ManyQuadrupeds: Learning a Single Locomotion Policy for Diverse Quadruped Robots

Milad Shafiee[1], Guillaume Bellegarda[1], and Auke Ijspeert[1]

***Abstract*— Learning a locomotion policy for quadruped robots has traditionally been constrained to a specific robot morphology, mass, and size. The learning process must usually be repeated for every new robot, where hyperparameters and reward function weights must be re-tuned to maximize performance for each new system. Alternatively, attempting to train a single policy to accommodate different robot sizes, while maintaining the same degrees of freedom (DoF) and morphology, requires either complex learning frameworks, or mass, inertia, and dimension randomization, which leads to prolonged training periods. In our study, we show that drawing inspiration from animal motor control allows us to effectively train a single locomotion policy capable of controlling a diverse range of quadruped robots. The robot differences encompass: a variable number of DoFs, (i.e. 12 or 16 joints), three distinct morphologies, a broad mass range spanning from 2 kg to 200 kg, and nominal standing heights ranging from 16 cm to 100 cm. Our policy modulates a representation of the Central Pattern Generator (CPG) in the spinal cord, effectively coordinating both frequencies and amplitudes of the CPG to produce rhythmic output (Rhythm Generation), which is then mapped to a Pattern Formation (PF) layer. Across different robots, the only varying component is the PF layer, which adjusts the scaling parameters for the stride height and length. Subsequently, we evaluate the sim-to-real transfer by testing the single policy on both the Unitree Go1 and A1 robots. Remarkably, we observe robust performance, even when adding a 15 kg load, equivalent to 125% of the A1 robot's nominal mass.**

I. Introduction and Related Work

The oldest group of vertebrates which have appendages (i.e. fins or legs) are the elasmobranchs (sharks and rays). These vertebrates have followed a separate evolutionary path from mammals for over 420 million years. However, neural circuits controlling elasmobranch fins and mammalian limbs have exhibited remarkable similarities at the molecular, cellular, and behavioral levels [1]. This suggests that the neural substrate responsible for limb control was already conserved over 420 million years ago, and that the motor control scheme of tetrapods is preserved across various vertebrate species, each with their own unique size, inertia, and morphology [2]. In contrast, in robotics, it is common practice to design and train a new control policy for each new specific robot. In this paper, we demonstrate how employing a biology-inspired motor-control scheme can streamline the training process, enabling the development of a single control policy applicable to quadruped robots with diverse sizes, inertias, morphologies, and degrees of freedom (DoF).

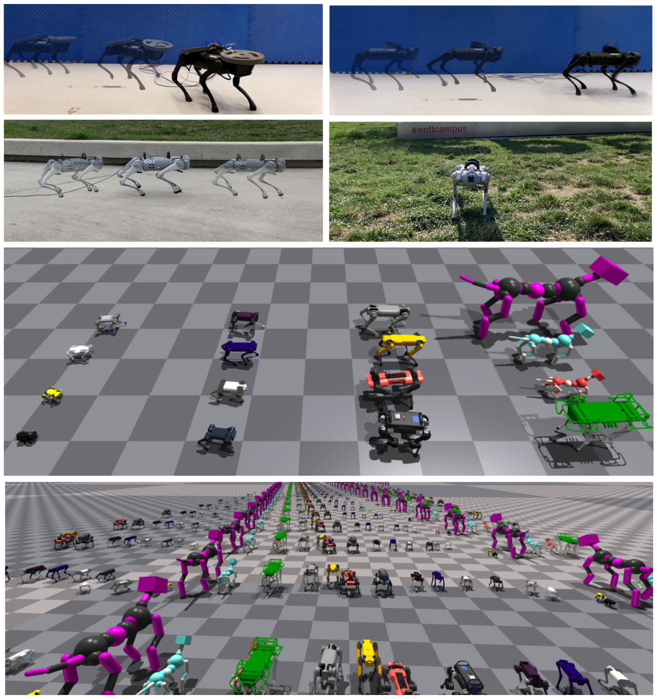


Fig. 1: Simulation and experiment snapshots of training and deploying a single policy to control 16 different robots. Videos: `https://miladshafiee.github.io/ManyQuadrupeds/`

A. Central Pattern Generators

Quadruped animal motor control can be described as an intricate interplay between the Central Pattern Generator (CPG), sensory feedback, and supraspinal drive signals from the brain. In robotics, abstract models of CPGs are commonly used for locomotion pattern generation [3]–[6], as well as to investigate biological hypotheses [7], [8]. Besides the intrinsic oscillatory behavior of CPGs, several other properties such as robustness and implementation simplicity make CPGs desirable for locomotion control [9]. For legged robots, the CPG is usually designed for feedforward rhythm generation, and dynamic balancing is achieved with optimization [3], or hand-tuned feedback [4], [10].

B. Learning Locomotion

Deep Reinforcement Learning (DRL) has emerged as a powerful approach for training robust legged locomotion control policies in simulation, and deploying these sim-to-real on hardware [11]–[24]. Most of these works view the trained artificial neural network (ANN) as a "brain" which has full authority to directly control the joint movements. These methods involve training custom policies from the ground up for each new quadruped's unique morphology. Recent research aims to eliminate the need for policy

[1] This research is supported by the Swiss National Science Foundation (SNSF) as part of project No.197237. The authors are with the BioRobotics Laboratory, Ecole Polytechnique Federale de Lausanne (EPFL). (e-mail: `firstname.lastname@epfl.ch`)

retraining by investigating the potential of graph learning [25]–[28]. In these approaches, the agent's morphology is typically considered as a graph, with the graph's structure mirroring the agent's physical body. Most of these methods utilize agent-agnostic reinforcement learning combined with transfer learning techniques. While these methods based on graph learning exhibit promise in simulating toy examples, they have yet to be validated in real-world robotic systems [25]–[27]. Additionally, the gaits generated in these simulations often lack a natural sense of locomotion, which could make the transition from simulation to the real world challenging. Furthermore, such approaches demand significant computational resources and can require days of training on a standard GPU [27].

Feng et al. [29] introduced a unified policy training approach for learning locomotion through animal motion imitation for a diverse set of quadrupeds, encompassing a range of sizes and masses, yet sharing identical DoFs. This method demonstrated successful simulation-to-reality transfer capabilities; however, it requires a two-week training period on a 16-core CPU, and is limited to robots with the same number of DoFs per leg.

C. Contribution

In this paper, we employ a biology-inspired learning framework to learn a single locomotion policy able to control a diverse range of quadrupeds, encompassing significant variations in size, mass, morphology, and DoF. Our proposed framework seamlessly integrates Central Pattern Generators and Deep Reinforcement Learning [30]–[32], where we use a simple Multi-Layer Perceptron (MLP) to represent higher control centers. This control center orchestrates the precise modulation of the CPG within the spinal cord and adeptly maps the rhythm generation network onto a pattern formation layer. We list below the advantages of the proposed architecture:

Generality and Versatility: The pattern formation layer, which shapes the feet trajectories in task-space, allows us to train a single locomotion policy for different robots with only a few parameter adjustments specific to each robot. Therefore, the action space does not rely on the number of joints or morphology, and we do not need to include joint information in the observation either. This leads to a constant size for both the action and observation spaces for all four-limb robots, which facilitates training robots with varying morphologies and DoFs with a simple MLP architecture. Mapping feet trajectories to joint positions is accomplished through inverse kinematics (IK). Solving the IK problem for legged robots, which fall under the category of serial manipulators, is a well-established research area. Numerous numerical and analytical methods are readily available for efficiently solving the IK problem for various types of serial limbs, each with different DoFs. By entrusting the task of handling morphology to IK, in contrast to the current state-of-the-art in learning quadrupedal locomotion for various robots [29], our method accommodates diverse morphologies and varying DoF. Furthermore, we tested the generalization of the framework by excluding three robots with extreme mass and size properties across three different morphologies from the training process. During these tests, we observed a stable gait for the previously unseen robots, even though the policy had not been specifically trained for them.

Computational Efficiency: Additionally, our approach doesn't necessitate extensive dynamics randomization or motion imitation [29], and it relies on a simple MLP architecture for the Artificial Neural Network. This architectural simplicity, in addition to training robot policies in parallel on a single GPU in Isaac Gym [33], results in high computational efficiency, enabling us to train a single policy for 16 diverse robots in less than two hours, giving orders of magnitude speed-up over previous works.

Robustness: Furthermore, we achieve stable trotting in sim-to-real quadruped experiments, even with an additional load of $15.0\ \mathrm{kg}$, equivalent to 125% of the A1 robot's nominal mass. To the best of our knowledge, this accomplishment represents the pinnacle of robustness against additional loads ever achieved on the A1 robot. In comparison, learning based methods managed a load of up to 12 kg [18], and a model-based approach achieved an 11 kg load capacity [34].

II. Central Pattern Generators

The vertebrate locomotor system is structured such that spinal CPGs are responsible for generating primary rhythmic patterns, while higher-level centers, such as the motor cortex, cerebellum, and basal ganglia, adjust these patterns in response to environmental conditions [1]. Rybak et al. [35] propose that biological CPGs exhibit a dual-level functional organization, with a half-center rhythm generator (RG) determining locomotion frequency, and pattern formation (PF) circuits shaping the precise form of muscle activation signals [35]. This two-tier functional model has also found application in robotics, particularly for quadruped locomotion research [30], [36].

A. Rhythm Generator (RG) Layer

We utilize non-linear phase oscillators to model the RG layer of the CPG circuits in the spinal cord. These oscillators have been effectively applied in the control of quadrupedal locomotion [3], [7], [30] with the following dynamics:

$$\ddot{r}_i = \alpha\left(\frac{\alpha}{4}(\mu_i - r_i) - \dot{r}_i\right) \tag{1}$$

$$\dot{\theta}_i = \omega_i \tag{2}$$

where θ_i is the phase of the oscillator, r_i is the amplitude of the oscillator, ω_i and μ_i are the intrinsic frequency and amplitude, α is a positive convergence factor. Notably, we do not consider explicit phase coupling between different oscillators. Consequently, the phase relationships between the legs must be learned and managed by the control policy by effectively modulating the intrinsic frequency for each limb. The control policy also learns to manipulate stride length by modifying the intrinsic amplitude.

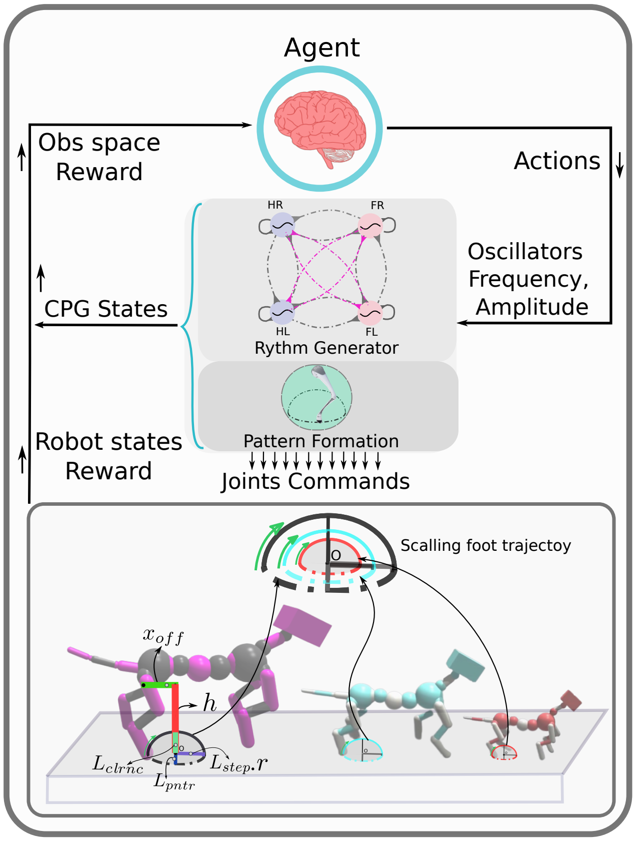


Fig. 2: Quadrupedal locomotion in animals is governed by interactions between the spinal CPG, sensory feedback, and supraspinal brain signals. In our approach, we employ DRL to train a neural network policy that mimics supraspinal drive behavior, enabling it to modulate CPG dynamics. To simulate the CPG within the spinal cord, we utilize nonlinear amplitude-controlled phase oscillators to represent the Rhythm Generator (RG) layer. The RG's outputs are then transformed into foot positions and, using inverse kinematics, converted into motor commands via a Pattern Formation (PF) layer.

B. Pattern Formation (PF) Layer

To establish a mapping from the output of the RG layer to joint commands, we first determine desired foot positions, and then we employ inverse kinematics to compute the corresponding desired joint positions. We formulate the desired foot position coordinates as follows:

$$x_{i,\text{foot}} = \quad x_{off,i} - L_{step}(r_i)\cos(\theta_i) \tag{3}$$

$$z_{i,\text{foot}} = \begin{cases} z_{off,i} - h + L_{clrnc}\sin(\theta_i) & \text{if } \sin(\theta_i) > 0 \\ z_{off,i} - h + L_{pntr}\sin(\theta_i) & \text{otherwise} \end{cases} \tag{4}$$

where L_{step} denotes the nominal step length, h represents the scaling factor for body height, L_{clrnc} indicates the maximum ground clearance during the swing, L_{pntr} signifies the maximum ground penetration during stance phase, and x_{off} serve as set-point altering the equilibrium point of oscillation in the x direction. L_{step}, L_{clrnc}, L_{pntr}, h, and x_{off} are parameters that vary between different robots and are selected based on the relative size of each robot, and these values remain constant throughout the training process.

III. Learning Framework

In this section, we present our hierarchical bio-inspired learning framework for training a single policy to control various quadruped robots. We represent the supraspinal controller as an ANN, which we train with DRL so the agent can learn to modulate the intrinsic frequencies and amplitudes of each limb oscillation to produce stable gaits. We formulate the problem as a Markov Decision Process (MDP), which consists of observations, actions, and rewards. The proposed action-space modulates feet trajectories, and we do not include joint information in the observation space. This leads to a constant size for action and observation space for all four-limb robots, which facilitates training robots with various morphologies and DoF with a simple MLP architecture. We detail the MDP components below.

A. Action Space

We incorporate one RG layer for each limb, as defined by Equations (1) and (2). The RG output is then utilized in a PF layer to generate spatio-temporal foot trajectories in task space, as described in Equations (3) and (4). Notably, we do not include explicit neural coupling, with the intuition that inter-limb coordination will be controlled by the supraspinal drive. As in [30], [32], our action space modulates the intrinsic amplitudes and frequencies of the CPG, by continuously updating μ_i and ω_i for each leg. Therefore, our action space is represented as $\mathbf{a} = [\boldsymbol{\mu}, \boldsymbol{\omega}] \in \mathbb{R}^8$. The agent selects these parameters at a rate of 100 Hz, varying them at each step based on sensory inputs. During training, we impose the following limits on each input: $\mu \in [0.5, 4]$ and $\omega \in [0, 5]$ Hz. We emphasize that our action space modulates the feet trajectories in task-space, and these trajectories are subsequently mapped to joint space through inverse kinematics. Therefore, the proposed learning architecture is independent of the robot's morphology and DoF, as long as we solve the inverse kinematics separately for each robot.

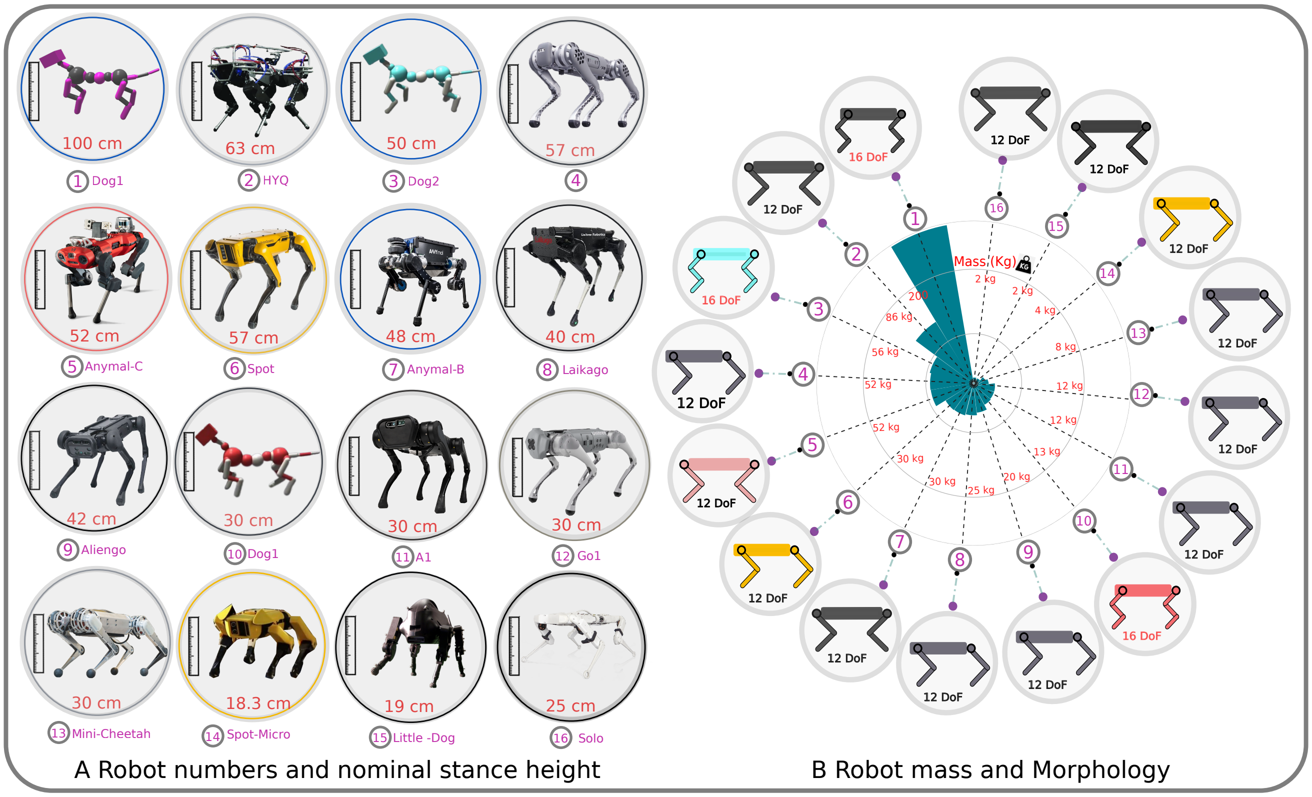
B. Observation Space

Our observation space includes body orientation, body linear and angular velocity, foot contact booleans, relative feet positions with respect to the body frame, the preceding action selected by the policy network, and the CPG states $\{\boldsymbol{r}, \dot{\boldsymbol{r}}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\}$. It is worth noting that, unlike most other RL approaches, we omit proprioceptive information like joint positions and velocities. Instead, we employ a forward kinematics approach based on the current joint positions to determine relative foot positions with respect to the body frame, and then we incorporate these foot positions into the observation space. This design choice simplifies the training process and ensures it remains independent of specific morphologies and DoFs.

C. Reward Function

We use the following reward function to promote viability with forward progress, minimize changes in body orientation, and encourage energy efficiency:

$$\begin{aligned} r_1 = \alpha_1 \cdot \min(f_x, d_{max}) + \alpha_2 \cdot ||\boldsymbol{o}_{base} - \boldsymbol{o}_{zero}|| \\ + \alpha_3 \cdot |\boldsymbol{\tau} \cdot (\dot{\boldsymbol{q}}_t - \dot{\boldsymbol{q}}_{t-1})| \end{aligned}$$



A Robot numbers and nominal stance height

B Robot mass and Morphology

Fig. 3: Characteristics and parameters of 16 diverse quadruped Robots. These robots exhibit variations in mass, ranging from 2 to 200 kg, nominal height from 18 to 100 cm, and come in three different morphologies, with two types of DoF, either 12 or 16.

TABLE I: PPO Hyperparameters and neural network architecture used with Isaac Gym.

Parameter	Value	Parameter	Value
Batch size	98304	GAE discount factor	0.95
Mini-bach size	24576	KL-divergence kl^*	0.01
Number of epochs	5	Learning rate α	adaptive
Clip range	0.2	NN Layers	[512, 256, 128]
Entropy coefficient	0.01	Activation	elu
Discount factor	0.99	Framework	Torch

- *Viability in forward progress*: In the first term, f_x represents the robot's forward progress in the world (along the x-direction). To prevent the exploitation of simulator dynamics and the attainment of unrealistic speeds, we constrain this term. The constraint is set to ensure that the robot is rewarded for moving forward with a certain maximum distance during each control cycle, where d_{max} denotes this maximum distance ($\alpha_1 = 8.0$).
- *Base orientation penalty*: The second term penalizes non-zero body orientation ($\alpha_2 = -0.25$).
- *Power*: The third term penalizes power to encourage energy-efficient gaits, with $\boldsymbol{\tau}$ and $\dot{\boldsymbol{q}}$ representing joint torques and velocities, respectively ($\alpha_3 = -0.00001$).

IV. Results

In this section, we present results from learning a single unified policy to control multiple diverse quadrupeds in both simulation and hardware experiments. Section IV-A details the implementation settings for both simulations and experiments. We then delve into the outcomes of training the policy in Section IV-B. Section IV-C focuses on sim-to-real hardware results. For clear visualizations of the experiments, we encourage the reader to watch the supplementary video.

A. Implementation Setting

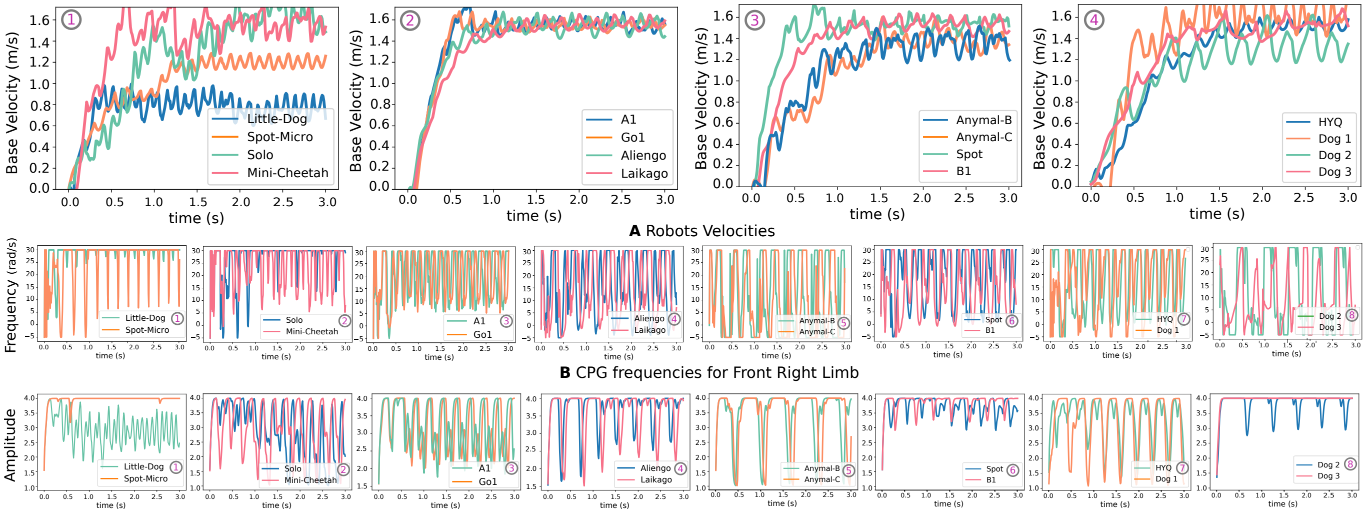
We use Isaac Gym [33], [37] for training and simulating 16 different quadruped robots. These robots exhibit variations in mass, ranging from 2 to 200 kg, nominal height from 18 to 100 cm, and come in three different morphologies, with two types of DoF, either 12 or 16. The robots in our study include the commercial robots Unitree A1, Go1, Aliengo, Laikago, B1, Boston Dynamics Spot, ANYbotics ANYmal-B and ANYmal-C, MIT mini-cheetah, Little-Dog, Spot-Micro, Solo, and HYQ, as well as three customized three-segmented leg quadruped robots. Characteristics and parameters for each robot are detailed in Table II and Figure 3. We consider the following three main morphologies in simulation:

Elbow-Up configuration for all limbs (3-DoF): This configuration, used in robots like Spot, MIT mini-Cheetah, A1, Go1, B1, Laikago, Aliengo, and Spot-micro, consists of a 2-segmented elbow-up for both front and hind legs. Each leg has three Degrees of Freedom (DoF): one for adduction-abduction and two for hip and knee flexion/extension. We solve the inverse kinematics analytically with constraints to an elbow-up configuration.

Elbow-Up for front and Elbow-Down for hind limbs (3-DoF): Similar to the first category, the front limbs maintain an elbow-up configuration, but the hind limbs adopt an elbow-down configuration for the knee. We utilized the same analytical inverse kinematics solution as the first

TABLE II: Characteristics and parameters of diverse quadruped robots, these parameters are fixed during training. The bold values represent the highest and lowest mass and dimensions, as well as a morphology that is unconventional for legged robots, resembling the morphology of animals more closely.

Robot	Height h [cm]	L_{step} [cm]	L_{clrnc} [cm]	L_{pntr} [cm]	x_{offset} [cm]	DoF - Morphology	Mass [kg]	K_p	K_d
Little Dog	19.0	5.0	4.7	0.5	1.1	12 - 2	2.9	20.0	0.3
Spot-Micro	**18.3**	5.0	3.7	0.5	1.0	12 - 1	4.8	20.0	0.3
Solo	25.0	10.0	5.0	0.5	3.7	12 - 2	**2.5**	20.0	0.3
Mini-Cheetah	30.0	13.0	7.0	1.0	0.0	12 - 1	8.4	100.0	2.7
A1	30.0	13.0	7.0	1.0	0.0	12 - 1	12.0	100.0	2.7
Go1	30.0	13.0	7.0	1.0	0.0	12 - 1	12.0	100.0	2.7
Aliengo	42.0	16.0	7.0	1.0	0.0	12 - 1	20.6	100.0	2.7
Laikago	40.0	16.0	7.0	1.0	0.0	12 - 1	25.0	100.0	2.7
Anymal-B	48.0	17.0	7.0	0.0	10.0	12 - 2	30.0	430.0	20.7
Anymal-C	52.0	18.0	7.0	1.0	12.0	12 - 2	52.1	430.0	20.7
Spot	57.0	20.0	9.0	1.0	0.0	12 - 1	30.0	430.0	20.7
B1	57.0	18.0	9.0	1.0	0.0	12 - 1	52.7	430.0	20.7
HYQ	63.0	20.0	9.0	1.0	8.7	12 - 2	86.7	430.0	20.7
Dog1	30.0	13.0	7.0	1.0	0.0	**16 - 3**	13.8	100.0	2.7
Dog2	57.0	18.0	7.0	1.0	0.0	**16 - 3**	56.0	200.0	10.7
Dog3	**100.0**	36.0	9.0	2.0	0.0	**16 - 3**	**200.0**	1400.0	140.7



A Robots Velocities

B CPG frequencies for Front Right Limb

C CPG Amplitudes for Front Right Limb

Fig. 4: Simulation results for training a single policy for 16 different quadrupeds: **Top:** Base locomotion velocity. **Middle:** CPG Frequencies for Front Right limb. **Bottom:** CPG Amplitudes for Front Right limb. We use the FR limb since the robots exhibit a trot gait where the feet closely repeat the same pattern.

category but applied an elbow-down setting for the hind limbs. This configuration is used in robots like ANYmal-B, ANYmal-C, Solo, HYQ, and Little-Dog.

Quadrupedal Animal-like Configuration (4-DoF): In this morphology, inspired by quadruped animals, each leg has four DoF: one for adduction-abduction and three for hip, knee, and foot flexion/extension. We also use an analytical inverse kinematics solution for this configuration, which is applied to three animal-like robots with varying sizes and mass properties. For a visual reference, please see Figure 3.

To train the policies, we use Proximal Policy Optimization (PPO) [38], and Table I lists the PPO hyperparameters and neural network architecture. The control frequency of the policy is 100 Hz, and the torques computed from the desired joint positions are updated at 1 kHz. The equations for each of the oscillators (Equations 1 and 2) are thus also integrated at 1 kHz. All policies are trained for 14.0×10^7 samples on a NVIDIA GeForce RTX 3070 8GB.

B. Simulation Results

In this section, we present simulation results from training the 16 quadruped robots to learn forward locomotion. We consider four parameters for each robot which are fixed during training: nominal standing height, nominal step length, ground clearance of feet during the swing phase, and feet penetration into the ground during stance phase, as shown in Table II. These values are heuristically scaled for each robot based on its height in the zero joints position (i.e. when the legs are fully extended). Figure 4 illustrates the base velocity and CPG frequency and amplitude of the Front Right limb for this first trained policy across all quadruped robots. In this scenario, the reward function encourages forward movement as much as possible while penalizing velocities exceeding 1.5 $\mathrm{m/s}$. Small-sized robots, such as Little-Dog and Spot-Micro, can only reach speeds of 0.8 and 1.2 $\mathrm{m/s}$, respectively (Figure 4-A-1). The frequency trajectory corresponding to small robots (Figure 4-B-1 and 2) shows that, in comparison to the larger robots (Figure 4-B-3, 4, 5, 6), the policy tends to use the highest possible frequency to increase the velocity of small-sized robots. However, increasing velocity for the larger robots does not necessarily require very high frequencies, as they have greater flexibility to increase their base velocity by

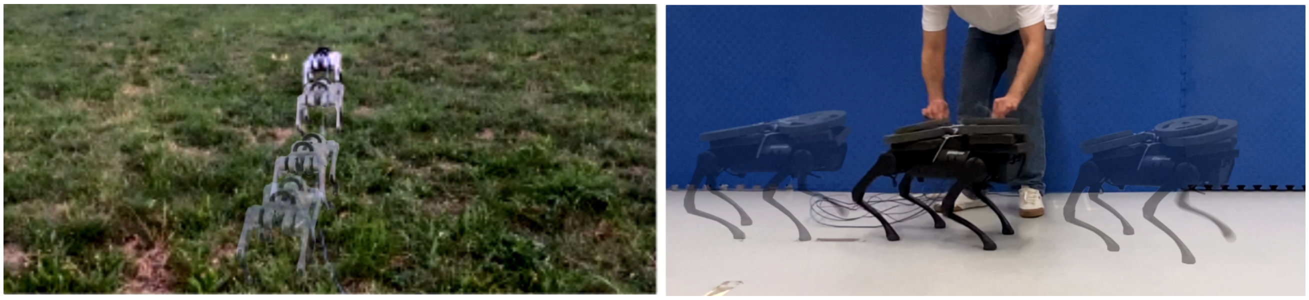


Fig. 5: Left: Snapshots of Go1 trotting on uneven grass. Right: Snapshots of A1 carrying 10 to 15 kg. The robot starts trotting with a 10 kg mass, and then an additional 5 kg is added. The robot successfully walks despite never having encountered any such disturbances during training.

extending their stride length compared to smaller robots.

Our results show that robots with the second type of configuration such as Little-Dog and ANYmal-B and ANYmal-C reach smaller velocities (0.8, 1.4 and 1.4 $\mathrm{m/s}$) respectively) with respect to similar sized robots with the first type morphology such as Spot-Micro, B1 and Spot (1.2, 1.55 and 1.55 $\mathrm{m/s}$ respectively). Furthermore, Figure 4-C-5 and 6 shows that Anymal-B and C locomote with a smaller average amplitude, while Spot and Unitree B1 tend to have the highest amplitude possible, resulting in a higher stride length. Despite the similar mass properties of the compared robots, this observation needs a proper investigation for different nominal parameters to shed some light on the effect of morphology and mass properties in agility and efficiency, for future work.

An interesting observation is that Dog3, the largest simulated robot with a weight of 200 kg and a nominal height of 100 cm, utilizes the highest amplitude. In contrast to other robots, the amplitude reaches its maximum at the starting point of the movement and does not change, allowing for the use of the longest possible stride length, while maintaining a low locomotion frequency. This observation intuitively suggests that robots with larger and heavier limbs tend to adopt a gait characterized by a low frequency and the longest possible stride length, which helps minimize the CoT (penalized in our reward function). At the other extreme, Spot-Micro, which is the smallest robot, exhibits a similar strategy for amplitude. It utilizes very high amplitude with small changes but takes many steps with a high control frequency. This behavior is driven by Spot-Micro's priority to increase speed due to its limitations caused by its small limbs. In contrast, increasing speed for Dog3 is less critical, so it selects a gait that minimizes energy consumption.

Furthermore, we trained a single policy for 13 quadruped robots, excluding HYQ, Dog3, and B1, during training. We deliberately selected these robots to represent the extreme mass and size properties of each morphology to test the generalization capabilities of the framework. We observed reasonable locomotion behavior for these robots, even though the policy had not been specifically trained on them. Please refer to the supplementary video for visual reference.

C. Experimental Results

For the experiments, we trained the single policy to locomote with a maximum velocity of 1 $\mathrm{m/s}$, without incorporating any domain randomization or noise during the training process. We transferred the trained policy sim-to-real to the Go1 and A1 robots, which are the only quadruped robots available in the lab. Figures 1 and 5 show both simulation and experiments snapshots. We tested the control policy on the Go1 robot in two outdoor environments: one on a concrete pavement, and the other on uneven grass. In both scenarios, we observed a very smooth trotting gait. Notably, the grass surface is quite uneven, with holes and bumps that were not encountered during the training process.

We tested the same policy on the A1 robot in two types of experiments: normal walking, and load-carrying scenarios. It's noteworthy that the addition of loads was not encountered during training. We achieved stable trotting in all experiments, even with an additional load of 15.0 kg, equivalent to 125% of the robot's nominal mass. To the best of our knowledge, this accomplishment represents the highest level of robustness against additional loads ever achieved with the A1 robot. In comparison, learning-based methods managed a load of up to 12 kg [18], and a model-based approach achieved an 11 kg load capacity [34].

V. Conclusion

Biological studies have shown that vertebrates utilize very similar motor control architectures, despite differences in morphology, size, mass, and DoFs [2]. Drawing inspiration from this fact, we have presented a biology-inspired learning framework based on Central Pattern Generators and Deep Reinforcement Learning (CPG-RL) with the capability to train a single policy for controlling diverse quadruped robots. These robots vary in the number of DoFs, with variations of 12 and 16, encompass three distinct morphologies, have a wide mass range of 2 Kg to 200 Kg, and nominal standing heights ranging from 16 cm to 100 cm. The proposed framework acts in task-space to modulate and control the feet trajectories, and it does not rely on observing joint information in the observation space, which facilitates training robots with different DoFs and morphologies. Moreover, we are able to train a single policy that works for 16 different robots in less than two hours. In our sim-to-real hardware experiments, we successfully demonstrated stable trotting even with an additional load of 15.0 kg on the Unitree A1 robot, which is equivalent to 125% of the robot's nominal mass. To the best of our knowledge, this accomplishment demonstrates the highest level of robustness against additional loads ever achieved on the A1 robot.

For future work, we plan to expand our framework to include omni-directional motion planning on uneven terrain.

We also aim to further explore how morphology and DoFs influence the agility and efficiency of quadruped locomotion in both animals and robots.

Acknowledgements

We would like to thank Alessandro Crespi for assisting with hardware setup.

References

[1] S. Grillner and A. El Manira, "Current principles of motor control, with special reference to vertebrate locomotion," *Physiological reviews*, vol. 100, no. 1, pp. 271–320, 2020.

[2] S. Grillner, "Evolution: vertebrate limb control over 420 million years," *Current Biology*, vol. 28, no. 4, pp. R162–R164, 2018.

[3] A. Spröwitz, A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri, and A. J. Ijspeert, "Towards dynamic trot gait locomotion: Design, control, and experiments with cheetah-cub, a compliant quadruped robot," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 932–950, 2013.

[4] M. Ajallooeian, S. Pouya, A. Sproewitz, and A. J. Ijspeert, "Central pattern generators augmented with virtual model control for quadruped rough terrain locomotion," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 3321–3328.

[5] S. Aoi, P. Manoonpong, Y. Ambe, F. Matsuno, and F. Wörgötter, "Adaptive control strategies for interlimb coordination in legged robots: a review," *Frontiers in neurorobotics*, vol. 11, p. 39, 2017.

[6] H. Kimura, Y. Fukuoka, and A. H. Cohen, "Adaptive dynamic walking of a quadruped robot on natural ground based on biological concepts," *The International Journal of Robotics Research*, vol. 26, no. 5, pp. 475–490, 2007.

[7] A. J. Ijspeert, A. Crespi, D. Ryczko, and J.-M. Cabelguen, "From swimming to walking with a salamander robot driven by a spinal cord model," *Science*, vol. 315, no. 5817, pp. 1416–1420, 2007.

[8] R. Thandiackal, K. Melo, L. Paez, J. Herault, T. Kano, K. Akiyama, F. Boyer, D. Ryczko, A. Ishiguro, and A. J. Ijspeert, "Emergence of robust self-organized undulatory swimming based on local hydrodynamic force sensing," *Science Robotics*, vol. 6, no. 57, 2021.

[9] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: A review," *Neural Networks*, vol. 21, no. 4, pp. 642–653, 2008, robotics and Neuroscience.

[10] L. Righetti and A. J. Ijspeert, "Pattern generators with sensory feedback for the control of quadruped locomotion," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 819–824.

[11] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.

[12] A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke, "Policies modulating trajectory generators," in *Conference on Robot Learning*. PMLR, 2018, pp. 916–926.

[13] 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.

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

[15] T. Miki, J. Lee, J. Hwanbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, 2022.

[16] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind bipedal stair traversal via sim-to-real reinforcement learning," *arXiv preprint arXiv:2105.08328*, 2021.

[17] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," 2020.

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

[19] G. Ji, J. Mun, H. Kim, and J. Hwangbo, "Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4630–4637, 2022.

[20] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *arXiv preprint arXiv:2205.02824*, 2022.

[21] G. Bellegarda, Y. Chen, Z. Liu, and Q. Nguyen, "Robust high-speed running for quadruped robots via deep reinforcement learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 10 364–10 370.

[22] 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.

[23] Y. Shao, Y. Jin, X. Liu, W. He, H. Wang, and W. Yang, "Learning free gait transition for quadruped robots via phase-guided controller," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1230–1237, 2022.

[24] W. Yu, C. Yang, C. McGreavy, E. Triantafyllidis, G. Bellegarda, M. Shafiee, A. J. Ijspeert, and Z. Li, "Identifying important sensory feedback for learning locomotion skills," *Nature Machine Intelligence*, vol. 5, no. 8, pp. 919–932, 2023.

[25] W. Huang, I. Mordatch, and D. Pathak, "One policy to control them all: Shared modular policies for agent-agnostic control," in *International Conference on Machine Learning*. PMLR, 2020, pp. 4455–4464.

[26] V. Kurin, M. Igl, T. Rocktäschel, W. Boehmer, and S. Whiteson, "My body is a cage: the role of morphology in graph-based incompatible control," *arXiv preprint arXiv:2010.01856*, 2020.

[27] B. Trabucco, M. Phielipp, and G. Berseth, "Anymorph: Learning transferable polices by inferring agent morphology," in *International Conference on Machine Learning*. PMLR, 2022, pp. 21 677–21 691.

[28] J. Whitman, M. Travers, and H. Choset, "Learning modular robot control policies," *IEEE Transactions on Robotics*, 2023.

[29] G. Feng, H. Zhang, Z. Li, X. B. Peng, B. Basireddy, L. Yue, Z. Song, L. Yang, Y. Liu, K. Sreenath, *et al.*, "Genloco: Generalized locomotion controllers for quadrupedal robots," in *Conference on Robot Learning*. PMLR, 2023, pp. 1893–1903.

[30] G. Bellegarda and A. Ijspeert, "CPG-RL: Learning central pattern generators for quadruped locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12 547–12 554, 2022.

[31] M. Shafiee, G. Bellegarda, and A. Ijspeert, "Puppeteer and marionette: Learning anticipatory quadrupedal locomotion based on interactions of a central pattern generator and supraspinal drive," *2023 IEEE International Conference on Robotics and Automation*, 2023.

[32] M. Shafiee, G. Bellegarda, and A. Ijspeert, "Deeptransition: Viability leads to the emergence of gait transitions in learning anticipatory quadrupedal locomotion skills," *arXiv preprint arXiv:2306.07419*, 2023.

[33] 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.

[34] M. Sombolestan, Y. Chen, and Q. Nguyen, "Adaptive force-based control for legged robots," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 7440–7447.

[35] I. A. Rybak, N. A. Shevtsova, M. Lafreniere-Roula, and D. A. McCrea, "Modelling spinal circuitry involved in locomotor pattern generation: insights from deletions during fictive locomotion," *The Journal of physiology*, vol. 577, no. 2, pp. 617–639, 2006.

[36] A. Fukuhara, D. Owaki, T. Kano, R. Kobayashi, and A. Ishiguro, "Spontaneous gait transition to high-speed galloping by reconciliation between body support and propulsion," *Advanced robotics*, vol. 32, no. 15, pp. 794–808, 2018.

[37] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proceedings of the 5th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Faust, D. Hsu, and G. Neumann, Eds., vol. 164. PMLR, 08–11 Nov 2022, pp. 91–100. [Online]. Available: https://proceedings.mlr.press/v164/rudin22a.html

[38] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017.