2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

MQE: Unleashing the Power of Interaction with Multi-agent Quadruped Environment

Ziyan Xiong* Bo Chen Shiyu Huang† Wei-Wei Tu Zhaofeng He Yang Gao

***Abstract*—The advent of deep reinforcement learning (DRL) has significantly advanced the field of robotics, particularly in the control and coordination of quadruped robots. However, the complexity of real-world tasks often necessitates the deployment of multi-robot systems capable of sophisticated interaction and collaboration. To address this need, we introduce the Multi-agent Quadruped Environment (MQE), a novel platform designed to facilitate the development and evaluation of multi-agent reinforcement learning (MARL) algorithms in realistic and dynamic scenarios. MQE emphasizes complex interactions between robots and objects, hierarchical policy structures, and challenging evaluation scenarios that reflect real-world applications. We present a series of collaborative and competitive tasks within MQE, ranging from simple coordination to complex adversarial interactions, and benchmark state-of-the-art MARL algorithms. Our findings indicate that hierarchical reinforcement learning can simplify task learning, but also highlight the need for advanced algorithms capable of handling the intricate dynamics of multi-agent interactions. MQE serves as a stepping stone towards bridging the gap between simulation and practical deployment, offering a rich environment for future research in multi-agent systems and robot learning. For open-sourced code and more details of MQE, please refer to https://ziyanx02.github.io/multiagent-quadruped-environment/.**

***Index Terms*—quadrupedal locomotion, multi-agent reinforcement learning, hierarchical reinforcement learning**

I. Introduction

In recent years, the field of robotics has witnessed remarkable advancements in the locomotion control of quadruped robots. These strides have been largely propelled by the utilization of deep reinforcement learning (DRL) in simulation environments [1], [2], enabling quadrupeds to perform a wide array of tasks with unprecedented agility and efficiency, from trespassing [3]–[5] rugged terrains to carrying out manipulative tasks [6]–[8], the capabilities of current quadrupeds represent a significant leap forward in robotic autonomy and versatility.

Ziyan Xiong and Yang Gao are with the Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China (e-mail: xiongzy20@mails.tsinghua.edu.cn, gaoyangiiis@mail.tsinghua.edu.cn).

Bo Chen and Zhaofeng He are with AI Institute, Beijing University of Posts and Telecommunications, Beijing, China (e-mail: {chenbo, zhaofenghe}@bupt.edu.cn).

Shiyu Huang is with Zhipu AI, Beijing 100084, China (e-mail: shiyu.huang@aminer.cn).

Wei-Wei Tu is with 4Paradigm Inc., Beijing 100084, China (e-mail: tuweiwei@4paradigm.com).

Yang Gao is with Shanghai Artificial Intelligence Laboratory and Shanghai Qi Zhi Institute, Shanghai, China.

*Work was done during the intern at 4Paradigm Inc..

†Work was done while at 4Paradigm Inc..

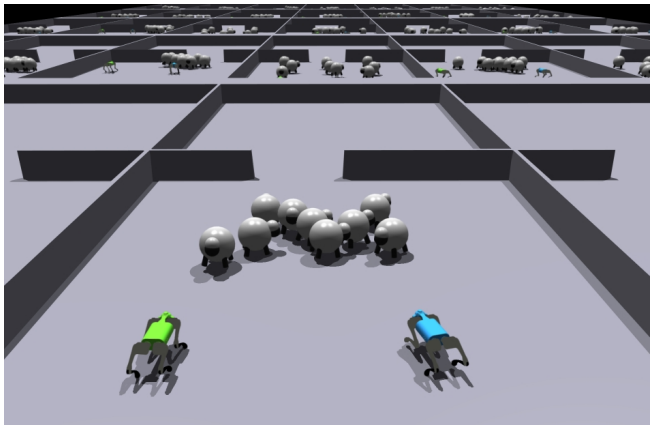


Fig. 1. Agents learning to herd sheep in hundreds of parallel environments.

Despite these advancements, the intricacy and variety of real-world tasks often surpass the capability of single-robot systems. These systems encounter fundamental constraints in performing tasks in situations such as large-scale environmental monitoring [9] and intricate logistical challenges within warehousing environments [10]–[12], highlighting the critical need for the development and deployment of multi-robot systems. Such scenarios demand a level of coordination, collaboration, and perception that single-robot systems, no matter how complex they are, cannot achieve on their own.

Addressing the need for coordinated multi-robot systems, the research community has done much research on multi-agent reinforcement learning in different multi-agent environments, including the Multi-Agent Particle Environment (MPE) [13], [14], Google Research Football [15], [16], and the StarCraft Multi-Agent Challenge (SMAC) [17]. While these environments have significantly contributed to the advancement of MARL by providing platforms for developing and evaluating algorithms, we still lack tangible intermediaries to apply multi-agent policies to real-world tasks. The simplified or even frictional interactions in these environments often fall short of replicating the complex interference and dynamics present in real-world scenarios, such as imperfect action execution caused by direct or indirect interactions among real robots. These issues highlight a gap between the current research efforts and practical real-world applications.

In response to these limitations, we propose the Multi-

agent Quadruped Environment to bridge this gap, emphasizing 1) complex and dynamic interactions between objects and robots, 2) a hierarchical structure of policies that transforms complex control problems into simple multi-agent tasks, and 3) realistic and challenging evaluation scenarios of MARL focusing on either assessing the viability of simplifying complex multi-robot tasks through the hierarchical structure of policies or exploring robust locomotion policies amidst the unpredictable interactions among robots.

To explore the capabilities of our environment and the effectiveness of current MARL methodologies, we experiment with multi-agent reinforcement learning algorithms, comparing those with a hierarchical structure of policies intended to streamline the learning against those without such a structure. Our findings reveal that even advanced algorithms struggle to solve the more difficult tasks presented in our environment. This underscores not only the complexity and realism of the challenges we introduce but also the pressing need for our environment as a tool for developing more sophisticated and capable RL algorithms. As inspiration for future works, we also showcase several promising research directions in this environment, i.e. hierarchical RL and RL enhanced by self-play strategies.

II. Related Works

A. Reinforcement Learning for Quadruped Control

The exploration of learning-based locomotion control has significantly advanced the capabilities of quadruped robots, enabling them to perform dynamic tasks such as robust walking on complex terrains [3], [18], [19], high-speed running [20]–[22], executing parkour abilities [23], climbing on obstacles [18], [24]–[26], and playing football [6], [7]. These achievements highlight the importance of simulation environments [27], [28] in facilitating the development of these complex locomotion strategies. Various simulators such as Gazebo, PyBullet, IsaacGym, and MATLAB have been employed, each offering different levels of fidelity, ease of use, and computational efficiency. Notably, IsaacGym, with its GPU-accelerated architecture, stands out for its enhanced sample efficiency, which is crucial for the rapid training of robots.

Despite these advancements, the focus has largely been on single-agent scenarios. However, as tasks become more complex and demand higher levels of interaction, the need for multi-agent simulation environments becomes evident. The emergence of multi-robot simulation environments in other domains, such as drones [29]–[31] and dexterous manipulators [32], [33] highlights the necessity for similar advancements in quadruped robotics. Addressing this, our work introduces a novel multi-agent quadruped environment, aiming to bridge this gap and facilitate the exploration of cooperative and competitive dynamics among robots, thus expanding the scope of quadruped research into multi-robot operations and opening new avenues for the application of these technologies.

B. Multi-agent Reinforcement Learning

Multi-agent reinforcement learning has increasingly received much research interest, propelled by its potential to tackle complex tasks through the collaborative and competitive interactions of multiple agents. The field distinguishes between two principal categories of MARL algorithms: value-based and policy-based. Value-based algorithms, including QMIX [34], VDN [35], and QPLEX [36], focus on estimating the value of actions in given states to determine the best course of action. On the other hand, policy-based algorithms like MADDPG [13], MAPPO [37], and MAT [38] directly learn a policy that maps states to actions, potentially offering more flexibility in continuous action spaces.

The development of MARL algorithms is established on the environments used for training and testing. Current MARL environments, such as MPE and SMAC, often present simplified, idealized scenarios, lacking complicated dynamics, i.e. in robotics simulation, to evaluate the capability of MARL algorithms. However, existing multi-robot environments primarily assess MARL algorithms through navigation tasks [39]–[42], which significantly differ from the simplified settings commonly employed in MARL research. This discrepancy highlights the necessity for integrating both kinds of environments.

III. Multi-agent Quadruped Environment

The Multi-agent Quadruped Environment is developed atop the widely recognized reinforcement learning framework for legged robots, *legged_gym*, which operates on NVIDIA's Isaac Gym. An overview of MQE can be found in Fig. 1, accompanied by a comparative analysis of MQE against existing legged robot simulations in Table I. In addition to the wide range of quadruped models available through *legged_gym*, MQE introduces several notable features:

a) Multi-robot interaction: Our environment is designed to seamlessly incorporate actuated robots, stationary and manipulable objects, and non-player characters (NPCs) which are objects that adhere to predetermined policies within each parallel-simulated environment. The inclusion of multiple robots and interactive objects intensifies the complexity of learning as each robot, or agent must navigate interactions with others, necessitating collaboration or evasion to mitigate the impacts of those interactions.

b) Modular terrain registration: Similar to many other legged robot environments, the simulated ground is segmented into parallel tracks to facilitate domain randomization of terrain or to implement curriculum learning strategies.

TABLE I
Comparison between Quadruped Environments

	Multi-agent	Hierarchical policy	Simulator
champ	✓	✓	Gazebo
Rex-gym	✓	✗	PyBullet
Genloco	✗	✗	PyBullet
legged_gym	✗	✓	IsaacGym
MQE (ours)	✓	✓	IsaacGym

Notably, our environment further divides each track into blocks, simplifying the creation and adaptation of terrain for various tasks. These blocks can be efficiently repurposed and reconfigured to meet diverse requirements.

c) Alternative hierarchical policy: Recognizing the challenges associated with learning a locomotion policy from scratch in a multi-agent task setting, our environment incorporates a robust walking policy as an optional foundational layer. This low-level policy responds to commands from the high-level policy. This structure allows for the simplification of some tasks as straightforward multi-agent tasks with complex movements, given that robot dynamics might not strictly follow the commands due to potential collisions, sliding, or command alterations.

d) Parallel Simulation: Leveraging the GPU-accelerated capabilities of Isaac Gym, our environment supports the execution of numerous simulations in parallel. This feature significantly enhances the sample efficiency of reinforcement learning, facilitating the achievement of high-performance policy within a constrained time.

IV. Benchamarking Tasks

Based on the previously outlined environment, we have designed 12 tasks with varying degrees of complexity and distinct characteristics for benchmarking purposes. These tasks are evenly divided into collaborative and competitive categories, with their details illustrated in Figure 2. The objective of collaborative tasks includes: 1) requiring agents to adopt asymmetric policies to achieve their goals, i.e. creating space to prevent potential collisions, and 2) necessitating cooperation among agents, i.e. passing and stopping a football. On the other hand, competitive tasks are designed to foster self-improvement among agents through self-play, as opposed to learning in a solitary agent setting, i.e. dribbling a football under the opposing defender's defense.

Actions are torques of the robot's actuated Degrees of Freedom (DoFs), or target positions that will be converted to torques through PD control. To accelerate the learning of high-level planning, we forgo the extensive training of a locomotion policy in favor of incorporating a pre-trained walk policy specifically designed for the Unitree Go1 robot [43]. This advanced locomotion policy takes 18 dimensions of command and robot states, subsequently determining the target positions for each actuated DoF, which enables precise control over the Go1 robot's movement, ensuring it walks according to the specified gait and commands. For uniformity and simplicity in our benchmarking results, we will exclusively utilize the Unitree Go1 robot.

All collaborative benchmarking tasks are formulated as decentralized partially observable Markov Decision Processes (Dec-POMDPs) with selected observation and shared rewards, although the simulator itself is fully observable. We provide all available observations within the simulator to facilitate privileged observation in actor-critic algorithms or policy distillation. Proprioception is provided to the aforementioned low-level locomotion policy and thereby is omitted from the observations delivered to the high-level policy within the benchmarking tasks. This omission aims to evaluate the effectiveness of a hierarchical policy framework, where the high-level policy operates independently from low-level locomotion control.

A. Collaborative Tasks

a) Narrow Gate: A narrow gate that permits only one robot to go through splits a room. Positioned on the same side, two robots are instructed to walk through this gate sequentially, avoiding any collisions. This task necessitates an understanding of potential collisions and the management of routine to avoid collisions, requiring different agents to learn asymmetric policies of the timing and order for optimal rewards.

b) Climb on Seesaw: Positioned next to a suspended platform, a flat seesaw's design allows one end to be elevated to the platform's height, allowing the robot to climb onto the platform. Due to the mechanism of the seesaw, successful ascension requires coordination wherein one robot occupies the farther end of the seesaw, maintaining its stability, while the second robot ascends along the seesaw. The absence of this synchrony results in the seesaw collapsing.

c) Sheep Dog (Easy): In *Sheep Dog* tasks, we introduce sheep as non-player characters (NPCs) that exhibit autonomous movement based on the following principles: 1) sheep always maintain a distance from dogs, which are the robots controlled by agents, and will accelerate to keep away from the dogs, 2) sheep always tends to get close to the central of the herd, and 3) sheep always move with randomness in velocity. Similar to *Narrow Gate*, *Sheep Dog-Easy* introduces a sheep being placed on the same side as the dogs. The objective of this task is to guide the sheep through the gate by utilizing the movement features of the sheep.

d) Sheep Dog (Hard): Building on *Sheep Dog (Easy)*, *Sheep Dog-Hard* escalates the challenge by requiring the dogs to herd nine sheep, significantly increasing the complexity of the sheep's movements. This complexity challenges agents' perception of the position and movement of sheep and their corresponding ability to really "understand" the situation. It necessitates the agents follow the sheep's behavior patterns rather than relying on the memorization of the specific trajectories of sheep and dogs.

e) Push Box: Similar to the design of the *Sheep Dog* tasks, *Push Box* utilizes the terrain established in *Narrow Gate*, with the sheep replaced by a heavy box that requires the collaborative effort of two robots to push. This task challenges to navigate and push the box through the gate. Due to the weight of the box, a dedicated locomotion policy for the pushing task will simplify the tasks. Furthermore, the task asks for cooperative behavior from both robots to maintain the box's trajectory and prevent its rotation.

f) Football (2 vs 1): We create a football field enclosed by walls that replace sidelines and goals for *Football* tasks. A defender, which is a robot serving as an NPC in this task, is programmed to position itself at the midpoint between the ball and the goal. Two robots are placed on the opposite side against the defender with a football. The target of the two

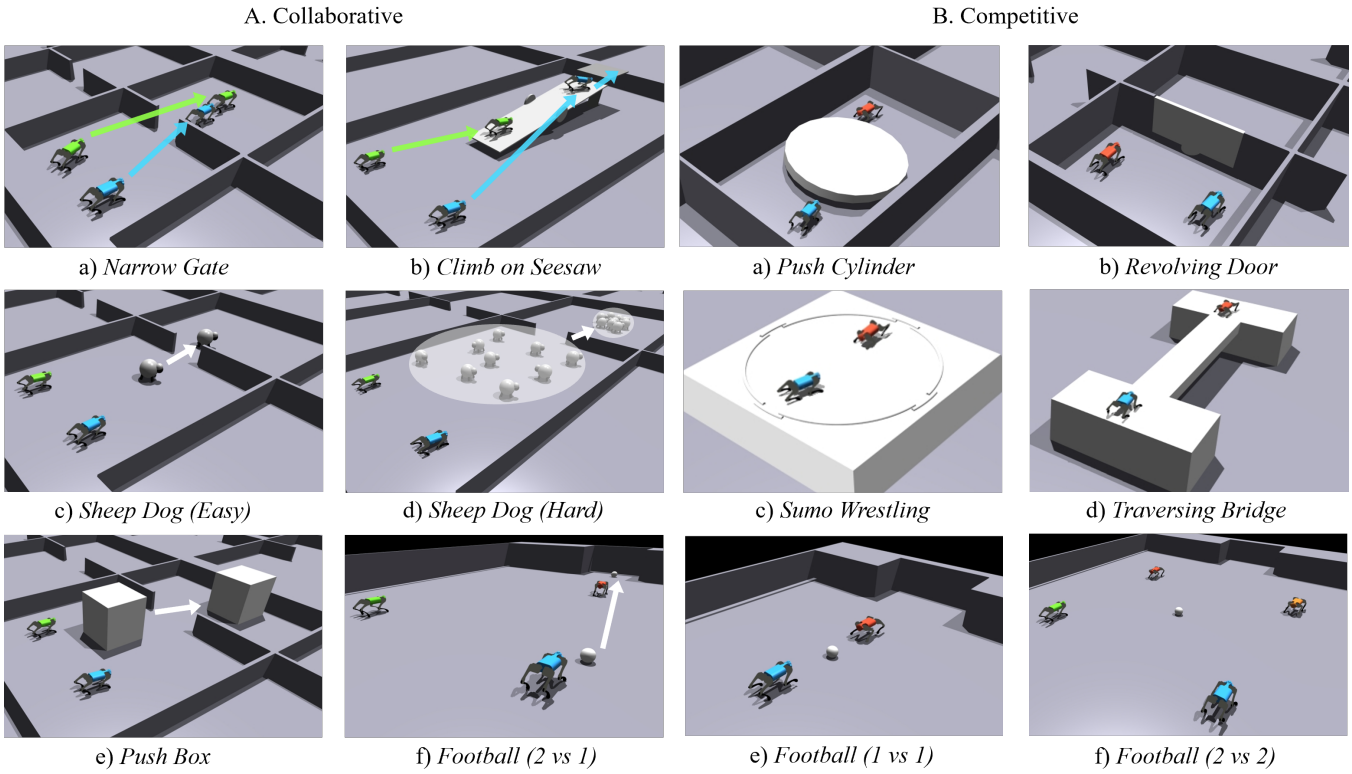


Fig. 2. Demonstration of benchmarking tasks. Blue and green robots are assigned to accomplish collaborative tasks, while red and orange robots will play against them. Arrows of different colors illustrate the intended movements of agents and objects in each task. Generally, tasks demonstrated lower are harder due to the rising demands of advanced locomotion control and environment awareness.

robots is to kick the football into the goal. With the defender maintaining a position between the goal and the ball, this task not only requires the two dogs to possess dribbling skill but also necessitates strategic passing to circumvent the defender's blockades and successfully score, demanding both refined locomotion policies and effective cooperative policy formulation.

B. Competitive Tasks

a) Push Cylinder: A cylinder centrally placed in an arena can only move sideways. Two quadruped robots, positioned on opposite sides, compete by pushing the cylinder towards each other. The robot that pushes the cylinder towards the opponent's side earns a reward. This task challenges the robots to identify optimal directions and positions for pushing, leveraging the cylinder's circular cross-section for axial force.

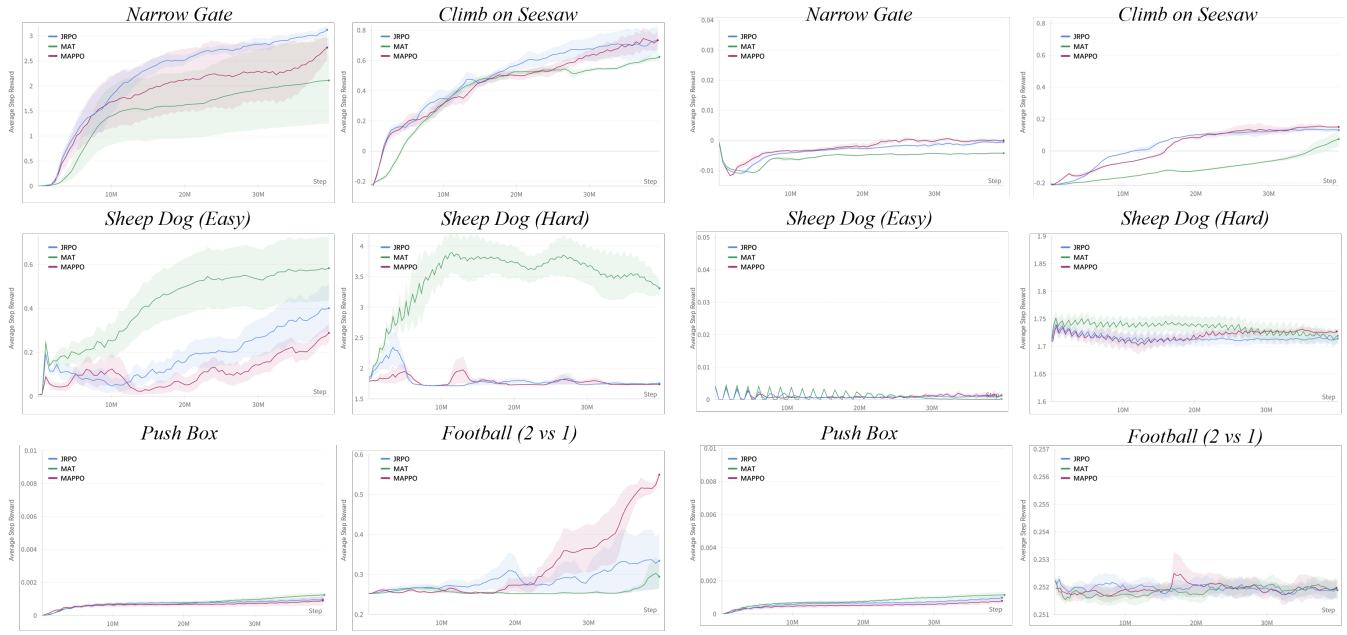
b) Revolving Door: Centered on a wall, a revolving gate permits rotation and passage from both sides, albeit one at a time. Two robots placed on the same side compete to be the first to pass through the door by manipulating the door's rotation and the wall close to the door. This task demands that the robots understand the revolving door's mechanics and strategically counter their opponent's moves.

c) Sumo Wrestling: Two robots are placed in a sumo wrestling match on a featured platform. The objective for each robot is twofold: to destabilize and cause their opponent to fall or to push them out of the designated circle. This task challenges the robots' ability to maintain balance when subjected to external forces, as well as their capability to tactically disrupt the opponent's movement without compromising their own stability.

d) Traverse Bridge: With the intention of crossing a narrow bridge to the other side, two robots are placed at opposite ends. The bridge's limited width permits only one robot to pass at a time. Thus, the primary objective for each robot is to push its opponent off the bridge to secure passage, asking the robots to make use of every inch of space to maintain equilibrium while attempting to displace the opposing robot.

e) Football (1 vs 1): On the football field introduced in *Football (2 vs 1)*, we ask two robots to compete for control of the football and try to goal, requiring the robots with capabilities to dribble, tackle and shoot. Goals only happen when there is a huge strength gap between two robots.

f) Football (2 vs 2): By increasing the number of robots on both sides, *Football (1 vs 1)* becomes a simple football match. The target of each side is simply winning the match by a goal, utilizing the ability of dribbling, passing, tackling, shooting and cohesive team collaboration. There are many complex strategies in a 2 vs 2 football game, making this task a challenge for both locomotion control and strategies.



A. Hierarchical RL

B. From Scratch

Fig. 3. Learning curves of 6 tasks under 2 settings: results using pre-trained locomotion policy is illustrated left and results learning from scratch is illustrated right. Rewards of tasks related to sheep and boxes are jiggling on a small scale due to the reset of corresponding objects.

V. Experiments

With the help of the pre-trained locomotion policy, our environment serves as a fair comparative basis for different MARL algorithms, as we get rid of the possible variance coming from different locomotion policies and focus fully on the planning capability provided by MARL algorithms. In the following sections, we showcase the performance of different multi-agent reinforcement learning algorithms using the 6 collaborative tasks with and without the pre-trained locomotion policy.

A. Simulation Performance

We select 4 tasks with different objects to illustrate the efficient simulation capabilities of our environment, which are *Narrow Gate*, *Push Box*, *Sheep Dog (Easy)*.

Thanks to Isaac Gym's high-performance parallel simulating capabilities, our environment can achieve $10000+$ frames per second with pre-trained locomotion policy running, which contains two MLPs with 4 layers and 3 layers. Means and standard deviations of FPS (frame per second) are shown in Table II, and all the results are obtained on a server with NVIDIA RTX3090, CUDA 11.4, and Isaac Gym Preview 4. The high speed of simulation extremely accelerates the learning process in the wall-clock time, with all data processes accomplished on GPUs.

TABLE II
Simulation Performance of Our Environment in FPS

# Envs	*Narrow Gate** 2 agents	*Climb on Seesaw** 2 agents & 1 object	*Sheep Dog (Easy)** 2 agents & 1 NPC
500	18307 ± 154	14083 ± 63	16164 ± 68
1000	27720 ± 80	22856 ± 80	25129 ± 82

*with the pre-trained locomotion policy running.

B. Environment Setup

Currently, we evaluate the performance of MAPPO, JRPO, and MAT implementation in OpenRL [44] on all collaborative tasks. Reward specifications are listed in Table III, while two settings of actions are 1) commands of horizontal velocity and rotation velocity of yaw or 2) target positions of actuated DoFs, aiming to simulate two reinforcement learning settings: 1) utilizing pre-trained locomotion policy and hierarchical structure to fully focus on the high-level collaboration, and 2) learning from scratch. Subtle rewards designed for learning a walking policy are excluded because such rewards require sampling of target velocity [43] that is impossible to achieve in a task-oriented environment. We train each algorithm in 500 parallel environments for 40 Million environment steps with 5 different random seeds.

C. Quantitative Results

The performance of each algorithm on different tasks is shown in Figure 3. By comparing the results under two settings, it's undeniable that hierarchical structure greatly simplifies tasks that can be solved by utilizing pre-trained locomotion policy through high-level commands, i.e. *Narrow Gate* and *Climb on Seesaw* which only requires different agents to simply walk around, but it's impossible to solve tasks that demand more specialized locomotion policies like *Push Box* and *Football-Defender* that, intuitively, requires

a policy for pushing instead of walking and policies for dribbling, passing and stopping footballs to successfully solve the tasks. The poor performance of RL from scratch demonstrates that end-to-end learning in multi-robot tasks is impossible. Because of the intricate NPC introduced in *Sheep Dog* tasks, MAT achieves notably better performance compared to JRPO and PPO, illustrating that environment awareness takes an important role in realistic multi-agent settings.

According to the results, we can divide the 6 collaborative tasks into 3 categories with rising complexity: 1) tasks that require the agents to perceive other agents' positions and collaborate according to the positions, including *Narrow Gate* and *Climb on Seesaw*; 2) tasks that require the agents to perceive both other agents' positions and have a general knowledge of how the environment will change according to all agents' involvement, i.e. *Sheep Dog-Easy* and *Sheep Dog-Hard*; and 3) tasks that demand specialized locomotion policy as well as collaboration capabilities required by previous 2 categories, including *Push Box* and *Football-Defender*. The different complexity between categories implies that a more sophisticated method of hierarchical RL other than simply outputting state-unrelated commands is necessary.

VI. Conclusion and Future Work

In this paper, we introduced the Multi-agent Quadruped Environment, our new reinforcement learning environment that supports the convenient design of multi-agent tasks for quadruped robots, along with twelve predefined tasks, covering cooperative, adversarial, low-level control, and high-level planning learning. Leveraging the powerful parallel computing capabilities of IsaacGym, our environment is capable of simulating hundreds of environments in parallel, significantly accelerating the process of reinforcement learning. Within our environment, we explored the feasibility of hierarchical RL in the control of quadruped robots and easily completed some cooperative tasks with the help of a low-level locomotion policy. This demonstrates the viability of hierarchical RL in the field of quadruped robots and its relative speed advantage over end-to-end learning. However, the naive hierarchical methods we employed were insufficient for solving complex tasks, prompting us to explore and discuss some viable future work here.

a) Environment awareness plays a critical role in tasks featuring complex dynamics. However, the lack of observations in the real world as well as the difficulty in understanding dynamics, highlights the importance of enhancing environment perception and prediction capabilities in multi-agent tasks, marking it as a promising area for future research.

b) Varied low-level policies will greatly simplify the process of solving any task through a hierarchical structure. We aspire to see an increasing number of low-level policies proven to be effective in the hierarchical structure as well as novel tasks demanding higher agility of quadrupeds.

TABLE III
Reward Specification

Task	Description of Reward	Scale
Narrow Gate	agent walked through the gate	5.
	decreased distance to the gate	1.
	distance between agents (clipped)^2	0.025
	collision penalty	-2
Climb on Seesaw	agent reached the destination	10.
	height of agent	1.
	agent's movement along the seesaw	5
	agent's distance from the seesaw^2	-0.5
	collision penalty	-2.
	falling penalty	-2.
	distance between agents (clipped)^2	0.25
Sheep Dog (Easy)	sheep moved through the gate	1.
	decreased sheep's distance to the gate	2.
Sheep Dog (Hard)	sheep moved through the gate	1.
	exp(-sheep's distance to the gate)	1.
Push Box	box moved through the gate	1.
	decreased box's distance to the gate	10.
Football (2 vs 1)	goal	10.
	exp(distance between ball and the goal)	3.

Appendix

Rewards for collaborative tasks are listed in Table III, where quadratic terms will be labeled by ^{2} and exponential terms will be labeled by exp.

There are two kinds of rewards: 1) rewards calculated based on change of states, i.e. Distance to Target Reward based on the decreased distance between agents and a specified target point behind the gate in *Narrow Gate*, are designed to guide the agent at the beginning stage of learning as they are easy to acquire, and 2) rewards calculated based on states, i.e. Success Reward in *Narrow Gate* provided when the agent has crossed the gate, are always closely related to success states of the tasks and result in much higher returns as they are provided repeatedly. Intuitively, the agent will first follow the change-of-state-based rewards, which may not provide an optimal guide to the target state, to discover the state space including success states, and then the state-based rewards illustrating the success states will dominate the learning. Figure 4 shows the effectiveness of the first kind of reward at the beginning stage and then the domination of the second kind of reward.

Fig. 4. Learning curves of two selected rewards in PPO learning of *Narrow Gate* within first 20M environment steps (average of 5 seeds).

REFERENCES

[1] 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.

[2] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 91–100.

[3] 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.

[4] 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.

[5] S. Gai, S. Lyu, H. Zhang, and D. Wang, "Continual reinforcement learning for quadruped robot locomotion," *Entropy*, vol. 26, no. 1, p. 93, 2024.

[6] 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," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 1479–1486.

[7] Y. Ji, G. B. Margolis, and P. Agrawal, "Dribblebot: Dynamic legged manipulation in the wild," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5155–5162.

[8] Z. Fu, X. Cheng, and D. Pathak, "Learning a unified policy for whole-body control of manipulation and locomotion," in *6th Annual Conference on Robot Learning*, 2022.

[9] T. N. Nguyen, T. B. Nguyen, T. Van Chien, and T. H. Nguyen, "Utilizing deep reinforcement learning to control uav movement for environmental monitoring," *International Journal of Electrical and Electronic Engineering & Telecommunications*, vol. 12, no. 5, pp. 317–325, 2023.

[10] A. P. Pandian, "Artificial intelligence application in smart warehousing environment for automated logistics," *Journal of Artificial Intelligence*, vol. 1, no. 02, pp. 63–72, 2019.

[11] S. Li, W. Guo *et al.*, "Supervised reinforcement learning for ulv path planning in complex warehouse environment," *Wireless Communications and Mobile Computing*, vol. 2022, 2022.

[12] H. Lee and J. Jeong, "Mobile robot path optimization technique based on reinforcement learning algorithm in warehouse environment," *Applied sciences*, vol. 11, no. 3, p. 1209, 2021.

[13] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[14] K. Liu, Y. Zhao, G. Wang, and B. Peng, "Sa-matd3: Self-attention-based multi-agent continuous control method in cooperative environments," *arXiv preprint arXiv:2107.00284*, 2021.

[15] K. Kurach, A. Raichuk, P. Stanczyk, M. Zajkac, O. Bachem, L. Espeholt, C. Riquelme, D. Vincent, M. Michalski, O. Bousquet *et al.*, "Google research football: A novel reinforcement learning environment," *arXiv preprint arXiv:1907.11180*, 2019.

[16] F. Lin, S. Huang, T. Pearce, W. Chen, and W.-W. Tu, "Tizero: Mastering multi-agent football with curriculum learning and self-play," *arXiv preprint arXiv:2302.07515*, 2023.

[17] M. Samvelyan, T. Rashid, C. S. De Witt, G. Farquhar, N. Nardelli, T. G. Rudner, C.-M. Hung, P. H. Torr, J. Foerster, and S. Whiteson, "The starcraft multi-agent challenge," *arXiv preprint arXiv:1902.04043*, 2019.

[18] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged locomotion in challenging terrains using egocentric vision," in *Conference on robot learning*. PMLR, 2023, pp. 403–415.

[19] 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.

[20] Y. Jin, X. Liu, Y. Shao, H. Wang, and W. Yang, "High-speed quadrupedal locomotion by imitation-relaxation reinforcement learning," *Nature Machine Intelligence*, vol. 4, no. 12, pp. 1198–1208, 2022.

[21] 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.

[22] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *The International Journal of Robotics Research*, p. 02783649231224053, 2022.

[23] X. Cheng, K. Shi, A. Agarwal, and D. Pathak, "Extreme parkour with legged robots," *arXiv preprint arXiv:2309.14341*, 2023.

[24] P. Fankhauser, M. Bjelonic, C. D. Bellicoso, T. Miki, and M. Hutter, "Robust rough-terrain locomotion with a quadrupedal robot," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5761–5768.

[25] H. Lee, Y. Shen, C.-H. Yu, G. Singh, and A. Y. Ng, "Quadruped robot obstacle negotiation via reinforcement learning," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE, 2006, pp. 3003–3010.

[26] M. P. Austin, M. Y. Harper, J. M. Brown, E. G. Collins, and J. E. Clark, "Navigation for legged mobility: dynamic climbing," *IEEE Transactions on Robotics*, vol. 36, no. 2, pp. 537–544, 2019.

[27] N. Heess, D. Tb, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami *et al.*, "Emergence of locomotion behaviours in rich environments," *arXiv preprint arXiv:1707.02286*, 2017.

[28] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," *arXiv preprint arXiv:1812.11103*, 2018.

[29] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics: Results of the 11th International Conference*. Springer, 2018, pp. 621–635.

[30] Y. Song, S. Naji, E. Kaufmann, A. Loquercio, and D. Scaramuzza, "Flightmare: A flexible quadrotor simulator," in *Conference on Robot Learning*. PMLR, 2021, pp. 1147–1157.

[31] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement learning for uav attitude control," *ACM Transactions on Cyber-Physical Systems*, vol. 3, no. 2, pp. 1–21, 2019.

[32] Y. Chen, Y. Geng, F. Zhong, J. Ji, J. Jiang, Z. Lu, H. Dong, and Y. Yang, "Bi-dexhands: Towards human-level bimanual dexterous manipulation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[33] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 5026–5033.

[34] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4295–4304.

[35] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls *et al.*, "Value-decomposition networks for cooperative multi-agent learning," *arXiv preprint arXiv:1706.05296*, 2017.

[36] J. Wang, Z. Ren, T. Liu, Y. Yu, and C. Zhang, "Qplex: Duplex dueling multi-agent q-learning," *arXiv preprint arXiv:2008.01062*, 2020.

[37] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative multi-agent games," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 611–24 624, 2022.

[38] M. Wen, J. G. Kuba, R. Lin, W. Zhang, Y. Wen, J. Wang, and Y. Yang, "Multi-agent reinforcement learning is a sequence modeling problem," *arXiv preprint arXiv:2205.14953*, 2022.

[39] L. Wen, Y. Liu, and H. Li, "Cl-mapf: Multi-agent path finding for car-like robots with kinematic and spatiotemporal constraints," *Robotics and Autonomous Systems*, vol. 150, p. 103997, 2022.

[40] X. Yang, S. Huang, Y. Sun, Y. Yang, C. Yu, W.-W. Tu, H. Yang, and Y. Wang, "Learning graph-enhanced commander-executor for multi-agent navigation," *arXiv preprint arXiv:2302.04094*, 2023.

[41] H. Surmann, C. Jestel, R. Marchel, F. Musberg, H. Elhadj, and M. Ardani, "Deep reinforcement learning for real autonomous mobile robot navigation in indoor environments," *arXiv preprint arXiv:2005.13857*, 2020.

[42] B. Xu, F. Gao, C. Yu, R. Zhang, Y. Wu, and Y. Wang, "Omnidrones: An efficient and flexible platform for reinforcement learning in drone control," *IEEE Robotics and Automation Letters*, vol. 9, no. 3, pp. 2838–2844, 2024.

[43] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," in *Conference on Robot Learning*. PMLR, 2023, pp. 22–31.

[44] S. Huang, W. Chen, Y. Sun, F. Bie, and W.-W. Tu, "Openrl: A unified reinforcement learning framework," *arXiv preprint arXiv:2312.16189*, 2023.