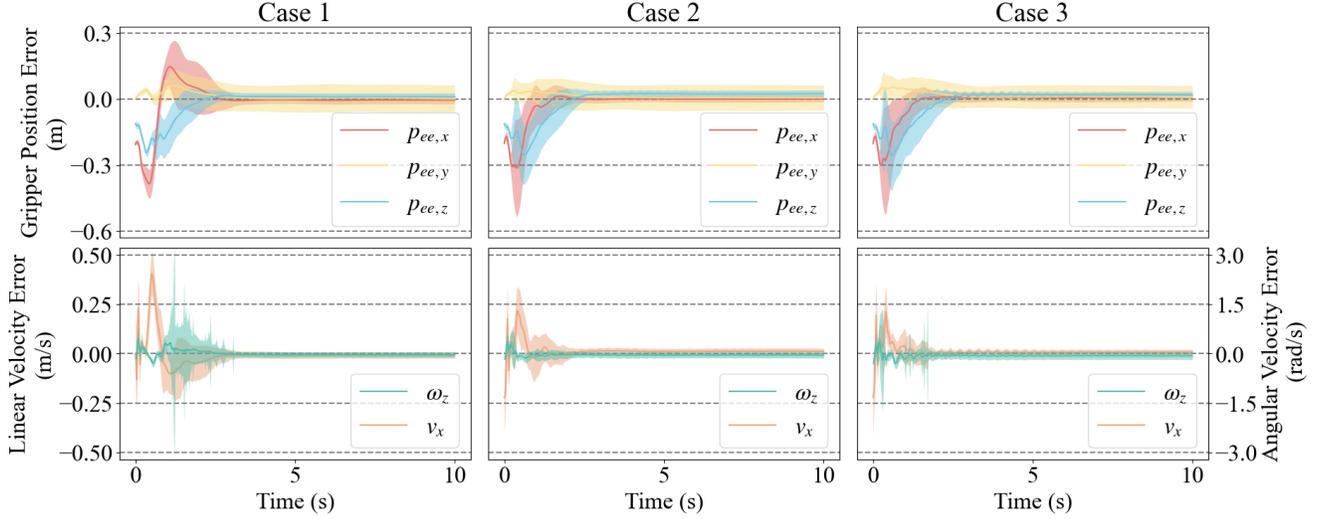


Fig. 4. The tracking results in simulation.

$S_t^{\text{cmd}} = \{v_x^{\text{cmd}}, \omega_z^{\text{cmd}}, p_{ee}^{\text{cmd}}, R_{ee}^{\text{cmd}}\} \in \mathbb{R}^8$ includes the control commands. The meanings of each parameter are:

- $h \in \mathbb{R}^1$: base height.
- $v \in \mathbb{R}^3$: base linear velocity.
- $\omega \in \mathbb{R}^3$: base angular velocity.
- $R \in \mathbb{R}^3$: base orientation.
- $q_{\text{leg}} \in \mathbb{R}^4$: leg joints position.
- $\dot{q}_{\text{leg}} \in \mathbb{R}^4$: leg joints velocity.
- $\dot{q}_{\text{wheel}} \in \mathbb{R}^2$: wheel joints velocity.
- $q_{\text{arm}} \in \mathbb{R}^6$: arm joints position.
- $\dot{q}_{\text{arm}} \in \mathbb{R}^6$: arm joints velocity.
- $p_{ee} \in \mathbb{R}^3$: gripper position.
- $R_{ee} \in \mathbb{R}^3$: gripper orientation.
- $v_x^{\text{cmd}} \in \mathbb{R}^1$: x-axis linear velocity.
- $\omega_z^{\text{cmd}} \in \mathbb{R}^1$: z-axis angular velocity.
- $p_{ee}^{\text{cmd}} \in \mathbb{R}^3$: gripper target position.
- $R_{ee}^{\text{cmd}} \in \mathbb{R}^3$: gripper target orientation.

2) **Action Space**: The action vector $A = \{A_\tau, A_{\text{arm}}\} \in \mathbb{R}^{12}$ of the policy is constructed by the joint torques $A_\tau \in \mathbb{R}^6$ to control the wheeled legs and the joint position $A_{\text{arm}} \in \mathbb{R}^6$ to drive the arm. The vector of joint positions is further processed by a low-level controller to obtain the torque required to drive the arm.

3) **Reward**: As shown in Tab. I, the locomotion rewards and manipulation rewards are defined separately for wheeled legs and the arm, both considered the tracking performance and safety.

B. Simulation Tests

The proposed structure is deployed in simulation first. We use Isaac Gym as our simulator and train the policy with 6000 environments simultaneously. The control commands for the body, target linear and angular velocities of the base, are selected from uniform distributions over the intervals $v_x^{\text{cmd}} = [-2, 2]$ m/s, and $\omega_z^{\text{cmd}} = [-0.5, 0.5]$ rad/s at the beginning of the training. The robot then learned to track this velocity until termination. Upon meeting the termination criteria, which means the robot learned to track the current

TABLE I

REWARD FUNCTION SETTINGS

Locomotion Rewards	Expression
Linear velocity tracking	$\exp(-7.5 \cdot \ v_x^{\text{cmd}} - v_x\ ^2)$
Angular velocity tracking	$\exp(-1.25 \cdot \ \omega_z^{\text{cmd}} - \omega_z\ ^2)$
Acceleration limits	$-0.1 \cdot \ v_x^{\text{last}} - v_x\ ^2$
Orientation penalty	$-1.2 \cdot \ R_y\ ^2 - 1.2 \cdot \ R_x\ ^2$
Energy penalty	$-10^{-5} \cdot \ \tau\ ^2$, τ : motor torque
Leg motion	$-10^{-7} (\ \dot{q}_{\text{leg}}\ ^2 - 2.5 \ \dot{q}_{\text{leg}}\ ^2)$
Manipulation Rewards	Expression
Gripper position tracking	$\exp(-5 \cdot \ p_{ee} - p_{ee}^{\text{cmd}}\ ^2)$
Body position tracking	$\exp(-0.05 \cdot \ p_{\text{base}} - p_{\text{base}}^{\text{cmd}}\ ^2)$
Arm position upper limits	$-10 \sum \max(q_{\text{arm}, i, t} - q_{\text{arm}, i}^{\text{upper}}, 0)^2$
Arm position lower limits	$-10 \sum \min(q_{\text{arm}, i, t} - q_{\text{arm}, i}^{\text{lower}}, 0)^2$

velocities, the target values were resampled. In terms of the manipulator, we employed the proposed reward-aware curriculum learning process as introduced in Sec. IV-C. The target position for the gripper is sampled from the normal distribution with 0-mean and standard deviations within a narrow range initially, which is $[0.5, 0.1, 0.2]$ m for p_{ee}^{cmd} . Once the reward exceeds 90% of the maximum reward, the range of standard deviations is expanded and the expanding step size is $[0.5, 0.1, 0.1]$ m for the x-, y-, and z- directions of the end-effector.

Before transferring to the physical robot, we validate the performance of the algorithm in the simulation. Three cases of command velocities for the base are selected as {Case 1: $v_x^{\text{cmd}} = 0.5$ m/s, $\omega_z^{\text{cmd}} = 0$ rad/s}, {Case 2: $v_x^{\text{cmd}} = 0$ m/s, $\omega_z^{\text{cmd}} = 0.5$ rad/s}, and {Case 3: $v_x^{\text{cmd}} = 0.5$ m/s, $\omega_z^{\text{cmd}} = 0.5$ rad/s}. They keep constant during the test procedure. On the other hand, the initial values of the desired target position for the manipulator are randomly selected from a cube of size $[0.25 \times 0.2 \times 0.35]$ m³ with the origin at $[0.25, 0, 0.15]$ m. Then the initial target is propagated randomly within this range with a relatively small step size to produce a continuous target trajectory with a length of 500. The sampling rate is 50hz and the test is run repeatedly for 2024 times to avoid bias. The test is designed to mimic the

scenario that the robot tracks a dynamic target while moving its base. As shown in Fig. 4, our approach can track the command with a small error. Even for a moving target, it can converge quickly to the desired trajectory.

C. Real-Robot Tests

We further validate the trained policy by testing it on the physical robot to complete various tasks, as illustrated in Fig. 1. In contrast to the random generation during the simulation process, the control commands required for the base velocities and manipulation arm poses are provided through user inputs to accomplish different tasks. Based on the sources of these commands, our experiments can be categorized as follows:

- 1) **Teleoperation:** The control commands are provided using a remote control handle. The robot can track these instructions to accomplish the following tasks.
 - a) *Door-opening task:* This task involves approaching the doorknob, rotating it, pushing or pulling the door, and moving.
 - b) *Fan-twitching task:* This task involves approaching the fan, pushing and twitching the knob.
- 2) **Dynamic Manipulation:** Other than commands generated with human-in-the-loop, the challenging task of dynamic tracking is accomplished by obtaining the commands calculated from feedback provided by a camera. An RGB camera is installed on the gripper at the end of the robotic arm to guide the robot towards the relay baton and capture it. The robotic arm swiftly tracks the movement of the relay baton and ultimately grasps it using the proposed structure, highlighting the benefits of our algorithm in integrating the robot's chassis mobility with the dynamic movement of the robotic arm.

D. Ablation Study

To evaluate the importance of different factors of our proposed structure, we did the ablation tests on two of the most important components in our method.

1) *With/Without the Curriculum Learning:* Fig. 5 illustrates the reward and mean action noise standard deviation throughout the training process. It is evident that our framework, incorporating the proposed reward-aware curriculum learning, achieves higher rewards and exhibits reduced randomness in actions.

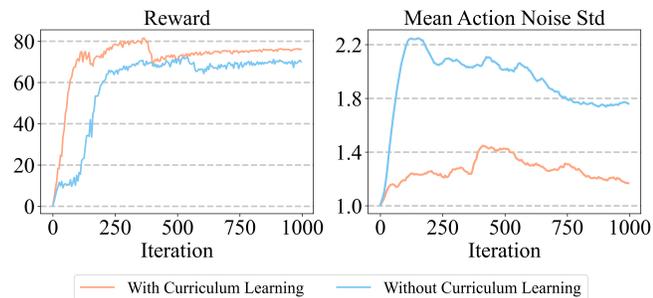


Fig. 5. Effect of curriculum learning for rewards and actions

2) With/without the Arm-Constrained Critic Network:

To evaluate the significance of the constraint network, we conducted a comparative analysis with our approach against the reward-only PPO framework, with all training parameters kept identical. As depicted in Fig. 6, our method notably promotes safer movements and reduces oscillations across all joints of the robotic arm. The actions of the robotic arm are less likely to violate joint limits for most of the joints when compared with the baseline method (only the J6 joint, which connects to the gripper, seems to violate the limitation more than the baseline, but it can be the result of trying to balance the arm centroid and the base centroid).

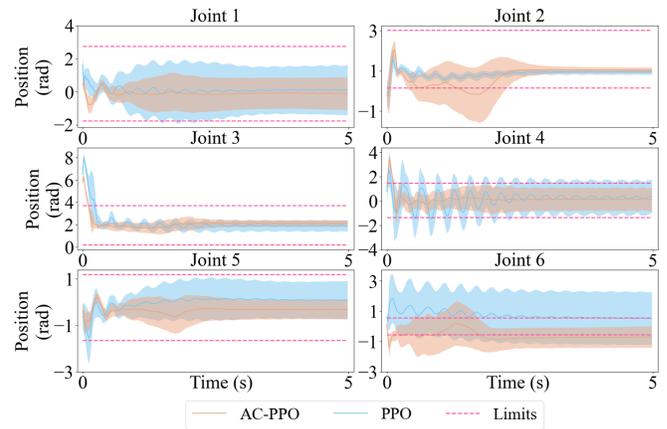


Fig. 6. Effect of the arm-constraints

VI. CONCLUSION

In summary, this paper presents a reinforcement learning framework for the loco-manipulation of wheel-legged robots, enabling them to perform a range of complex manipulation tasks in highly dynamic situations. We emphasize the additional challenges posed by incorporating the arm into the system. Firstly, an AC-PPO is designed to ensure safety and stability in control performance. Secondly, a reward-aware curriculum learning algorithm is proposed to address differences in reward settings between the arm and the base. The structure demonstrates relatively high tracking accuracy in simulation. Finally, we showcase the proficiency of the architecture in the real world by completing basic teleoperation tasks and dynamic manipulation tasks. In the future, we are going to implement and extend the architecture to the multi-agent collaboration tasks.

REFERENCES

- [1] F. Jenelten, J. He, F. Farshidian, and M. Hutter, "Dtc: Deep tracking control," *Science Robotics*, vol. 9, no. 86, p. eadh5401, 2024.
- [2] M. Geilinger, S. Winberg, and S. Coros, "A computational framework for designing skilled legged-wheeled robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3674–3681, 2020.
- [3] Z. Fu, X. Cheng, and D. Pathak, "Deep whole-body control: Learning a unified policy for manipulation and locomotion," in *Conference on Robot Learning*. PMLR, 2023, pp. 138–149.
- [4] F. Raza, W. Zhu, and M. Hayashibe, "Balance stability augmentation for wheel-legged biped robot through arm acceleration control," *IEEE Access*, vol. 9, pp. 54 022–54 031, 2021.

- [5] Q. Chang, X. Liu, W. Xu, L. Yan, and B. Yang, "The design and experiments of a small wheel-legged mobile robot system with two robotic arms," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 2590–2595.
- [6] S. Wang, L. Cui, J. Zhang, J. Lai, D. Zhang, K. Chen, Y. Zheng, Z. Zhang, and Z.-P. Jiang, "Balance control of a novel wheel-legged robot: Design and experiments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 6782–6788.
- [7] E. F. Morales, R. Murrieta-Cid, I. Becerra, and M. A. Esquivel-Basaldua, "A survey on deep learning and deep reinforcement learning in robotics with a tutorial on deep reinforcement learning," *Intelligent Service Robotics*, vol. 14, no. 5, pp. 773–805, 2021.
- [8] C. Zhang, J. Jin, J. Frey, N. Rudin, M. E. Mattamala Aravena, C. Cadena, and M. Hutter, "Resilient legged local navigation: Learning to traverse with compromised perception end-to-end," in *41st IEEE Conference on Robotics and Automation (ICRA)*, 2024.
- [9] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," 09 2021.
- [10] V. Tsounis, M. Alge, J. Lee, F. Farshidian, and M. Hutter, "Deepgait: Planning and control of quadrupedal gaits using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3699–3706, 2020.
- [11] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.
- [12] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.
- [13] Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, and V. Sindhwani, "Data efficient reinforcement learning for legged robots," in *Conference on Robot Learning*. PMLR, 2020, pp. 1–10.
- [14] L. Schneider, J. Frey, T. Miki, and M. Hutter, "Learning risk-aware quadrupedal locomotion using distributional reinforcement learning," in *41st IEEE Conference on Robotics and Automation (ICRA)*, 2024.
- [15] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.
- [16] S. Choi, G. Ji, J. Park, H. Kim, J. Mun, J. H. Lee, and J. Hwangbo, "Learning quadrupedal locomotion on deformable terrain," *Science Robotics*, vol. 8, no. 74, p. eade2256, 2023.
- [17] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," in *Robotics: Science and Systems*, 07 2020.
- [18] E. Farnioli, M. Gabiccini, and A. Bicchi, "Toward whole-body locomotion: Experimental results on multi-contact interaction with the walk-man robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1372–1379.
- [19] Y. Wu, P. Balatti, M. Lorenzini, F. Zhao, W. Kim, and A. Ajoudani, "A teleoperation interface for loco-manipulation control of mobile collaborative robotic assistant," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3593–3600, 2019.
- [20] Y. Ma, F. Farshidian, and M. Hutter, "Learning arm-assisted fall damage reduction and recovery for legged mobile manipulators," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 12 149–12 155.
- [21] J.-P. Sleiman, F. Farshidian, and M. Hutter, "Versatile multicontact planning and control for legged loco-manipulation," *Science Robotics*, vol. 8, no. 81, p. eadg5014, 2023.
- [22] Y. Ma, F. Farshidian, T. Miki, J. Lee, and M. Hutter, "Combining learning-based locomotion policy with model-based manipulation for legged mobile manipulators," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2377–2384, 2022.
- [23] E. Altman, *Constrained Markov decision processes*. Routledge, 2021.
- [24] Y. Kim, H. Oh, J. Lee, J. Choi, G. Ji, M. Jung, D. Youm, and J. Hwangbo, "Not only rewards but also constraints: Applications on legged robot locomotion," *arXiv preprint arXiv:2308.12517*, 2023.
- [25] J. Lee, L. Schroth, V. Klemm, M. Bjelonic, A. Reske, and M. Hutter, "Evaluation of constrained reinforcement learning algorithms for legged locomotion," *arXiv preprint arXiv:2309.15430*, 2023.
- [26] L. Zhang, L. Shen, L. Yang, S. Chen, X. Wang, B. Yuan, and D. Tao, "Penalized proximal policy optimization for safe reinforcement learning," 07 2022, pp. 3719–3725.
- [27] Z. Xie, H. Y. Ling, N. H. Kim, and M. van de Panne, "Allsteps: curriculum-driven learning of stepping stone skills," in *Computer Graphics Forum*, vol. 39, no. 8. Wiley Online Library, 2020, pp. 213–224.
- [28] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.
- [29] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.