Reinforcement Learning for Collaborative Quadrupedal Manipulation of a Payload over Challenging Terrain

Yandong Ji[1], Bike Zhang[2] and Koushil Sreenath[2]

***Abstract*— Motivated towards performing missions in unstructured environments using a group of robots, this paper presents a reinforcement learning-based strategy for multiple quadrupedal robots executing collaborative manipulation tasks. By taking target position, velocity tracking, and height adjustment into account, we demonstrate that the proposed strategy enables four quadrupedal robots manipulating a payload to walk at desired linear and angular velocities, as well as over challenging terrain. The learned policy is robust to variations of payload mass and can be parameterized by different commanded velocities. (Video[1])**

I. Introduction

Quadrupedal robots have been deployed in many real-world applications, such as extreme environment exploration and industrial inspection [1], [2]. While a single quadrupedal robot can handle many tasks, it is still limited by its own onboard power. Multiple quadrupedal robots can collaborate together so that they are able to deal with complex tasks. Besides, a group of quadrupedal robots are more fault-tolerant and reliable than an individual quadruped.

Collaborative quadrupedal robots will be essential for hazardous and critical tasks such as planetary exploration. Collaborative quadrupedal teams will offer force multiplication and redundancy for efficient task completion. Motivated by this, we consider a typical scenario: four quadrupedal robots collaboratively manipulating a payload while locomoting over challenging terrain, as shown in Fig. 1.

A. Related Work

1) Quadrupedal Manipulation: Legs for quadrupedal robots can not only be used for locomotion purposes but also for manipulation tasks. For instance, a quadruped robot can manipulate a ball by its legs from different configurations [3], [4] or guide a human via a leash [5]. A whole body strategy for quadruped manipulation was studied by separating two legs as arms [6]. Moreover, two additional prongs can be designed for a quadrupedal robot to perform manipulation tasks [7]. Quadrupeds can also be equipped with robotic manipulators for general manipulation tasks [8], [9].

[1] Y. Ji is with the College of Artificial Intelligence, Nankai University, Tianjin, 300350, CHN, jiyandong0204@gmail.com

[2] B. Zhang and K. Sreenath are with the Department of Mechanical Engineering, University of California, Berkeley, CA, 94720, USA, {bikezhang, koushils}@berkeley.edu

This work was supported by National Science Foundation CMMI-1944722.

[1] https://youtu.be/i8kZSYdi9Nk

Fig. 1: Four quadrupeds traverse over challenging terrain at a desired velocity while collaboratively carrying a payload in RaiSim.

2) Collaborative Quadrupeds: Multiple quadrupedal robots have the potential to execute tasks that are beyond the capability of a single robot, e.g., robotic soccer using multiple quadrupedal robots has been popularized since the last decade [10]. A model-based approach was developed for cooperative transportation by two quadrupedal robots [11]. Recently, distributed feedback controllers for cooperative locomotion was developed in [12]. However, this collaborative manipulation strategy for quadrupeds needs two additional arms to be attached.

3) Learning Quadruped Locomotion: Reinforcement learning based locomotion policy has shown promising results of performing robust and dynamic locomotion tasks [13]–[21]. Different kinds of quadrupedal locomotion skills were learned by imitating animals through reinforcement learning [15]. A privileged learning strategy is used to teach a quadrupedal robot to traverse over challenging terrain [16]. A contact-adaptive controller was learned for quadrupedal robots to walk more efficiently and robustly on various terrain, including on slippery ground [22]. While most of the learning quadrupedal locomotion works focus on an individual quadruped robot, we develop a collaborative quadruped policy using multiple quadrupedal robots.

4) Multi-Agent Reinforcement Learning (MARL): MARL methods have been investigated for multi-agent's gaming and collaboration tasks. Independent Q-Learning algorithm (IQL) is a basic value-based approach wherein each agent's network is updated separately [23]. COMIX approximates the coupling relationship between each agent with a mixing network, the output of which is a mixing Q function [24]. However, these works have not considered a multiple high-dimensional quadrupedal collaboration scenario.

B. Contribution

The contributions of our work are as follows.

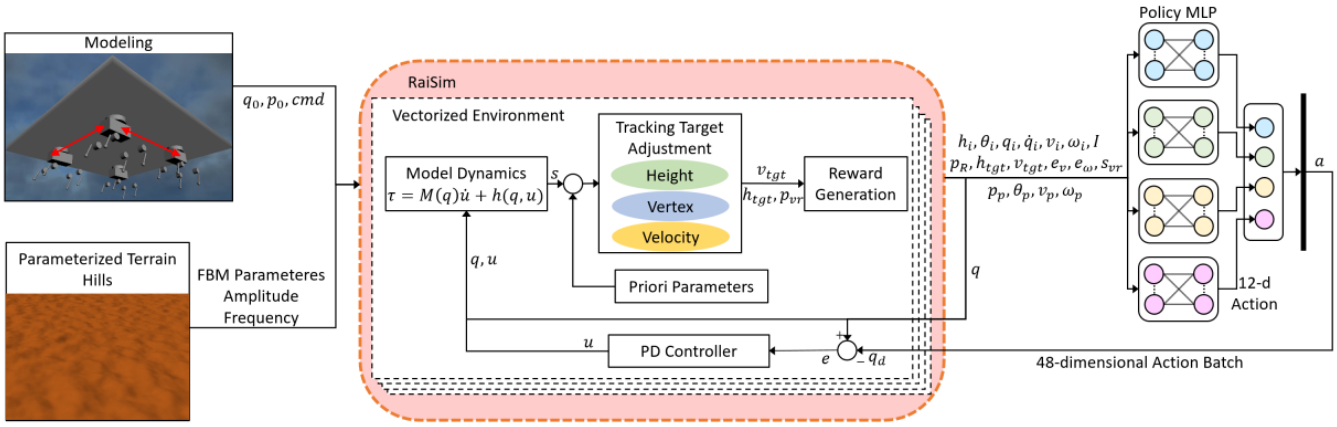

Fig. 2: Control architecture of collaborative quadrupedal manipulation. The initial agent parameter q_0, terrain parameter p_0 and command cmd are the input of this architecture. System input p_0 describes the initial position of agents and payload, q_0 denotes initial joint configuration, and cmd is the linear velocity command for the payload. Priori parameters include p_{vi} in Eq. (2) and h_0, h_{tr} in Eq. (3). Our terrain is generated by Fractal Brownian Motion (FBM) and based on which, we tune the terrain curve's amplitude and frequency. The model dynamics takes the robot's generalized coordinates q and inputs u to produce the simulated state $\mathbf{s}$. This is then used by the tracking target adjustment to generate target height h_{tgt}, vertex position p_{vr} and agent velocity v_{tgt}, which are used for the reward function. The decentralized MLPs obtain observation and rewards from the environment and each generates a 12-dimensional output, the desired joint positions. Adding up the 4 outputs gets a 48 dimensional action batch q_d. Then, a PD controller is utilized to convert desired joint positions to torques u.

- We introduce a reinforcement learning-based strategy for collaborative quadrupeds.
- We demonstrate that our proposed strategy enables four quadrupeds to collaboratively carry a payload while locomoting over rough terrain.
- Our learned policy can be parameterized by different velocities for the payload.
- We also propose a decentralized policy that eliminates jerky behaviors seen with centralized policies.

C. Paper Structure

The rest of the paper is organized as follows. Sec. II introduces the reinforcement learning formulation. The collaborative quadrupedal manipulation policy design is presented in Sec. III. Sec. IV demonstrates the simulation results in different scenarios. Sec. V provides concluding remarks.

II. Reinforcement Learning Formulation

We now introduce our reinforcement learning (RL) formulation for the collaborative quadrupedal manipulation task. Our RL framework is based on the work [14], [16], and the overall framework is illustrated in Fig. 2.

A. Background

We formulate the control process as a Markov Decision Process (MDP) represented by the tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$. $\mathcal{S}$ denotes the state space, $\mathcal{A}$ is the action space, $\mathcal{P}$ is the transition probability, and r is the reward function. In each control cycle, the agent observes a state $\mathbf{s} \in \mathcal{S}$ from the vectorized environment, then it samples an action $\mathbf{a} \in \mathcal{A}$ according to the state and the policy π. The agent applies this action, which results in a reward $r(\mathbf{s}, \mathbf{a}, \mathbf{s}')$. Repeating this process generates a trajectory $\tau = \{(\mathbf{s}_0, \mathbf{a}_0, r_0), (\mathbf{s}_1, \mathbf{a}_1, r_1), ...\}$. Based on the gradient of the loss function $\mathrm{L}(\theta)$, the Adam optimizer updates the parameters of the policy and value networks after each episode to maximize the value of accumulated returns $J(\pi)$,

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right], \tag{1}$$

where T represents time steps in one episode, γ is the discount factor, $p(\tau|\pi)$ is the probability of obtaining the trajectory τ after applying the policy π.

We use proximal policy optimization (PPO) [25] to train the policy. Hyperparameters are shown in Table. I.

TABLE I: PPO Hyperparameters

Parameter	Value
Optimizer	Adam
Number of epochs	4
Discount (γ)	0.996
GAE factor (λ)	0.95
Learning rate	$5{\cdot}10^{-4}$
Clip parameter(ϵ)	0.2
Network layer number	3
Activation Function	Leaky ReLU
No. of nodes in each layer	[128, 256, 256]

B. Observation and Action Space

The observation $\mathbf{o}$ for each agent i includes body height h_i, body orientation $\boldsymbol{\theta}_i$, joint position $\mathbf{q}_i$, joint velocity $\dot{\mathbf{q}}_i$, body linear velocity $\mathbf{v}_i$, body angular velocity $\boldsymbol{\omega}_i$, contact state $\mathbf{I}$, position in payload frame $\mathbf{p}_R$, target height h_{tgt}, target linear velocity $\mathbf{v}_{tgt}$, linear and angular velocity tracking difference $\mathbf{e}_v$ and $\mathbf{e}_\omega$, distance between vertices and agent's CoM s_{vr}. Target velocity $\mathbf{v}_{tgt}$ refers to the target velocity assigned to each agent, specified in Appendix A. The observation $\mathbf{o}$ for the payload consists of position $\mathbf{p}_p$, orientation $\boldsymbol{\theta}_p$, linear velocity $\mathbf{v}_p$ and angular velocity $\boldsymbol{\omega}_p$ of the payload in the world frame. Adding up all 4 agents and

the payload's observations, we generate a 196 dimensional observation vector for policy and value networks.

We utilize a low-level joint position controller with an action $\mathbf{a} \in \mathbb{R}^{48}$ as its input, which is a desired position vector for the 4 agents' total 48 joints. Action space for each policy network is the corresponding agent's 12 joint positions.

III. Method

Having established the RL framework, we next present our collaborative quadrupedal manipulation policy design. We first present agent tracking target adjustment, followed by the decentralized training architecture. Lastly, we apply a terrain curriculum to gradually train a difficult task.

A. Agent Tracking Target Adjustment

We assume that the payload placement over four quadrupeds shown in Fig. 1 is a desired configuration for payload support and control. The agents provide pushing and pivoting force through the contact between the payload and the agents' upper surface. The reward function is an abstraction of the interactive relationship between the system and the environment. Setting a guiding reward function properly helps improve the training efficiency. When the reward function is only set concerning the payload, the optimization efficiency is limited. Thus, we assign rewards to agents and the payload.

Three major parts of the policy design include each agent's vertex tracking, velocity tracking, and height adjustment. Vertex tracking enables a stable relative position between the payload and each agent, which is similar to position control in traverse plane. Velocity tracking provides momentum for pushing and pivoting. While vertex tracking encourages the robot to stay in the target position without moving, velocity tracking drives robots to move at a desired velocity. Thus, these two components are in conflict with each other. We introduce the third component: height adjustment, which alleviates this conflict by adjusting the support force with respect to the payload's lower surface. The tracking targets are updated in every step.

1) Vertex Tracking: This component encourages the agents to track their corresponding vertices on the payload. The specification of the vertex tracking is presented in Appendix B.

$$\hat{\mathbf{P}}_{vertex} = \mathbf{P}_{payload} + \mathbf{R}_z[\mathbf{p}_{v1}, \mathbf{p}_{v2}, \mathbf{p}_{v3}, \mathbf{p}_{v4}], \tag{2}$$

where $\hat{\mathbf{P}}_{vertex}$ is the position matrix of four vertices, and $\mathbf{P}_{payload}$ is the position of payload, $\mathbf{R}_z$ denotes the rotation matrix around z axis, $\mathbf{p}_{vi}$ denotes vertex i's position in the payload frame. The intuition is that multiple agents in a fixed shape provide stable supporting force for the payload.

2) Velocity Tracking: In order to control the payload to rotate at a desired angular velocity and to recover from an undesired yaw angle, we design reward components to encourage each agent to follow its own desired velocity, which enables the payload to be manipulated along a curve. We assign different velocity targets to different agents as their desired curve trajectories differ from each other. Velocity tracking specification is detailed in Appendix B.

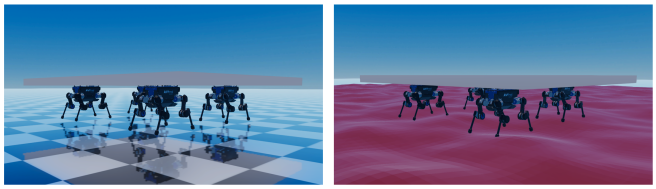

Fig. 3: Snapshots of simulation setup for flat ground (left) and uneven terrain (right).

3) Height Adjustment: When the agents are moving, the payload might sway. The resulting sliding friction between the payload and the agent causes changes to the agents' position. We eliminate this problem by adjusting the height, which indirectly changes the interaction force between the payload and agents. Since vertex and velocity tracking may cause sliding friction, height adjustment helps balance these two components.

The desired height of the agent is adjusted when it is not under a critical line as shown in Appendix B. This line is assumed as a circle with a radius equal to the distance between the center of the payload and a target vertex. Standing under this line, four agents can support the payload robustly.

$$\hat{h} = h_{tr} + \begin{cases} h_0 + k_h(r_c - p_{diff}), & \text{if } \Delta_h \in (h_{min}, h_{max}) \\ h_{max}, & \text{if } \Delta_h > h_{max}, \\ h_{min}, & \text{if } \Delta_h < h_{min}. \end{cases} \tag{3}$$

Here, h_{tr} is the terrain height beneath an agent, h_0 denotes the initial agent height, r_c equals to the critical line's radius, p_{diff} denotes the distance between the payload center and the agent in traverse plane, k_h maps the range of difference to a feasible one, h_{max} and h_{min} are the estimated maximum and minimum stable heights, $\Delta_h = \hat{h} - h_{tr}$.

B. Decentralized Training Architecture

Inspired by the independent Q-learning algorithm (IQL) [23], in which each agent learns an individual action-value network and selects actions greedily, our method provides each agent with its own network while sharing a reward function, see Fig. 2. Agents' trained policies are distinct as they have different relative positions. The observation space for each agent is 196-dimensional. We also implement a centralized training architecture as our baseline, which uses one network for all the agents.

C. Adaptive Terrain Curriculum

Complex tasks generate large negative rewards at early stage of training, which leads to a tendency of standing still for our task. In order to train quadrupeds traversing over challenging terrain with gradual difficulty, we adopt an adaptive terrain curriculum [16]. The policy first learns to walk on a terrain with less roughness, and after every 200 time steps, the old terrain is replaced by a new one with more roughness. In RaiSim [26], we use zScale to indicate the roughness of the terrain. For each update, we increase zScale by 0.05.

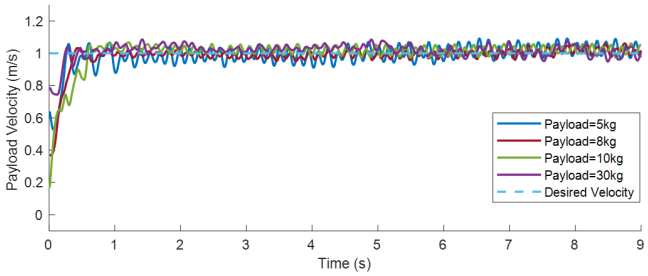

Fig. 4: Payload velocity tracking results of straight motion. The policy is trained with payload mass = 8 kg and tested with payload mass = 5, 8, 10, and 30 kg.

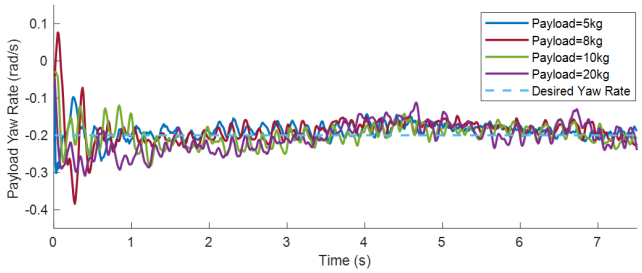

Fig. 5: Payload yaw rate tracking results of curve motion. The policy is trained with payload mass = 8 kg and tested with 5, 8, 10, and 20 kg.

IV. Results

In this section, we first describe our simulation setup where we evaluate the proposed method using four simulated quadrupedal robots with a payload in RaiSim [26]. Then we present results of the learned policy in three scenarios: straight motion, curve motion, and rough terrain as well as robustness tests. We next demonstrate the results of parameterized velocities. Finally, we compare the performance between our proposed decentralized method and a centralized baseline method.

A. Simulation Setup

We choose ANYmal as our simulated testbed. ANYmal is a highly mobile and dynamic quadrupedal robot [27]. The detailed parameters for ANYmal can be found in Table. II. We use four ANYmal robots and a payload which has a dimension of 4m $\times$ 4m $\times$ 0.2m. A gray cylinder, which weighs 1.96 kg, is fixed on top of each ANYmal in order to support the payload. The height of the cylinder is 10 cm and its radius is 20 cm. The payload is moved by the contact force between the cylinder and itself. Friction coefficients between the payload and each agent, terrain and each agent are all 0.7, and corresponding restitution coefficients are 0.5 and 0.2. The environment setup for flat ground and rough terrain are shown in Fig. 3. A 400 steps trajectory is utilized for training in every episode. The control frequency of the policy is 100 Hz, and the simulation time step is 0.0025s. We use RaiSim as our simulation environment [26]. The model is trained on a machine with intel i7-10875H CPU and a Geforce RTX 3060 GPU.

TABLE II: Physical Parameters of ANYmal

Parameter	Symbol	Value	Units
Mass	m	30	kg
Body Length	l_{body}	0.53	m
Body Width	d_{body}	0.27	m
Body Height	h_{body}	0.5	m
Leg Link Lengths	$l_{thigh/shank}$	0.25	m

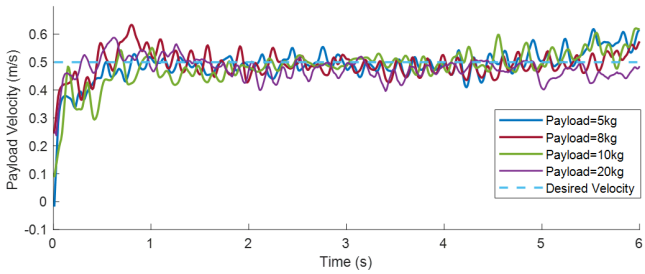

Fig. 6: Payload velocity tracking results on a 0.5 zScale terrain. The policy is trained with payload mass = 8 kg and tested with 5, 8, 10, and 20 kg.

B. Different Traversal Scenarios

1) Straight Motion: The walking forward motion is shown in Fig. 4. The system achieves stable velocity within 2 seconds. The limited velocity oscillation of the payload contributes a stable straight motion.

2) Curve Motion: The result for curve motion is shown in Fig. 5. The desired yaw rate is -0.2 rad/s, and the actual yaw rate converges within 1 second. The supplementary video also shows that the orientation of each agent aligns with the payload's orientation.

3) Challenging Terrain: Fig. 6 demonstrates the payload velocity tracking result for the challenging terrain scenario. The parameterized hill terrain is generated by Fractal Brownian Motion. Our control policy enables collaborative quadrupeds to traverse a rough terrain while tracking a desired velocity.

4) Robustness: We test the robustness of the control policy by varying the payload mass. For the above three scenarios, we use a 8 kg payload during training and test with different payload mass. Collaborative quadrupedal robots are able to carry a 30 kg payload during straight and can afford a 20 kg payload during curve motion and when traversing on rough terrain.

C. Parameterized Velocity

Fig. 7 describes the parameterized linear velocity tests. The learned parameterized policy enables the agents with a payload to walk at a commanded velocity. During training, the commanded linear velocity is sampled from a uniform distribution. We demonstrate that when the commanded velocity is 0.82, 1.0, or 1.28 m/s, the system can track these commands well. The tracking error increases when the commanded velocity reaches to the upper limit.

D. Decentralized Architecture Comparison

Fig. 8 compares the left front knee joint angle results for both centralized and decentralized methods. While both policies can achieve the traversal task over challenging terrain,

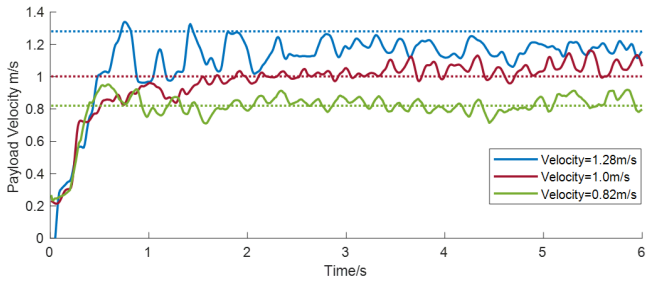

Fig. 7: Parameterized linear velocity tests. The commanded linear velocities used for tests are 0.82, 1.00, and 1.28 m/s (dashed lines). The actual velocity profiles are depicted in solid lines.

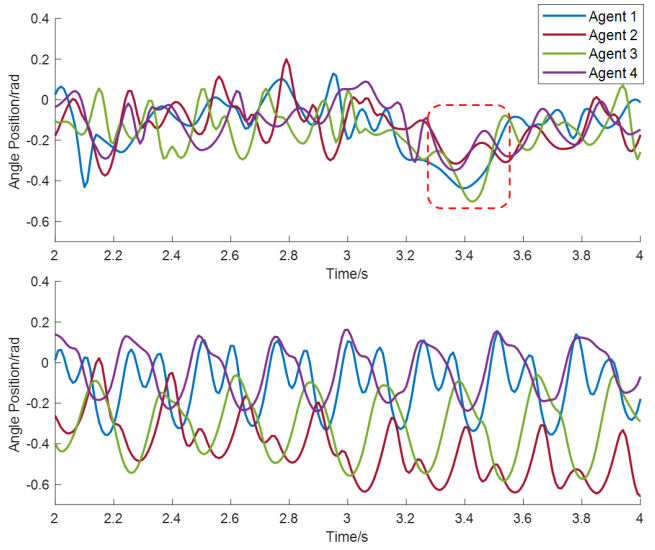

Fig. 8: Left front knee angles for four agents while traversing over challenging terrain. **Top:** Centralized policy results. **Bottom:** Decentralized policy results.

the behavior from the centralized policy is less natural. The dashed red rectangular marks the reactive behavior of the centralized policy when agents are moving uphill. In this case, the joint angles for all agents increase, which shows a centralized training effect.

V. Conclusion and Future Work

In this paper, we presented a reinforcement learning-based strategy for collaborative quadrupedal manipulation. The learned policy enables four quadrupedal robots carrying a payload while executing a locomotion task. We validated this policy design through straight, curve, and rough terrain tests as well as variations of payload mass. There are two major shortcomings. The payload is manipulated through contact. However, the simulated contact force in RaiSim might not be realistic [28]. Another shortcoming is that we require a large amount of observations, and part of which might not be easy to measure for real-world experiments. For the future work, we plan to systematically investigate multi-agent reinforcement learning to develop efficient and robust collaborative quadrupedal behaviors. We also plan to deploy our policy on the real quadrupedal robots.

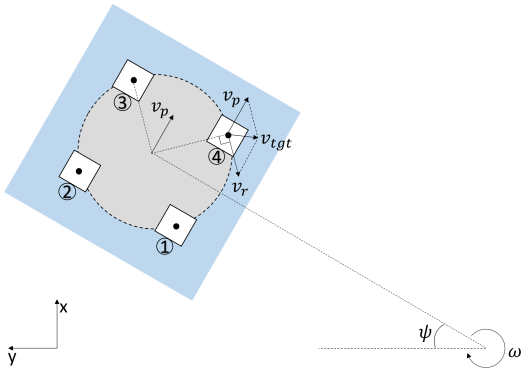

Fig. 9: Agents placement, sequence and velocity tracking targets.

Acknowledgement

We would like to thank Xian Guo for insightful discussion and Hongpeng Wang for the computing support.

Appendix

A. Agents and Payload Placement

The vertex tracking component in the reward function encourages each agent to track its corresponding virtual vertex, which are black points as shown in Fig. 9. The dashed circle represents the critical line in the height adjustment section. When an agent stays within the grey area, a lower height is set while blue area represents a higher agent height. v_{tgt} denotes the desired velocity which includes the target velocity in tracking target adjustment section, v_p represents the payload's desired velocity, and v_r denotes an agent's rotation speed around the center of the payload.

B. Reward Function

Reward functions encourage position and velocity tracking and penalize unnatural behaviours. In our setting, the total number of joints k is 48, the number of feet j for each agent is 4, and the number of agents i is 4. $\hat{\mathbf{P}}_{vertex}$ and $\mathbf{P}$ denote the desired position of tracking points with respect to the payload and each agent's center of mass (CoM) in the traverse plane, $\hat{\mathbf{v}}$ and $\mathbf{v}$ describe each agent's desired and actual linear velocity, ψ_p and ψ denote the payload and agent's yaw angle, $\hat{\dot{\psi}}$ and $\dot{\psi}_H$ represent payload's desired yaw rate and yaw rate history with a length of 12, $\hat{\mathbf{h}}$ and $\mathbf{h}$ describe agent's target height and actual height, $\boldsymbol{\tau}_k$ denotes generalized torque of joint k, $h_{i,tr}$ represents height of terrain beneath agent i, $h_{i,jfoot}$ and $\mathbf{v}_{i,jfoot}$ denote height and velocity of agent i's j^{th} foot, ϕ_i denotes agent i's pitch angle, $\|\cdot\|$ denotes an L_2 norm.

1) Straight Motion:

$$r_s = 0.05r_{vr} + 0.15r_{lv} + 0.02r_{ya} + 0.01r_{aa} + 0.05r_h + 5 \cdot 10^{-7}c_t,$$

where r_{vr} encourages each agent to track its corresponding vertex, r_{lv} and r_{yr} respectively contributes to linear and angular velocity tracking, r_{ya} encourages the yaw angle of each agent close to the payload's yaw angle, r_h rewards for consistence of the agents' height and the target height,

TABLE III: Reward Function Details

Item	Symbol	Value
Vertex Track	r_{vr}	$e^{-\delta_{vr}\left\|\hat{\mathbf{P}}_{vertex}-\mathbf{P}\right\|}$
Linear Velocity	r_{lv}	$e^{-\delta_{lv}\left\|\hat{\mathbf{v}}-\mathbf{v}\right\|}$
Yaw Align	r_{ya}	$e^{-\delta_{ya}\left\|\boldsymbol{\psi}_p-\boldsymbol{\psi}\right\|}$
Yaw Rate	r_{yr}	$e^{-\delta_{yr}\left\|\hat{\dot{\boldsymbol{\psi}}}-\dot{\boldsymbol{\psi}}_H\right\|}$
Angle Adjust	r_{aa}	$e^{-\delta_{aa}\left\|\boldsymbol{\psi}_H\right\|}$
Height	r_h	$e^{-\delta_h\left\|\hat{\mathbf{h}}-\mathbf{h}\right\|}$
Torque	c_t	$-\sum_k \left\|\boldsymbol{\tau}_k\right\|^2$
Foot Clearance	c_{clr}	$-\sum_{i,j}(0.1+h_{i,tr}-h_{i,jfoot})^2\cdot\left\|\mathbf{v}_{i,jft}\right\|$
Slip	c_{slp}	$-\sum_{i,j}\left\|\mathbf{v}_{i,jft}\right\|$
Irregular Pitch	c_{pc}	$-\sum_i \phi_i^2$

r_{aa} encourages a low yaw angle to ensure that the payload is heading forward, and the torque penalty c_t curbs energy consumption.

2) Curve Motion:

$$r_c = 0.04r_{vr} + 0.15r_{lv} + 0.05r_{ya} + 0.045r_{yr} + 0.06r_h + 5\cdot 10^{-7}c_t,$$

where r_{lv} contributes to the tracking of linear velocity v_{tgt}, r_{yr} encourages payload's yaw rate tracking. Agents' linear velocity provide angular momentum for the payload so we set a larger weight parameter for linear velocity tracking reward.

3) Challenging Terrain:

$$r_t = 0.04c_{clr} + 0.005c_{slp} + 0.1c_{pc}.$$

Here, foot clearance cost c_{clr} penalizes the swing motion with a low step height. Slip cost c_{slp} discourages a slipping motion during stance period. Irregular pitch cost c_{pc} contributes to rectifying irregular pitch angle. The reward function is summarized in Table. III.

REFERENCES

[1] A. Bouman, M. F. Ginting, N. Alatur, M. Palieri, D. D. Fan, T. Touma, T. Pailevanian, S. K. Kim, K. Otsu, J. Burdick, and A. a. Agha-Mohammadi, "Autonomous spot: Long-range autonomous exploration of extreme environments with legged locomotion," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2518–2525, 2020.

[2] P. Fankhauser and M. Hutter, "Anymal: a unique quadruped robot conquering harsh environments," *Research Features*, no. 126, pp. 54–57, 2018.

[3] 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," *arXiv preprint arXiv:2011.08811*, 2020.

[4] C. Yang, B. Zhang, J. Zeng, A. Agrawal, and K. Sreenath, "Dynamic legged manipulation of a ball through multi-contact optimization," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7513–7520, 2020.

[5] A. Xiao, W. Tong, L. Yang, J. Zeng, Z. Li, and K. Sreenath, "Robotic guide dog: Leading a human with leash-guided hybrid physical interaction," *arXiv preprint arXiv:2103.14300*, 2021.

[6] T. Omata, K. Tsukagoshi, and O. Mori, "Whole quadruped manipulation," *IEEE International Conference on Robotics and Automation*, vol. 2, pp. 2028–2033, 2002.

[7] W. Wolfslag, C. McGreavy, G. Xin, C. Tiseo, S. Vijayakumar, and Z. Li, "Optimisation of body-ground contact for augmenting whole-body loco-manipulation of quadruped robots," *arXiv preprint arXiv:2002.10552*, 2020.

[8] P. Ewen, J.-P. Sleiman, Y. Chen, W.-C. Lu, M. Hutter, and R. Vasudevan, "Generating continuous motion and force plans in real-time for legged mobile manipulation," *arXiv preprint arXiv:2104.11685*, 2021.

[9] J.-P. Sleiman, F. Farshidian, M. V. Minniti, and M. Hutter, "A unified mpc framework for whole-body dynamic locomotion and manipulation," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4688–4695, 2021.

[10] J. Dalgliesh and M. Lawther, "Playing soccer with quadruped robots," 1999.

[11] M. Hara, M. Fukuda, H. Nishibayashi, Y. Aiyama, J. Ota, and T. Arai, "Motion control of cooperative transportation system by quadruped robots based on vibration model in walking," in *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No. 99CH36289)*, vol. 3. IEEE, 1999, pp. 1651–1656.

[12] K. A. Hamed, V. R. Kamidi, A. Pandala, W.-L. Ma, and A. D. Ames, "Distributed feedback controllers for stable cooperative locomotion of quadrupedal robots: A virtual constraint approach," *2020 American Control Conference (ACC)*, pp. 5314–5321, 2020.

[13] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," *Robotics: Science and Systems*, 2018.

[14] 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, 2019.

[15] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *Robotics: Science and Systems*, 07 2020.

[16] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, 2020.

[17] G. Bellegarda and Q. Nguyen, "Robust high-speed running for quadruped robots via deep reinforcement learning," *arXiv preprint arXiv:2103.06484*, 2021.

[18] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots, "Fast and efficient locomotion via learned gait transitions," *arXiv preprint arXiv:2104.04644*, 2021.

[19] T. Li, R. Calandra, D. Pathak, Y. Tian, F. Meier, and A. Rai, "Planning in learned latent action spaces for generalizable legged locomotion," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2682–2689, 2021.

[20] Z. Xie, X. Da, B. Babich, A. Garg, and M. van de Panne, "Glide: Generalizable quadrupedal locomotion in diverse environments with a centroidal model," *arXiv preprint arXiv:2104.09771*, 2021.

[21] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," *Robotics: Science and Systems*, 2021.

[22] 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," *arXiv preprint arXiv:2009.10019*, 2020.

[23] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proceedings of the tenth international conference on machine learning*, 1993, pp. 330–337.

[24] C. S. de Witt, B. Peng, P.-A. Kamienny, P. Torr, W. Böhmer, and S. Whiteson, "Deep multi-agent reinforcement learning for decentralized continuous cooperative control," *arXiv preprint arXiv:2003.06709*, 2020.

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

[26] J. Hwangbo, J. Lee, and M. Hutter, "Per-contact iteration method for solving contact dynamics," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 895–902, 2018. [Online]. Available: www.raisim.com

[27] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, *et al.*, "Anymal-a highly mobile and dynamic quadrupedal robot," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 38–44, 2016.

[28] Y. Jiang, T. Zhang, D. Ho, Y. Bai, C. K. Liu, S. Levine, and J. Tan, "Simgan: Hybrid simulator identification for domain adaptation via adversarial reinforcement learning," *arXiv preprint arXiv:2101.06005*, 2021.