AllGaits: Learning All Quadruped Gaits and Transitions

Guillaume Bellegarda, Milad Shafiee, Auke Ijspeert

***Abstract*— We present a framework for learning a single policy capable of producing all quadruped gaits and transitions. The framework consists of a policy trained with deep reinforcement learning (DRL) to modulate the parameters of a system of abstract oscillators (i.e. Central Pattern Generator), whose output is mapped to joint commands through a pattern formation layer that sets the gait style, i.e. body height, swing foot ground clearance height, and foot offset. Different gaits are formed by changing the coupling between different oscillators, which can be instantaneously selected at any velocity by a user. With this framework, we systematically investigate which gait should be used at which velocity, and when gait transitions should occur from a Cost of Transport (COT), i.e. energy-efficiency, point of view. Additionally, we note how gait style changes as a function of locomotion speed for each gait to keep the most energy-efficient locomotion. While the currently most popular gait (trot) does not result in the lowest COT, we find that considering different co-dependent metrics such as mean base angular velocity and joint acceleration result in different 'optimal' gaits than those that minimize COT. We deploy our controller in various hardware experiments, focusing on 9 quadruped animal gaits, and demonstrate generalizability to novel and unseen gaits during training, and robustness to leg failures. Video results can be found at `https://youtu.be/OLoWSX_R868`.**

I. Introduction

Quadruped robots are displaying complex motor skills with different gaits to locomote at varying speeds and across challenging terrains, including combinations of discrete capabilities like running and jumping [1]–[9]. While several works study transitions between such gaits (for example), the optimal transition times, speeds, and between which discrete gaits remains an open question. Additionally, for frameworks that do learn, or transition between gaits, the parameters must often be re-tuned for each (MPC) [10], may have heuristics for transitioning [11], or may otherwise be non-optimal, as the cyclic motions may affect the body and joints differently. For example, in contrast with most robots (with some exceptions [12], [13]), animals do not bound with a rigid spine.

While several common quadrupedal gaits have been successfully demonstrated on quadruped hardware, previous work requires either explicit parameter tuning in MPC [10], extensive reward function tuning [11], [14], specific training schemes [15], or expert demonstrations from animals or MPC to imitate [16]. In contrast, we show all quadruped gaits (Fig. 1) and their transitions can be realized without reward function tuning or any expert demonstrations. We center our scientific investigation around three fundamental biological and robotics locomotion questions:

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). {firstname.lastname}@epfl.ch

Fig. 1: **AllGaits**: snapshots from learning all quadruped gaits with Central Pattern Generators and deep reinforcement learning.

1. Which gaits are most efficient at which speeds, and when should gait transitions occur?
2. How should parameters like body height, posture, and swing foot trajectories change for different gaits at different speeds?
3. Can we produce novel gaits not seen during training, and how robust is the policy to leg failures?

II. Method

In order to answer the questions above, we present a hierarchical bio-inspired architecture (Figure 2) consisting of a policy trained with deep reinforcement learning (higher centers), a network of coupled oscillators mapped to task space foot trajectories (rhythm generator and pattern formation layers of the spinal cord), and sensory feedback from onboard sensors and internal CPG states (efference copy of the spinal cord). Details are in Appendices A, B, C. We explicitly enforce a gait through the coupling matrix, and the locomotion style through the pattern formation parameters (i.e. body height, swing foot ground clearance, foot offsets). We leverage this architecture to produce all quadrupedal gaits (Walk, Amble, Trot, Pace, Bound, Pronk, Canter, Transverse Gallop, Rotary Gallop (Figure 4)), determine when the

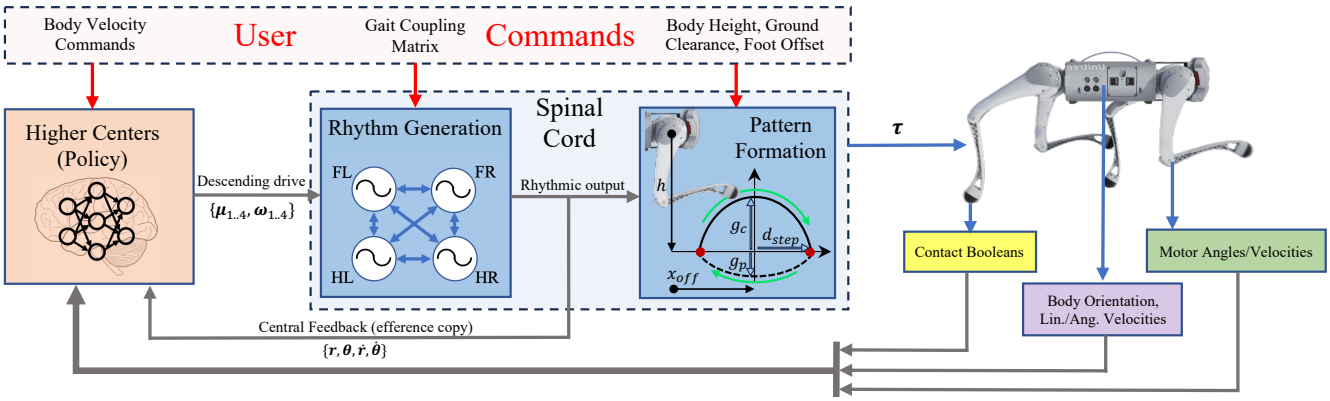

Fig. 2: **AllGaits**: Control architecture for learning central pattern generators to locomote at all gaits for quadruped robots. The observation consists of velocity commands, proprioceptive measurements, and the current CPG states (efference copy of the spinal cord), which the policy network uses to select CPG parameters μ and ω for each leg i (Front Right (FR), Front Left (FL), Hind Right (HR), Hind Left (HL)) to coordinate the Rhythm Generation. A gait coupling matrix is input from the user to set a particular gait. The resulting CPG states are then mapped to desired foot positions in a Pattern Formation layer, which the user can also directly modulate by setting body height h, swing foot ground clearance g_c, and foot offset from the hip x_{off}. This task-space mapping is converted to desired joint angles with inverse kinematics, and tracked with joint PD control to produce torques τ. The control policy selects actions at 100 Hz, and all other blocks operate at 1 kHz.

optimal transitions between gaits should occur, and with which locomotion style. Additionally, we are able to realize novel gaits that were not seen during training, and have not been previously shown, without any modifications directly in hardware experiments. Furthermore, our framework is robust to failures of either one or two disabled legs.

III. Experimental Results and Discussion

Which Gaits and Styles are Most Energy-Efficient? As described in Appendix D, after training we consider 100 gait styles spanning the ranges of possible body heights h, swing foot ground clearances g_c, and foot offsets x_{off} for each of the 9 gaits. Figure 3-A shows the best possible COT for every gait at every velocity at top, and the corresponding gait style parameters at bottom (Fig. 3-B). Between gaits, the walk gait is most energy-efficient from 0.3-0.9 m/s, and then the pace gait results in the lowest COT from 0.9-3.0 m/s, while the amble gait is second most-efficient in this range.

Notably, as also seen in nature, as locomotion speed changes, different gait styles are more optimal from an energy-efficiency perspective. For all gaits, there does not exist a single set of gait parameters that is optimal at all velocities. However, there are certain important trends. The most optimal parameters at all velocities all use the lowest ground clearance, 0.02 m. For most gaits, especially at higher speeds, the highest nominal body height 0.34 m gives the most efficient locomotion. However, we see variability in the hip offset, which changes most variably within each gait as a function of speed to subtly improve the COT.

Is Energy-Efficiency the Most Important Gait Metric? While energy-efficiency is often the most popular explanation for different gait styles at different speeds, why do walk and pace gaits not naturally emerge when training locomotion policies with end-to-end deep reinforcement learning for quadruped robots? Indeed, most recent results showing robot locomotion on flat or rough terrain show a trot gait both at high and low velocities [2], [17]–[19]. However, in addition to velocity tracking, typically the reward function includes many terms related to base stability

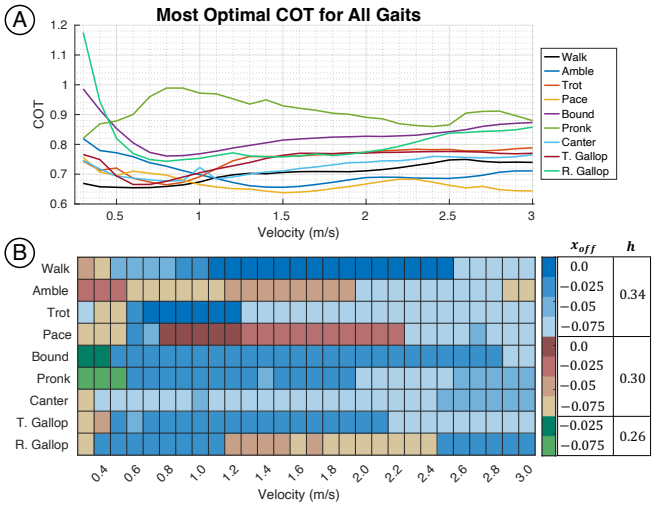

Fig. 3: Most optimal Cost of Transport (COT) for all quadruped gaits, and corresponding gait style parameters for all velocities with our framework. **(A)**: minimum COT possible for each gait. Walk is most optimal for velocities below 0.9 m/s, and pace is most optimal for higher velocities. **(B)**: corresponding gait style parameters for the minimum COTs in **(A)**, with varying body heights h and foot offsets x_{off} for all gaits. The lowest COTs here were all achieved with the lowest swing foot ground clearance $g_c = 0.02$ m.

to keep smooth motions and encourage a "natural" gait.

In Figure 5, we evaluate our policy for each gait with all possible styles with respect to two different metrics: mean angular velocity, and mean joint acceleration. From the plots, we can observe that a gait with the lowest COT at a particular velocity may not be optimal with respect to other metrics. For example, at low velocities, the trot gait has the lowest mean angular velocity, indicating a stable base, while having only a slightly higher COT with respect to the walk gait. While the amble gait has the highest mean angular velocity at high speeds, we see that it has the lowest mean joint acceleration, and has the second best COT (Figure 3). Therefore, depending on the importance a user gives to different metrics, this directly changes the "optimal" gait.

Qualitative Evaluation Appendix F discusses simulation and hardware results from deploying our single policy to accomplish all gaits, including transitions between any gaits at any velocity, as well as novel gaits not seen during training.

REFERENCES

[1] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "Anymal parkour: Learning agile navigation for quadrupedal robots," *Science Robotics*, vol. 9, no. 88, p. eadi7566, 2024.

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

[3] 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.

[4] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao, "Robot parkour learning," in *Conference on Robot Learning (CoRL)*, 2023.

[5] I. M. A. Nahrendra, B. Yu, and H. Myung, "Dreamwaq: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5078–5084.

[6] G. Bellegarda, C. Nguyen, and Q. Nguyen, "Robust quadruped jumping via deep reinforcement learning," *Robotics and Autonomous Systems*, p. 104799, 2024.

[7] G. Bellegarda, M. Shafiee, M. E. Özberk, and A. Ijspeert, "Quadruped-Frog: Rapid online optimization of continuous quadruped jumping," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 1443–1450.

[8] G. B. Margolis, T. Chen, K. Paigwar, X. Fu, D. Kim, S. bae Kim, and P. Agrawal, "Learning to jump from pixels," in *5th Annual Conference on Robot Learning*, 2021.

[9] L. M. Smith, J. C. Kew, T. Li, L. Luu, X. B. Peng, S. Ha, J. Tan, and S. Levine, "Learning and adapting agile locomotion skills by transferring experience," in *Robotics: Science and Systems*, 2023.

[10] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

[11] 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.

[12] S. Seok, A. Wang, M. Y. Chuah, D. Otten, J. Lang, and S. Kim, "Design principles for highly efficient quadrupeds and implementation on the mit cheetah robot," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 3307–3312.

[13] M. Khoramshahi, A. Spröwitz, A. Tuleu, M. N. Ahmadabadi, and A. J. Ijspeert, "Benefits of an active spine supported bounding locomotion with a small compliant quadruped robot," 2013, pp. 3329–3334.

[14] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," *Conference on Robot Learning*, 2022.

[15] Z. Fu, A. Kumar, J. Malik, and D. Pathak, "Minimizing energy consumption leads to the emergence of gaits in legged robots," in *5th Annual Conference on Robot Learning*, 2021.

[16] X. Bin Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *Robotics: Science and Systems (RSS)*, pp. 12–16, 2020.

[17] 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.

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

[19] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *5th Annual Conference on Robot Learning*, 2021.

[20] 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.

[21] 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.

[22] Unitree Robotics. Go1. https://www.unitree.com/products/go1/.

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

APPENDIX

A. Rhythm Generation and Pattern Formation

We use CPG-RL [20] as a basis, where the abstract oscillators which form our Rhythm Generation layer are defined as:

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

$$\dot{\theta}_i = \omega_i + \sum_j r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij}) \tag{2}$$

where r_i is the current amplitude of the oscillator, θ_i is the phase of the oscillator, μ_i and ω_i are the intrinsic amplitude and frequency, a is a positive constant representing the convergence factor. Couplings between oscillators are defined by the weights w_{ij} and phase biases ϕ_{ij}. For a quadruped robot with a single oscillator corresponding to each leg, the coupling matrices $\mathbf{\Phi}$ can be defined by following the timings from Figure 4. These matrices define the offsets between different oscillators to converge to the desired gaits. For example, in the trot gait, 0.5 represents a π offset between the sets of diagonal limbs. With appropriately high (strong) coupling weights, i.e. $w_{ij} = 10$, these coupling matrices enforce the gait.

To map from the oscillator states to joint commands, we first compute corresponding desired foot positions, and then calculate the desired joint positions with inverse kinematics. The desired foot position coordinates are given as follows:

$$x_{i,\text{foot}} = x_{off} - d_{step}(r_i - 1)\cos(\theta_i) \tag{3}$$

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

where d_{step} is the maximum step length, x_{off} is the foot offset with respect to the hip, h is the robot height, g_c is the max ground clearance during swing, and g_p is the max ground penetration during stance. A visualization of the foot trajectory for a set of these parameters is shown in the Pattern Formation block of Figure 2.

We re-sample h, x_{off}, g_c, and g_p during training so the agent can learn to locomote with varying base heights, foot offsets, swing foot ground clearances, and stance foot ground penetrations. We use the following ranges during training: $h \in [0.18, 0.35]$, $x_{off} \in [-0.08, 0.03]$, $g_c \in [0.02, 0.12]$, $g_p \in [0, 0.015]$. This is important to vary in order to find the optimal combination, which is unlikely to be the same for each different gait. The agent does not receive any explicit observation of these parameters, and the user can specify each of these parameters during deployment.

B. Markov Decision Process

1) Action Space: Our action space provides an interface for the agent to directly modulate the intrinsic oscillator amplitudes and phases, by learning to modulate μ_i and ω_i for each leg. This allows the agent to adapt each of these states online in real-time depending on sensory inputs. However, in contrast with our previous work [20], the strong coupling enforces the relative offsets between different oscillators, meaning the agent is forced to learn parameters to locomote

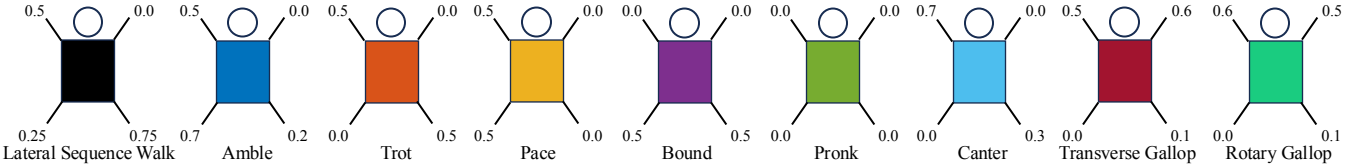

Fig. 4: Contact timing for each foot with the ground as a percentage of a single gait cycle for various quadruped gaits: Lateral Sequence Walk, Amble, Trot, Pace, Bound, Pronk, Canter, Transverse Gallop (T.G.), Rotary Gallop (R.G.). These timings are converted to matrices that denote phase offsets between different limbs in column order: Front Right (FR), Front Left (FL), Hind Right (HR), Hind Left (HL), as they appear in Equation 2.

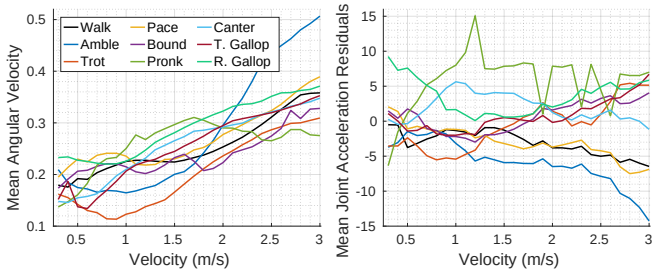

Fig. 5: Minimum mean angular velocity of the base (left), and joint acceleration (right), across all gaits and gait style parameters for all velocities. The residuals with respect to the mean joint acceleration for all gaits at that same velocity is shown for clarity purposes, with negative values indicating gaits with lower joint acceleration.

TABLE I: Reward function terms. $(\cdot)^*$ represents a desired command, and $f(x) := \exp(-\frac{||x||^2}{0.25})$. $dt = 0.01$ is the control policy time step.

Name	Formula	Weight
Linear velocity tracking $v^*_{b,x}$	$f(v^*_{b,x} - v_{b,x})$	$3dt$
Linear velocity penalty $\boldsymbol{v}_{b,yz}$	$-\|\|\boldsymbol{v}_{b,yz}\|\|^2$	$2dt$
Angular velocity penalty $\boldsymbol{\omega}_{b,xyz}$	$-\|\|\boldsymbol{\omega}_{b,xyz}\|\|^2$	$0.1dt$
Power	$-\|\boldsymbol{\tau} \cdot \dot{\boldsymbol{q}}\|$	$0.001dt$

with each particular gait. During training, we resample the coupling matrices randomly among each of the 9 gaits so the agent can learn to locomote with all gaits, as well as transition between different gaits without falling. Our action space can be summarized as $\mathbf{a} = [\boldsymbol{\mu}, \boldsymbol{\omega}] \in \mathbb{R}^8$. The agent selects these parameters at 100 Hz, and we use the following action space ranges during training: $\mu \in [1,2]$, $\omega \in [0,8]$ Hz.

2) Observation Space: Our observation space includes velocity commands, the body state (orientation, linear and angular velocities), joint state (positions, velocities), and foot contact booleans. We also include the last action chosen by the policy network and CPG states (i.e. efference copy of the spinal cord) $\{\boldsymbol{r}, \dot{\boldsymbol{r}}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\}$ as feedback to the policy (i.e. higher centers). Notably, the agent is not directly aware of any coupling matrices (i.e. gaits), nor mapping parameters h, x_{off}, g_c, g_p.

3) Reward Function: Our reward function primarily rewards tracking body linear and angular velocity in the base frame. In addition to forward velocity tracking, we add terms to minimize other undesired base velocities (lateral/vertical oscillations in the base y and z directions, and base roll, pitch, and yaw rates). To minimize energy consumption, we penalize the total power. The terms and respective weights are summarized in Table I. We emphasize that we do not need to add any reward terms beyond those fully specifying the base motion behavior. Notably, we do not need to specify any 'style' rewards to try to enforce any particular gait, base height, foot ground clearance, etc.

TABLE II: Randomized parameters during training and their ranges.

Parameter	Lower Bound	Upper Bound	Units
$v^*_{b,x}$	0.2	3	m/s
Joint Gain K_p	30	100	-
Joint Gain K_d	0.5	2	-
Mass (each body link)	70	130	%
Added base mass	0	5	kg
Coefficient of friction	0.3	1	-

C. Training Details

We use Isaac Gym and PhysX as our training environment and physics engine [19], [21], and the Unitree Go1 quadruped [22]. This framework enables us to simulate 4096 Go1s in parallel on a single NVIDIA RTX 3090 GPU, which allows us to learn control policies within minutes with Proximal Policy Optimization [23]. We use the same hyperparameters and neural network architecture as in [20].

We train on flat terrain, and we reset the environment for an agent if the base or a thigh comes in contact with the ground, or if the episode length reaches 20 seconds. With each reset, we sample new parameters h, g_c, g_p, and x_{off} for mapping the oscillator states to motor commands, allowing the agent to learn continuous locomotion behavior at varying body heights, step heights, and postures. New velocity commands $v^*_{b,x}$ are sampled every 5 seconds, and the gait coupling matrix $\boldsymbol{\Phi}$ is re-sampled every 3 seconds. As in our previous work, we apply domain randomization on the physical mass properties and coefficient of friction (Table II). Finally, an external push of up to 0.5 m/s is applied in a random direction to the base every 15 seconds.

The policy network outputs modulation signals at 100 Hz, and the torques computed from the mapped desired joint positions are updated at 1 kHz. The equations for each of the oscillators (Equations 1-2) are thus also integrated at 1 kHz. During training we re-sample joint PD controller gains at each environment reset as described in Table II.

D. Gait Style Parameter Efficiency

We investigate the effects of different gaits and style parameters on the Cost of Transport ($\text{COT} = \frac{|P|}{m.g.v}$, where P is the average power, m is the mass of the system, g is gravitational acceleration, and v is the average velocity). After training, we consider the following style parameters (body height h, foot ground clearance g_c, foot offset x_{off}) for each gait:

$$h = \{0.18,\ 0.22,\ 0.26,\ 0.30,\ 0.34\}$$
$$g_c = \{0.02,\ 0.05,\ 0.08,\ 0.12\}$$
$$x_{off} = \{-0.075,\ -0.05,\ -0.025,\ 0.0,\ 0.025\}$$

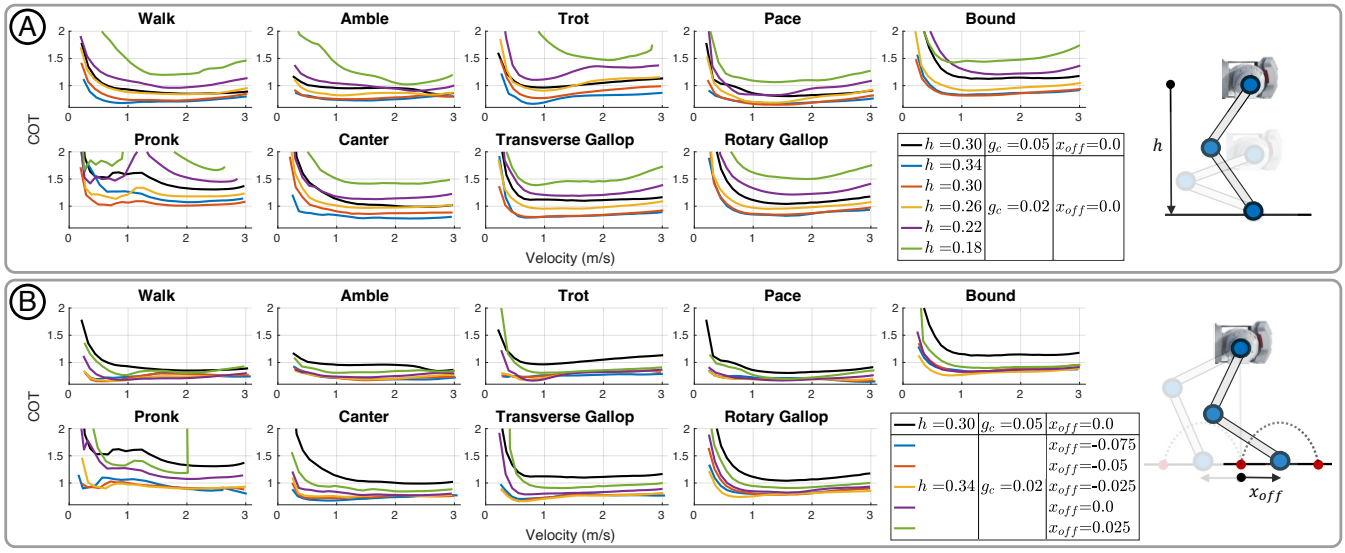

Fig. 6: Effects on the Cost of Transport for all gaits from modulating nominal (A) body height, and (B) foot offset relative to the hip around which oscillations occur. While the policy is capable of locomotion with all of these gait styles (with notable difficulty for the pronk gait with foot offset 0.025 m in front of the hip above 2 m/s), each of these parameters significantly affects the COT, and different combinations of these result in better energy-efficiency at different velocities, most obviously with respect to the foot offset in (B).

For each of the possible 100 combinations of these three parameters, we command the robot to locomote at 0.3 m/s to 3.0 m/s, in increments of 0.1 m/s. For each of these 28 velocities, we run the policy for 5 seconds across 100 robots in parallel, and compute the mean Cost of Transport. For the purpose of this data collection, we do not include any noise in the simulation environment.

Figure 6 shows the effects of varying the three parameters on the Cost of Transport, with respect to baseline parameters seen in previous works such as [20]: $h = 0.3$, $g_c = 0.05$, $x_{off} = 0$, shown by the black line. Figure 6-A shows the effects of different nominal body heights, from 0.34 m to 0.18 m, on the Cost of Transport. As can be expected, a more upright posture generally leads to more efficient locomotion, with lower COT, as less power is needed at the thigh and knee joints to maintain the body height. While almost all gaits have the lowest COT throughout all velocities for the highest, most upright posture, the pronk gait is a notable exception, with the most efficient locomotion for a slightly lower nominal base height parameter, 0.3 m. However, many gaits have an almost identical COT curve when locomoting with body height 0.3 m or 0.34 m. In the video, we also investigate the effects on the COT of changing the nominal swing foot ground clearance from 0.02 m to 0.12 m. As can be expected, a lower swing foot ground clearance results in a more efficient gait, as it requires more energy to bring the foot higher off the ground, and this trend is consistent across all gaits.

Figure 6-B shows the effects of changing the nominal center point around which the oscillations take place with respect to the hip in the x direction range from -0.075 m behind the hip, to 0.025 m in front of the hip. Due to the configuration of the legs and body, the overall Center of Mass (COM) of the robot lies behind the geometric center of the body. Therefore, an offset behind the hips helps to keep the COM more at the center of the foot contacts, thereby (generally) decreasing energy required to remain upright during locomotion. While there is more variability among gaits, and even within a single gait as the locomotion velocity changes, a general observation is that it is more energy-efficient to locomote with the oscillation x center point behind the hips.

E. CPG Parameter Modulation Across Different Gaits

One interesting observation for the pronk gait is the reverse curvature with respect to nominal COT plots, which typically have positive parabola-like shapes, with a minimal COT at a particular velocity. For the pronk gait, at low speeds, the policy has learned to modulate the CPG parameters to slowly lean forward, then take a short and fast hop, to more efficiently track the mean desired velocity. This is further shown in Figure 7 by the mean CPG amplitude and frequency selected by the policy to locomote as a function of speed, which changes for each gait and style. The amplitude and frequency of the oscillators directly correspond to mean step length and mean step frequency. Interestingly, we see several inflection points for several gaits, where the strategy changes for increasing locomotion speed. This is most obvious for the pace and amble gaits, which actually decrease the step frequency at higher speeds, while instead locomoting faster by increasing the step length. The policy has learned this different coordination among gaits in order to maximize the returns from our reward function, and without explicit knowledge of the coupling parameters or gait style, but which it can indirectly deduce through observing the CPG and joint states. Thus, our framework can also be used as a tool to evaluate different locomotion strategies given a particular gait.

F. Hardware Qualitative Evaluation

1) All Gaits and Transitions: In the video, we show both simulation and hardware results from deploying our single policy to accomplish all 9 gaits, with a variety of

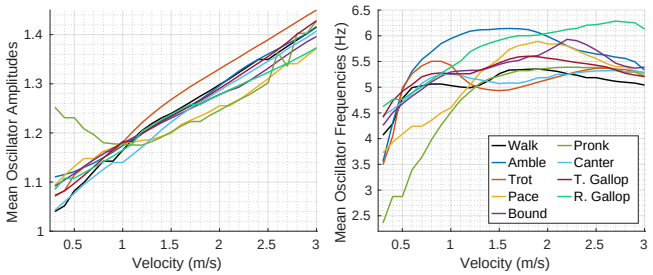

Fig. 7: Mean oscillator amplitudes and frequencies for style parameters $h = 0.34$, $g_c = 0.02$, $x_{off} = -0.075$, which is optimal for many gaits and at many velocities. Notably, the policy coordinates the amplitudes and frequencies differently for different gaits.

styles (i.e. varying body height, swing foot height, and foot offsets). Notably, we are able to transition between any gaits at any velocity, just by re-sampling the coupling matrix, without any explicit input to the policy network. We can also modulate the gait style parameters within the same gait, or switch these along with the gait, by changing the pattern formation parameters.

2) Novel Gaits: We test the robustness and ability of the policy to produce and transition between novel gaits not seen during training. Since we had already trained for all quadruped gaits, we develop several artificial gaits (not found in nature) by creating new coupling matrices. For example, we test new gaits where three limbs are coupled in phase, and one is out of phase with a π offset. The out of phase leg can be either a front or rear leg, and the video shows that although these gaits and observations were never seen during training, we can still effectively locomote with such new gaits at test time in hardware experiments. We also test different timing patterns for several gaits, for example modifying the trot or bound with different timing to create more uneven gaits, which still result in successful locomotion and velocity tracking.

3) Leg Failure Robustness: Lastly, we show that our single policy is robust among different gaits to disabling either one or two legs. We design an experiment where we transition from trot, to pace, to bound, to pronk; and then repeat the experiment with disabling (locking) one or two rear legs, so that they stay fixed in a nominal extension position. The robot does not fall down and continues to locomote during these experiments, despite never having encountered any of these situations during training.