

Learning multiple gaits of quadruped robot using hierarchical reinforcement learning

Yunho Kim, Bukun Son, and Dongjun Lee

Abstract—There is a growing interest in learning a velocity command tracking controller of quadruped robot using reinforcement learning due to its robustness and scalability. However, a single policy, trained end-to-end, usually shows a single gait regardless of the command velocity. This could be a suboptimal solution considering the existence of optimal gait according to the velocity for quadruped animals [1], [2]. In this work, we propose a hierarchical controller for quadruped robot that could generate multiple gaits (i.e. pace, trot, bound) while tracking velocity command. Our controller is composed of two policies, each working as a central pattern generator and local feedback controller, and trained with hierarchical reinforcement learning. Experiment results show 1) the existence of optimal gait for specific velocity range 2) the efficiency of our hierarchical controller compared to a controller composed of a single policy, which usually shows a single gait. Codes are publicly available [link](#).

I. INTRODUCTION

Designing a velocity command tracking controller of a quadruped robot is crucial for related applications. However, it is a complex problem due to rich contact with environment that is hard to model or predict. Reinforcement learning is an interesting solution for this due to its robustness and scalability which are shown in previous works [4], [5]. Previous works used a simple framework composed of a single policy and trained it in end-to-end manner. However, this simple framework usually results a single gait regardless of the command velocity.

According to the previous works done in biomechanics community, quadruped animals show different optimal gaits according to the velocity [1]. Similar results were also shown in simulated simple quadruped robot [2]. Thus, the result of simple framework, which shows a single gait, could be a suboptimal solution. There could be advantages in perspective of energy usage and velocity tracking accuracy if we use multiple gaits. Thus, we present a novel hierarchical controller that could learn both single or multiple gaits. The details of the proposed controller will be explained in Section III

II. RELATED WORKS

A. Locomotion control

Designing a controller for agile locomotion, including gait pattern generation, is currently an active area of research in both simulation and real world robots. We could categorize previous approaches in three categories: model based analytic

control, data driven control with reference motions, data driven control without reference motions.

First is model based analytic control. After mathematically modeling the dynamics and interactions of the robot, optimal solutions of the formulated problem could be used as control signal [8], [9], [10], [11], [12]. However, inaccurate dynamic model, complex optimization problem, stochastic environment, and online computation limits cause the approaches limited in robustness and scalability. Second is data driven control with reference motions. Reinforcement learning is often used in data driven control. Previous approaches designed simple reward to track reference motion and alleviated the difficulty of learning [13], [17]. The method was also successfully implemented in real world quadruped robot [14]. However, using reference motions and tracking reward signal to learn multiple behaviors could result suboptimal solution due to gradient conflicts and suboptimal reference motions. Some approaches alleviated the objective of strictly following reference motions using generative adversarial network [15], [16], [26]. Last is data driven control without reference motions. Designing specific reward signal enables learning desired behavior and often generates unexpected behavior [4], [18]. However, designing specific reward signal requires lots of trial and error and often difficult to learn multiple behaviors in a single policy.

B. Gait generation

Humans and animals show various gait patterns and smooth transitions depending on the environment and desired velocity. Researchers actively worked on generating and controlling multiple gaits considering the relationship between them [20], [25]. Erden et al. designed specific reward based on heuristics and learned adaptive gaits in limited environment [19]. Singla et al. manually time shifted 'walk' gait reference data to generate different gait data, and successfully restored other gaits using kinematic motion primitives [20]. Siekmann et al. proposed a simple reward signal to learn several bipedal gaits considering the periodicity of gait control signal [21]. However, previous approaches are quite limited in robustness and scalability because it is based on manual data generation process and limited heuristics. Furthermore, the methods didn't consider the relationship between the gait pattern and desired velocity, and rather designed the policy to take desired gait pattern as input signal.

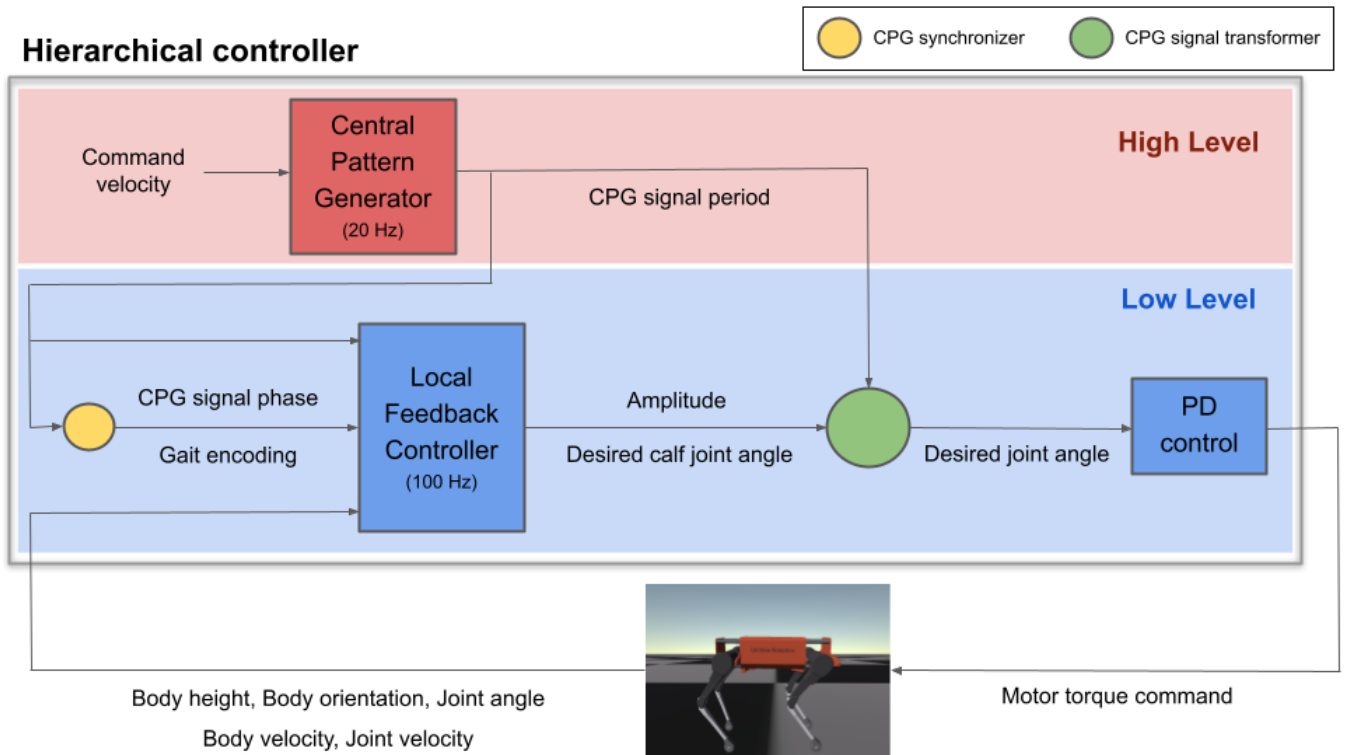


Fig. 1: Proposed hierarchical controller for quadruped robot

C. Quadruped gait

Quadruped animals show various gaits (i.e. walk, pace, trot, bound etc). Previous researches experimentally showed the reason for gait selection and transition. Hoyt et al. discovered the relationship between gaits, velocity, and energy consumption in horse [1]. They found that each gait shows convex function of energy consumption according to the velocity and the minimum energy consumption region differs for each gait. Xi et al. found similar result with simulated 4 DOF quadruped robot [2] and showed the necessity of using appropriate gaits for specific velocity range. According to these works, quadruped animals show walk gait in low velocity range, trot and pace gait in intermediate velocity range, bound and gallop gait in high velocity range.

III. METHOD

Our objective is to build a velocity command tracking controller that could show single or multiple gaits according to the command velocity. Previous approaches, which trained single policy in end-to-end manner were easily trapped to a single gait. We thought the reason for this was due to the distance between different gaits in joint angle domain, which is the output domain of the single policy. Different gaits correspond to different joint angle changes over time and they are often conflicting, which implies the large distance between different gaits in joint angle domain. Thus, the network could be easily trapped in a single gait region while not moving across different gait region. As mentioned in the previous work, using a single gait could result energy inefficient behavior and using multiple gaits appropriately

could result more optimal behavior. To let the controller take advantage of diverse gaits and use each of them for appropriate velocity range, we propose a novel hierarchical controller (Fig 1). In this work, the controller was designed for 8 DOF (Degrees of Freedom) quadruped robot which has four thigh joints and four calf joints. However, it could also be applied to 12 DOF quadruped robot which also includes extra four hip joints.

A. Framework

Proposed controller is composed of two parts, the high level controller and low level controller. The high level controller plans and selects appropriate gait according to the command velocity. The role is similar to "Central Pattern Generator"(CPG) which is a biological neural circuits that produce simple rhythmic outputs for rhythmic behaviors like walking. The controller is composed of single policy and due to the similarity of role, we will call it as the central pattern generator. We parameterized CPG signal with simple sinusoidal function (Eq 1) and designed the central pattern generator to output the five parameters of sinusoidal function, B for CPG signal period and C_i ($i = 0, 1, 2, 3$) for CPG signal phase where each corresponds to one leg, based on the command velocity (Eq 2). We set the CPG signal period of four legs the same to facilitate learning natural rhythmic behavior. In this work, C_i ($i = 0, 1, 2, 3$), which are the CPG signal phase parameters, were manually selected and learning them from scratch will be left for future work.

$$CPG_i(t) = \sin(Bt + C_i) \quad \forall i, i \in \{0, 1, 2, 3\} \quad (1)$$

$$\pi_{high}(v) = [B, C_0, C_1, C_2, C_3] \quad (2)$$

The low level controller enables forward walking or running by using the CPG signal while also considering feedback signal from the environment. The controller is composed of two parts, a single policy, which we will call as local feedback controller, and a PD controller. To directly connect CPG signal with gait pattern, we selected four thigh joints as CPG targets by assuming that they are more crucial for gait generation than calf joints. Then we parameterized the thigh joint angles using the CPG signal which will also result simple sinusoidal function with only difference in amplitude (Eq 4). The amplitude A roles as a domain transfer parameter from CPG signal domain to joint angle domain. We designed the local feedback controller to output five parameters, a domain transfer parameter A and desired joint angles of four calf joints, based on the input composed of robot configuration and CPG feature (Eq 3). Details of the input are written in Figure 1. Then eight desired joint angles are computed by transforming CPG signal to joint angle domain using the domain transfer parameter A (Eq 4, 5). The desired joint angles are passed to the PD controller to compute desired motor torques and are applied to the robot.

$$\pi_{low}(s) = [A, Calf_1, Calf_2, Calf_3, Calf_4] \quad (3)$$

$$Thigh_i(t) = A \times CPG_i(t) \quad \forall i, i \in \{0, 1, 2, 3\} \quad (4)$$

$$Calf_i(t) = \pi_{low}(s)[i + 1] \quad \forall i, i \in \{0, 1, 2, 3\} \quad (5)$$

The low level controller is called more frequently than high level controller. In this work, the frequencies of high and low level controller were set to 20Hz, 100 Hz, however it could be changed freely based on the computation power of on-board computer of the robot.

In online scenario, there could be an abrupt change of CPG signal period due to the change of command velocity. If CPG signal shows discontinuity or sudden slope change, it could result unnatural behavior or even severe damages in real robot. Thus, we added a simple CPG signal synchronizer before the local feedback controller which works as Algorithm 1. The effect of CPG signal synchronizer is shown in Figure 2.

The whole process of proposed hierarchical controller are summarized in Algorithm 2.

B. Training

The controller was trained end-to-end using hierarchical reinforcement learning. For each policy, PPO algorithm, the state of the art model free reinforcement learning algorithm, was used [3]. Similar cost terms with Hwangbo et al. and Lee et al. were used for realistic behavior with small changes [4], [5] (Table I). Each cost terms were weighted differently and integrated over time. The specific cost term equations and weighting parameters used for the experiment are summarized in APPENDIX (Eq 10-19, Table III). Cost scale k_c was gradually increased using $k_{c,j+1} \leftarrow (k_{c,j})^{k_d}$ $k_c, k_d \in (0, 1)$ update rule and this let k_c reach 1 gradually. Gradually increasing the cost scale avoids the robot from

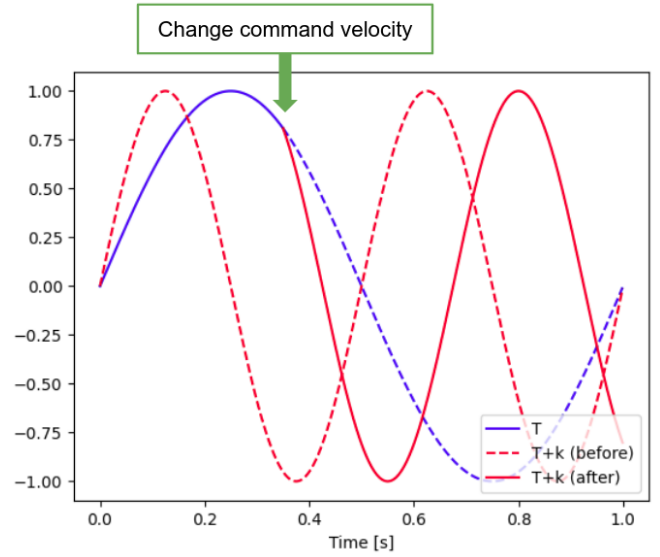


Fig. 2: Effect of CPG signal synchronizer (T: current time, k: period of the high level controller)

angular velocity of the base cost	c_1
linear velocity of the base cost	c_2
torque cost	c_3
joint speed cost	c_4
foot vertical velocity cost	c_5
foot clearance cost	c_6
foot slip cost	c_7
orientation cost	c_8
smoothness cost	c_9
leg phase cost	c_{10}
FINAL COST	$c_{final} = \sum_{i=1}^{10} w_i \cdot c_i$

TABLE I: Cost terms used for training our hierarchical controller

learning stopping behavior which is a suboptimal result. The update rule could be also thought of as part of curriculum learning, and had shown good result in previous works [4], [6].

IV. EXPERIMENT

We deigned the experiment to validate the necessity of multiple gaits and the efficiency of proposed hierarchical controller. Particularly we aim to answer following three questions.

- 1) Does there exist optimal gait for 8 DOF quadruped robot according to velocity?
- 2) Can the proposed hierarchical controller work as a multiple gait controller?
- 3) How does the multiple gait controller perform compared to a single gait controller?

A. Setup

We experimented our hierarchical controller in RAISIM simulator [7]. Laikago quadruped robot model from Unitree Robotics was used for training. Although laikago has 12

Algorithm 1 CPG signal synchronization

```
1: procedure SYNCHRONIZE( $B, C, period, step, \Delta t$ )
2:    $B_{old} \leftarrow B$ 
3:    $C_{old} \leftarrow C$ 
4:    $B_{new} \leftarrow 2\pi/period$ 
5:    $C_{new} \leftarrow (B_{old} - B_{new}) \times step \times \Delta t + C_{old}$ 
6:   return  $B_{new}, C_{new}$  ▷ synchronized CPG parameter
```

Algorithm 2 Hierarchical controller

```
1: Set  $\pi_{CPG}$  : Central pattern generator
2: Set  $\pi_{local}$  : Local feedback controller
3: Initialize  $B, C$ 
4: Set parameter  $T_{CPG}$  :  $\pi_{CPG}$  period
5: Set parameter  $\Delta t$  :  $\pi_{local}$  period
6:
7: while controller running do
8:   Get command velocity
9:   if  $(step \times dt) \equiv 0 \pmod{T_{CPG}}$  then
10:     $period \leftarrow \pi_{CPG}(v)$ 
11:     $B, C \leftarrow SYNCHRONIZE(B, C, period, step, \Delta t)$  ▷ CPG signal synchronization (Algorithm 1)
12:    Generate CPG signal with  $B, C$ 
13:   Get current observation
14:   Compute current CPG phase and gait encoding
15:    $state \leftarrow observation, period, CPG_{phase}, gait\ encoding$ 
16:    $A, Calf\ joint\ angle \leftarrow \pi_{local}(state)$ 
17:    $Thigh\ joint\ angle \leftarrow A \times CPG$  ▷ CPG signal transformation
18:   Set  $Thigh$  and  $Calf$  joint angle as target joint angle and compute motor torque with PD control
19:   Execute the computed torque
20:    $step \leftarrow step + 1$ 
```

DOF, we fixed four hip joints, resulting 8 DOF, and focused on learning just forward moving behavior. However, as the proposed controller has no limited constraint for number of joints, expanding the work by including four hip joints will be left for future work. For computation, single GPU was enough due to the light weight network model.

Hyperparameters and network architectures are written in APPENDIX Table III. Same hyperparameters and architectures were used for the whole experiments.

B. Comparison between different gaits

To check the existence of optimal gait according to the velocity, we learned different gaits separately using our controller. Then we defined optimality with velocity tracking error and energy consumption and measured the values for each gait. Trot, pace, and bound gaits were chosen to be learned because they are the three most typical gaits of quadruped animals. CPG signal phase parameter, C_i in Equation 1, 2, were manually fixed with values shown in Table II.

Every gaits were almost successfully learned as shown in Figure 3. Bound gait showed incomplete result due to contact with the environment which caused CPG signal phase shift of RR and RF thigh joints. For each learned gaits, we measured the mean velocity tracking error and energy consumption.

	phase
Trot	$[\pi, 0, 0, \pi]$
Pace	$[\pi, 0, \pi, 0]$
Bound	$[\pi, \pi, 0, 0]$

TABLE II: CPG signal phase for each gait. The order of leg phases is [FR, FL, RR, RL]

0.3, 0.6, 0.9, 1.2, 1.5 m/s command velocity were given as input, which are uniformly selected velocities in the trained velocity range. For the energy consumption, we divide the whole measured velocity range by 0.2 and clustered using 0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5 m/s as median velocity for each equalized range. Then the mean energy consumption for each range were computed. The results are shown in Figure 4. Pace and trot showed smaller energy usage in low velocity range while bigger in high velocity range compared to bound. Pace and trot also failed to learn high velocity tracking while bound succeeded. These results are similar to previous works with horse and simulated 4 DOF quadruped robot [1], [2]. From the result we could conclude that pace and trot are optimal for intermediate velocity range, while bounds are optimal for high velocity range.

C. Performance of multiple gait controller

Based on the performance of different gaits examined in previous experiment, we learned a multiple gait controller



Fig. 3: Single gait learned using our hierarchical controller

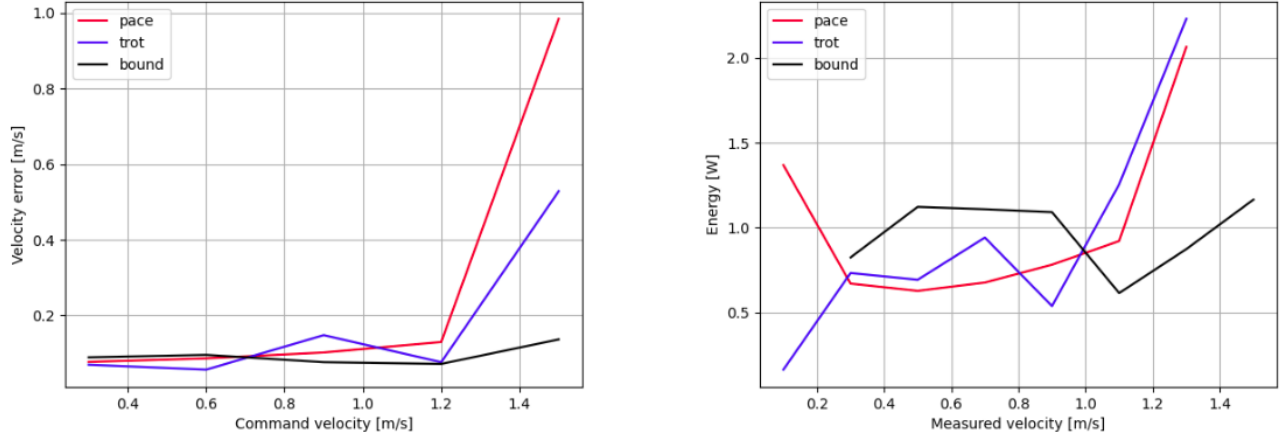


Fig. 4: Velocity tracking error (left) and Energy consumption (right) for each gait

using our framework by defining a function that determines CPG signal phase feature according to the velocity as Equation 6. Then we used it for the high level controller output as Equation 7. The purpose of the defined function was to generate trot, pace, bound in $[0, 0.5]$, $[0.5, 1]$, $[1, 1.5]$ velocity range.

$$f(v) = \begin{cases} [\pi, 0, 0, \pi] & 0 < v \leq 0.5 \\ [\pi, 0, \pi, 0] & 0.5 < v \leq 1 \\ [\pi, \pi, 0, 0] & 1 < v \leq 1.5 \end{cases} \quad (6)$$

$$\pi_{high}(v) = [B, f(v)] \quad (7)$$

The velocity tracking result and CPG signal period of the learned multiple gait controller are shown in Figure 5. By using appropriate gait for each velocity range, we nearly tracked the command velocity. Furthermore, the CPG signal period, which is the output of high level controller, decreased as the command velocity increased. From the result, we could realize that our proposed hierarchical controller can learn multiple gaits by taking the advantage of both high and low level controller each working as a central pattern generator and local feedback controller. Furthermore, each policies, composing the controller, converged to a reasonable outcome.

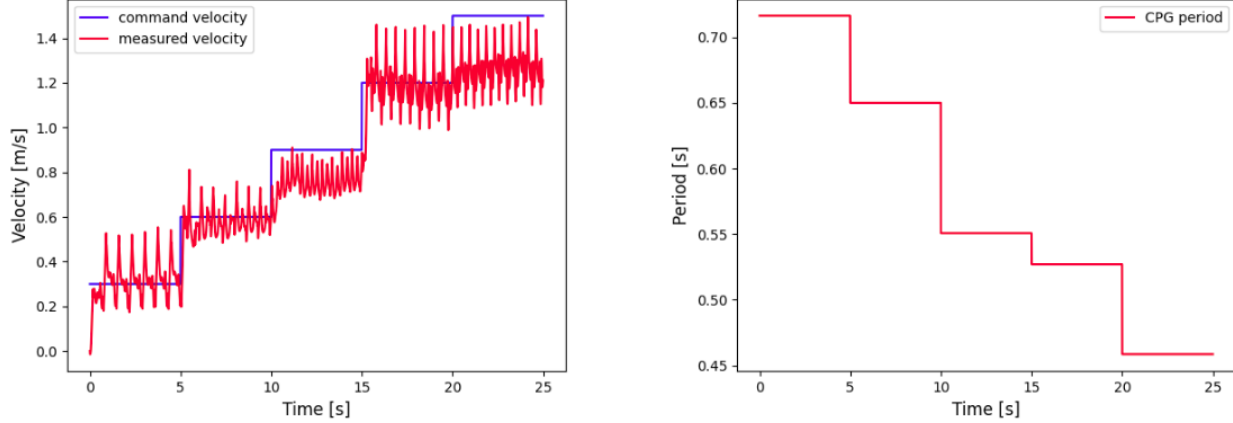


Fig. 5: Velocity tracking result (left) and CPG signal period (right) of multiple gait controller

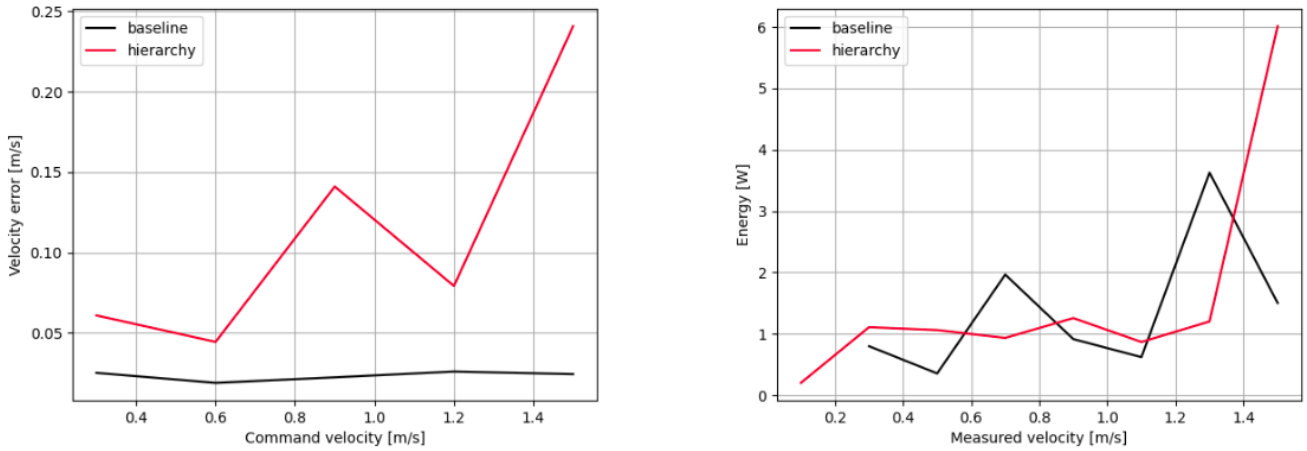


Fig. 6: Velocity tracking error (left) and Energy consumption (right) of multiple gait controller and baseline controller

We compared the performance of multiple gait controller with a simple framework composed of a single policy. The baseline was trained with PPO using the same reward signal shown in Table I. However, as we couldn't define swing and stance phase in the baseline method, we set the leg phase cost coefficient to zero. The cost coefficients for other costs were set to the same value as in Table III. The trained baseline method converged to a single gait as expected and showed unnatural behavior.

We compared the performance of multiple gait controller and the baseline by measuring velocity tracking error and energy consumption, same as previous experiment. The results are shown in Figure 6. Multiple gait controller showed energy efficiency in some velocity region compared to the baseline controller. However, the velocity tracking error of multiple gait controller was much higher compared to the baseline. This is due to abrupt change of CPG signal which corresponds to abrupt change of gait. Algorithm 1 could handle abrupt change of CPG period, but not the abrupt change of CPG phase which occurred due to gait transition. These discontinuity of CPG signal results degraded performance of tracking the command velocity. We will expand Algorithm 1

to handle difference in CPG phase for smooth gait transition as a future work.

V. CONCLUSIONS

We propose a novel hierarchical controller where high and low level controller collaborate to generate single or multiple gaits. The high level controller is composed of central pattern generator which plans and selects gaits. The low level controller is composed of local feedback controller and PD controller which enable walking or running by using the CPG signal and the environment feedback. In experiment, we showed the existence of optimal gait for 8 DOF quadruped robot according to the velocity. Furthermore, we empirically showed that our proposed hierarchical controller could learn multiple gaits that could be effective compared to single gait controller. However, the efficiency of multiple gait controller should be more thoroughly studied using various baselines including analytic model based controller. Also the performance of multiple gait controller should be improved by considering smooth gait transition and CPG signal phase shift problem. We will leave further improvements as future work.

APPENDIX

A. Cost terms

Below are the cost terms used for training the proposed hierarchical controller. To control command tracking error, which is the main objective of the controller, two different logistic kernels were used for angular velocity and linear velocity cost (Eq 8, 9).

$$K_{angular}(x) = -\frac{1}{e^{10x} + 2 + e^{-10x}} \quad (8)$$

$$K_{linear}(x) = -\frac{1}{e^x + 2 + e^{-x}} - \frac{1}{e^{10x} + 2 + e^{-10x}} \quad (9)$$

Notation

k_c	cost scale.
k_d	curriculum factor.
v_{AB}^C	linear velocity of B respect to A expressed in C
ω	angular velocity
$\hat{\cdot}$	desired quantity
τ	joint torque
ϕ	angular quantity
v_{ft}	tangential velocity of a foot (x, y components)
v_{fz}	vertical velocity of a foot (z components)
p_f	linear position of a foot
g_i	contact function of i_{th} foot (0: not in contact, 1: in contact)
G_i	Leg phase function of i_{th} foot (0: swing phase, 1: stance phase)
a_t	action at t step
$ \cdot $	cardinality of a set or l_1 norm
$\ \cdot\ $	l_2 norm

angular velocity of the base cost

$$c_1 = K_{angular}(k_c \|\omega_{IB}^I - \hat{\omega}_{IB}^I\|^2) \quad (10)$$

linear velocity of the base cost

$$c_2 = K_{linear}(|v_{IB}^I - \hat{v}_{IB}^I|) \quad (11)$$

torque cost

$$c_3 = k_c \|\tau\| \quad (12)$$

joint speed cost

$$c_4 = k_c |\dot{\phi}^i|^2 \quad \forall i \in \{1, 2, \dots, 8\} \quad (13)$$

foot vertical velocity cost

$$c_5 = k_c |v_{fz,i}|^2, \quad \forall i, i \in \{0, 1, 2, 3\} \quad (14)$$

foot clearance cost ($\hat{p}_{f,i,z} = 0.07$ m)

$$c_6 = k_c (\max(0, \hat{p}_{f,i,z} - p_{f,i,z}))^2 \|v_{ft,i}\|, \quad \forall i, g_i = 0, i \in \{0, 1, 2, 3\} \quad (15)$$

foot slip cost

$$c_7 = k_c \|v_{ft,i}\|, \quad \forall i, g_i = 1, i \in \{0, 1, 2, 3\} \quad (16)$$

orientation cost

$$c_8 = k_c \|[0, 0, 1]^T - \phi_g\| \quad (17)$$

smoothness cost

$$c_9 = k_c \|a_{t-1} - a_t\| \quad (18)$$

leg phase cost

$$c_{10} = \frac{1}{4} (g_i G_i + (1 - g_i)(1 - G_i)) \quad \forall i, i \in \{0, 1, 2, 3\} \quad (19)$$

B. Hyperparameters and Network architecture

Hyperparameter

v	Uniform(0.1, 1.5) [m/s]
w_1	120.
w_2	500.
w_3	0.5
w_4	0.02
w_5	1.0
w_6	1.5×10^4
w_7	200.
w_8	100.
w_9	0.5
w_{10}	300.
$k_{c,0}$	0.3
k_d	0.999

Network architecture

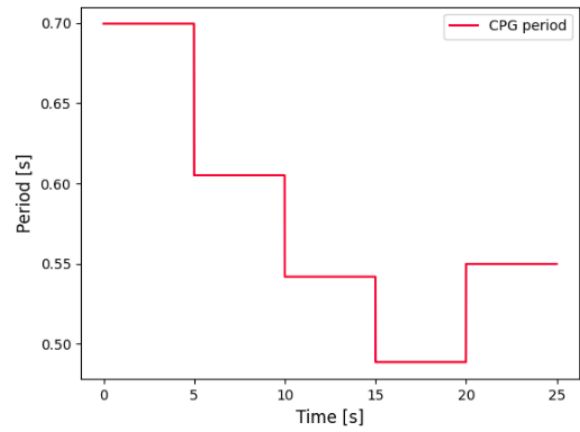
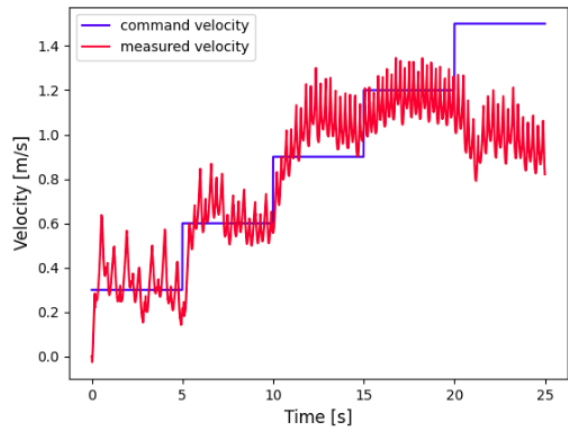
hidden units (high)	[128]
hidden units (low)	[128, 128]
activation (high)	LeakyReLU
activation (low)	LeakyReLU

TABLE III: Hyperparameters and network architecture

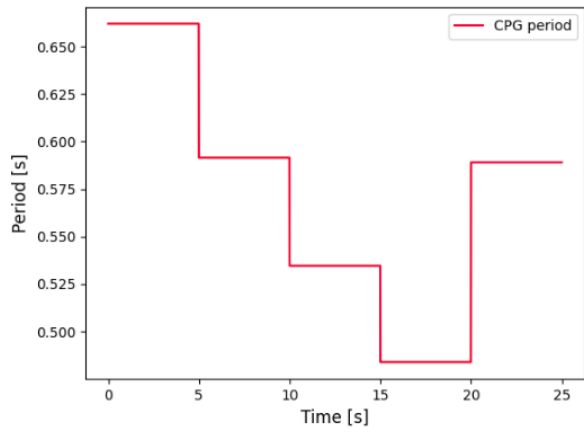
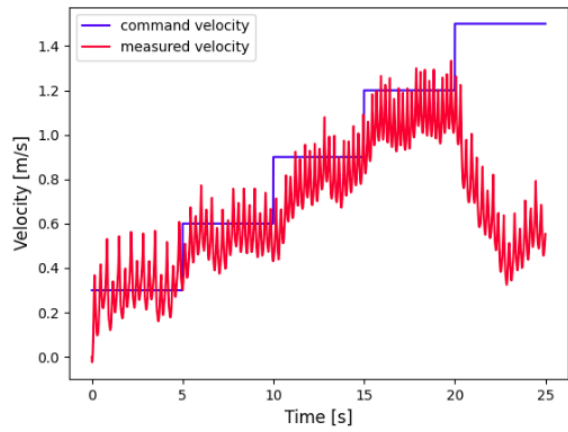
C. Additional plots for learned single gait



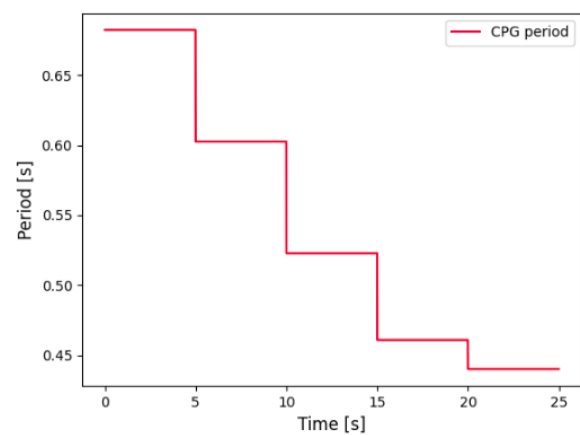
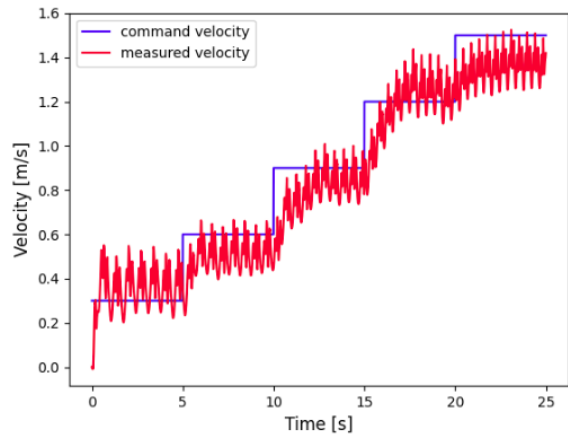
Fig. 7: Contact plot for each gait



(a) Trot



(b) Pace



(c) Bound

Fig. 8: Velocity tracking result (left) and CPG signal period (right) for each gait

REFERENCES

- [1] Hoyt, Donald F., and C. Richard Taylor. "Gait and the energetics of locomotion in horses." *Nature* 292.5820 (1981): 239-240.
- [2] Xi, Weitao, Yevgeniy Yesilevskiy, and C. David Remy. "Selecting gaits for economical locomotion of legged robots." *The International Journal of Robotics Research* 35.9 (2016): 1140-1154.
- [3] Schulman, John, et al. "Proximal policy optimization algorithms." arXiv preprint [arXiv:1707.06347](https://arxiv.org/abs/1707.06347) (2017).
- [4] Hwangbo, Jemin, et al. "Learning agile and dynamic motor skills for legged robots." *Science Robotics* 4.26 (2019).
- [5] Lee, Joonho, et al. "Learning quadrupedal locomotion over challenging terrain." *Science Robotics* 5.47 (2020).
- [6] Y. Bengio, J. Louradour, R. Collobert, J. Weston, Curriculum learning, *Proceedings of the 26th annual International Conference on Machine Learning*, 41–48 (ACM, 2009).
- [7] Hwangbo, Jemin, Joonho Lee, and Marco Hutter. "Per-contact iteration method for solving contact dynamics." *IEEE Robotics and Automation Letters* 3.2 (2018): 895-902.
- [8] Zhorniyak, L., Emami, M. (2020). Gait Optimization for Quadruped Rovers. *Robotica*, 38(7), 1263-1287. doi:10.1017/S0263574719001413
- [9] A. W. Winkler, C. D. Bellicoso, M. Hutter and J. Buchli, "Gait and Trajectory Optimization for Legged Systems Through Phase-Based End-Effector Parameterization," in *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560-1567, July 2018, doi: 10.1109/LRA.2018.2798285.
- [10] Owaki Dai, Kano Takeshi, Nagasawa Ko, Tero Atsushi and Ishiguro Akio 2013 Simple robot suggests physical interlimb communication is essential for quadruped walking. *J. R. Soc. Interface*. 102012066920120669 <http://doi.org/10.1098/rsif.2012.0669>
- [11] Owaki, D., Ishiguro, A. A Quadruped Robot Exhibiting Spontaneous Gait Transitions from Walking to Trotting to Galloping. *Sci Rep* 7, 277 (2017). <https://doi.org/10.1038/s41598-017-00348-9>
- [12] Fukuoka, Y., Habu, Y. Fukui, T. A simple rule for quadrupedal gait generation determined by leg loading feedback: a modeling study. *Sci Rep* 5, 8169 (2015). <https://doi.org/10.1038/srep08169>
- [13] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. Deepmimic: Example guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)*, 37(4):143, 2018.
- [14] Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Edward Lee, Jie Tan, Sergey Levine. Learning Agile Robotic Locomotion Skills by Imitating Animals. *Robotics: Science and Systems (RSS 2020)*
- [15] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, Angjoo Kanazawa. AMP: Adversarial Motion Priors for Stylized Physics-Based Character Control. *ACM Transactions on Graphics (Proc. SIGGRAPH 2021)*
- [16] Josh Merel, Yuval Tassa, Sriram Srinivasan, Jay Lemmon, Ziyu Wang, Greg Wayne, and Nicolas Heess. 2017. Learning human behaviors from motion capture by adversarial imitation. arXiv preprint [arXiv:1707.02201](https://arxiv.org/abs/1707.02201) (2017)
- [17] Z. Xie, G. Berseth, P. Clary, J. Hurst, and M. van de Panne, "Feedback control for cassie with deep reinforcement learning," *CoRR*, <http://arxiv.org/abs/1803.05580>
- [18] 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. *CoRR*, abs/1804.10332, 2018.
- [19] Erden MS and Leblebicioglu K (2008) Free gait generation with reinforcement learning for a six-legged robot. *Robotics and Autonomous Systems* 56(3): 199–212.
- [20] A. Singla et al., "Realizing Learned Quadruped Locomotion Behaviors through Kinematic Motion Primitives," 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 7434-7440, doi: 10.1109/ICRA.2019.8794179.
- [21] J. Siekmann, Y. Godse, A. Fern, and J. Hurst, "Sim-to-real learning of all common bipedal gaits via periodic reward composition," arXiv preprint [arXiv:2011.01387](https://arxiv.org/abs/2011.01387), 2020.
- [22] Ian Mason, Sebastian Starke, He Zhang, Hakan Bilen, and Taku Komura. 2018. Few-shot learning of homogeneous human locomotion styles. *Computer Graphics Forum* 37, 7, 143–153
- [23] Harrison Jesse Smith, Chen Cao, Michael Neff, and Yingying Wang. 2019. Efficient neural networks for real-time motion style transfer. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 2, 2, Article 13 (2019), 17 pages. <https://doi.org/10.1145/3340254>
- [24] Glen Berseth, Cheng Xie, Paul Cernek, and Michiel Van de Panne. 2018. Progressive reinforcement learning with distillation for multi-skilled motion control. In *ICLR '18*.
- [25] Yamamoto, H.; Kim, S.; Ishii, Y.; Ikemoto, Y. Generalization of movements in quadruped robot locomotion by learning specialized motion data. *ROBOMECH J.* 2020, 7, 29.
- [26] Goodfellow, Ian J., et al. "Generative adversarial networks." arXiv preprint [arXiv:1406.2661](https://arxiv.org/abs/1406.2661) (2014).
- [27] Peng, Xue Bin, et al. "Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning." *ACM Transactions on Graphics (TOG)* 36.4 (2017): 1-13.