SYNLOCO: Synthesizing Central Pattern Generator and Reinforcement Learning for Quadruped Locomotion

Zhang, Xinyu; Xiao, Zhiyuan; Zhang, Qingrui; Pan, Wei

DOI
10.1109/CDC56724.2024.10886438

Publication date
2025

Document Version
Final published version

Published in
Proceedings of the IEEE 63rd Conference on Decision and Control, CDC 2024

2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

SYNLOCO: Synthesizing Central Pattern Generator and Reinforcement Learning for Quadruped Locomotion

Xinyu Zhang[1*], Zhiyuan Xiao[1*], Qingrui Zhang[1], and Wei Pan[2]

2024 IEEE 63rd Conference on Decision and Control (CDC) | 979-8-3503-1633-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/CDC56724.2024.10886438

***Abstract*— The Central Pattern Generator (CPG) is adept at generating rhythmic gait patterns characterized by consistent timing and adequate foot clearance. Yet, its open-loop configuration often fails to adjust the system's control performance in response to environmental variations. On the other hand, Reinforcement Learning (RL), celebrated for its model-free properties, has gained significant traction in robotics due to its inherent adaptability and robustness. However, initiating traditional RL approaches from the ground up presents a risk of converging to suboptimal local minima and slow learning convergence. In this paper, we propose a quadruped locomotion framework–called SYNLOCO–by synthesizing CPG and RL, which can ingeniously integrate the strengths of both methods, enabling the development of a locomotion controller that is both stable and natural with partial state observations (*e.g.*, no velocity measurements). To optimize the learning trajectory of SYNLOCO, a two-phase training strategy is presented. Both ablation analysis and experimental comparison are performed using a real quadruped robot under varied conditions, including distinct velocities, terrains, and payload capacities. The experiments showcase SYNLOCO's efficiency in producing consistent and clear-footed gaits across diverse scenarios, despite no velocity measurements. The developed controller exhibits resilience against substantial parameter variations, underscoring its potential for robust real-world applications.**

I. Introduction

Quadruped robots are expected to become more common in industrial and daily life due to their extraordinary mobility competence in rough terrain and load-carrying capacity. In particular, quadruped robots offer the best advantage in terms of mobility and versatility in various harsh and dangerous scenarios. In these challenging tasks, quadruped robots must traverse different terrains with different payloads. Therefore, robust locomotion capabilities are highly expected for quadruped robots in real-world applications.

An interesting solution for quadruped locomotion is based on Central Pattern Generators (CPGs), which are circuits in the self-contained integrative nervous system of animals that generate repetitive movements such as walking, swimming, and crawling [1], [2]. Therefore, a CPG can generate rhythmic joint signals for locomotion control without using model information [3]–[5]. CPG-based locomotion control is simple, stable, and time-efficient, making it suitable for real-time online implementations. However, most CPG-based locomotion control methods are open-loop, thus leading to poor adaptation to environmental changes. Parameter tuning is tedious and non-trivial for all CPGs, often requiring a large amount of expert knowledge or the use of advanced optimization techniques, e.g., genetic algorithms [5]. It should also be noted that CPG-based quadruped locomotion often performs poorly on unstable quadrupeds [6]. However, the CPG signal is suitable as a baseline signal to generate a rhythmic gait pattern, while a closed-loop quadruped locomotion controller should be additionally considered.

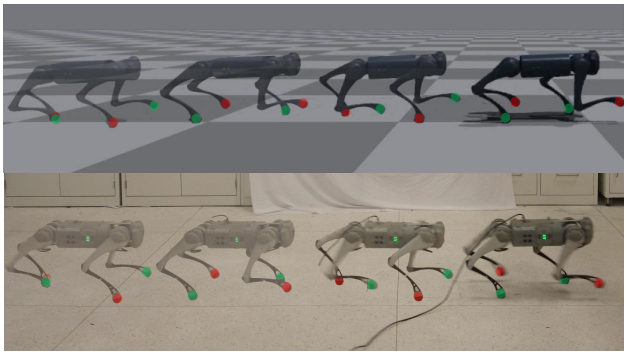

Fig. 1: Training and evaluation snapshots. Red points mark the stance foot; green points indicate the swing foot.

Reinforcement learning (RL) does not require a mathematical model similar to CPGs, which has recently shown great potential in quadruped locomotion control [7]–[10]. In RL, a locomotion control policy is learned using input and output data collected through robot interactions with environments [7]. The learning process of RL is completely data-driven, so no model information is required. Although learning an end-to-end controller by RL is promising for quadruped robots, the learned policy requires inferring actions under partial observation. For example, trunk velocity cannot be measured by onboard proprioceptive sensors. Learning a policy directly by vanilla RL algorithm under partial observation is challenging. A vanilla RL algorithm is prone to get stuck in local minima that exhibit strange locomotion. Therefore, many previous works explore the solution in terms of learning procedure, complex reward function formulation, and additional system identification modules in an attempt to harness more efficient learning by reducing the risk of getting stuck in a local minimum. In this paper, we are inspired by CPGs, leading us to question: *How could we embed the CPG*

*These authors contributed equally to this work.

This work is supported by State Key Laboratory of Robotics and Systems (HIT) under Grant SKLRS-2024-KF-05, in part by Shenzhen Science and Technology Program JCYJ20220530145209021, and in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2024A1515012408.

[1]School of Aeronautics and Astronautics, Shenzhen Campus of Sun Yat-sen University, Shenzhen 518107, P.R. China. Correspondence to: Qingrui Zhang (zhangqr9@mail.sysu.edu.cn)

[2]Department of Computer Science, The University of Manchester, UK and Department of Cognitive Robotics, Delft University of Technology, Netherlands.

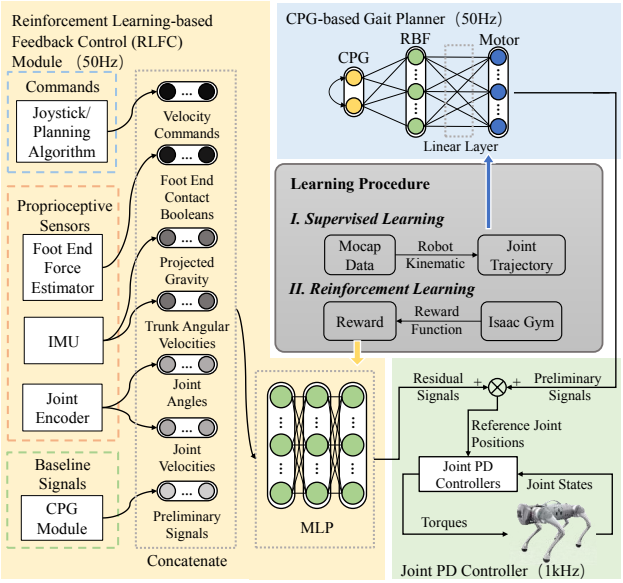

Fig. 2: The SYNLOCO framework synthesizes RL and a CPG for quadruped locomotion.

knowledge into RL to enhance its performance in learning natural and robust locomotion for quadruped robots with partial observations?

Hence, we aim to learn a control policy for stable, natural, and robust locomotion of quadruped robots under partial observation. To achieve this, we propose an integrated framework called SYNLOCO, which combines a CPG-based gait planner with a reinforcement learning-based feedback control (RLFC) module, as illustrated in Fig. 2. This framework leverages the strengths of both CPGs and RL, enabling efficient learning of quadruped locomotion control without velocity measurements. The resulting policy allows the robot to respond effectively to various commands and terrains with a stable, natural gait. Extensive experiments and ablation analysis confirm the competence of the proposed approach. The contributions of this paper are threefold:

1) A quadruped locomotion framework, SYNLOCO, is developed by combining CPG and RL. CPG generates rhythmic gait patterns to guide RL in avoiding local minima, while RL refines these patterns using proprioceptive feedback. SYNLOCO can achieve stable, natural, and robust quadruped locomotion under partial observations.
2) A two-step training mechanism is introduced for SYNLOCO, where the CPG is first trained offline via supervised learning to clone animal behavior, followed by RL training to adjust the CPG with sensor feedback. This approach reduces the difficulty of policy learning and speeds up training, leading to more natural locomotion.
3) SYNLOCO's efficiency is analyzed through ablation studies and real-world experiments, confirming its ability to avoid local minima without complex reward functions or system identification. It outperforms vanilla RL, ensuring stable locomotion without trunk velocity measurements.

II. Related Works

A. Central pattern generator for quadruped locomotion

CPGs can generate rhythmic signals for regulating repetitive movements in bio-inspired robots. Numerous artificial CPG models, such as the Hopf [11], Kimura [12], and $SO(2)$ oscillator models [5], have been proposed for locomotion control tasks. The $SO(2)$ model, in particular, is favored for its computational efficiency for online implementation [6], [13], [14]. However, parameter tuning remains difficult for CPGs. Combining a CPG with other approaches has been discussed to build a general auto-tuning framework for locomotion. In [6], a $SO(2)$ oscillator model is synthesized with a radial basis function (RBF) network that is optimized using a black-box method. A modular CPG-RBF network is introduced to realize legged robot control with fast learning [14]. However, these works focus on the hexapod robot without feedback control and need better performance for quadruped robots. Bellegarda *et al.* proposed a framework using an RL method to adjust CPG parameters [4]. RL learns the gait pattern and foot trajectory, which enables a quadruped robot to move with different foot clearance and body height. Shi *et al.* proposed a framework that includes a learned CPG-based evolutionary trajectory generator with an RL-trained network to perform multitask locomotion [13]. These auto-tuned CPGs cannot guarantee the learned gait pattern, since defining a reward function to specify the coupled relationship between legs is difficult. These works bring a viable approach combining CPG and other data-driven methods, inspiring us with a hybrid approach to design a controller.

B. Reinforcement learning-based locomotion control

In recent years, RL has been successfully applied to quadruped locomotion [7], [15]. RL-trained networks demonstrated high-speed locomotion capabilities on challenging and deformable terrains [8], [9]. Agile locomotion can be learned via DRL with an adaptive curriculum on commands and online system identification [16]. RL-based locomotion control requires a lot of data for policy training, which is time-consuming and risky. Some data is even impossible to collect in the real world. Therefore, policy training in the form of simulated data is preferred for robot control tasks. [7], [8], [15], [17]. Considering the discrepancy between the simulated and real environments, this approach raises concern about the possibility of successful sim-to-real transfer. Peng *et al.* realized the robot arm policy transfer by a domain randomization method that randomly changes the parameters of the model in training [18]. Tan *et al.* analyzed the performance of domain randomization in quadruped locomotion [19]. The domain randomization method allows the trained policy to be deployed to the robot without any other re-tuning, which applies to the work in this paper.

C. Policy learning under partial observation

It is a challenge for RL to learn a feasible policy with partial observation, particularly in quadruped locomotion control. The limited information provided by onboard proprioceptive sensors hampers learning efficiency and exacerbates the issue

of local minima. In [20], an off-policy asymmetric Actor-Critic (AC) method is proposed, in which the critic receives full states as input while the actor uses partial observations. Another solution is Privileged Learning (PL), which employs a teacher-student architecture [8], [16]. The teacher policy is trained using vanilla RL with privileged full-state information. The student policy mimics the behaviors of the teacher policy using limited onboard measurements. However, PL's training is complex, and its success hinges on the quality of the teacher. A third solution is to define complex reward functions to guide policy learning under limited observation [17]. This approach requires tedious weight tuning to balance the importance of different rewards. This paper presented a knowledge-informed solution by synthesizing a CPG with RL to address policy learning under limited observation. The proposed solution is compatible with both on-policy and off-policy RL and can be combined with the above-mentioned methods for further performance improvement.

III. CPG-BASED GAIT PLANNER

A CPG-based gait planner is introduced to generate preliminary signals for quadruped robots. Following the approach in [6], [14], we utilize a CPG-RBF network as the gait planner. This network is capable of producing the desired gait patterns and frequencies necessary for quadruped locomotion. The planner is optimized through supervised learning for behavior cloning.

A. Network architecture

The CPG-RBF network consists of a CPG layer, a Radial Basis Function (RBF) layer, and a motor layer. The CPG layer is a simple $SO(2)$ oscillator designed to generate stable rhythmic signals resilient to perturbations. It includes two neurons with a sigmoid activation function, which can be expressed as

$$\begin{bmatrix} o_0(t+1) \\ o_1(t+1) \end{bmatrix} = \tanh\left(\alpha \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} o_0(t) \\ o_1(t) \end{bmatrix} \right), \quad (1)$$

where ϕ is the desired oscillation velocity, $o_i(t)$ represents the neuron state, α is a hyper-parameter set to be 0.01. The parameter ϕ is set to $\frac{\pi}{60}$ to match the gait frequency of the demonstrated locomotion.

The RBF layer takes o_0 and o_1 as inputs and outputs a Gaussian-like expression.

$$R_h = exp\left(-\frac{(o_0-\mu_{h,0})^2 + (o_1-\mu_{h,1})^2}{\sigma_{RBF}^2} \right), \quad (2)$$

where $\mu_{h,j}$ is the position, and σ_{RBF} is the width of RBF neuron h. A bell-shaped curve R_h is generated with a center $\mu_{h,n}$ and width σ_{RBF}^2. The center of each RBF neuron is uniformly distributed along the limit cycle of the CPG output, represented as $\mu_{h,n} = o_n\left(\frac{hT}{H}\right)$, where T is the signal period and H is the number of RBF neurons in the layer. We set $H = 20$ and $\sigma_{RBF} = 0.1$ for this layer.

The motor layer is a fully connected neural network with one layer without activation functions. The output of the motor layer is a 12-dimensional vector representing the baseline position reference signals for 12 joints.

TABLE I: Reward functions and weights

Name	Expression	Weight
Linear velocity tracking	$\exp(-\frac{\|\|v^*_{b,xy}-v_{b,xy}\|\|^2}{0.25})$	1dt
Angular velocity tracking	$\exp(-\frac{\|\|\omega^*_{b,z}-\omega_{b,z}\|\|^2}{0.25})$	0.5dt
Linear velocity penalty	$v^2_{b,z}$	-2dt
Angular velocity penalty	$\|\|\omega_{b,xy}\|\|^2$	-0.05dt
Trunk orientation	$\|\|\hat{g}_x\|\|^2 + \|\|\hat{g}_y\|\|^2$	-5dt
Trunk height	$1-\exp(-\frac{\|\|h_b-h^*_b\|\|^2}{8.1\times10^{-4}})$	-1dt
Joints acceleration	$\|\|\frac{\dot{q}_{j-1}-\dot{q}_j}{dt}\|\|^2$	-1×10^{-7}dt
Action rate	$\|\|q^*_{j-1}-q^*_j\|\|^2$	-0.005dt
Self collision	$n_{collisions}$	-0.001dt
Feet airtime	$\sum_{f=0}^{4}(t_{air,f}-0.5)$	1.5dt
Feet position	$\sum_{n=1}^{4}\exp(-\frac{\|\|p_f-p_d\|\|^2}{0.02})$	0.3dt

B. Supervised learning for behavior cloning

The weights of the motor layer are optimized through behavior cloning to replicate gaits from demonstrations. The demonstration data, sourced from a dog trotting at 1.5 Hz as described in [21], includes foot trajectories. These foot trajectories are then transformed into joint trajectories using the robot's kinematics. The behavior cloning process is conducted via supervised learning, aiming to minimize the mean square error (MSE) loss between the demonstrated joint trajectories and the gait planner's generated outputs.

IV. RL-BASED FEEDBACK CONTROLLER

SYNLOCO leverages RL to develop an RL-based feedback control policy (RLFC) that fine-tunes the CPG-based gait planner through residual signals. This improves quadruped locomotion using proprioceptive feedback. Further details of the RLFC are provided in this section.

Observations: The observation space includes high-level commands $\{v^*_x, v^*_y, \omega^*_z\}$, trunk angular velocity $\{\omega_x, \omega_y, \omega_z\}$, gravity unit vector in the body frame $\hat{g}$, joint positions and velocities $\{q_0, ..., q_{11}, \dot{q}_0, ..., \dot{q}_{11}\}$, foot contact states $\{c_0, c_1, c_2, c_3\}$, previous actions, and gait planner signals. All observations are normalized. Unlike prior works [8], [15], linear trunk velocity is not necessary.

Actions: The action space is 12-dimensional, representing residual joint command signals for the 12 actuators of a quadruped robot. The final joint position command q_t is the sum of the CPG-based gait planner output $q_{CPG,t}$ and the RLFC module output $q_{RLFC,t}$, or $q_t = q_{CPG,t} + q_{RLFC,t}$. A first-order low-pass filter smooths q_t before it reaches the PD controllers, expressed as $s_t = \alpha q_t + (1-\alpha)s_{t-1}$ with $\alpha = 0.7$. Joint PD controllers then compute torque using $K_p = 75$ and $K_d = 1.5$, which actuates the robot's locomotion.

Reward functions: The total reward is a weighted sum of 11 terms, as shown in TABLE I. This reward encourages the robot to follow target velocities and maintain energy efficiency while penalizing unsteady trunk motion and unexpected collisions. The 'Feet airtime' term promotes a longer swing phase to prevent dragging, and the 'Feet position' term regulates deviations from the CPG-generated trajectory.

RL policy: The Proximal Policy Optimization (PPO) algorithm [22] is used to train the feedback control policy. Both the actor and critic are parameterized by a three-layer

TABLE II: Parameters for domain randomization

	Parameter	Range
Randomized dynamics	Trunk mass	[-1,1] kg
	Foot friction coefficient	[0.5,1.25]
Trunk impulse	External force magnitude v_{xy}	[-1.8,1.8] m/s
	External force interval	15 s
Sensor noises	Trunk angular velocity noise	[-0.05,0.05] rad/s
	Gravity vector noise	[-0.05,0.05]
	Joint positions noise	[-0.01,0.01] rad
	Joint velocities noise	[-0.075,0.075] rad/s

Multi-Layer Perceptron (MLP) with ELU activation function. The hidden layers are sized at 512, 256, and 128, respectively.

Domain randomization: In each training episode, body mass and terrain friction are randomized to simulate various environments. An instantaneous linear velocity change to the trunk is also randomly applied to train a policy robust to disturbances. A curriculum learning strategy, similar to [7], is employed, gradually increasing perturbations as locomotion robustness improves. Noise is added to sensor feedback to improve the controller's robustness against sensor errors. The randomized parameters are detailed in TABLE II and follow a uniform distribution.

Training setup: The training is conducted in the Isaac Gym environment [23]. As shown in Fig. 1, the quadruped robot is trained to follow high-level commands on flat and gently sloping terrain with 4096 parallel agents. The simulation runs at 200 Hz, while the policy operates at 50 Hz. Each episode lasts up to 20 seconds (1000 time steps) and terminates early if the robot's trunk collides with the ground. Omnidirectional velocity commands are randomly sampled from a uniform distribution every 10 seconds. At the start of each episode, a robot spawns in a default pose with randomized commands.

V. Results and Analysis

This section presents the results of the ablation study and real-world experiments. Ablation studies analyze the contribution of each module, and experimental comparisons highlight SYNLOCO's advantages over vanilla RL. Experiments validate the robustness of the learned locomotion control across different commands, terrains, and payloads.

A. Ablation study on the effect of RLFC

An ablation study of the RLFC module is conducted on a real-world quadruped to assess its necessity. As shown in Fig. 3, the quadruped is unable to maintain trunk stability using only the CPG module, demonstrating that open-loop CPG signals are insufficient for quadruped locomotion. While some studies achieve stable locomotion with open-loop CPG, this approach is effective only for robots with static stability, such as hexapods [5], [14] or salamander robots [11]. In our case, the quadruped is statically unstable with a low-frequency trot and a long step length of up to 0.5 m, making an RL-based feedback controller essential.

Fig. 3: Locomotion with only the CPG module. The robot is unable to maintain trunk stability and falls over as the front right foot gets stuck.

TABLE III: Trunk states at different velocity commands

Trunk state		Velocity command (m/s)				
		Varying	0.25	0.5	0.75	1.00
Velocity (m/s)	Mean	/	0.233	0.472	0.704	0.874
	Std. Dev.	/	0.014	0.020	0.033	0.040
Height (m)	Mean	0.317	0.318	0.319	0.317	0.316
	Std. Dev.	0.005	0.004	0.005	0.006	0.007
Pitch (deg)	Mean	2.745	1.872	2.754	2.622	3.702
	Std. Dev.	2.480	0.666	2.846	7.082	4.101
Roll (deg)	Mean	-0.050	0.183	-0.162	-0.113	0.059
	Std. Dev.	1.361	1.371	1.289	1.192	1.514

B. Ablation study on the effect of CPG

An ablation study of the CPG module is conducted through real-world experiments. The CPG signal is replaced by the default joint angle, making the RLFC output the primary control signal rather than a residual one. The action scale of the RLFC is increased to promote better exploration. Policy training is repeated five times with different seeds. Both simulations and experiments show that the learning process frequently became trapped in local minima. The experiments confirm that RLFC alone could not learn a rhythmic gait without trunk velocity measurements. Despite penalties for action rate and joint acceleration, the quadruped still exhibit unnatural, infeasible locomotion, as shown in Fig. 4.

C. Locomotion control for varying velocity commands

In the first scenario, the robot tracks constant velocity commands of 0.25 m/s, 0.5 m/s, 0.75 m/s, and 1 m/s, as shown in TABLE III. Data is collected using the OptiTrack motion capture system. Experiments reveal that tracking performance declines at higher speeds due to increased stride length, reducing locomotion stability and accuracy. In the

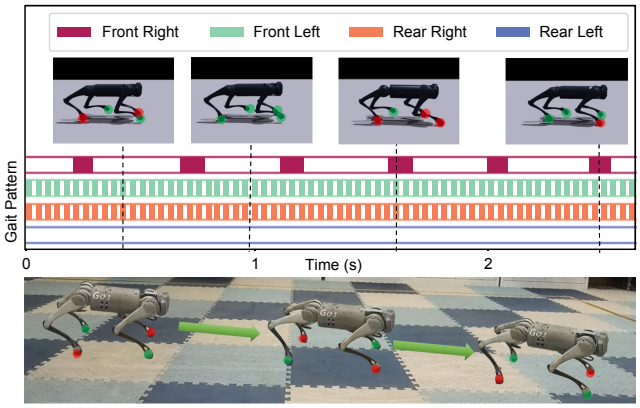

Fig. 4: Locomotion with only the RLFC module. The robot exhibits high-frequency gait (25 Hz) but with small step length, resulting in dragging its feet on the ground.

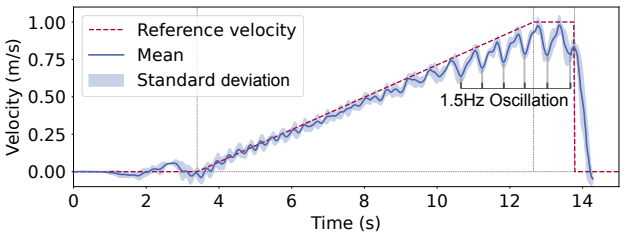

Fig. 5: Trunk velocity responses to a varying command across five trials. At higher speeds, periodic oscillations in trunk velocity correspond to the gait pattern frequency.

second scenario, the robot can track a time-varying velocity command from 0 m/s to 1 m/s, as depicted in Fig. 5. The robot maintains a steady trotting gait, with the stance phase slightly longer than the swing phase to improve stability, as shown in Fig. 6.

The robot maintains sufficient ground clearance at various speeds, with front legs clearing 7 cm and rear legs 4 cm, as shown in Fig. 7. The foot remains stable during the stance phase, while step length smoothly adjusts with changes in trunk velocity. As seen in Fig. 8, the CPG-based gait planner generates a baseline foot trajectory, which is regulated by the RLFC to adjust step length based on velocity commands. This demonstrates the stability and smoothness of locomotion under the SYNLOCO algorithm, as shown in Fig. 1.

The relationship between the target foot trajectory generated by SYNLOCO and the actual trajectory under varying velocity commands is shown in Fig. 9. The actual trajectory closely matches the desired foot clearance and step length, while the target trajectory is non-convex and incompatible with the robot's kinematics. This indicates that the RL module compensates for ground impact by adjusting the target trajectory and generating feed-forward torque.

D. Locomotion control at different tasks

As shown in Fig. 10, the SYNLOCO algorithm is tested in various environments. Commands for forward velocity ($[0,1]$ m/s) and yaw rate command($[-1,1]$ rad/s) are provided via joystick. The robot maintains a stable trotting gait, with smooth transitions between forward trotting and turning. It traverses diverse terrains, including rigid floors, soft mattresses, slopes, and ramps. The control policy's robustness is further assessed by adding an 11 kg payload, nearly

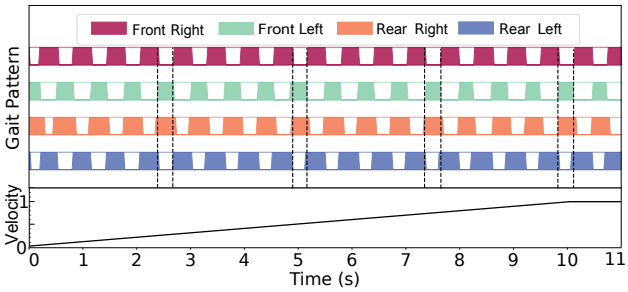

Fig. 6: Robot gait pattern under varying commands. The robot maintains a steady trotting gait with a frequency of approximately 1.5 Hz at different speeds.

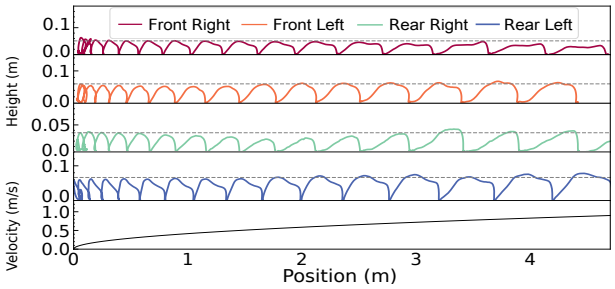

Fig. 7: Foot trajectories in the XOZ plane of the local frame under varying commands. The foot does not slip during the stance phase, with adequate clearance during the swing phase.

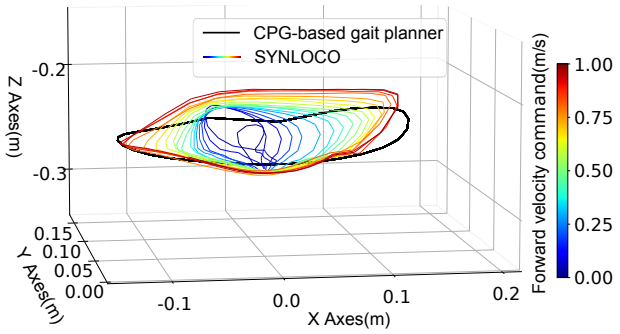

Fig. 8: Rear left foot trajectory under varying commands by SYNLOCO. The framework adaptively regulates the foot trajectory in response to commands while maintaining sufficient foot clearance.

equivalent to the robot's torso mass, demonstrating the SYNLOCO controller's robustness in handling different tasks.

E. Comparison between SYNLOCO and vanilla RL

For comparison, we train a baseline locomotion control policy using vanilla RL. To ensure fairness, the observation space of the baseline matches that of SYNLOCO, excluding trunk linear velocity measurements. The policy configuration follows [15], utilizing identical action and reward functions, with joint PD controller parameters set to $P = 20$ and $D = 0.5$ to encourage smooth locomotion. As shown in Fig. 11, SYNLOCO produces stable, natural movement, while vanilla RL causes the robot to crawl with low trunk height and spread legs. Vanilla RL results in violent locomotion with excessive actuator torque, making it unsuitable for real quadruped.

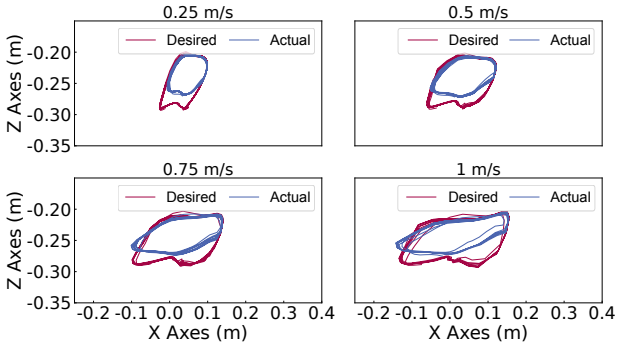

Fig. 9: XZ trajectories of the front right foot in the body frame by SYNLOCO algorithm at different velocity commands.

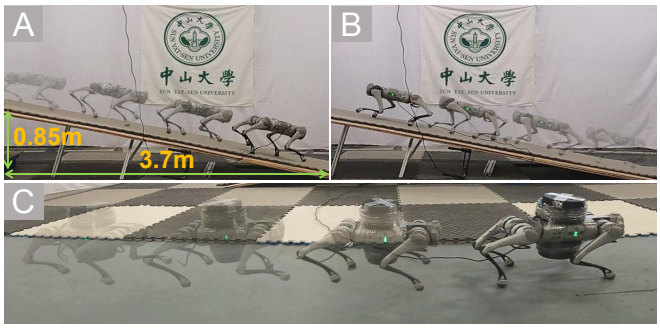

Fig. 10: Experiments. (A) Downhill. (B) Uphill. (C) Trotting with an 11 kg payload at 0.8 m/s.

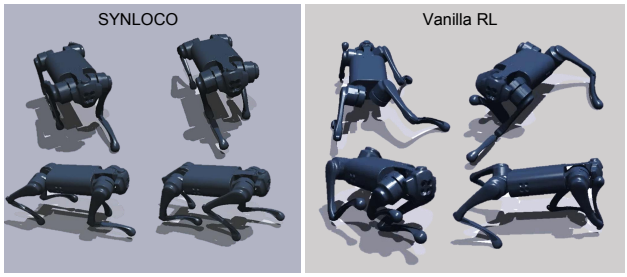

Fig. 11: Comparison between SYNLOCO and vanilla RL in simulation.

This comparison highlights SYNLOCO's advantage over vanilla RL under partial observation conditions. As noted in [24], learning a locomotion policy without trunk linear velocity observations increases the complexity of policy learning. With limited proprioception, it becomes difficult to accurately infer the robot's state. Vanilla RL struggles to explore optimal gaits through trial and error in such conditions. In contrast, SYNLOCO leverages gait pattern priors from a CPG module, reducing the risk of local minima and enabling natural, stable locomotion. Furthermore, CPG parameters can be efficiently learned via imitation, offering an autonomous method to design natural locomotion without manual tuning.

VI. Conclusions

The article introduces SYNLOCO, a framework combining a CPG-based gait planner with an RL-based Feedback Control (RLFC) module to enhance quadruped robot locomotion. The gait planner generates preliminary signals, while the RLFC module adjusts them using sensor feedback. This method improves performance over standard RL and CPG methods, especially in partially observed environments. Ablation studies and real-world experiments confirmed the efficiency of SYNLOCO, although performance declined at higher velocities due to the lack of velocity measurements. The system demonstrated robust locomotion under various conditions, including heavy payloads. Future work will focus on the adaptation to diverse terrains, *e.g.*, stairs and gaps.

References

[1] R. Yuste, J. N. MacLean, J. Smith, and A. Lansner, "The cortex as a central pattern generator," *Nature Reviews Neuroscience*, vol. 6, no. 6, pp. 477–483, 2005.

[2] F. Delcomyn, "Neural basis of rhythmic behavior in animals," *Science*, vol. 210, no. 4469, pp. 492–498, 1980.

[3] J. Wang, C. Hu, and Y. Zhu, "CPG-Based hierarchical locomotion control for modular quadrupedal robots using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7193–7200, 2021.

[4] 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.

[5] M. Thor and P. Manoonpong, "A fast online frequency adaptation mechanism for CPG-based robot motion control," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3324–3331, 2019.

[6] M. Thor, T. Kulvicius, and P. Manoonpong, "Generic neural locomotion control framework for legged robots," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 9, pp. 4013–4025, 2021.

[7] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[8] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, p. eabc5986, 2020.

[9] S. Choi, G. Ji, J. Park, H. Kim, J. Mun, J. H. Lee, and J. Hwangbo, "Learning quadrupedal locomotion on deformable terrain," *Science Robotics*, vol. 8, no. 74, p. eade2256, 2023.

[10] Z. Xiao, X. Zhang, X. Zhou, and Q. Zhang, "PA-LOCO: Learning perturbation-adaptive locomotion for quadruped robots," 2024. [Online]. Available: https://arxiv.org/abs/2407.04224

[11] 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.

[12] H. Kimura, S. Akiyama, and K. Sakurama, "Realization of dynamic walking and running of the quadruped using neural oscillator," *Autonomous Robots*, vol. 7, no. 3, pp. 247–258, 1999.

[13] H. Shi, B. Zhou, H. Zeng, F. Wang, Y. Dong, J. Li, K. Wang, H. Tian, and M. Q.-H. Meng, "Reinforcement learning with evolutionary trajectory generator: A general approach for quadrupedal locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3085–3092, 2022.

[14] M. Thor and P. Manoonpong, "Versatile modular neural locomotion control with fast learning," *Nature Machine Intelligence*, vol. 4, no. 2, pp. 169–179, 2022.

[15] 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*, Auckland, New Zealand, 2022, pp. 91–100.

[16] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," in *Proceedings of Robotics: Science and Systems*, New York, USA, 2022.

[17] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," in *Proceedings of The 6th Conference on Robot Learning*, Atlanta, USA, 2023, pp. 22–31.

[18] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia, 2018, pp. 3803–3810.

[19] 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," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, USA, 2018.

[20] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," in *Proceedings of Robotics: Science and Systems*, vol. 14, Pittsburgh, USA, 2018.

[21] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "DeepMimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 143:1–143:14, 2018.

[22] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

[23] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance GPU-based physics simulation for robot learning," 2021. [Online]. Available: https://arxiv.org/abs/2108.10470

[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.