

Robust Pedipulation on Quadruped Robots via Gravitational-moment Minimization

Heechan Shin, Minsung Yoon, Jeil Jeong, and Sung-eui Yoon*

Abstract: This paper presents adaptive locomotion and manipulation (ALaM), a novel approach for quadruped robots that enables robots to pedipulate with any leg as a manipulator while maintaining balance and locomotion with the remaining three legs. Unlike previous methods that only utilize a designated leg as a manipulator, ALaM enables any leg as manipulator based on the task goal position. For this end, we propose a gravitational moment minimization reward for training pedipulation using the centroid of feet as a pivot point. Experimental results demonstrate that ALaM achieves better stability during manipulation, reduces tracking error by approximately 50% compared to state-of-the-art methods, and completes multi-goal tasks more efficiently by dynamically switching manipulation legs.

Keywords: Adaptive locomotion, locomotion, pedipulation, quadruped robot.

1. INTRODUCTION

Quadruped robots have traditionally focused on locomotion tasks, with control strategies evolving from classical dynamic models to modern reinforcement learning approaches. Recent advancements have enabled these robots to perform more complex behaviors beyond just walking or running. One particularly interesting capability is using an additional manipulator by attaching it to the back of the robot. However, attaching additional manipulators to the robot's back incurs extra costs and takes up additional space. Instead, we focused on pedipulation. Pedipulation involves using one of the existing four legs as a manipulator, which does not require any additional equipment or money.

In this paper, we introduce adaptive locomotion and manipulation (ALaM), which is briefly introduced at [1], a novel method that allows quadruped robots to use any of their legs as a manipulator while preventing falling with the remaining three legs. The main contribution is proposing a reward that minimizes the gravitational moment based on the centroid of locomotion feet. Through this, we were able to maintain a robust posture even while the manipulation leg was moving, and achieve a goal tracking error up to two times lower than state-of-the-art research.

2. RELATED WORKS

2.1. Reinforcement learning based quadruped locomotion

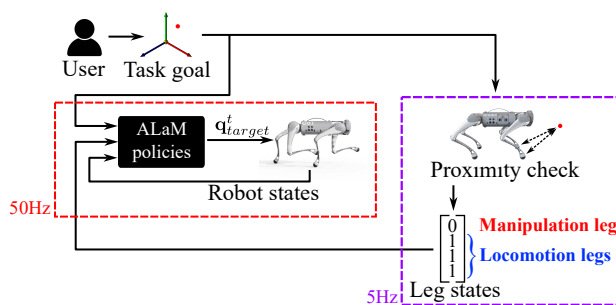


Fig. 1. Overall structure of our system. Any leg close to the task goal can be a manipulation leg. ALaM policies enable the robot to locomote only with locomotion legs.

Traditionally, legged-robot control relied on precise dynamic models to compute control strategies for responsive motions. Approaches ranged from the linear inverted pendulum model [2] to trajectory optimization [3] and Model Predictive Control (MPC) [4,5]. Recently, significant advancements in learning-based methodologies, particularly reinforcement learning, have revolutionized locomotion techniques for legged robots.

Notable research by Hwangbo *et al.* [6] has enabled more realistic locomotion simulations by modeling real-world actuators using neural networks. Rudin *et al.* [7] leveraged GPUs to run simulations, allowing the concurrent training of thousands of agents. This combination of realistic simulation and computational efficiency has significantly improved reinforcement learning techniques for

Manuscript received June 15, 2025; revised August 17, 2025; accepted September 10, 2025. Recommended by Associate Editor Jongdae Jung under the direction of Senior Editor Hyun Myung.

Heechan Shin, Minsung Yoon, and Sung-eui Yoon are with the School of Computing, KAIST, Korea (e-mails: _____@_____, _____@_____, sungeui@kaist.edu). Jeil Jeong is with the Robotics Program, KAIST, Korea (e-mail: _____@_____).

* Corresponding author.

quadruped robot locomotion, inspiring numerous subsequent studies.

An example is a study by Kumar *et al.* [8]. This study introduces a knowledge distillation technique for privileged information, which is data that are not available in real world environments, utilizing observation history instead of privileged information. This approach has been used in several studies, including [9-11], and our work, to effectively incorporate privileged information into locomotion training, bridging the gap between simulation and real-world performance.

2.2. Legged manipulation

Legged manipulation, which enables quadruped robots to perform various tasks by integrating a manipulator, is a subfield within quadruped locomotion research. This approach allows quadruped robots to navigate diverse terrains while simultaneously performing specific tasks [12,13]. Notable contributions in this area include the work of Sleiman *et al.* [13] employed task and motion planning to derive whole-body trajectories. Fu *et al.* [12] utilized *Advantage Mixing* to learn a unified policy covering both locomotion and manipulation.

However, equipping an external manipulator increases costs and reduces the robot’s load-carrying capacity because of its placement on the quadruped’s back. To address these limitations, our study uses a technique called *pedipulation* [14], which uses one of the existing legs as a manipulator while using the other three legs for locomotion. This approach eliminates the need for an additional limb. Although our study also focuses on pedipulation, unlike [14], our approach allows any leg to be used as a manipulator.

2.3. Pedipulation

Over the past few years, numerous studies have explored the utilization of quadruped robot legs for task execution [15-17]. However, these studies primarily focused on performing a single specific task. For instance, Ji *et al.* [15] employed legs to kick a ball. Cheng *et al.* [16] separately trained policies for locomotion and task execution, nonetheless, the robot uses only one skill at a time, thereby making it challenging to perform a task with one leg while walking simultaneously. In [17], the legs were solely used for dribbling. Recently, however, Philip *et al.* [14] have tackled this problem by distinguishing between locomotion legs and manipulation legs, enabling simultaneous locomotion and target goal tracking for task execution. In [14], after predefining locomotion and manipulation legs, the robot is trained. However, this *predefined* strategy restricts the ability of using undefined legs for task goal tracking. To overcome the predefined strategy, our study uses a technique named *adaptive locomotion* which has been studied in the field of fault-tolerant locomotion to walk with available legs, allowing any of the four legs to function as a manipulator.

2.4. Adaptive locomotion

Adaptive locomotion involves maintaining balance and performing locomotion with the remaining three legs when one leg becomes inoperable due to malfunctions or other issues. There are studies [18,19] that consider zero moment point (ZMP) for maintaining stability. The goal is to ensure that the point where all moments applied to the CoM become zero falls within the support polygon formed by the contact feet. Recently, Luo *et al.* [10] studied utilizing the center of pressure (CoP) for fault-tolerant locomotion. They achieved balance by aligning the CoP with the center of mass (CoM) through variable height inverted pendulum (VHIP) [20] modeling. However, unlike [10], where the malfunctioning limb remains fixed or moves minimally, our study involves significant shifts of the CoM as the manipulation leg moves during task execution. To tackle this challenge, we propose a method that enables robust balancing and locomotion while moving one leg.

3. ADAPTIVE LOCOMOTION AND MANIPULATION

In this section, we describe our pedipulation system where a quadruped robot uses one leg as a manipulator to perform tasks while maintaining balance and locomotion with the remaining three legs. The leg used as a manipulator is called the *manipulation leg*, while the other three legs used for locomotion are called *locomotion legs*. Unlike state-of-the-art pedipulation work [14], which can only use a designated leg as the manipulation leg, our proposed method allows any leg to be selected as the manipulation leg based on the given task during execution time. Then, the remaining three legs automatically become locomotion legs to perform movement. Additionally, in this paper, we focus on the goal tracking task, which is fundamental to pedipulation tasks with the manipulation leg. This means that the end effector of the manipulation leg tracks the given task goal while maintaining balance with the locomotion legs.

This section consists of three main components. The Overall Structure (Subsection 3.1) explains our reinforcement learning framework that separates locomotion and manipulation parts for effective pedipulation. The Adaptive Locomotion (Subsection 3.2) component introduces novel techniques for maintaining balance using locomotion legs, introducing gravitational moment reward and feet area reward. Finally, the Adaptive Manipulation (Subsection 3.3) component presents a method for target goal tracking using the manipulation leg through the use of a reachability map.

3.1. Overall structure

We divide our approach into two parts (Fig. 2). One part computes actions for locomotion, while the other calculates actions for a manipulation leg and base commands. This division is due to the different purposes of each part. The

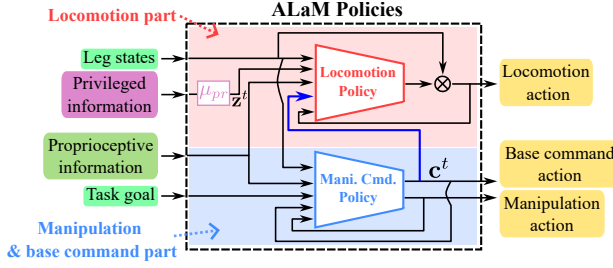


Fig. 2. Detailed description of ALaM policies. The privileged information is embedded into \mathbf{z}^t through encoder μ_{pr} and passed to the locomotion policy. The output of the locomotion policy is performed element-wise multiplication for action selection (See Subsection 3.2.3). The base command \mathbf{c}^t , generated from the manipulation & base command policy, is input into the locomotion policy to help approach the task goal.

locomotion part generates motions to maintain balance in response to given base commands and leg states. In contrast, the manipulation and base command part generates actions for approaching to task goal. Combining them in a single network may introduce competing gradient signals. Thus, we decouple the policies to reduce ambiguity in the learning signals. During training, we further isolate gradients using locomotion action selection (Subsection 3.2.3), which masks action dimensions assigned to the manipulation leg so they do not backpropagate into the locomotion policy. In this study, the task is to track the goal using end-effector of the manipulation leg.

3.1.1 Observations and actions

The policies of our system employ several categories of inputs, depicted in Fig. 2. The first category comprises proprioceptive information. Each element represents a specific aspect of the system’s state, see Table 1. The second category incorporates encoded privileged information, expressed as $\mathbf{z}^t = \mu_{pr}(\mathbf{v}_{cof}^t, \mathbf{p}_{com,B}^t, \mathcal{R})$, where μ_{pr} is a privileged encoder and \mathbf{v}_{cof}^t denotes the linear velocity of centroid of locomotion feet. By considering \mathbf{v}_{cof}^t , we can reduce the unnatural behavior reported by Bellegarda *et al.* [21]. The term \mathcal{R} encapsulates randomized parameters such as payload, friction, and motor strength. During training, these parameters are randomized within predefined ranges to model uncertainty in real-world conditions. At evaluation time, however, fixed parameter values are used, and the learned policy adapts implicitly without requiring explicit knowledge of the exact values. This design ensures robustness while allowing fair experimental comparisons. We use privileged inputs only during training. At the evaluation time, \mathbf{z}^t is produced from observable histories. The third category includes command information, task goal position \mathbf{G}_B^t , leg states ℓ^t , and base velocity command \mathbf{c}^t .

Table 1. The variables used in this paper consist of proprioceptive information, privileged information, command information, and actions.

Proprioceptive information	
$\mathbf{g} \in \mathbb{R}^3$	Projected gravity
$\dot{\theta}_B \in \mathbb{R}^3$	Base angular velocity in base frame (rad/s)
$\mathbf{q} \in \mathbb{R}^{12}$	Joint position (rad)
$\dot{\mathbf{q}} \in \mathbb{R}^{12}$	Joint velocity (rad/s)
$\mathbf{b} \in \mathbb{R}^4$	Foot contact indicator (contact: 1) $\mathbf{b} = \langle \mathbf{b}_{FR}, \mathbf{b}_{FL}, \mathbf{b}_{RR}, \mathbf{b}_{RL} \rangle^T$ F : front, R : rear, $_R$: right, $_L$: left
$\mathbf{r}_B \in \mathbb{R}^{12}$	Foot position in base frame, $\hat{\mathbf{r}}_B \in \mathbb{R}^3$ is foot position of manipulation leg in base frame
Privileged information	
$\mathbf{v}_{cof} \in \mathbb{R}^3$	Linear velocity of centroid of locomotion feet
$\mathbf{p}_{com,B} \in \mathbb{R}^3$	Center of mass in base frame
$\mathcal{R} \in \mathbb{R}^{14}$	Domain randomization parameters. payload $\in \mathbb{R}^1$, friction $\in \mathbb{R}^1$ motor strength $\in \mathbb{R}^{12}$
Command information	
$\mathbf{G}_B \in \mathbb{R}^3$	Task goal position in base frame. $\hat{\mathbf{G}}_B$ is projected goal in base frame (Subsection 3.3).
$\ell \in \mathbb{R}^4$	Leg states indicator $\ell = \langle \ell_{FR}, \ell_{FL}, \ell_{RR}, \ell_{RL} \rangle^T$ locomotion leg: 1, manipulation leg: 0
$\mathbf{c} \in \mathbb{R}^3$	Base velocity command $\mathbf{c} = \langle \mathbf{v}_{xy}^{des}, \dot{\theta}_{yaw}^{des} \rangle^T$ \mathbf{v}_{xy}^{des} is desired xy-axes linear velocities $\dot{\theta}_{yaw}^{des}$ is yaw angular velocity
Actions	
$\mathbf{a}_{loc} \in \mathbb{R}^{12}$	Actions used for locomotion
$\mathbf{a}_{man} \in \mathbb{R}^3$	Actions used for manipulation
$\mathbf{a}_{cmd} \in \mathbb{R}^3$	Actions for base command

A detailed explanation of each input is provided in Table 1. The superscript t denotes the value at time step t . For simplicity, we omit the superscript t unless we need to clearly distinguish between different time steps, such as $t - 1$.

In this study, we train two distinct policies. The locomotion policy, that generates a locomotion action, $\mathbf{a}_{loc} \in \mathbb{R}^{12}$, corresponding to the twelve degrees of freedom of a quadrupedal robot’s joints. The target joint positions for the locomotion legs are $\alpha \times \mathbf{a}_{loc} + \mathbf{q}_{init}$, where, α is action scale factor and \mathbf{q}_{init} is initial joint configuration.

The manipulation and base command policy generates two types of actions: one is for the manipulation leg and the other one is for the base command. The manipulation action, $\mathbf{a}_{man} \in \mathbb{R}^3$, is scaled by the action scaler and added by the previous manipulation joint position. Furthermore, to match dimensions with the target joint positions, we zero-pad with function $\mathcal{Z}(\cdot)$. Thus, we obtain the target joint position for the manipulation leg with $\mathcal{Z}(\alpha \times \mathbf{a}_{man}^t + \mathbf{q}_{man}^{t-1})$, where \mathbf{q}_{man}^{t-1} is previous manipulation joint position.

Finally, the target joint position for the PD controller is computed as $\mathbf{q}_{target}^t = (\alpha \times \mathbf{a}_{loc} + \mathbf{q}_{init}) \oplus \mathcal{Z}(\alpha \times \mathbf{a}_{man}^t + \mathbf{q}_{man}^{t-1})$, where \oplus is elementwise addition. The final control

joint torque is determined by: $\kappa_P \times (\mathbf{q}_{target} - \mathbf{q}) - \kappa_D \times \dot{\mathbf{q}}$, where κ_P represents the P gain and κ_D the D gain of the PD control and we use zero target joint velocity. We employ $\kappa_P = 28.0$ and $\kappa_D = 0.7$, suggested in [22].

3.1.2 Rewards

The locomotion policy training uses three types of rewards: a task reward to ensure the following of the given command, a balance reward to maintain body equilibrium, and an auxiliary reward to achieve consistent and smooth motion. The auxiliary rewards are similar to a normalization reward proposed by Arm *et al.* [14], or a fixed auxiliary reward, as described by Margolis *et al.* [23].

In Table 2, we use σ_1 as 0.25, and σ_2 as 0.02. \mathcal{F} is ground reaction force vector for each foot. \mathbf{v}_f and $\dot{\theta}_f$ are linear and angular velocity vectors of feet and \mathbf{v}_f^{th} and $\dot{\theta}_f^{th}$ are slip threshold of the velocity vectors, respectively. $\mathbf{q}_{violation}$ is number of limit violated joints.

The training of the manipulation and base command policy also incorporates task rewards and auxiliary rewards. Similar to the task rewards of the locomotion policy, the task rewards for the manipulation and base command pol-

Table 2. Rewards for training the locomotion policy.

Locomotion rewards			
Task rewards			
\mathbf{v}_{xy} tracking $w_1 \times e^{-\ \mathbf{v}_{xy}^{des} - \mathbf{v}_{xy}\ /\sigma_1}$		$\dot{\theta}_{yaw}$ tracking $w_2 \times e^{-\ \dot{\theta}_{yaw}^{des} - \dot{\theta}_{yaw}\ /\sigma_1}$	
\mathbf{v}_{cof} tracking $w_3 \times e^{-\ \mathbf{v}_{xy}^{des} - \mathbf{v}_{cof}\ /\sigma_1}$			
Balance rewards			
gravitational moment Subsection 3.2.1		locomotion feet area Subsection 3.2.2	
Auxiliary rewards			
Orientation $w_4 \times e^{-\ \theta_{pitch,roll}\ /\sigma_2}$		Penalize \mathbf{v}_z $w_5 \times \ \mathbf{v}_z\ ^2$	
Penalize $\dot{\theta}_{roll,pitch}$ $w_6 \times \ \dot{\theta}_{roll,pitch}\ ^2$		Penalize $\ddot{\mathbf{q}}$ $w_7 \times \ \ddot{\mathbf{q}}\ $	
Penalize τ $w_8 \times \ \tau\ $		Penalize joint power $w_9 \times \ \dot{\mathbf{q}}^T \cdot \tau\ $	
Action rate $w_{10} \times \ \mathbf{a}_{loc}^t - \mathbf{a}_{loc}^{t-1}\ $		Action smoothness $w_{11} \times \ \mathbf{a}_{loc}^t - 2\mathbf{a}_{loc}^{t-1} + \mathbf{a}_{loc}^{t-2}\ $	
Soft contact $w_{12} \times \ \mathbf{b}^T \cdot \mathcal{F}\ $		Feet slip $w_{13} \times \ \mathbf{b}^T \cdot \mathbf{1}(\ \mathbf{v}_f\ > \mathbf{v}_f^{th} \vee \ \dot{\theta}_f\ > \dot{\theta}_f^{th})\ $	
Collision $w_{14} \times \text{collision}$		Penalize joint limit $w_{15} \times \mathbf{q}_{violation}$	
Weights			
$w_1=1$	$w_2=1$	$w_3=1$	$w_4=1$
$w_5=-1$	$w_6=-0.05$	$w_7=-8e^{-4}$	$w_8=-5e^{-4}$
$w_9=-8e^{-3}$	$w_{10}=-0.04$	$w_{11}=-0.04$	$w_{12}=-2e^{-3}$
$w_{13}=-0.12$	$w_{14}=-1$	$w_{15}=-1$	

Table 3. Rewards for training the manipulation and base velocity command policy.

Manipulation and Base Command Rewards			
Task rewards			
Goal position tracking $w_{16} \times e^{-\ \mathbf{G}_B\ /\sigma_1}$		Projected goal position tracking see Subsection 3.3	
Auxiliary rewards			
Action $w_{17} \times \ \mathbf{a}_{mc}\ $		Action rate $w_{18} \times \ \mathbf{a}_{mc}^t - \mathbf{a}_{mc}^{t-1}\ $	
Action smoothness $w_{19} \times \ \mathbf{a}_{mc}^t - 2\mathbf{a}_{mc}^{t-1} + \mathbf{a}_{mc}^{t-2}\ $		Penalize $\dot{\mathbf{v}}$ $w_{20} \times \Sigma \dot{\mathbf{v}}_B^2$	
Penalize $\ddot{\theta}_B$ $\ddot{\theta}_B: w_{21} \times \Sigma \ddot{\theta}^2$		Collision $w_{22} \times \text{collision}$	
Penalize joint limit $w_{23} \times \mathbf{q}_{violation}$		Penalize $\dot{\mathbf{q}}$ $w_{24} \times \ \dot{\mathbf{q}}\ $	
Penalize $\ddot{\mathbf{q}}$ $\ddot{\mathbf{q}}: w_{25} \times \ \ddot{\mathbf{q}}\ $			
Weights			
$w_{16}=1$	$w_{17}=-0.04$	$w_{18}=-0.04$	$w_{19}=-0.04$
$w_{20}=-0.01$	$w_{21}=-5e^{-6}$	$w_{22}=-1$	$w_{23}=-8e^{-4}$
$w_{24}=-8e^{-2}$	$w_{25}=-1$		

icy also require tracking the given goal. Higher rewards are given as the robot gets closer to the task goal and maintains contact for a longer duration.

In Table 3, \mathbf{a}_{mc} is concatenated vector of \mathbf{a}_{man} and \mathbf{a}_{cmd} . $\mathbf{q}_{violation}$ is number of limit violated joints.

3.2. Adaptive locomotion

3.2.1 Gravitational moment reward

The objective of locomotion policy is to calculate target joint positions for the locomotion actions, enabling the robot to follow a given base command while preventing falling.

In the state-of-the-art work [10], preventing falls is achieved by minimizing two factors. One is the variable height inverted pendulum (VHIP) angle, which is an angle between z-axis of the global frame and a vector from the center of pressure (CoP) to the center of mass (CoM). The other one is the acceleration of the angle. While minimizing the VHIP angle helps the robot not to fall, it only considers feet that are in contact with the ground for balancing, as the CoP only can be calculated using the contact feet. However, it is crucial to recognize that the feet, which are not currently in contact with the ground, can still contribute to prevent falling in subsequent timesteps by preparing future contact feet position. Balancing should be viewed as a continuous process rather than a series of discrete states. Furthermore, our findings from Subsection 4.2 revealed that the adaptive locomotion method considering the VHIP angle, as proposed by [10], becomes unstable when the CoM is continuously shifted due to the movement of the

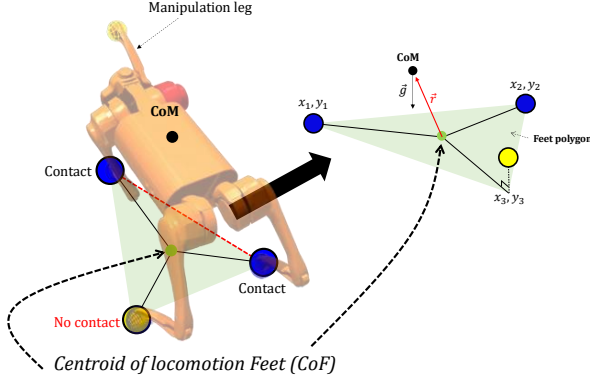


Fig. 3. This figure illustrates the front right leg being used as the manipulation leg. Blue circles indicate contact state, while yellow circles indicate non-contact state. The light green triangular area represents the locomotion feet polygon, which is formed by the projected end-effectors of the locomotion legs onto the xy plane. The centroid of locomotion feet (CoF) is marked with a green circle at the centroid of the polygon.

manipulation leg while pedipulation.

To address these issues, our study proposes a novel method for preventing falls that incorporates the centroid of locomotion feet (CoF), thus taking into account the potential contributions of all locomotion feet, whether in contact with the ground or not. We can calculate the CoF using the leg states ℓ , where 1 represents a locomotion leg and 0 represents a manipulation leg. Thus, the CoF is

$$CoF = \frac{1}{\sum_{i=0}^3 \ell_i} \left\langle \sum_{j=0}^3 \ell_j x_j, \sum_{j=0}^3 \ell_j y_j \right\rangle.$$

Where, ℓ_i is i^{th} leg state indicator. Since we consider one manipulation leg in this paper, $\sum_{i=0}^3 \ell_i > 0$ is always satisfied.

Consequently, we define a reward for the gravitational moment at the CoM with respect to the CoF as a pivot point as $w_{\mathbf{M}_g} \times e^{-|\mathbf{M}_g|/\sigma_g}$, where $\mathbf{M}_g = \vec{r} \times \vec{g}$. \vec{r} is a vector from CoF to CoM, \vec{g} is gravity vector, see Fig. 3. We use $w_{\mathbf{M}_g} = 0.5$, $\sigma_g = 10$.

In this paper, we assume the terrain is flat. On flat terrain with a known gravity direction, minimizing $|\mathbf{M}_g| = |\vec{r} \times \vec{g}|$ about the CoF promotes alignment between \vec{r} and \vec{g} , which reduces the horizontal displacement between the ground projected CoM and CoF (Fig. 3). We therefore use $w_{\mathbf{M}_g} \times e^{-|\mathbf{M}_g|/\sigma_g}$ as a practical surrogate objective under these conditions. This means, by providing the CoF as an appropriate CoM position for maintaining balance, the robot can prevent falling.

Although the CoF calculation considers only the locomotion legs because the manipulation leg's role is not to contribute to locomotion but to perform goal tracking, since all robot bodies including the manipulation leg are consid-

ered in calculating the CoM, balance can be maintained by minimizing the gravitational moment at the CoM using the CoF as a pivot point, even when the manipulation leg moves freely.

3.2.2 Locomotion feet area reward

To prevent a robot falling in flat terrain, it is helpful that the CoM ground projection remains within the support polygon, which is a polygon formed by the contact feet. For this purpose, Luo *et al.* [10] assigned penalties proportional to the maximum distance between the projected CoM and the edge of the support polygon. Although this method moves the CoM away from the polygon's edge, it can have an unintended effect: the contact points might shift to minimize the maximum CoM-to-edge distance, which reduces the polygon's area. This reduced area compromises falling, as it increases the risk of the CoM projection moving outside the polygon during pedipulation movements.

Moreover, as mentioned in Subsection 3.2.1, since feet that are not in contact can still contribute to prevent falling in subsequent timesteps, we considered the locomotion feet polygon rather than the support polygon. The locomotion feet polygon is formed by projecting the end-effectors of the locomotion legs onto the xy -plane. By including feet in the polygon calculation even when they are not in contact, we enable non-contacting feet to contribute to prevent falling. With this strategy, we propose a method that both expands the polygon's area and maintains the projected CoM at a safe distance from the polygon's edges. Our approach is to provide a reward for increasing the area of the triangle, which contains the closest distance to the edge, formed when connecting the CoF to each vertex of the locomotion feet polygon. If the triangle has the shortest height measured from the CoF to its base, the height is equivalent to the smallest distance from the CoF to the edge of the locomotion feet polygon. By increasing the area of this triangle, we can simultaneously increase both the shortest distance from the CoF to the edges of the polygon and the overall area of the locomotion feet polygon. Thus, the locomotion feet area reward is defined as: $w_{\mathcal{A}} \times \min(e^{\mathcal{A}/\sigma_a} - 1, 1)$ where \mathcal{A} represents the area of the triangle that has shortest height, which is discussed above, and we use $w_{\mathcal{A}} = 0.2$, $\sigma_a = 0.05$. Maximizing the selected triangle's area increases the shortest distance from the CoF to the polygon edges, enlarging a geometric stability margin against edge crossings during pedipulation.

3.2.3 Locomotion action selection

When implementing a unified locomotion policy for both normal four-legged locomotion and adaptive locomotion, it is crucial to ensure that actions for the manipulation leg do not influence the actions for the locomotion legs. Thus, we propose a method named *locomotion action selection* to address this issue.

When a quadruped robot performs normal locomotion using all four legs as locomotion legs, it uses all 12 dimensions of actions, $\mathbf{a}_{loc} \in \mathbb{R}^{12}$, output from the locomotion policy for locomotion. However, when one leg is used as a manipulation leg, 3 dimensions among the 12-dimensional actions from the locomotion policy are not used for locomotion. Therefore, if these unused actions are backpropagated, it would result in the backpropagation of dummy derivative values, since these values were not actually used in the locomotion process.

To address this issue, we leverage leg states ℓ that indicate whether each leg is used for locomotion or manipulation to filter out irrelevant information during backpropagation of the locomotion policy training. We apply element-wise multiplication, \otimes in Fig. 2, between the locomotion actions \mathbf{a}_{loc} and the leg states ℓ to make the actions and their derivative corresponding to manipulation legs to be zero. This gating prevents dummy derivatives from unused action channels and reduces gradient conflicts between stance stabilization and manipulation objectives during backpropagation.

3.3. Adaptive manipulation

Through adaptive locomotion, Subsection 3.2, we achieve the ability to maintain balance and locomote using only the locomotion legs. Additionally, we can perform pedipulation by tracking a target goal using the manipulation leg.

For target goal tracking, the manipulation foot must reach a given target goal in 3D space. Goal tracking can be achieved easily by designing a reward that reduces the distance between the target goal and the end-effector. However, the challenge lies in where the distance to the goal does not change uniformly with the robot movement. For instance, when a quadruped robot moves its entire body to the target goal, the distance to the goal continuously decrease maximizing the goal tracking reward. However, when only the manipulation leg moves while the base position remains fixed, the goal tracking reward stops increasing once the manipulation foot has extended maximally toward the target goal. This issue leads to the robot fixing its manipulation leg position and moving only its base for target goal tracking.

To address this problem, we utilize a reachability map of the manipulation leg. In this work, the reachability map is approximated as a sphere centered at the hip joint, with a radius equal to the maximum kinematic extension of the leg. When a task goal lies outside this spherical region, it is projected onto the sphere’s boundary to ensure that the manipulation policy always receives feasible goals. Through this approach, independent from approaching the task goal through base velocity commands, we can reduce the task goal tracking error by moving the manipulation leg. If we do not use the projected goal, for distant task goals, the reward changes from moving the base position become

more significant than the reward changes from moving the manipulation leg. This is because once the manipulation leg is fully extended toward the task goal, moving only the manipulation leg cannot increase the reward anymore. Therefore, by using the projected goal, we can generate meaningful manipulation leg actions for all goals. Once the goal comes within range, the manipulation foot executes the final approach to precisely reach the target. Thus, we define the projected goal position tracking reward as $w_{\hat{\mathbf{G}}_B} \times e^{-\|\hat{\mathbf{G}}_B - \hat{\mathbf{r}}_B\|/\sigma_1}$, where $\hat{\mathbf{G}}_B$ is projected goal position in base frame, $\hat{\mathbf{r}}_B$ is foot position of manipulation leg in base frame, and we use $w_{\hat{\mathbf{G}}_B} = 5$.

4. EXPERIMENTS

In this section, we study the effectiveness of our proposed method by conducting experiments of task goal tracking for pedipulation. For successful task goal tracking, three requirements must be met: 1) The robot body must prevent falling while performing the task, 2) The tracking error must be low to execute the given task precisely, and 3) The robot is able to perform the given task as quickly as possible.

In Subsection 4.2, we show how robustly the robot can control its posture during task execution. In Subsection 4.3, we demonstrate lower tracking error for randomly sampled task goals compared to state-of-the-art research, and in Subsection 4.4, we show that when given consecutive task goals, our method can perform tasks more quickly and effectively compared to the state-of-the-art method.

Because pedipulation is an emerging topic that has only recently attracted broad interest, the number of directly comparable studies remains limited. Accordingly, we focus on two representative families: 1. a state-of-the-art pedipulation method with a fixed manipulation leg [14], and 2. CoP-based stabilization for adaptive locomotion [10]. These baselines align with our problem formulation and provide a fair point of comparison within the scope of this study.

4.1. System and simulator

This study was conducted using the Unitree Go1 robot [24]. Training was performed in Isaac Gym [25], which enables massively parallel simulation on GPUs and significantly accelerates reinforcement learning. For evaluation, we employed RaiSim [26] because of its more accurate physics engine and its established use as a benchmark environment for legged locomotion. Furthermore, by conducting training and evaluation in different simulation platforms, we demonstrate that our algorithm performs robustly across diverse environments. Training was performed in the Isaac Gym, and evaluation was performed in the RaiSim. A total of 4096 agents were trained in parallel with an episode duration of 20 seconds. All policies were

trained using PPO (Proximal Policy Optimization) [27] and learned through the actor-critic methodology. Both the actor network and critic network consist of MLPs with sizes [512, 256, 128], and they receive the same values as input. All policies were trained for 2000 iterations using an NVIDIA GeForce 4090 GPU. Domain randomization was applied only during training to encourage robustness to variations in payload, friction, and motor strength. For evaluation, fixed parameters were used to enable consistent comparisons across methods, following established practices in legged locomotion research [6].

4.2. Moving manipulation leg while standing

In the first experiment, we tested how well the robot could maintain balance while moving its manipulation leg in a standing position. During pedipulation tasks, even when standing still, the robot’s center of mass (CoM) shifts as the manipulation leg moves. This shift in CoM can cause the robot to lose balance and fall. In this experiment, we set the base command to zero to keep the robot stationary, then continuously moved the task goal position within the reachable region to test how robustly the robot could maintain its posture while moving the manipulation leg. To achieve robust balance, we proposed a gravitational moment minimization reward that uses the centroid of locomotion feet as a pivot point.

Fig. 4 shows the experimental results for four methods.

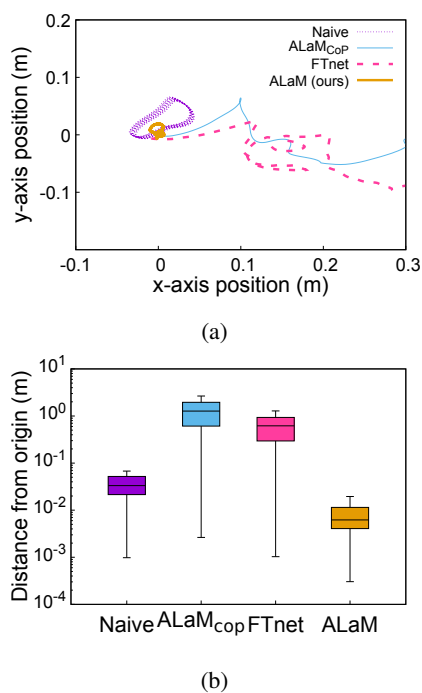


Fig. 4. (a) Changes of the center of mass (CoM) position when moving the manipulation leg while standing. (b) Distribution of distances between each point on the trajectory and the origin.

The Naïve approach uses the reward from LeggedGym [7] locomotion learning which is widely used as a locomotion baseline. While this method doesn’t explicitly include rewards for balance control only with three legs, we found that it learned to balance using three legs to avoid ground collision during locomotion. ALaM_{cop} uses our proposed rewards but uses the center of pressure instead of the centroid of feet as the pivot point for gravitational moment. FT-net [10], which is the state-of-the-art method in fault-tolerant locomotion, uses a reward that minimizes the variable height inverted pendulum (VHIP) angle to maintain balance with three legs when one leg becomes unusable. As shown in Fig. 4(a), both the Naïve and our approach, ALaM, maintain a standing posture, while ALaM_{cop} and FT-net fail to maintain standing and shift their base position for posture control. This occurs because while our method attempts to align the CoM with a geometrically fixed point, centroid of feet CoF, ALaM_{cop} and FT-net try to align the CoM with the center of pressure (CoP). Since the CoP can continuously change even in a standing state according to the ground reaction force of each foot, these methods adjust the CoP position in response to CoM changes, losing balance when the CoM moves outside the support polygon.

As a result, our proposed method was able to maintain body balance while performing tasks and showed the lowest CoM position variation among the comparison groups.

4.3. Measuring goal tracking error

To evaluate task goal tracking error of our proposed method, we measured task goal tracking error for 2,000 random task goal positions within ranges of -5 m to 5 m along both x and y axes, and 0.2 m to 0.5 m along the z -axis from the world frame’s origin. One example is shown in Fig. 5(a) as red sphere. The measurement was calculated by averaging the tracking error during 1-second after 14 seconds from the episode start, considering the time needed to reach far task goals. As a result, we obtained an average tracking error of 2.12 cm, which is approximately half of the 4.3cm error reported in [14]. Fig. 5(b) shows the tracking error for each height. Since the standing height of the Unitree Go1 robot used in the experiment is 0.3 m, we found that it shows low error rates for task goals up to its body height, but relatively higher error rates for task goals around 0.4 m-0.5 m, which are higher than its body.

4.4. Task goal touch

While the previous experiment evaluated the accuracy of task goal tracking, in this section, we conducted two types of experiments to verify how quickly and efficiently each task goal can be tracked when performing consecutively given tasks: the shuttle run test and the radial reach test.

4.4.1 Shuttle run test

The shuttle run test involves repeatedly touching given task goals positioned in front of and behind the robot. The

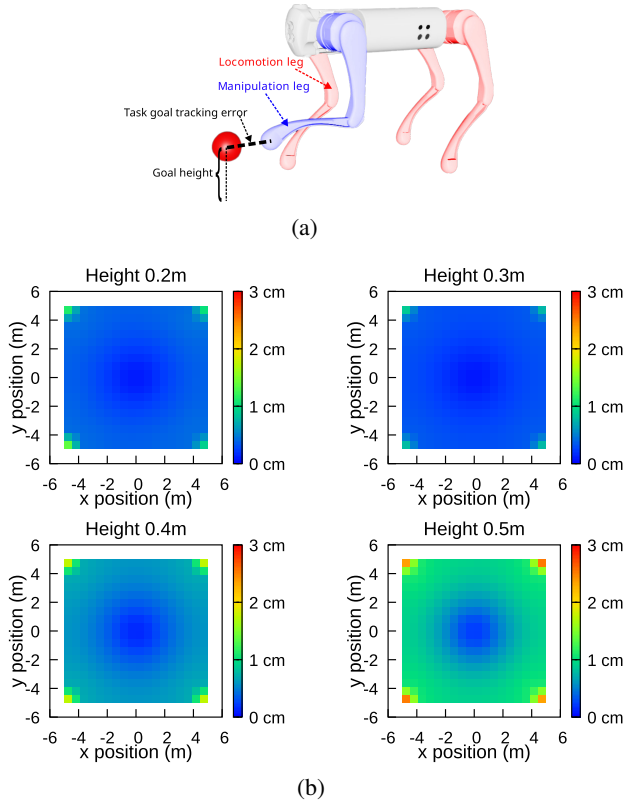


Fig. 5. Task goal tracking error. Totally, 2000 task goals are evaluated. Each point represents task goal position and the color represents tracking error. Average tracking error is approximately 2.12 cm.

task goals are located 2m in front and 2m behind the robot, and the time taken to touch these goals a total of 10 times is measured. Our method demonstrates the ability to quickly move back and forth while switching the manipulation leg according to the position of the given task goal. In contrast, [14] assigns fixed roles to each leg, either locomotion or

manipulation. This means only one specific leg can be used for manipulation, and it cannot switch between legs. As a result, the robot must turn its entire body after touching each task goal. Due to this difference, while the control group took 42.2 seconds, our method completed the task in just 28.76 seconds.

4.4.2 Radial reach test

The radial reach test is similar to the shuttle run test but evaluates reaching performance in all directions. In this test, task goals are positioned at distances ranging from 1m to 5m, placed every 30 degrees. The test results are shown in Fig. 6.

In the radial graph, the color of each cell indicates the time taken to reach the task goal placed at that position. Comparing Figs. 4(a) and 4(b), we can see that Fig. 4(a) shows similar color distributions between front and rear areas, Fig. 4(b) shows darker colors in the rear compared to the front. This difference shows that both methods take similar time for forward task goals, [14] takes longer for rear task goals. This difference arises because our proposed method can adaptively select which leg to use as the manipulator leg based on the relative position of the task goal, whereas [14] must use a fixed leg as the manipulator.

This distinction becomes even clearer when comparing Figs. 4(c) and 4(d). These graphs show the trajectories taken by our method and [14] to reach radial task goals at 5 meters distance. The key points to observe are the shape and direction of the arrows forming each trajectory. Each arrow indicates which leg is being used as the manipulation leg. Our method quickly reaches task goals by selecting appropriate legs for manipulation based on the task goal's direction, while [14] uses the same leg for manipulation in all cases. Furthermore, the arrow direction indicates the robot's facing direction. Considering that all trajectories move from the center to outward, we can see that the robot which uses [14] faces the task goal while moving. On

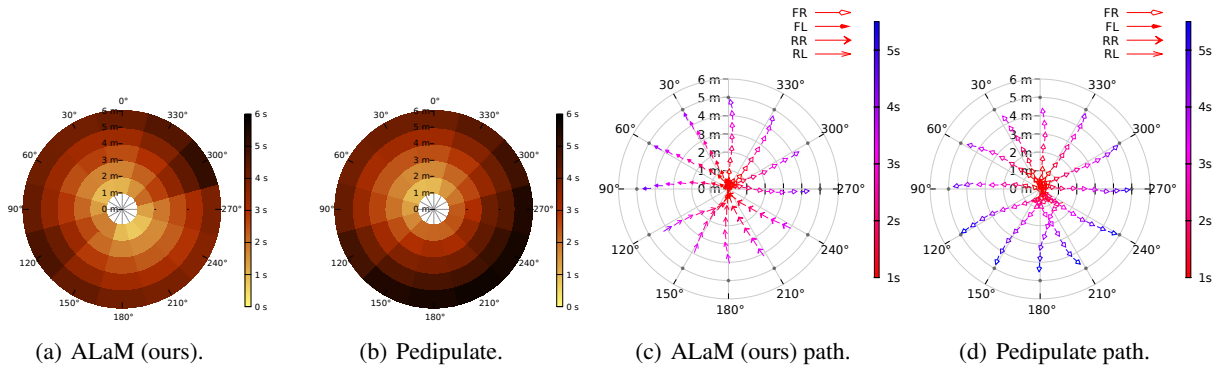


Fig. 6. (a) and (b) are radial graph of the time taken to reach task goals at each position for ALaM (ours) and pedipulate [14], respectively. (c) and (d) illustrate the trajectory of our method and [14], respectively. Each arrow indicates which leg was used as the manipulation leg.

the other hand, with our method, the robot keeps its body direction for the selected manipulation leg to be close to the task goal.

4.5. Far away goal

This experiment compares state-of-the-art pedipulation research [14] with our proposed method, specifically evaluating their goal tracking performance when the task goal position significantly exceeds the distances used during training. Both methods utilize the task goal position as input to their policies, with a maximum training distance of 2 m. The experiment calculated the success rate by performing pedipulation up to the task goal placed at 10 meters with 0.5 m intervals. A success was defined as when the end-effector of the robot’s manipulation leg came within 0.05 m of the task goal. Each task goal was executed 100 times. Pedipulate [14] generates movements for both manipulation and locomotion legs simultaneously, while our method employs a two-step approach. This architectural difference proves crucial for long-distance performance. By generating base command firstly rather than directly generating joint action, our method can handle long-distance target goal. As shown in Fig. 7, pedipulate’s tracking success rate declines when the task goal extends beyond the maximum training distance, whereas our method maintains consistent tracking performance regardless of the task goal distance.

5. CONCLUSIONS

In this paper, we conducted research on task goal tracking using a manipulation leg while maintaining balance when given a task goal for pedipulation, which is to use one leg of a quadruped robot as a manipulator. For efficient task goal tracking, three requirements must be met: 1) maintaining robust balance while performing tasks, 2) reaching task goals precisely, and 3) approaching task goals quickly and efficiently. To achieve this, we proposed a gravitational moment minimization reward that uses the centroid of lo-

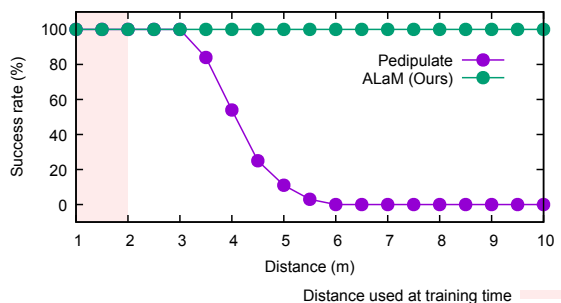


Fig. 7. Task goal tracking success rate based on distance. Our method, in green, shows no change in success rate according to distance. In contrast, the success rate of pedipulate, in purple, decreases beyond 3 meters, where the goal is not used for training.

comotion feet as a pivot point. Unlike center of pressure (CoP) based balancing [10], our Centroid of locomotion Feet (CoF) formulation remains stable during pedipulation despite CoM shifts induced by the manipulation leg. Additionally, through task goal tracking experiments comparing with state-of-the-art research [14], which learns to use a fixed leg as a manipulator for pedipulation, we showed that our method’s ability to adaptively select manipulation legs based on goal position enables more precise, faster, and efficient task execution compared to existing research.

While proposed our gravitational moment reward is conceptually related to CoP-based balancing, the proposed CoF formulation introduces a distinct extension by defining stability with respect to the centroid of locomotion feet rather than only ground contacts. Unlike CoP, which becomes undefined during reconfiguration or flight, CoF remains applicable and allows the policy to prepare stable postures before touchdown. Although we do not provide a rigorous theoretical proof in this work, we demonstrate through extensive experiments that the CoF formulation effectively stabilizes adaptive pedipulation. A formal theoretical analysis is left as an important direction for future work.

In future research, we can conduct studies on utilizing large language model and vision language model to perceive surrounding environments and understand high-level commands to generate task goals. Since we gained the ability to track task goals through this research, if we can generate task goal points or task goal trajectories based on user requirements, we will be able to perform fully automated pedipulation in response to high-level demands.

DECLARATIONS

Conflict of Interest

The authors declare that there is no competing financial interest or personal relationship that could have appeared to influence the work reported in this paper.

Authors’ Contributions

Heechan Shin proposed and developed the core idea, and implemented the simulation environment and code for training the method. Minsung Yoon implemented the baseline models for experimental comparison. Jeil Jeong conducted investigation and supported the experiments. Sung-eui Yoon supervised the overall research and contributed to the editing and reviewing the paper.

Funding

This work was supported by the IITP(Institute of Information & Coummunications Technology Planning & Evaluation)-ITRC(Information Technology Research Center) grant funded by the Korea government(Ministry of Science and ICT)(IITP-2025-RS-2020-II201460) and the

National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2023-00208506)

REFERENCES

- [1] H. Shin, M. Yoon, J. Jeong, and S. Yoon, "Alam: Adaptive locomotion and manipulation for quadruped robot," *Proc. of the 25rd International Conference on Control, Automation and Systems (ICCAS 2025)*, 2025.
- [2] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation," *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, 2001.
- [3] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560-1567, 2018.
- [4] G. Bleidt, M. J. Powell, B. Katz, J. D. Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.
- [5] Y. Ding, A. Pandala, C. Li, Y. H. Shin, and H. W. Park, "Representation-free model predictive control for dynamic motions in quadrupeds," *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1154-1171, 2021.
- [6] 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, eaa5872, 2019.
- [7] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," *Proc. of Conference on Robot Learning*, 2022.
- [8] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid motor adaptation for legged robots," *Robotics: Science and Systems XVII*, 2021.
- [9] Z. Fu, A. Kumar, J. Malik, and D. Pathak, "Minimizing energy consumption leads to the emergence of gaits in legged robots," in *Proc. of Conference on Robot Learning*, 2022.
- [10] Z. Luo, E. Xiao, and P. Lu, "FT-Net: Learning failure recovery and fault-tolerant locomotion for quadruped robots," *IEEE Robotics and Automation Letters*, vol. 8, no. 12, pp. 8414-8421, 2023.
- [11] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *The International Journal of Robotics Research*, vol. 43, pp. 572-587, 2024.
- [12] Z. Fu, X. Cheng, and D. Pathak, "Deep whole-body control: Learning a unified policy for manipulation and locomotion," *Proc. of Conference on Robot Learning*, 2023.
- [13] J. P. Sleiman, F. Farshidian, and M. Hutter, "Versatile multi-contact planning and control for legged loco-manipulation," *Science Robotics*, vol. 8, eadg5014, 2023.
- [14] P. Arm, M. Mittal, H. Kolvenbach, and M. Hutter, "Pedipulate: Enabling manipulation skills using a quadruped robot's leg," *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2024.
- [15] Y. Ji, Z. Li, Y. Sun, X. B. Peng, S. Levine, G. Berseth, and K. Sreenath, "Hierarchical reinforcement learning for precise soccer shooting skills using a quadrupedal robot," *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.
- [16] X. Cheng, A. Kumar, and D. Pathak, "Legs as manipulator: Pushing quadrupedal agility beyond locomotion," *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2023.
- [17] Y. Ji, G. B. Margolis, and P. Agrawal, "Dribblebot: Dynamic legged manipulation in the wild," *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2023.
- [18] C. D. Bellicoso, F. Jenelten, P. Fankhauser, C. Gehring, J. Hwangbo, and M. Hutter, "Dynamic locomotion and whole-body control for quadrupedal robots," *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.
- [19] X. Li, X. Zhang, J. Niu, and C. Li, "A stable walking strategy of quadruped robot based on zmp in trotting gait," in *Proc. of IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 858-863, 2022.
- [20] J. Liu, H. Chen, P. M. Wensing, and W. Zhang, "Instantaneous capture input for balancing the variable height inverted pendulum," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7421-7428, 2021.
- [21] G. Bellegarda, and A. Ijspeert, "CPG-RL: Learning central pattern generators for quadruped locomotion," *IEEE Robotics and Automation Letters*, vol. 7, pp. 12547-12554, 2022.
- [22] I. M. A. Nahrendra, B. Yu, and H. Myung, "DreamWaQ: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning," *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2023.
- [23] G. B. Margolis, and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," *Proc. of Conference on Robot Learning*, 2023.
- [24] "Go1." Unitree, <https://www.unitree.com/go1>.
- [25] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac Gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint*, arXiv:2108.10470, 2021.
- [26] R. T. Inc, <https://raisim.com/>, in *RaiSim v1.1.7*.
- [27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint*, arXiv:1707.06347, 2017.



Heechan Shin received his Ph.D. degree from the School of Computing from KAIST in 2025. His research interests include pedipulation of quadruped robot, locomotion, and motion planning. He currently joins Samsung Research of Samsung Electronics.



Minsung Yoon received his M.S. degree from the School of Computing from KAIST in 2022 and is currently a Ph.D. candidate student. His research interests include learning-based initialization, and adaptive control of a quadruped robot on moving platform.



Jeil Jeong received his M.S. degree in robotics program from KAIST in 2025 and is currently a Ph.D. candidate student. His research interests include posture-adaptive locomotion of quadruped robot over unstructured environments.



Sung-eui Yoon received his Ph.D. degree in computer science from the University of North Carolina at Chapel Hill in 2005. His research interest is techniques for Rendering, Robotics, and Computer vision. He is a professor at KAIST and leads the SGVR laboratory.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.