Learning Multi-Agent Collaborative Manipulation for Long-Horizon Quadrupedal Pushing

Yuming Feng[1,*], Chuye Hong[1,*], Yaru Niu[1,*], Shiqi Liu[1], Yuxiang Yang[2], and Ding Zhao[1]

***Abstract*— Recently, quadrupedal locomotion has achieved significant success, but their manipulation capabilities, particularly in handling large objects, remain limited, restricting their usefulness in demanding real-world applications such as search and rescue, construction, and industrial automation. This paper tackles the task of obstacle-aware, long-horizon pushing by multiple quadrupedal robots. We propose a hierarchical multi-agent reinforcement learning framework with three levels of control. The high-level controller integrates an RRT planner and a centralized adaptive policy to generate subgoals, while the mid-level controller uses a decentralized goal-conditioned policy to guide the robots toward these sub-goals. A pre-trained low-level locomotion policy executes the movement commands. We evaluate our method against several baselines in simulation, demonstrating significant improvements over baseline approaches, with 48.1% higher success rates and 24.7% reduction in completion time than the best baseline. Our framework successfully enables long-horizon, obstacle-aware manipulation tasks like Push-Cube and Push-T on Go1 robots in the real world. Videos of this work can be found at https://chrisyrniu.github.io/mqpush.**

I. Introduction

Recent advances in quadrupedal robots have significantly improved their ability to traverse challenging terrains [1]–[6]. While many studies have focused on enhancing their mobility and stability of locomotion, the manipulation capabilities of these robots remain relatively limited. Efforts have been made to improve the quadrupedal capabilities in prehensile manipulation through attaching grippers or robotic arms on the robot [7]–[12], and non-prehensile manipulation by using legs [13]–[16] or the head [17], [18] as the end-effectors. Although these advancements enable quadrupeds to handle some routine tasks, their limited ability to manipulate large and heavy objects still restricts their usefulness in demanding fields like search and rescue, construction, and industrial automation, where both dexterity and strength are essential. To address these challenges, researchers have explored adding support structures to the robots [19], [20], coordinating whole-body movements [21], and using multiple robots [22], [23] to strengthen contact forces and expand operational dimensions. However, achieving long-horizon manipulation of large objects in cluttered environments remains a largely unexplored and challenging task.

In this work, we focus on addressing the challenge of obstacle-aware, long-horizon pushing by coordinating the whole-body motions of multiple quadrupedal robots. We build our work upon recent works of quadrupedal pushing that demonstrate impressive results. As shown in Table I, while many approaches utilize multiple robots to enhance manipulation abilities, few focus on long-horizon pushing and obstacle avoidance, both of which are critical for real-world tasks. Additionally, the limited use of whole-body motions (e.g., relying solely on heads to push) [18], [22], [23] restricts the contact patterns between robots and objects, making it difficult for the robots to perform diverse movements and avoid collisions with obstacles.

TABLE I: Comparisons between our proposed method and previous methods of quadrupedal pushing.

Method	Collaborative	Long-Horizon	Whole-Body	Obstacle-Avoidance
Sombolestan et al. [18]	✗	✗	✗	✗
Jeon et al. [21]	✗	✗	✓	✗
Sombolestan et al. [22]	✓	✗	✗	✗
Nachum et al. [23]	✓	✓	✗	✓
An et al. [24]	✓	✗	✓	✗
Xiong et al. [25]	✓	✗	✓	✗
Ours	✓	✓	✓	✓

To achieve collaborative, obstacle-aware, long-horizon quadrupedal pushing through whole-body motions, we propose a hierarchical multi-agent reinforcement learning (MARL) framework with three levels of controllers. The high-level controller integrates an Rapidly-exploring Random Tree (RRT) planner [26] and a centralized adaptive policy, which processes the reference trajectory, environment, and agent information to generate subgoals for the object. The mid-level controller learns a decentralized goal-conditioned policy, enabling multiple robots to coordinate and push the object toward the sequential subgoals proposed by the high-level controller. The low-level controller is a pre-trained locomotion policy that executes commands from the mid-level controller. We validate our approach through a series of experiments in both simulation and real-world tests on Go1 robots, a few of which are visualized in Figure 1. Our results show that the proposed method outperforms the best baseline approaches by 48.1% in success rate and 24.7% in completion time. Furthermore, our method can be deployed on real robots to successfully complete obstacle-aware, long-horizon Push-Cube and Push-T tasks. The main contributions of this paper can be summarized as follows.

- We propose a hierarchical MARL framework with three hierarchies that can handle long-horizon collaborative quadrupedal pushing in an environments with obstacles.
- We benchmark our proposed method against baselines on various long-horizon pushing tasks involving obsta-

[1] Carnegie Mellon University
[2] University of Washington
* Equal contributions.

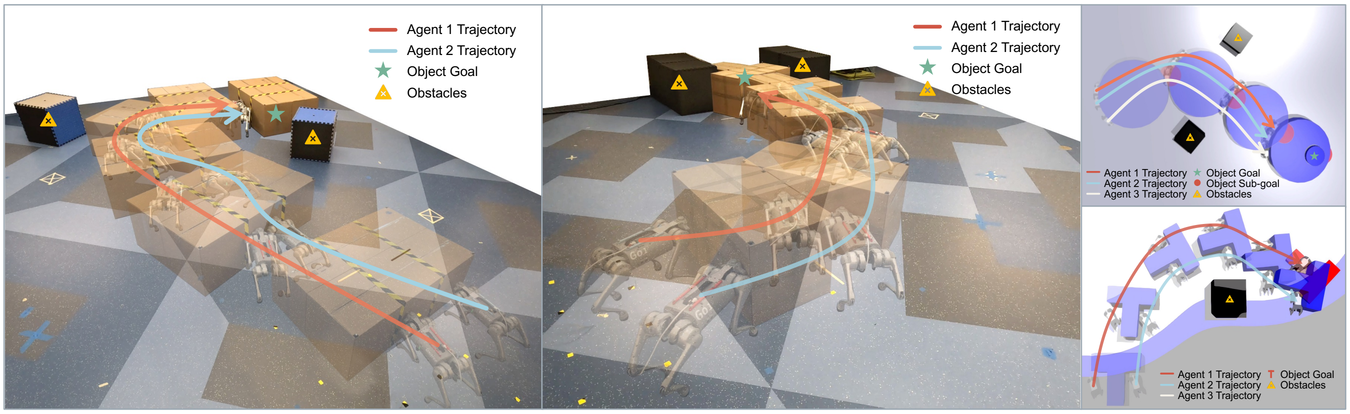


Fig. 1: Our proposed method enables long-horizon collaborative pushing by multiple quadrupedal robots in environments with obstacles. The high-level controller within our hierarchical MARL framework generates adaptive subgoals to guide the lower-level policies during the collaborative manipulation of large objects of varying shapes. The agents' adaptive coordination ensures smooth obstacle avoidance and successful task completion, showcasing the robustness and flexibility of our hierarchical framework.

cles in IsaacGym [27], demonstrating that our method significantly outperforms the baselines.

- We deploy our trained hierarchical policy on real robots, successfully completing the collaborative long-horizon Push-Cube and Push-T tasks with coordinated whole-body motions.

II. Related Work

A. Loco-Manipulation for Legged Robots

Researchers have proposed various optimization-based methods for prehensile loco-manipulation [7]–[9], [28]. These approaches often use hierarchical structure to coordinate locomotion and gripper motions [7], decompose tracking objectives [9], or abstract object information for planning [8]. Optimization-based methods have also been applied to single-robot non-prehensile manipulation tasks [17]–[20], [29], many of which rely on modeling and optimizing contacts with either the object or the ground. Murooka et al. demonstrate how humanoid robots can push large, heavy objects through contact posture planning [29], while Polverini et al. introduce a multi-contact controller for a centaur-type humanoid robot to handle similar tasks [19]. Rigo et al. introduce a hierarchical MPC framework for optimizing contact in quadrupedal loco-manipulation, where the robot is constrained to using its head for pushing [17].

Recently, learning-based methods have demonstrated its effectiveness in loco-manipulation for legged robots. Specifically, reinforcement learning (RL) has been used to train short-horizon quadrupedal pushing skills [15], [30], [31], and other non-prehensile loco-manipulation skills such as dribbling a soccer ball [14], manipulating a yoga ball [13] pressing buttons [16], opening doors [32], and carrying boxes [33]. Jeon et al. propose a hierarchical reinforcement learning framework for quadrupedal whole-body manipulation of large objects, capable of inferring manipulation-relevant privileged information through interaction history [21]. Moreover, learning-based whole-body controllers are trained for prehensile manipulation that requires grasping various objects [11], [34] and consuming visual inputs [10], [12], [35]. Our work focuses on quadrupedal pushing, coordinating whole-body motions using RL-trained policies without explicitly modeling contacts.

B. Multi-Agent Collaborative Manipulation

Optimization-based methods have proven effective in multi-agent collaborative manipulation across various robotic embodiments, such as mobile robots [36]–[39], robotic arms [40], quadrotors [41], and six-legged robots [42]. Some works explore utilizing Model Predictive Control (MPC) to achieve cooperative locomotion for multiple quadrupedal robots holonomically constrained to one another [43]–[45] or collaborative loco-manipulation with objects rigidly attached to each robotic hand [46]. However, these approaches might lack generalizability to more typical scenarios due to their reliance on specific inter-robot connections. The work in [22] introduces a hierarchical adaptive control method enabling multiple quadrupeds to cooperatively push an object with unknown properties along a predetermined path, though the robots are constrained to use their head to push the objects.

Moreover, MARL are employed in cooperative bimanual manipulation for robotic arms [47]–[50] and dexterous hands [51], [52], and collaborative loco-manipulation for quadrupeds [23]–[25], [53], snake robots [54] and bipedal robots [55]. Nachum et al. propose a two-level hierarchical policy in which the high-level policy generates subgoals for each robot to navigate toward [23]. Xiong et al. benchmark MARL with a two-level hierarchical structure in cooperative and competitive tasks, but the methods struggle in a simple box-pushing scenario [25]. An et al. introduce a permutation-invariant network architecture that enables short-horizon multi-object pushing for wheeled-legged quadrupeds [24]. However, these approaches primarily focus on generating effective robot-centric commands for locomotion controllers, making them limited in longer-horizon manipulation tasks.

Our approach addresses these limitations by enabling multiple robots to coordinate whole-body motions for long-horizon pushing tasks in the environments with obstacles.

C. Hierarchical Reinforcement Learning

Hierarchical reinforcement learning (HRL) is often utilized to tackle challenging long-horizon decision making problems. In HRL methods, the high-level policy usually learns to set subgoals for the low-level [56]–[59], or learns to combine and chain behavior primitives [50], [60], [61]. In multi-agent settings, the high-level policy in hierarchical MARL generates goals or commands in a decentralized manner [24], [25], [62], or through a centralized controller [23], [63]–[65]. Meanwhile, many learning-based controllers for legged robots follow a hierarchical structure, where a high-level RL policy provides intermediate commands to the low-level controllers, such as torso velocities [21], [24], foot landing positions [66], [67], target poses [68], [69], gait timing [70], [71] or a combined of several [72], [73]. In our approach, we use a centralized high-level controller to propose a shared object-centric goal for all robots, while decentralized mid-level controllers send torso velocity commands to each robot's low-level policy.

III. Methodology

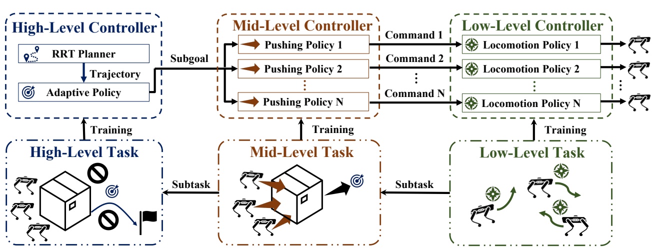


Fig. 2: The framework of our proposed hierarchical MARL method for quadrupedal pushing.

A. Hierarchical Reinforcement Learning for Long-Horizon Multi-Robot Pushing

To enable quadrupedal robots to collaboratively perform long-horizon pushing tasks in environments with obstacles, we propose a hierarchical reinforcement learning framework, as illustrated in Figure 2. This framework consists of three layers of controllers. At the top level, an RRT planner generates a geometrically feasible trajectory without accounting for the robots' pushing capabilities or the dynamics of multiple robots and the object. The high-level adaptive policy then uses this trajectory as a reference to assign a subgoal for the target object, based on the dynamic states of the environment, object, and robots. Using this common subgoal, each robot's mid-level pushing policy provides velocity commands to its corresponding low-level policy. Due to the computational demands of the RRT planner, it is executed only once at the start of each episode. Both the high-level adaptive policy and the mid-level controller operate at a frequency of 50 Hz, with the higher frequency at the high level proving beneficial for more adaptive behavior in our settings. The low-level locomotion policy also runs at 50 Hz, while the PD controller is implemented at 200 Hz in simulation and 2000 Hz on the physical robot. In the following sections, we will introduce each of these three hierarchies in detail.

B. Low-Level Controller

The low-level controller controls each robot individually to track the mid-level velocity commands. More specifically, each low-level controller $\pi_{\varphi}^{l,i} : \mathcal{O}^{l,i} \to \mathcal{A}^{l,i}$ computes motor commands $a^{l,i}$ to track the mid-level velocity command $a^{m,i} = (v_x^i, v_y^i, v_{\text{yaw}}^i)$. Despite recent progress of learning-based low-level controllers [73], we find these controllers to suffer from a large sim-to-real gap, and cannot accurately track the velocity commands, especially when the robot is pushing a heavy object. Instead, we use Unitree's built-in low-level controller, which tracks the velocity commands significantly more robustly in the real world. For efficient policy training in simulation, we train a learned low-level policy to mimic the behavior of Unitree's built-in controller.

We create the simulated low-level controller based on *Walk-These-Ways* (WTW) [73]. As an RL framework for low-level motor control, WTW can learn locomotion behaviors with configurable body pose, gait timing, and reference velocity. We measure these parameters on the built-in controller, and reproduce them in WTW to learn similar behaviors. During upper-level policy training, we invoke the low-level WTW policy in parallel on GPU, which significantly reduced the training time.

C. Mid-Level Controller

The mid-level controller $\pi_{\phi}^{m,i} : \mathcal{O}^{m,i} \to \mathcal{A}^{m,i}$ is a decentralized policy of agent i, where $\mathcal{O}^{m,i}$ represents the mid-level local observation space of robot i, and $A^{m,i}$ is the action space of the mid-level policy of agent i. This decentralized policy takes as input the high-level action a^h, the local observation of robot i, $o^{m,i} \in \mathcal{O}^{m,i}$, which consists of the local observations of the target object state s_{object}^i, obstacle state s_{obstacle}^i, and the state of other robots, $\{s_j^i\}_{j=1,j\neq i}^N$, all computed in the local torso frame of the robot i. For example, s_{object}^i can be expanded as $(x_{\text{object}}^i, y_{\text{object}}^i, \psi_{\text{object}}^i)$, where $(x_{\text{object}}^i, y_{\text{object}}^i)$ is the 2D position of the object, and ψ_{object} is its yaw angle, both in the local frame of robot i. The mid-level policy of agent i will output a mid-level action $a^{m,i} \sim \pi_{\phi}^{m,i}(a^{m,i}|o^{m,i}, a^h) \in \mathcal{A}^{m,i}$ in a decentralized manner to the low-level controller of robot i.

In practice, we train a mid-level policy shared by all robots, noted as π_{ϕ}^m. Following the scheme of centralized training and decentralized execution, it is trained by MAPPO [74] to optimize the objective function $\mathcal{J}^m(\theta) = \mathbb{E}_{\tau\sim\rho_\pi}\left[\sum_{t=0}^T \gamma^t r^m(s_t, a_t^h, \{a_t^{m,i}\}_{i=1}^N)\right]$, where s_t is a joint state at time t, τ is the trajectory sampled from a distribution ρ_π induced by policy π_{ϕ}^m, initial state ρ_0^m and the transition probability p^m that are defined by the mid-level task. Here, $r^m(\cdot)$ represents the reward function for the mid-level controller. During training, we randomly sample the subgoals of the object as a^h and freeze the low-level policy. Meanwhile, we specialize the domain randomization for frictions to reduce the Sim2Real gap of pushing.

D. High-Level Controller

The high-level controller is composed of two elements, a RRT planner $\mathcal{P} : \mathcal{M} \times \mathcal{G} \to \mathcal{T}$ and a centralized adaptive policy $\pi_\theta^h : \mathcal{M} \times \mathcal{T} \times \mathcal{S}_{\text{object}} \times \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \mathcal{S}_N \to \mathcal{A}^h$, where $\mathcal{M}$ represents the map information space, $\mathcal{G}$ represents the goal space of the target object, T represents the trajectory space of the RRT planner, $\mathcal{S}_{\text{object}}$ denotes the object state space, $\mathcal{S}^i$ is the state space of robot i, and A^h is the action space of the high-level adaptive policy.

The RRT planner takes the desired goal position of the object $g_{\text{object}} \in \mathcal{G}$ and the map information $p_{\text{map}} \in \mathcal{M}$ encompassing the obstacle position and the initial position of the object as input and outputs a reference trajectory $\tau_r \in \mathcal{T}$ for the adaptive policy π_θ^h. The adaptive policy will use the desired goal g_{object}, each robot state $s_i \in \mathcal{S}_i$, the map information p_{map} and the dynamic global object pose $s_{\text{object}} \in \mathcal{S}_{\text{object}}$, as the input, and output a high-level action $a^h \sim \pi_\theta^h(a^h | g_{\text{object}}, p_{\text{map}}, s_1, s_2, \cdots, s_N, s_{\text{object}}, \tau_r) \in \mathcal{A}^h$ as the subgoal position of the target object to the mid-level policy π_ϕ^m. The high-level adaptive policy is a centralized policy and trained via PPO [75] to optimize the objective function $\mathcal{J}^h(\theta) = \mathbb{E}_{\tau \sim \rho_\pi}\left[\sum_{t=0}^{T} \gamma^t r^h(g_{\text{object},t}, p_{\text{map},t}, s_{\text{object},t}, \{s_{i,t}\}_{i=1}^N, a_t^h, \tau_r)\right]$, where τ is the trajectory sampled from a distribution ρ_π induced by policy π_θ^h, initial state ρ_0^h and the transition probability p^h that are defined by the high-level task, Here, $r^h(\cdot)$ represents the reward function for the high-level controller. During training, we freeze the mid-level and low-level controller.

E. Reward Design

1) High-Level Reward: The high-level reward function is composed of two terms: $r^h = r^h_{\text{task}} + r^h_{\text{penalty}}$. The high-level task reward r^h_{task} provides a sparse reward for reaching the final target and two dense rewards: one for minimizing the distance between the subgoal and the nearest point on the RRT trajectory, and the other for reducing the distance between the object and the final target. This guides the robots to follow the RRT trajectory while allowing minor deviations for handling push complexities.

The high-level penalty r^h_{penalty} includes penalties for close distances to obstacles and severe punishments for exceptions such as robot falls, collisions, object shaking, and timeouts.

2) Mid-Level Reward: Our mid-level reward function is formulated as $r^m = r^m_{\text{task}} + r^m_{\text{penalty}} + r^m_{\text{heuristic}}$. Similar to high-level reward, the mid-level task reward r^m_{task} rewards actions for approaching and reaching the target point, and the mid-level penalty r^m_{penalty} punishes close distances among agents, as well as exception situations described in Sec. III-E.1.

Additionally, the mid-level heuristic reward $r^m_{\text{heuristic}} = r^m_{\text{approach}} + r^m_{\text{vel}} + r^m_{\text{OCB}}$. The mid-level approaching reward r^m_{approach} encourages agents to approach the object, while the velocity reward r^m_{vel} is awarded when the object's velocity exceeds a set threshold, promoting various pushing actions and preventing oscillation near the object. Also, we design an occlusion-based [38] reward r^m_{OCB} to guide agents toward more favorable contact points. We reward or penalize agents based on if they are occluded by the surface of the object.

In particular, $r^m_{\text{heuristic}}$ plays a crucial role in the push process, given the large action space and the inherent uncertainty and complexity of interactions during push.

IV. Experiments

A. Simulation Setups

1) Environments and Tasks: We build our simulation environments in IsaacGym [27]. We consider a cluttered environment with randomly placed $1.0\,\text{m} \times 1.0\,\text{m}$ obstacles, where multiple quadrupeds need to push a target object to a desired goal. Unitree Go1 robots are utilized in simulation to match the physical robots, each with an approximate payload capacity of 5 kg. The robots are tested with three types of objects varying in shape and mass: a 4 kg cube ($1.2\,\text{m} \times 1.2\,\text{m}$), a 3 kg T-shaped block, and a 10 kg cylinder with a radius of 1.5 m. Different numbers of agents are evaluated across tasks, with two agents for the cube and T-shaped block, and up-to four agents for the cylinder. The initial positions and postures of the agents and target objects are randomly set within a small range on one side of the room, while the target goals for the object are generated on the other side. The task is considered successful if the center of the object is positioned within 1 m of the target. Failure occurs in the situation described in Sec. III-E.1 . The tasks are designed for challenging long-horizon pushing, with initial-to-target distances exceeding 10 m.

2) Baselines: We compare our proposed method with the following baselines.

Single-Agent (SA) retains the three hierarchical levels of the policy and the reward function design, but only a single quadrupedal robot is employed for each task.

High-Level+Low-Level (H+L) utilizes both a high-level and a low-level policy, where the high-level policy proposes subgoals for the robots, and the low-level policy aids the robots in navigating to these subgoals. We maintain r^h_{task} and r^h_{penalty} mentioned in Sec. III-E.1. This baseline follows a similar approach to [23], with a multi-agent implementation using MAPPO [74].

Mid-Level+Low-Level (M+L) retains the mid-level and low-level policies without using a high-level policy to provide subgoals, meaning the robots are guided directly by the distant final target. Similar to the methods proposed in [25] and [24], we maintain r^m_{task} and r^m_{penalty} mentioned in Sec. III-E.2. In addition, extra heuristic rewards, r^m_{approach} and r^m_{vel}, are added to promote the long-horizon pushing performance of the mid-level network.

B. Simulation Results and Analysis

1) Comparisons with Baselines: Training was carried out in 500 environments, accumulating 60 million steps, each 10 million steps taking approximately one hour of simulation time. The performance of each method is summarized in Table II. The success rate (S.R.) is evaluated over 50 trials, averaging results from four random seeds with a frozen low-level policy. The completion time (C.T.) represents the average time taken by the robots to complete the pushing task in 50 trials. If a failure occurs during the task, the completion time is recorded as the timeout duration.

TABLE II: Success rates and completion time ($\pm$ standard deviation) of our method and baselines in different settings in simulation. The completion time is scaled to $[0, 1]$ where 1 means taking up a full episode reaching the timeout.

Task		SA	H+L	M+L	**Ours**
Cube	S.R.↑	4.5±0.3%	0.23±0.02%	18.1±8.0%	**77.5±3.0%**
	C.T.↓	0.98±0.02	1.00±0.00	0.88±0.05	**0.66±0.04**
T-Shape	S.R.↑	21.2±9.3%	9.0±6.0%	18.8±8.6%	**63.5±7.7%**
	C.T.↓	0.96±0.04	0.94±0.04	0.86±0.06	**0.68±0.04**
Cylinder	S.R.↑	0.0±0.0%	3.0±4.0%	31.0±24.5%	**71.2±5.1%**
	C.T.↓	1.00±0.00	0.99±0.02	0.82±0.16	**0.48±0.01**

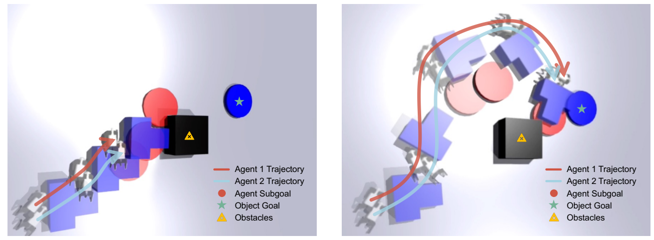
As shown in Table II, the proposed method achieves an average success rate exceeding 60% in all tasks involving three distinct objects. In contrast, the **Single-Agent** method exhibits a success rate below 25%, as a single robot lacks sufficient strength needed for effective pushing, underscoring the importance of multi-robot coordination. The **H+L** method also faces challenges, with a similarly low success rate. In some situations, the object movements towards the final goal are disrupted by the other one or two agents, even if the agents can usually reach their own subgoals. This highlights the difficulty of aligning agent-wise subgoals with effective push control, particularly given the complexity of object interactions, indicating the need for a more fine-grained coordination. The **M+L** method outperforms **H+L** by enabling more elaborate collaboration, but it exhibits greater variability across different seeds. Likewise, it still struggles with the long-horizon nature of the task, as its higher-level policy has difficulty effectively guiding the object towards the distant target.

TABLE III: The results of the OCB-reward ablation study. Two methods are evaluated under two timeout conditions.

	Timeout=20s		Timeout=40s	
	S.R.↑	C.T.↓	S.R.↑	C.T.↓
Ours	**57.0±6.1%**	**14.9±0.8**	**74.0±6.9%**	**22.5±2.4**
Ours w/o OCB	13.0±1.2%	18.3±0.2	19.0±2.6%	34.9±0.3

2) Ablation Study:

a) The OCB Reward: To assess the effectiveness of the OCB reward in training the mid-level controller for short-horzion pushing, we conduct an ablation study in a free space environment, where a 6 kg cube ($1.5\,\mathrm{m} \times 1.5\,\mathrm{m}$) is placed with random orientations. Two agents are randomly initialized, while the target object position (subgoal for the mid-level controller) is generated within a circular area 1.5 to 3.0 m from the cube's initial position randomly. As shown in Table III, our method significantly outperforms the ablation experiment in both success rate (S.R.) and completion time (C.T.). In particular, as the duration of the timeout increases, the success rate of our method improves more rapidly, indicating a better adaptability to adjust the direction of pushing when the object deviates, an issue that often causes failures with shorter timeouts.



(a) Ours w/o adaptive policy. (b) Ours.

Fig. 3: Comparison between our method and the one with only the RRT planner at the high-level controller.

b) The High-Level Adaptive Policy: To demonstrate the need for an adaptive high-level controller, we design a challenging scenario with obstacles placed directly between the start and target positions of the T-block. Although the RRT planning algorithm finds a short path, it often leads to trajectories that come too close to obstacles without considering the object shape, failing to account for the dynamics of the pushing process and lacking real-time adjustments based on the object's state (Fig. 3a) In contrast, our method deviates from obstacles in advance, allowing the robots to bypass them and reach the goal (Fig. 3b), underscoring the importance of the RL-trained high-level adaptive policy.

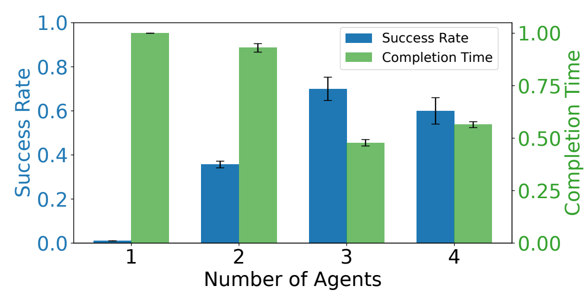


Fig. 4: The success rate and completion time of different numbers of robots in the task of cylinder pushing.

3) Scalability Analysis: To evaluate the scalability of the proposed method, we experiment with different numbers of robots to push the cylinder described in Sec. IV-A.1. As shown in Fig. 4, both the success rate and the completion time improve significantly as the number of robots increases. However, once the number of robots reaches four, the success rate decreases compared to that with three robots. This decline could be due to an increased risk of collisions, which causes robots to maintain greater distances from each other, which leads to less coordinated behaviors.

C. Real-World Setups

1) Environments and Task: We evaluate our policy on real hardware in a 7.5 m $\times$ 7.5 m room, utilizing two Unitree Go1 robots, each with an approximate payload of 5 kg. The entire room is equipped with 24 Primex 22 cameras, using OptiTrack's motion capture system to gather real-time data regarding robots and objects. We deploy both our

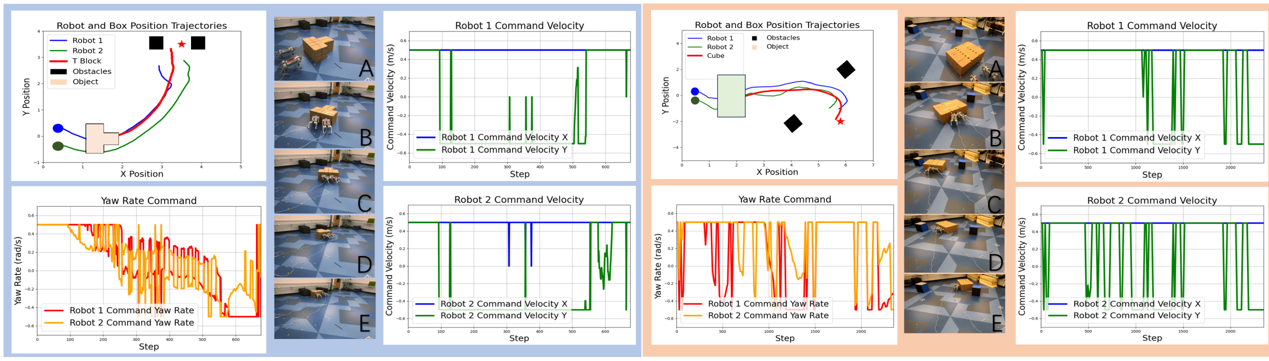


Fig. 5: The blue part demonstrates the effect of pushing the T-shaped block in the real environment, while the red part illustrates the cube-pushing results in the real environment. In each section, the top-left figure shows the movement trajectories of the robots and the object. The bottom-left figure displays the angular velocity commands received by the robots. The top-right figure shows the linear velocity commands received by robot 1, and the bottom-right figure shows the linear velocity commands received by robot 2. The middle part contains screenshots of the video corresponding to the completion of each task. We have extracted key frames that align well with the 2D trajectories, providing a clear visual comparison.

high-level and low-level policies, trained in simulation, on physical robots, utilizing the integrated locomotion policy of the Unitree Go1 as the low-level controller for real-world experiments. With domain randomization implemented for the higher-level policy and the low-level policy trained with Real-to-Sim considerations outlined in Section III-B, we observe that this zero-shot transfer demonstrates robustness.

We conduct two tasks for two robots: pushing a cube and pushing a T-shaped block. The cube utilized in the experiment measures $1.5\,\mathrm{m} \times 1.0\,\mathrm{m}$ and weighs 6.8 kg. The T-block consists of a main body measuring $0.5\,\mathrm{m} \times 1.0\,\mathrm{m}$, with a protruding section measuring $0.5\,\mathrm{m} \times 0.5\,\mathrm{m}$, and has an overall weight of 3.3 kg. Specifically, in the cube task, the target position is randomly set on the opposite side of the room, constrained within an x-range of 5.5 m to 6.5 m and a y-range of -3.5 m to 3.5 m. In the T-pushing task, the target position is set within an x-range of 3.5 m to 4.5 m and a y-range of -4 m to 4 m,. Even with a limited size of the physical environment, this setup ensures that our pushing process remain sufficiently long to meet the requirements of long-horizon tasks. Additionally, in the cube-pushing task, obstacles are randomly initialized within a narrow 2 m band surrounding the line connecting the starting and target positions of the box.

D. Real-World Results and Analysis

1) Pushing T: As shown in the blue part of Figure 5, the robots effectively control the turning of the T-shaped block, and successfully push the T-block to the target position. Due to the smaller size of the T-shaped block, finding the appropriate pushing point is crucial for the task. We observe that the two robots consistently push from the two ends of the block to apply a larger contact surface, which ensures the application of continuous forward force while also maintaining directional control.

Additionally, we find that the x direction velocity commands output by the mid-level policy almost always reach the upper bound of 0.5. It is reasonable because once the robots identify the correct pushing points, the pushing action is primarily forward, with turning achieved by adjusting their own yaw. Since the target is located in the positive y direction, we observe that the yaw command is initially positive and gradually returns to zero or negative after the turn is completed to straighten the robots. This is complemented by minor y direction velocity commands, which help adjust the robots' positions, aligning well with our expectations.

2) Pushing Cube: The push-cube task, a typical multi-robot collaboration challenge, is difficult due to the cube's large size and weight. It involves long-distance pushing with obstacles, testing the planning and coordination of our hierarchical framework. As shown in Figure 5, the robots push the box along an adaptive trajectory, avoiding obstacles and reaching the target. The cube's large surface offers ample contact points, allowing the robots to adopt different pushing postures as needed. At times, both push forward for speed, while at others, they push sideways to turn, consistent with our simulation results. The x-direction velocity commands often hit the 0.5 upper bound, similar to the push-T task, while y-direction and yaw rate commands adjust as expected due to minimal turning.

V. Conclusions

In this paper, we address the challenge of obstacle-aware, long-horizon object pushing by coordinating the whole-body motions of multiple quadrupedal robots. While previous studies make significant progress in improving quadrupedal mobility and some aspects of manipulation, their ability to handle large objects in complex, real-world environments remains limited. To overcome these limitations, we propose a hierarchical MARL framework, consisting of high-level, mid-level, and low-level controllers, to enable effective and coordinated pushing tasks.Through both simulation and real-world experiments, we demonstrate that our approach significantly outperforms the best baseline methods, achieving a 48.1% higher success rate and a 24.7% reduction in completion time. Our method effectively handles obstacle-aware, long-horizon tasks such as Push-Cube and Push-T, highlighting its potential for real-world applications.

References

[1] 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.

[2] 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.

[3] R. Yang, G. Yang, and X. Wang, "Neural volumetric memory for visual locomotion control," in *CVPR 2023*, 2023.

[4] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," *arXiv preprint arXiv:2107.04034*, 2021.

[5] B. Lindqvist, S. Karlsson, A. Koval, I. Tevetzidis, J. Haluška, C. Kanellakis, A.-a. Agha-mohammadi, and G. Nikolakopoulos, "Multimodality robotic systems: Integrated combined legged-aerial mobility for subterranean search-and-rescue," *Robotics and Autonomous Systems*, 2022.

[6] X. Cheng, K. Shi, A. Agarwal, and D. Pathak, "Extreme parkour with legged robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 11 443–11 450.

[7] C. D. Bellicoso, K. Krämer, M. Stäuble, D. Sako, F. Jenelten, M. Bjelonic, and M. Hutter, "Alma-articulated locomotion and manipulation for a torque-controllable robot," in *2019 International conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 8477–8483.

[8] M. Mittal, D. Hoeller, F. Farshidian, M. Hutter, and A. Garg, "Articulated object interaction in unknown scenes with whole-body mobile manipulation," in *2022 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2022, pp. 1647–1654.

[9] C. Lin, X. Liu, Y. Yang, Y. Niu, W. Yu, T. Zhang, J. Tan, B. Boots, and D. Zhao, "Locoman: Advancing versatile quadrupedal dexterity with lightweight loco-manipulators," *arXiv preprint arXiv:2403.18197*, 2024.

[10] H. Ha, Y. Gao, Z. Fu, J. Tan, and S. Song, "Umi on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers," *arXiv preprint arXiv:2407.10353*, 2024.

[11] 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.

[12] M. Liu, Z. Chen, X. Cheng, Y. Ji, R. Yang, and X. Wang, "Visual whole-body control for legged loco-manipulation," *arXiv preprint arXiv:2402.16796*, 2024.

[13] F. Shi, T. Homberger, J. Lee, T. Miki, M. Zhao, F. Farshidian, K. Okada, M. Inaba, and M. Hutter, "Circus anymal: A quadruped learning dexterous manipulation with its limbs," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2316–2323.

[14] 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.

[15] J. Stolle, P. Arm, M. Mittal, and M. Hutter, "Perceptive pedipulation with local obstacle avoidance," *arXiv preprint arXiv:2409.07195*, 2024.

[16] X. Cheng, A. Kumar, and D. Pathak, "Legs as manipulator: Pushing quadrupedal agility beyond locomotion," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5106–5112.

[17] A. Rigo, Y. Chen, S. K. Gupta, and Q. Nguyen, "Contact optimization for non-prehensile loco-manipulation via hierarchical model predictive control," in *2023 ieee international conference on robotics and automation (icra)*. IEEE, 2023, pp. 9945–9951.

[18] M. Sombolestan and Q. Nguyen, "Hierarchical adaptive loco-manipulation control for quadruped robots," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 12 156–12 162.

[19] M. P. Polverini, A. Laurenzi, E. M. Hoffman, F. Ruscelli, and N. G. Tsagarakis, "Multi-contact heavy object pushing with a centaur-type humanoid robot: Planning and control for a real demonstrator," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 859–866, 2020.

[20] W. J. Wolfslag, C. McGreavy, G. Xin, C. Tiseo, S. Vijayakumar, and Z. Li, "Optimisation of body-ground contact for augmenting the whole-body loco-manipulation of quadruped robots," in *2020*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020, pp. 3694–3701.

[21] S. Jeon, M. Jung, S. Choi, B. Kim, and J. Hwangbo, "Learning whole-body manipulation for quadrupedal robot," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 699–706, 2023.

[22] M. Sombolestan and Q. Nguyen, "Hierarchical adaptive control for collaborative manipulation of a rigid object by quadrupedal robots," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 2752–2759.

[23] O. Nachum, M. Ahn, H. Ponte, S. Gu, and V. Kumar, "Multi-agent manipulation via locomotion using hierarchical sim2real," *arXiv preprint arXiv:1908.05224*, 2019.

[24] T. An, J. Lee, M. Bjelonic, F. De Vincenti, and M. Hutter, "Solving multi-entity robotic problems using permutation invariant neural networks," *arXiv preprint arXiv:2402.18345*, 2024.

[25] Z. Xiong, B. Chen, S. Huang, W.-W. Tu, Z. He, and Y. Gao, "Mqe: Unleashing the power of interaction with multi-agent quadruped environment," *arXiv preprint arXiv:2403.16015*, 2024.

[26] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Research Report 9811*, 1998.

[27] 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.

[28] H. Ferrolho, V. Ivan, W. Merkt, I. Havoutis, and S. Vijayakumar, "Roloma: Robust loco-manipulation for quadruped robots with arms," *Autonomous Robots*, vol. 47, no. 8, pp. 1463–1481, 2023.

[29] M. Murooka, S. Nozawa, Y. Kakiuchi, K. Okada, and M. Inaba, "Whole-body pushing manipulation with contact posture planning of large and heavy object for humanoid robot," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 5682–5689.

[30] K. N. Kumar, I. Essa, and S. Ha, "Cascaded compositional residual learning for complex interactive behaviors," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4601–4608, 2023.

[31] T. Huang, N. Sontakke, K. N. Kumar, I. Essa, S. Nikolaidis, D. W. Hong, and S. Ha, "Bayrntune: Adaptive bayesian domain randomization via strategic fine-tuning," *arXiv preprint arXiv:2310.10606*, 2023.

[32] M. Zhang, Y. Ma, T. Miki, and M. Hutter, "Learning to open and traverse doors with a legged manipulator," *arXiv preprint arXiv:2409.04882*, 2024.

[33] C. Zhang, W. Xiao, T. He, and G. Shi, "Wococo: Learning whole-body humanoid control with sequential contacts," *arXiv preprint arXiv:2406.06005*, 2024.

[34] P. Arm, M. Mittal, H. Kolvenbach, and M. Hutter, "Pedipulate: Enabling manipulation skills using a quadruped robot's leg," in *41st IEEE Conference on Robotics and Automation (ICRA 2024)*, 2024.

[35] J. Zhang, N. Gireesh, J. Wang, X. Fang, C. Xu, W. Chen, L. Dai, and H. Wang, "Gamma: Graspability-aware mobile manipulation policy learning based on online grasping pose fusion," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 1399–1405.

[36] P. Culbertson and M. Schwager, "Decentralized adaptive control for collaborative manipulation," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 278–285.

[37] P. Culbertson, J.-J. Slotine, and M. Schwager, "Decentralized adaptive control for collaborative manipulation of rigid bodies," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1906–1920, 2021.

[38] J. Chen, M. Gauci, W. Li, A. Kolling, and R. Groß, "Occlusion-based cooperative transport with a swarm of miniature mobile robots," *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 307–321, 2015.

[39] Z. Tang, Y. Feng, and M. Guo, "Collaborative Planar Pushing of Polytopic Objects with Multiple Robots in Complex Scenes," in *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024.

[40] L. Yan, T. Stouraitis, and S. Vijayakumar, "Decentralized ability-aware adaptive control for multi-robot collaborative manipulation," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2311–2318, 2021.

[41] O. Shorinwa and M. Schwager, "Distributed contact-implicit trajectory optimization for collaborative manipulation," in *2021 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 2021, pp. 56–65.

[42] M. J. Mataric, M. Nilsson, and K. T. Simsarin, "Cooperative multi-

robot box-pushing," in *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, vol. 3. IEEE, 1995, pp. 556–561.

[43] R. T. Fawcett, L. Amanzadeh, J. Kim, A. D. Ames, and K. A. Hamed, "Distributed data-driven predictive control for multi-agent collaborative legged locomotion," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9924–9930.

[44] J. Kim and K. Akbari Hamed, "Cooperative locomotion via supervisory predictive control and distributed nonlinear controllers," *Journal of Dynamic Systems, Measurement, and Control*, vol. 144, no. 3, p. 031005, 2022.

[45] J. Kim, R. T. Fawcett, V. R. Kamidi, A. D. Ames, and K. A. Hamed, "Layered control for cooperative locomotion of two quadrupedal robots: Centralized and distributed approaches," *IEEE Transactions on Robotics*, 2023.

[46] F. De Vincenti and S. Coros, "Centralized model predictive control for collaborative loco-manipulation," *Proceedings of Robotics: Science and System XIX*, p. 050, 2023.

[47] G. Ding, J. J. Koh, K. Merckaert, B. Vanderborght, M. M. Nicotra, C. Heckman, A. Roncone, and L. Chen, "Distributed reinforcement learning for cooperative multi-robot object manipulation," *arXiv preprint arXiv:2003.09540*, 2020.

[48] M. Zhang, P. Jian, Y. Wu, H. Xu, and X. Wang, "Dair: Disentangled attention intrinsic regularization for safe and efficient bimanual manipulation," *arXiv preprint arXiv:2106.05907*, 2021.

[49] Y. Li, C. Pan, H. Xu, X. Wang, and Y. Wu, "Efficient bimanual handover and rearrangement via symmetry-aware actor-critic learning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 3867–3874.

[50] Y. Lee, J. Yang, and J. J. Lim, "Learning to coordinate manipulation skills via skill behavior diversification," in *International conference on learning representations*, 2019.

[51] B. Huang, Y. Chen, T. Wang, Y. Qin, Y. Yang, N. Atanasov, and X. Wang, "Dynamic handover: Throw and catch with bimanual hands," *arXiv preprint arXiv:2309.05655*, 2023.

[52] Y. Chen, T. Wu, S. Wang, X. Feng, J. Jiang, Z. Lu, S. McAleer, H. Dong, S.-C. Zhu, and Y. Yang, "Towards human-level bimanual dexterous manipulation with reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5150–5163, 2022.

[53] Y. Ji, B. Zhang, and K. Sreenath, "Reinforcement learning for collaborative quadrupedal manipulation of a payload over challenging terrain," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 899–904.

[54] Y. Zhang, Y. Niu, X. Liu, and D. Zhao, "Composer: Scalable and robust modular policies for snake robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 10 800–10 806.

[55] B. Pandit, A. Gupta, M. S. Gadde, A. Johnson, A. K. Shrestha, H. Duan, J. Dao, and A. Fern, "Learning decentralized multi-biped control for payload transport," *arXiv preprint arXiv:2406.17279*, 2024.

[56] O. Nachum, S. S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning," *Advances in neural information processing systems*, vol. 31, 2018.

[57] O. Nachum, S. Gu, H. Lee, and S. Levine, "Near-optimal representation learning for hierarchical reinforcement learning," *arXiv preprint arXiv:1810.01257*, 2018.

[58] A. Levy, G. Konidaris, R. Platt, and K. Saenko, "Learning multi-level hierarchies with hindsight," *arXiv preprint arXiv:1712.00948*, 2017.

[59] A. S. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, and K. Kavukcuoglu, "Feudal networks for hierarchical reinforcement learning," in *International conference on machine learning*. PMLR, 2017, pp. 3540–3549.

[60] S. Nasiriany, H. Liu, and Y. Zhu, "Augmenting reinforcement learning with behavior primitives for diverse manipulation tasks," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 7477–7484.

[61] R. Strudel, A. Pashevich, I. Kalevatykh, I. Laptev, J. Sivic, and C. Schmid, "Learning to combine primitive skills: A step towards versatile robotic manipulation §," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4637–4643.

[62] H. Tang, J. Hao, T. Lv, Y. Chen, Z. Zhang, H. Jia, C. Ren, Y. Zheng, Z. Meng, C. Fan *et al.*, "Hierarchical deep multiagent reinforcement learning with temporal abstraction," *arXiv preprint arXiv:1809.09332*, 2018.

[63] S. Ahilan and P. Dayan, "Feudal multi-agent hierarchies for cooperative reinforcement learning," *arXiv preprint arXiv:1901.08492*, 2019.

[64] J. Ma and F. Wu, "Feudal multi-agent deep reinforcement learning for traffic signal control," in *Proceedings of the 19th international conference on autonomous agents and multiagent systems (AAMAS)*, 2020, pp. 816–824.

[65] B. Liu, Q. Liu, P. Stone, A. Garg, Y. Zhu, and A. Anandkumar, "Coach-player multi-agent reinforcement learning for dynamic team composition," in *International Conference on Machine Learning*. PMLR, 2021, pp. 6860–6870.

[66] W. Yu, D. Jain, A. Escontrela, A. Iscen, P. Xu, E. Coumans, S. Ha, J. Tan, and T. Zhang, "Visual-locomotion: Learning to walk on complex terrains with vision," in *5th Annual Conference on Robot Learning*, 2021.

[67] S. Gangapurwala, M. Geisert, R. Orsolino, M. Fallon, and I. Havoutis, "Rloc: Terrain-aware legged locomotion using reinforcement learning and optimal control," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 2908–2927, 2022.

[68] W. Tan, X. Fang, W. Zhang, R. Song, T. Chen, Y. Zheng, and Y. Li, "A hierarchical framework for quadruped locomotion based on reinforcement learning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 8462–8468.

[69] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn, "Humanplus: Humanoid shadowing and imitation from humans," *arXiv preprint arXiv:2406.10454*, 2024.

[70] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots, "Fast and efficient locomotion via learned gait transitions," in *Conference on robot learning*. PMLR, 2022, pp. 773–783.

[71] X. Da, Z. Xie, D. Hoeller, B. Boots, A. Anandkumar, Y. Zhu, B. Babich, and A. Garg, "Learning a contact-adaptive controller for robust, efficient legged locomotion," in *Conference on robot learning*. PMLR, 2021, pp. 883–894.

[72] Y. Yang, G. Shi, X. Meng, W. Yu, T. Zhang, J. Tan, and B. Boots, "Cajun: Continuous adaptive jumping using a learned centroidal controller," in *Conference on Robot Learning*. PMLR, 2023, pp. 2791–2806.

[73] 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.

[74] 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.

[75] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.