Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

Learning a Faster Locomotion Gait for a Quadruped Robot with Model-Free Deep Reinforcement Learning

Biao Hu[♭], Shibo Shao[♭], Zhengcai Cao[♭], Qing Xiao[♭], Qunzhi Li[♮], Chao Ma[♮]

***Abstract*— Quadruped robots have great agility, flexibility and stability, which enables them to walk through uneven terrain. Motion control of legged robots is always a difficult problem. Previous approaches mostly either use a predefined gait that results in clumsy and unnatural behavior, or use reinforcement learning approach to generate a gait strategy that needs longtime computation and elegant network design. In this paper, we present an effective approach that uses deep reinforcement learning with prior knowledge to optimize the gait of quadruped robot. By using a specific quadruped robot walking gait as a priori knowledge, this approach adopts the technique of distributed proximal policy optimization to optimize the search for better gait. The proposed approach does not require modelling of complex robots, and has good network convergence speed and learning effect. Simulation results demonstrate that our proposed approach converges faster than other deep reinforcement learning methods without prior knowledge. Besides, our achieved gait has higher speed that is 50% faster than the trot gait without optimization.**

I. INTRODUCTION

Compared to wheeled robots, legged robots have better capabilities in adapting rough terrain, crossing obstacles, and keeping stability, which make them applicable for more varieties of applications, such as factory automation, construction service and exploration [1], [2]. A central problem in legged robots is how to coordinate their controlling, computing and leg motion because their kinematic and dynamic models are too complicated to design a proper controller. In general, legged robots can be categorized into monopod robots and multi-legged robots, where multi-legged robots are easier to keep balance than monopod robots. Multi-legged robots include the bipedal robot, quadruped robot, hexapod robot, eight-legged robot, where the quadruped robot is a representative type of robot that has a larger load capacity and a simpler structure. In this paper, we focus on the motion controlling problem of a quadruped robot.

Under the inspiration of the quadruped mammals' gaits, quadruped robot gaits are often designed to mimic quadruped mammals such that robots' locomotion is like mammals [3]. There are two typical approaches to design a bionic gait: drive function and foot trajectory. In [4], a circular-step and push-up gaits are designed to control 12-way steering gear of a 12 degrees of freedom robot. Simulation results show that the two types of gaits have low-stability. In [5], a trot gait is designed with Central Pattern Generators for a baby elephant-like quadruped robot to reproduce trot gaits, which enables the robot to change the walking frequency at runtime. The foot trajectory design is an important part of realizing bionic gaits. In [6], a trot pattern generated from the frequency modulated pattern generation is developed for controlling a hydraulic quadruped robot with heavy roads to quickly walk through the uneven terrain. In [7], a unified framework combining an active compliance control scheme with a trajectory generator is presented to produce effective trot-gait trajectories for a hydraulically actuated quadruped robot. A remarkable achievement is made on the MIT Cheetah, where a hierarchical controller using proprioceptive impedance control makes this robot to run with a stable and fast trot gait [8]. Although quadruped mammal-like bionic gaits are useful principles to design the controller of robot locomotion, they are often constrained into several specific trajectories that lack of generality in applying this approach to control robots with complex model and structure. Besides, as a controller is designed based on a certain type of terrain, the robot may not perform well on a terrain with varying slopes or frictions.

Recent years have witnessed the success of applying deep reinforcement learning (DRL) approaches to a variety of real-world robotic systems such as robot manipulator [9], autonomous driving [10], and legged locomotion [11]. Early studies on reinforcement learning do not consider using deep learning network, thus losing the generality. For instance, a form of policy gradient reinforcement learning is developed in [12] to optimize a quadrupedal trot gait to make the robot move the fastest. Because the reinforcement learning relies on the robot dynamic model, it is mostly used to optimize some parameters of a given trajectory pattern. The tremendous potential of deep learning network have motivated many researchers to explore using reinforcement learning together with deep learning network to generate a type of gait without prior gait configurations.

From simulation to reality, DRL has been demonstrated to be useful for the legged-robot locomotion in many cases. In [13], legged-robots are trained in simulations to walk through a rich terrain environment, where complex locomotion behaviors emerge on different types of legged-robots.

*This work was supported in part by the National Natural Science Foundation of China under Grant 91848103 and Grant 51975044, in part by the Talent Foundation of Beijing University of Chemical University under Grant XK1802-4.

[♭]Biao Hu and Zhengcai Cao are with College of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029, China (e-mail:hubiao@mail.buct.edu.cn, giftczc@163.com).

[♮]Qunzhi Li and Chao Ma are with Beijing Key Laboratory of Intelligent Space Robotic Systems Technology and Applications, Beijing Institute of Spacecraft System Engineering, Beijing, China (e-mail:liqunzhi78olive@139.com, machaodn@163.com).

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

However, deploying the learned behaviors from simulations to real-world robot is not an easy work because of model discrepancies between the simulated system and real robot system. Factors such as simplified dynamic model, inaccurate parameters cause this *reality gap*. Fortunately, several latest works have provided some ideas on how to bridge this gap. By improving the simulator and learning policies, trotting and galloping gaits learned from simulations are successfully deployed into a real-world robot in [14]. Another similar work is presented in [15] that realize several gaits including trot, gallop, walk and bound in a quadruped robot by using DRL. A remarkable progress is made on the ANYmal robot whose locomotion skills learned using DRL go beyond its previous best skills [11]. This quadruped robot can precisely and energy-efficiently follow high-level body velocity commands, run faster than before, and quickly recover from falling down.

Compared to model-based controller design approaches, a primary advantage of using DRL is that gait designers do not need to have a deep knowledge of robotics to generate an effective motion trajectory. However, DRL algorithms are also known as notoriously data inefficient, and they often require numerous attempts before learning to provide a strategy such as playing an Atari game [16]. Towards this problem, in this paper, we propose an approach that learns a locomotion behavior based on the priori knowledge of a quadruped robot Doggo. By using the same control strategy of the trot gait as the prior knowledge, controller parameters are automatically learned with DRL. The proposed approach only needs to monitor the robot internal state, without building its complex kinematic model. Experimental results demonstrate that the robot locomotion with our learned policies can move 1.5 faster than the non-optimized trot gait. Besides, compared to DRL without using prior knowledge, our algorithm converges also faster.

This paper is organized as follows: Second II introduces the Extended Learning Proximal Policy Optimization that we use. Section III introduces the foot structure and trot gait of the complex quadruped robot Doggo. Section IV proposes a deep reinforcement learning method based on prior knowledge to optimize the gait and speed of the Doggo robot. Section V presents our experimental results and Section VI concludes this paper.

II. DISTRIBUTED PROXIMAL POLICY OPTIMIZATION

In this section, we explore the Distributed Proximal Policy Optimization(DPPO) method for quadruped walking in simulations. DPPO is an end-to-end policy gradient learning algorithm that directly outputs actions instead of action values, which is very suitable for continuous control of complex models. DPPO is a distributed frameworks with high degree of parallelism, that can simulate in many learning environments at the same time to speed up the convergence.

A. Definition

The motion problem of a quadruped robot locomotion can be described as a Markov decision process(MDP), the current decision is only related to the current state, and the state at the next moment is determined only by the current state and the current decision. Therefore, a decision can be made only relying on the current state. A MDP is represented by $\{S, A, P, R, \gamma\}$, where S is the state space of the robot that collects all possible states of the robot; A represents the action space, which is a collection of all executable actions of the robot; $P : S*A*S \rightarrow [0, 1]$ is the transition probability function that models the evolution of states based on actions; $R : S*A \rightarrow R$ represents the reward given after a behavior and state; γ represents the forgetting factor in the range $[0, 1]$.

The goal of reinforcement learning is to implement a strategy π, expressed as $\pi : S*A \rightarrow [0, 1]$, a probability distribution acting on the state. The optimal strategy is a series of actions that maximize the cumulative return of the quadruped robot. The optimal strategy is represented as π_θ:

$$\pi_\theta = \arg\max P_{\pi_\theta}[R(s_t, a_t)], \tag{1}$$

where P_{π_θ} represents the state trajectory $\{s_0, s_1, ...s_T\}$ obtained when the policy π_θ is used. Strategy π_θ wants to get the maximum value of all returns and trajectories. The network structure is presented as the Figure 1.

DPPO has been proved to be very successful for obtaining optimal continuous action value. It is an on-policy algorithm based on the traditional Actor-Critic framework. The Actor is called the policy network and is responsible for calculating actions based on the current state. Critic is called the evaluation network and is responsible for evaluating the value of the current state. Each of them is designed by a neural network. We use $\hat{A}$ to represent the Advantage function, which represents how better the action we use than the others we give up, its given by $\hat{A} = r(s_t, a_t) + \gamma V(s_{s+1} - V(s))$.

DPPO collects up to $N = 1000$ episodes in the environment using the current policy. DPPO collects experience consisting of $\{s_t, a_t, s_{t+1}, r_t\}$, in each iteration DPPO selects collected 64 sets of data for stochastic gradient descent. DPPO optimizes network parameter using gradients and rewards

$$L(\theta) = \mathbb{E}_{\pi_{old}}[\frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)}\hat{A}], \tag{2}$$

In order to solve the problem that the learning rate cannot be determined in reinforcement learning, DPPO uses the following formula to optimize the parameters of the network

$$L^{CLIP}(\theta) = \mathbb{E}_t[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)], \tag{3}$$

where ϵ is a hyper-parameter, use the clip method to limit $r_t(\theta)$ to $1-\epsilon$,$1+\epsilon$. We take the minimum of the clipped and unclipped objective. With this scheme, we only ignore the change of probability ratio when it would make the objective better,and we use it when it makes the objective worse.

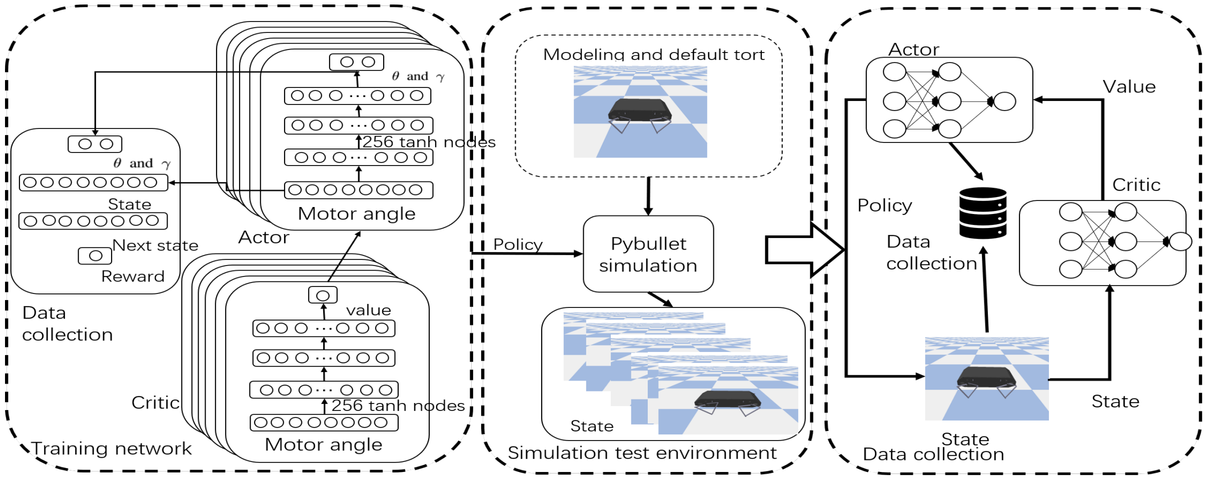


Fig. 1: Structure of the network

B. Optimization

The quadruped robot has a lot of data during training and the network structure is complicated. In order to speed up DPPO convergence, in each iteration of training, we regularize the Doggo robot data in the environment and then input it into the network for training. With the $\tanh$ activation function, the data can be kept in the most sensitive part of the activation function, which is beneficial to prevent the gradient from disappearing. Data regularization can improve the convergence speed of the network.

At the same time, we modify the reward function. When the quadruped robot has an accident instead of completing the specified forward mission and stops training. For example, a rollover occurs; the quadruped robot is too low from the ground; or the leg structure is stuck and cannot be stepped. The sum of the obtained bonus values is multiplied by a weight $\gamma \in [0, 1]$. But when the robot finishes the specified advance task and stops training, the weight is not multiplied. In this way, we make the reward for normal forward movement higher than the reward when the accident occurs. During the training, the quadruped robot adopts a strategy with higher reward value to reduce the probability of accident.

Fig. 2: The physical picture of the Doggo quadruped robot

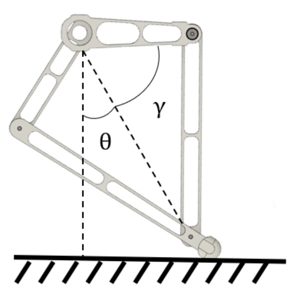


Fig. 3: The foot structure of the Doggo quadruped robot

III. THE LEG STRUCTURE OF DOGGO

The Doggo robot is an open source, complex quadruped robot from Stanford. Doggo uses a closed-chain leg structure with two degrees of freedom for each leg structure [17]. As shown in Figure 2. The motion control of Doggo robot mainly uses the default gait such as walk, trot, dance, backflip, *etc*. The Doggo robot is a complex quadruped robot, its dynamic structure is very complicated, and it is thus very difficult to directly model. Therefore, we do not directly model it in kinematics, but analyze its leg structure and the trot gait as a prior knowledge.

The leg model of the Doggo quadruped robot is shown in the Figure 3. Each leg is composed of two links, which are connected by a rotating shaft, and the two links form a closed chain structure. Each leg structure has two degrees of freedom, two motors control θ and γ of the leg respectively. θ and γ are two key angles to control the gait of the Doggo robot, where θ controls the height at which each leg structure is lifted and γ controls the length of the leg structure moving forward each time. θ and γ work together for the gait of the Doggo robot.

The trot gait is a bionic gait that mimics the scene in which the quadruped is running on a plane, it has high motion stability. In the default trot gait of the Doggo robot, the Doggo robot lifts and moves one of the leg structures each time and then lifts the foot structure on its diagonal. The Doggo robot uses the same θ and γ each time, cycles through this sequence and uses the fixed parameters allowing the robot to run forward. The Doggo robot using the trot gait does not perceive the environment, the gait is determined only by the specified θ and γ. Before the gait is executed, each of the following actions is determined. The trot gait does not change according to the change of the environment. Nor does it have a high speed and it does not fully meet our needs.

We use Doggo's open source trot gait as a priori knowledge of reinforcement learning, mainly because the structure of the trot gait is simple, we only need to control two parameters at a time to control the leg movement. It can also greatly simplify the network structure and speed up the network convergence. At the same time, because the trot is bionic gait, the gait is fixed when the foot movement order is fixed, θ and γ are limited to a certain range. We can use these prior knowledge to reduce the amount of manual design gait. The trot gait has high stability and adapts to various terrains, so the learned gait is highly portable and can adapt to different environments. Therefore, we use the method of reinforcement learning, based on the trot gait, and determine θ and γ according to the rotation angle of the current motor, which has achieved the goal of adapting to the environment and increasing the speed.

IV. REINFORCEMENT LEARNING FOR FASTER QUADRUPEDAL ROBOT

In this section, we mainly describe the methods we use and how to make the Doggo quadruped robot run faster.

A. Priori Knowledge

From the trot gait we know that by controlling θ and γ of the Doggo robot's leg, the gait of the Doggo robot can be controlled. θ and γ used for each step are determined by the rotation angle of all current motors. Therefore, in our proposed method, using the current rotation angle of all eight motors as the input of the neural network, a neural network with three layers of fully connected layers is established. The output of the neural network is θ and γ of the leg structure. We adjust the motion strategy of the robot by adjusting θ and γ to optimize the gait of the quadruped robot. The neural network uses the tanh function as a nonlinear activation function. After the output layer of the neural network, an activation layer of the tanh function is added to limit θ and γ to a certain range. The formulas for θ and γ we use during training are as follows:

$$\theta = \sin(t) * (N_0 * 0.5 + \theta_0 * 0.5)$$
$$\gamma = \sin(t + \pi) * (N_1 * 0.5 + \gamma_0 * 0.5), \tag{4}$$

where N_0 and N_1 represent the output of the neural network,and θ_0 and γ_0 represent the parameters of the default gait. We add the output of the neural network and the parameters of the default trot gait as the parameters of θ and γ , while we use the sin function to ensure the periodicity of the motion of the leg structure. This preserves the trot gait as a priori knowledge and uses neural networks to optimize the gait.

In order to maintain the stability of Doggo robot better, we hope that θ and γ should not be too large, so as to avoid the situation where Doggo robot rolls over or falls. Using neural networks can learn the strategies that should be implemented in this case, but learning this situation requires a lot of computation and time, so we artificially specify the maximum values of θ and γ, as follows:

if $N_0 * 0.5 + \theta_0 * 0.5 > 1$,

$$N_0 * 0.5 + \theta_0 * 0.5 = 1$$

if $N_1 * 0.5 + \gamma_1 * 0.5 > 1$,

$$N_1 * 0.5 + \gamma_1 * 0.5 = 1 \tag{5}$$

by this method, the maximum value of θ and γ are manually limited to increase the gait stability and accelerate the network convergence speed.

B. The Deep Reinforcement Learning Network

In the process of reinforcement learning, we hope that Doggo robot will move along a straight line quickly, so we define the reward function of reinforcement learning as follows:

$$R = \lambda_0 * (x_1 - x_0) - \lambda_1 * (y_1 - y_0) - \lambda_2 * (z_1 - z_0) - \lambda_3 * E$$
$$E = \sum_{i=1}^{n} T_i * V_i, \tag{6}$$

λ_0 represents the weight parameter of the reward robot forward, λ_1,λ_2, and λ_3 are used as the penalty parameters to punish the robot to yaw left and right, sway up and down, and consume energy, E represents the energy spent by the quadruped robot. T_i represents the torque of the current i-th motor output, and V_i represents the speed of the i-th motor at this time. x, y, z are the coordinates of the Cartesian coordinate system of the quadruped robot in the environment. Here we mainly consider the position of the centroid of the quadruped robot. x_1, y_1, z_1 are the current position, and x_0, y_0, z_0 are the coordinate value of the previous moment. The return value is mainly composed of four parts, the first part rewards the robot's forward behavior, the second part and the third part respectively punish the left and right yaw and the upper and lower shaking of the robot, and the fourth part limits the energy every time. The energy is mainly obtained by multiplying the torque of all the motors and the speed of motors. This part mainly prevents the robot from consuming too much energy each time, which is inconsistent with the real physical world. This reward function rewards the robot's rapid forward linear motion, while punishing the robot's yaw or sway, as well as limiting energy to prevent it from conforming to the real physical world.

We mainly observe the gait movement and speed changes of the Doggo robot, getting better experimental results by adjusting weight parameters and hyper-parameters. To illustrate the advantage we made for quadruped robot with the trot gait as a prior knowledge, we conducted a set of controlled experiments using the same network structure and hyper-parameters, and using the current rotation angle of

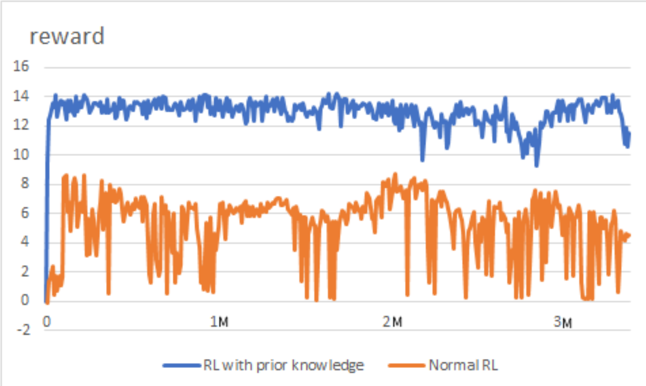


Fig. 5: The reward curve obtained when using reinforcement learning with prior knowledge on Doggo and the reward curve when using general reinforcement learning method. We stop training when the parameters are stable.The ordinate represents the return value and the abscissa represents the number of iterations.

each motor as the output of the neural network. The only difference is that the output of the neural network is no longer θ and γ of the robot's leg structure, but the angle of rotation of each eight motor at the next moment. In the same environment, the quadruped robot Doggo learns walking without prior knowledge and we collects the gait status and speed changes of the Doggo robot.

V. RESULT

In this section we collect reinforcement learning gaits with prior knowledge, reinforcement learning gaits without prior knowledge and Doggo robot default trot gait data. We compare of their motion gait, training speed and speed changes the situation has a quantitative demonstration of the approach we have proposed. At the same time, we will show the environmental adaptability of the gait we have learned.

A. The gait of Doggo quadruped

First we compare the gaits, as shown in Figure 4. It can be clearly seen that the gait of the Doggo robot obtained by the reinforcement learning using the prior knowledge is much better than that obtained by the reinforcement learning without using the prior knowledge. The gait obtained by using the prior knowledge has smaller yaw and sway, while avoiding collisions between the Doggo robot body or leg structure and the ground. In the comparison the default trot gait, the reinforcement learning gait using prior knowledge has a significant improvement in speed.

B. The training speed of reinforcement learning

Next, we compare the training speed of the reinforcement learning gait with prior knowledge and the reinforcement learning gait without prior knowledge in the learning process, as shown in the Figure 5. In the same training environment, reinforcement learning using prior knowledge has a good reward return value at the beginning, and reaches the maximum return value faster in the learning process than the reinforcement learning without prior knowledge. The maximum return value obtained is also twice more than the return value of reinforcement learning without prior knowledge .This shows that the reinforcement learning with prior knowledge uses a shorter time to search better reward than the reinforcement learning without prior knowledge.

C. The speed of Doggo

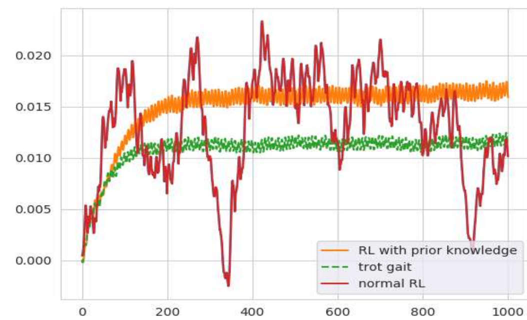


Fig. 6: Speed of three different gaits.The ordinate represents the speed and the abscissa represents the number of simulation steps.

Finally, we quantify the changes in the movement speed of the three, we define the speed as the distance the Doggo moves each simulation action. The formula is

$$V = x_1 - x_0, \tag{7}$$

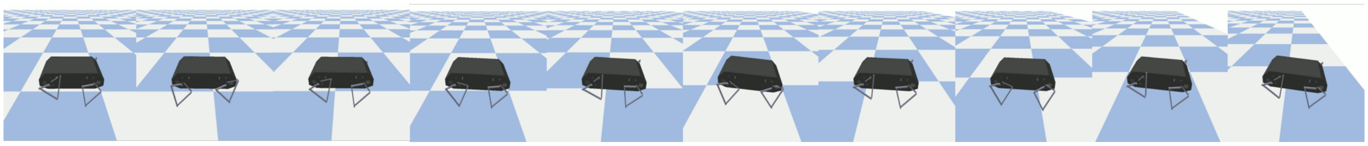
where x_1 is the position of the Doggo robot after the action is performed,x_0 is the position before the action is performed, which is also part of the reward function. As shown in the Figure 6. We can be clearly seen that the average speed of gait obtained by reinforcement learning using prior knowledge is higher than that of other gaits. It is higher than the default trot gait by about 50%. The gait obtained without prior knowledge is highly volatile in speed and more unstable than the other two gaits.

D. The environmental adaptability of the gait

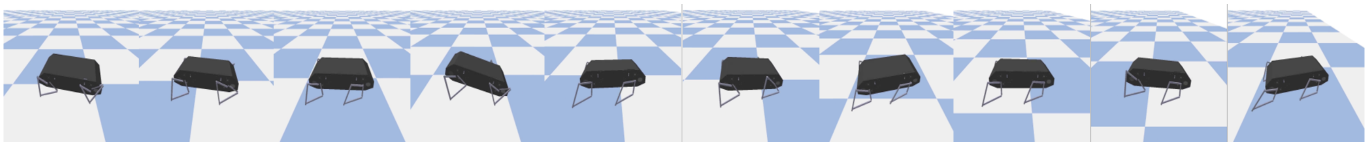
The gait we have learned also has environmental adaptability, as shown in the Figure 7. The former quadruped robot moves on the flat ground, and the latter quadruped robot moves on the 20 degree uphill. The robot completed two movements and maintained the stability of the speed.

VI. CONCLUSION

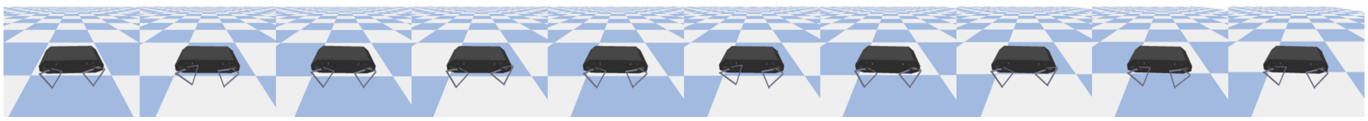
The quadrupedal robot locomotion is an important and difficult problem in legged-robot systems. In this paper, we have presented an approach that is able to effectively use prior knowledge with deep reinforcement learning to generate a good trot gait. With a specific quadruped robot gait as a priori knowledge, parameters for obtaining a better gait are searched with high efficiency by using distributed proximal policy optimization approach. Because this approach



(a) reinforcement learning with prior knowledge



(b) normal reinforcement learning



(c) default trot gait

Fig. 4: The gait of Doggo quadruped robot obtained by three methods. (a) the gait obtained by using reinforcement learning with prior knowledge; (b) the gait obtained by using normal reinforcement learning (b) and (c) the default trot gait of Doggo robot.

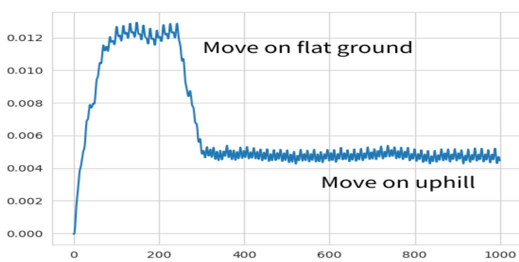


Fig. 7: The speed of moving on flat ground and uphill

does not rely on an accurate robot model, it is very useful for robots that are hard to model. Simulation results also demonstrate that the proposed approach converges faster, and the generated gait moves 0.5 times faster than a specific trot without optimization. Last, since the proposed approach only needs a prior knowledge of a gait, the proposed approach also fits for some other legged robots, such as hexapod robot or eight-legged robot.

REFERENCES

[1] S. Guo, M. Li, L. Shi, S. Mao, and C. Yue, "Performance evaluation on land of an amphibious spherical mother robot," in *IEEE International Conference on Mechatronics & Automation*, 2013.

[2] Y. Shen, G. Zhang, T. Yang, and S. Ma, "Development of a wheel-paddle integrated quadruped robot for rough terrain and its verification on hybrid mode," *IEEE Robotics & Automation Letters*, vol. PP, no. 99, pp. 1–1, 2018.

[3] R. Alexander, "The gaits of bipedal and quadrupedal animals," *International Journal of Robotics Research*, vol. 3, no. 2, pp. 49–59, 1984.

[4] L. Wang, X. Kong, S. Wu, X. Wang, Q. Wang, and Q. Hu, "Gait design and simulated analysis of quadruped robot," *Rev. Tec. Ing. Univ. Zulia*, vol. 39, no. 5, pp. 101–110, 2016.

[5] J. Zhang, G. Feng, X. Han, X. Chen, and X. Han, "Trot gait design and cpg method for a quadruped robot," *Journal of Bionic Engineering*, vol. 11, no. 1, pp. 18–25, 2014.

[6] H. K. Kim, D. Won, O. Kwon, T. J. Kim, S. S. Kim, and S. Park, "Foot trajectory generation of hydraulic quadruped robots on uneven terrain," *Ifac Proceedings Volumes*, vol. 41, no. 2, pp. 3021–3026, 2008.

[7] B. Ugurlu, I. Havoutis, C. Semini, and D. G. Caldwell, "Dynamic trot-walking with the hydraulic quadruped robot hyq: Analytical trajectory generation and active compliance control," in *IEEE/RSJ International Conference on Intelligent Robots & Systems*, 2013.

[8] D. J. Hyun, S. Seok, J. Lee, and S. Kim, "High speed trot-running: Implementation of a hierarchical controller using proprioceptive impedance control on the mit cheetah," *International Journal of Robotics Research*, vol. 33, no. 11, pp. 1417–1445, 2014.

[9] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation," in *https://arxiv.org/abs/1610.00633*, 2016.

[10] J. Chen, B. Yuan, and M. Tomizuka, "Model-free deep reinforcement learning for urban autonomous driving," in *https://arxiv.org/abs/1904.09503*, 2019.

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

[12] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *IEEE International Conference on Robotics & Automation*, 2004.

[13] N. Heess, T. B. Dhruva, S. Sriram, J. Lemmon, and D. Silver, "Emergence of locomotion behaviours in rich environments," in *https://arxiv.org/abs/1707.02286*, 2017.

[14] 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 *https://arxiv.org/abs/1804.10332*, 2018.

[15] A. Singla, S. Bhattacharya, D. Dholakiya, S. Bhatnagar, and S. Kolathaya, "Realizing learned quadruped locomotion behaviors through kinematic motion primitives," in *https://arxiv.org/abs/1810.03842*, 2018.

[16] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *Computer Science*, 2013.

[17] N. Kau, A. Schultz, N. Ferrante, and P. Slade, "Stanford doggo: An open-source, quasi-direct-drive quadruped," in *International Conference on Robotics & Automation*, 2019.