World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:15, No:9, 2021

Gaits Stability Analysis for a Pneumatic Quadruped Robot Using Reinforcement Learning

Soofiyan Atar, Adil Shaikh, Sahil Rajpurkar, Pragnesh Bhalala, Aniket Desai, Irfan Siddavatam

Open Science Index, Industrial and Manufacturing Engineering Vol:15, No:9, 2021 publications.waset.org/10012225.pdf

Abstract—Deep reinforcement learning (deep RL) algorithms leverage the symbolic power of complex controllers by automating it by mapping sensory inputs to low-level actions. Deep RL eliminates the complex robot dynamics with minimal engineering. Deep RL provides high-risk involvement by directly implementing it in real-world scenarios and also high sensitivity towards hyperparameters. Tuning of hyperparameters on a pneumatic quadruped robot becomes very expensive through trial-and-error learning. This paper presents an automated learning control for a pneumatic quadruped robot using sample efficient Deep Q learning, enabling minimal tuning and very few trials to learn the neural network. Long training hours may degrade the pneumatic cylinder due to jerk actions originated through stochastic weights. We applied this method to the pneumatic quadruped robot, which resulted in a hopping gait. In our process, we eliminated the use of a simulator and acquired a stable gait. This approach evolves so that the resultant gait matures more sturdy towards any stochastic changes in the environment. We further show that our algorithm performed very well as compared to programmed gait using robot dynamics.

Keywords—Model-based reinforcement learning, gait stability, supervised learning, pneumatic quadruped.

I. Introduction

THE mobile robots are usually associated with robots that consist of wheels to move around. There is a different category of mobile robots that possess the ability to overcome many terrains with ease. These are legged robots. These robots can interact more effectively with the environment. Quadruped robots possess the characteristic properties that give them an edge over another similar system. Designing locomotion controllers for pneumatic quadruped robots [1], [2] is a critical and long-lasting research task. Previous works [3], [4] for model-free reinforcement learning (RL) [5], [6] have shown that without prior knowledge of the dynamics of the system, still, the algorithm was able to optimize the target policy of the architecture. Recently, research on the automation of quadruped robot locomotion has heightened due to the more sober approach [5], [7]–[10]. State of the art approach for pneumatic quadruped robots depends on the stacks of state estimation, contact scheduling, control system, and dynamics of pneumatic cylinders. Implementing these pipelines leads to complex design, which incorporates complex dynamics model of the robot and complex controllers [11]. This model is challenging to acquire for real hardware. Many implementations of model-free methods are executed through simulated environments that do not lead to achievable solutions for real-world scenarios, considering training hours [4], [12]. Our main challenges are threefold. We were first decreasing the complexity of the quadruped robot resulting in more stability and more efficiency. The next challenge was to implement an accurate model-free algorithm that should result in long-distance locomotion. Our algorithm prepared an approach that used pre-trained weights using a supervised Neural network with basic gaits. Without prior knowledge of the robot's dynamics, the control algorithm can be implemented directly using deep RL. The pre-trained weights can be obtained using standard gait analysis, which can be opted from a simple walking gait or, in our case, an unstable crawling gait. Applying deep RL algorithms needs high samples [5] for learning the gait of each robot, which results in efficient locomotion. However, a large number of samples results in long hours of training. Thus, efficient locomotion with fewer samples was a challenge in our case. Moreover, these deep RL algorithms are often susceptible to hyperparameters and need multiple tuning [13]. Consequently, our approach does not require any hyperparameters tuning, which includes learning rate and discount factor. Thus, our policy makes the system more practical to apply deep RL directly to the real-world scenario. Our algorithm consists of simple neural networks which do not need too much computation processing, and limiting the data for learning results in low computational processing. Fig. 2 illustrates the end gait after training with our algorithm in the real world. This policy was developed for a flat surface, but the robust learning could tackle obstacles that resulted in vigorous gait. Our approach eliminates many gaits that the policy tries to learn while training. Thus, limiting it to the choice of gait we want, the procedure becomes less complicated and robust for real-world scenarios. In a simulation-based approach [5], [12], [14], [15], the model must also adapt to any stochastic noise seen in real-world scenes. Our training required about 400 episodes, equating to about two hours for real-world application. Our paper focuses on a framework that includes a deep Q learning algorithm with an asynchronous learning system, leading to data efficiency and sample efficiency for locomotion on a pneumatic quadruped robot, with a scarce computational expensive neural network. We demonstrate the framework by training a pneumatic quadruped robot for hopping gait. This algorithm also uses an asynchronous learning system, described in Fig. 5, enabling the deep RL algorithm to be independent of machine-related factors. Pneumatic quadruped is generally very unstable while traversing [16]. This algorithm

Dr. Irfan Siddavatam is with the Department Information Technology, K.J. Somaiya College of Engineering, Mumbai, India 400071 (e-mail: irfansiddavatam@somaiya.edu).

Soofiyan Atar, Adil Shaikh, Pragnesh Bhalala, Sahil Rajpurkar and Aniket Desai are with K.J. Somaiya College of Engineering.

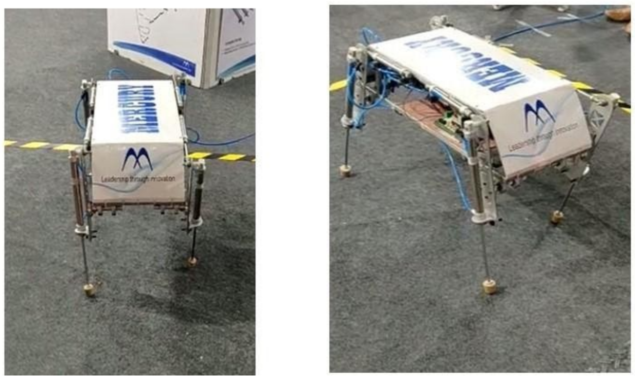
Fig. 1 Pneumatic quadruped robot

molded the robot to automatically learn the policy for stable locomotion even for unseen terrains.

II. Related Work

The current state-of-the-art quadruped robot typically adopts a pipe-lined complex controller. For example, the MIT Cheetah uses a state machine over contact conditions, generates simple reference trajectories, performs model predictive control [12] for optimizing desired contact forces for individual leg, and then uses dynamics and controller. The ANYmal robot [15] plans footholds based on the inverted pendulum model even our robot is based on the concept of invert pendulum, applies CMA-ES [17], [18] to optimize a parameterized controller, and solves a hierarchical operational space control problem to efficiently calculate individual joint torques, contact forces with the ground, and motion of whole quadruped body. With these methods, practical and robust gaits are possible, but they require significant knowledge of control, dynamics, kinematics for the locomotion task along with full-body control. However, our method aims to control the robot without substantial knowledge of the dynamics, kinematics, and control but just basic knowledge of gaits. There is no requirement for prior knowledge of trajectory planners or 3D modeling of the robot in our approach. For the implementation of initial gait using supervised learning, the requirement lies for supervised learning, which includes the dimensions of the quadruped robot and bounds for each actuator that is useful for state and action space. While in practice, knowledge of dynamics and control may accelerate the learning process [19] through which prior assumptions of gait will be broadly applicable with more diverse conditions. Implementing deep RL algorithm to adapt legged locomotion in simulation has been accomplished [5], [12], [15] along with sim-to-real transfer learning [10], [12]. Sim-to-real transfer learning needs a tuning curve which creates discrepancies between real-world environment and simulation environment. Using deep RL algorithms directly on real-world applications have been one of the main challenges in reinforcement learning. Moreover, our framework includes direct implementation of a deep RL algorithm with supervised learning on real hardware.

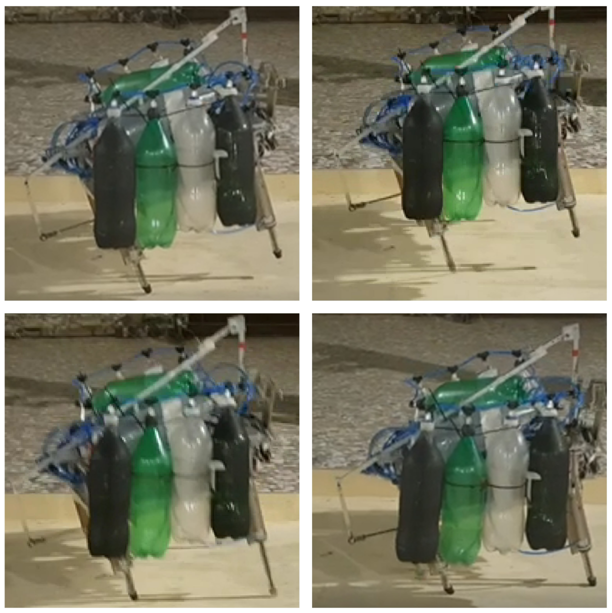
Fig. 2 Final trajectory of pneumatic quadruped robot

III. Mechanical Design

One of the main factors kept in mind while designing the legged robot is the weight of the robot and the availability of actuators capable of providing enough force/torque. The chassis made up of Aluminum in collaboration with ABS has an excellent weight to strength/stiffness ratio compared to other composites such as acrylic or poly-carbonate at the given speed and load. Each leg module has two pneumatic cylinders, a damper pad at the leg's foot that provides sufficient traction. Each leg has two active degrees of freedom provided by the two cylinders. The piston's nature helps attain the precise position required with the desired force using position and pressure sensors. The larger cylinder provides the robot with the necessary force to counterbalance the expected reaction and some excess force to lift the body off the ground. The lateral oscillatory motion of the larger piston is provided by the smaller cylinders that help the robot move forward. For this motion, the smaller cylinder is hinged on both sides. A smaller cylinder's rod is connected to the larger cylinder with the help of an eyeball joint, which is a form of a spherical joint. The hinges allow the smaller cylinder to compensate for the rotary movement of the larger piston freely. The air inside the cylinder acts like a virtual spring and provides compliance to the leg module. Fig 3 demonstrates the complete mechanical design used for this paper.

IV. Control Theory

A. Kinematics

For control strategy analysis of the quadruped robot, pertaining to gait development, it was necessary to track the contact point of the leg in the Cartesian plane concerning the body. We track the position of the point of contact/end-effector of the robot and find the jacobian of the following for finding

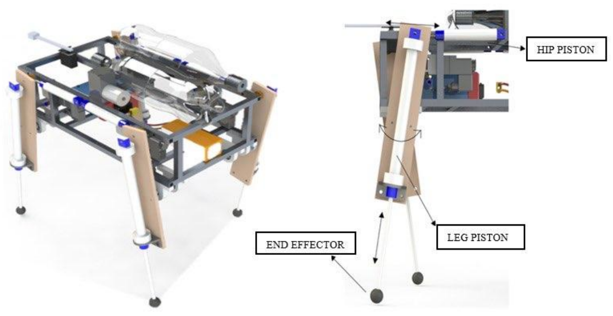

Fig. 3 Mechanical design overview

TABLE I
NAMING CONVENTION FOR SYMBOLS

Symbols	Description
Pressure in Pascal	P_{pa}
Stroke length	L_{ext}
Maximum stroke length	L_{max}
Area of piston	A
Joint twist	S_n
Joint linear/angular displacement	θ_n
Mass	m
Spring constant	K

the end-effector velocity/force.

$$T(\theta) = e^{[S_n]\theta_n}$$

B. Dynamics

The dynamic approach for the pneumatic quadruped robot is described in equation 2. Although these apply separately to the hip and the leg piston [20], the underlying concept is the same, so we mention the following for the leg piston concerning the body. Dynamics is applied to control the force applied by each pneumatic piston [21], [22].equation Lagrangian L to find the equations of motion with constraint system:

$$L = \frac{m}{2} + (\dot{L_{ext}}^2 + L_{ext}^2 * \dot{\theta}^2) - mgL_{ext}Scos\theta - \frac{K}{2} * L_{ext}^2 \quad (1)$$

The above equation is the basic Lagrange equation for dynamics on a pneumatic quadruped robot which can lead to a stable gait, but with lots of engineering and prior knowledge, In our approach, we have compared the gait by using Dynamical equation and reinforcement learning, and we found that this was not as stable as gait produced by deep RL. Here, the equation complexity also increases as we increase the robot's degree of freedom (DoF).

C. Reinforcement Learning

Recent breakthroughs in reinforcement learning [5], [13], [23], [24] and multi-legged locomotion have been dependent on efficiently training deep neural networks along with gradient descent. It is better to learn the network using deep neural networks by feeding extensive data in deep neural networks than handcrafted features from scratch. These merits motivate our approach to apply reinforcement learning in our quadruped robot. Our goal is to use a deep reinforcement-learning algorithm for molding trajectories without prior experience of dynamics and synchronization for all legs. For reinforcement learning, prior knowledge can act as a catalyst. The policy then results in better education of the policy. Q-learning architecture [8] was the starting point for our approach, but many input states and their respected actions were causing computational deficiency concerning our hardware. This algorithm is an off-policy algorithm because it acquires the policy with the help of greedy exploration, similar to taking random actions or maximizing the Q function. Therefore, an action policy is not required for selecting actions. Q learning is an off-policy [25] reinforcement learning algorithm that tries to find the best action according to the given input state or current state. Moreover, Q-learning tries to understand the approach by continuously maximizing the expected rewards fed through sensors data. Our algorithm, as shown in algorithm 1, utilizes a technique known as experience replay. We store all the previous experiences at each time-step, $e_1 = (\phi_t, a_t, r_t, \phi_{t+1})$ in a dataset $[D = e_1, ...e_N]$ many episodes are clubbed into a replay memory. Then we initialize an action-value function Q with random weights. In each episode, we select an action based on ϵ-greedy policy. As shown in Fig. 5, our neural network takes four input states of each leg, and actions are opening the pneumatic piston, closing it, or nothing. As shown, Fig. 4 summarizes the whole deep RL algorithm. It consists of four sections, which are starting from Data collection, which uses the target network for getting the action using greedy epsilon ϵ algorithm, and since using histories of variable length will be difficult because of memory constraint in real-world robots; instead, it works on the fixed-length representation of histories produced by a function ϕ; next, it measures all the sensory inputs and calculates rewards. These states, action, reward, and next state are stored in replay buffer *D* for every transition where each transition is $(\phi_t, a_t, r_t, \phi_{t+1})$ Then comes the gradient descent part where we calculate the loss using the equation as shown below,

$$\begin{aligned}\theta_{K+1} = \theta_K + \alpha[(y_{max_{a'}} Q(s', a'; \theta_{i-1}) \\ -Q(s, a, \theta_i) - \nabla\theta_i Q(s, a; \theta i))]\end{aligned} \quad (1)$$

The asynchronous design makes the system more robust towards hardware failure and issues related to communication. This design can help to continue our training even after any hardware failure in practice. We faced these issues during our training. Thus, this system was beneficial for continuous flow. This system can also be deployed for multiple machine setups where it will not be dependent on any system.

In the pre-processing pipeline, we are using supervised learning, as shown in Fig. 5. It consisted of 3 hidden layers along with input states and output as actions. Here actions are mapped to 12 by constraining the quadruped's gait. This design also decreases the complexity of the neural network. Input states are the state of each piston which includes eight total states. Moreover, the final gait was learned even faster than many implementations are done on simulator [5]. This same neural network is also used in the Deep Q learning algorithm.

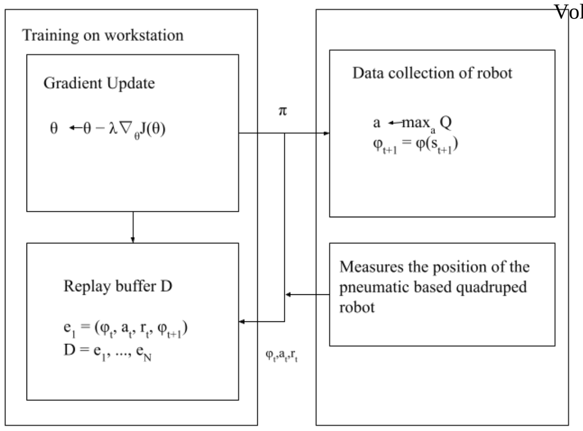

Fig. 4 Asynchronous Learning system

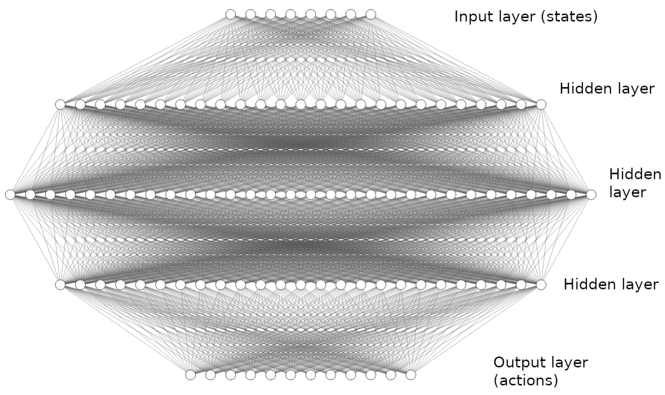

Fig. 5 Simulation Results

Algorithm 1: Deep Q learning with experience relay

```
1  Initialize replay memory D to capacity N
2  Initialize action-value function Q with random weights $\theta$
3  while $episode < M$ do
4      Initialize sequence $s_1 = x_1$ and prequel sequenced $\phi_1 = (s_1)$
5      while $t < T$ do
6          With probability epsilon $\epsilon$ select a random action
7          Otherwise select action $a_t = max_a Q*(\phi(s_t), a; \theta)$
8          Execute action and observe reward $r_t$ and next state $x_{t+1}$
9          Set $s_{t+1} = (s_t), a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
10         Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
11         Sample random mini-batch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
12         if $\phi_{j+1}$ is terminal then
13             $y_j = r_j$
14         end
15         if $\phi_{j+1}$ is not terminal then
16             $y_j = r_j + \gamma max_a Q*(\phi_{j+1}, a'; \theta)$
17         end
18         Perform a gradient descent step on $(y_j = -Q*(\phi_j, a_j; \theta))^2$
19     end
20 end
```

This approach has several advantages over other Deep RL algorithms. Experience relay [12] is used to update weights, which results in data efficiency compared to Q-learning. Also, randomizing the samples helps in efficiently updating weights because of the low correlation between samples. In this example, we store limited experience replay memory due to memory constraints. Deep Q learning algorithm was sampled efficiently for our pneumatic quadruped robot, which uses physical hardware. Thus, we used a supervised training approach using the same network as shown in Fig. 1, which consisted of three hidden layers; we trained this network with an unstable crawling gait. The input state consisted of eight inputs where the leg consisted of three values 0,1, and -1, which includes 0 for the intermediate state, as shown in Fig. 5. If we had taken all actions of a pneumatic quadruped robot, it would have become 3^4 actions, but our output state consisted of 12 actions. These 12 actions were implemented by considering the front two legs with the same actions, thus simultaneously reducing the action size and computation. This process gave us some pre-trained weights for the Deep reinforcement learning algorithm. For each state, the actions were mapped as the desired output. In this way, we achieved better sample efficiency and reduced the hardware damage caused by random weights.

V. Training and Stability

In a supervised learning algorithm (deep learning algorithm), tracking the training and evaluation of the model is comparatively easier than reinforcement learning algorithms. According to the actions recorded as per unstable crawling gait, the model was trained in the supervised learning technique. Our algorithm uses the mean of total rewards, which we periodically train using gradient descent. In Fig. 7, the rewards of the bot are mapped according to the distance of the bot concerning the origin. In Fig. 6, the stability of the bot has been plotted using the roll value of the IMU sensor. The stability is more in the deep RL model as compared to the dynamic model. Fig. 2 shows us the analysis of the height of the robot concerning the ground. Here, the robot is hopping at a consistent height [26], maintaining stability. In the training phase, the bot opted for this gait after 2 hours of training. These training hours were reduced by providing pre-trained weights on a similar gait.

VI. Experiments

We use the PyTorch framework to implement a neural network for deep reinforcement learning. We use the same algorithm as in [23]. Our rewards scheme was the distance between the bot, and the origin, using a laser mounted at the robot's back. During training, the behavior policy was -greedy

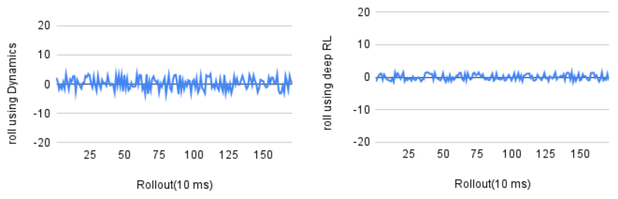

Fig. 6 IMU comparison with Dynamics and deep RL

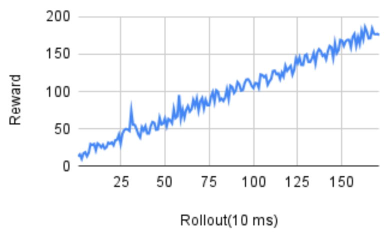

Fig. 7 Reward

with annealed linearly from 1 to 0.1 over the first few episodes and fixed at 0.1 after that [27].

VII. Results

In this framework, the gait was learned from 400 episodes, with each episode taking around 15 sec for each episode. Also, there were other termination criteria when the quadruped fails to update IMU reading according to a stable position. As per the framework, the most efficient gait was the hopping gait which we obtained. Then for the continuous motion, the same gait was periodically and synchronously scheduled. As we can compare the initial and final gait after the framework, there was a vast difference in stability and speed. This framework was also tested on different slopes, with all cases being succeeded except steep slopes. This robustness was seen only after training on flat terrain.

VIII. Conclusion

After the testing was conducted, it was concluded that such a design and policy learned by deep RL were indeed possible. A better damping model at the foot of the leg is needed to minimize the transmitted force. There were losses in the energy achieved due to the internal friction of the cylinder. However, these losses and inefficiencies were avoided by deep RL while training. The hopping gait obtained through the policy was tested, and relevant results were obtained. The robot works well with only two active DoF, but adding another DoF would give it much more controllability, eventually increasing policy complexity. The use of pneumatic provides greater control in an active two-DF configuration with a greater effective force, which is enhanced by the deep RL algorithm. More work needs to be done to apply other deep RL algorithms and analyze the true potential. After applying supervised learning to improve the deep RL model in two, DoF proved to be very efficient with minimal neural networks. Due to its energy efficiency, low cost of transportation, and reasonable force control, which can be increased through different policies with better deep RL networks and models. As per our research and knowledge, our application is the first example of applying deep RL directly to the real-world pneumatic quadruped robot. The next goal of this project can be to optimize the reinforcement learning process for another type of gaits and terrain to enhance its robustness further.

Acknowledgment

We thank Dr. Irfan Siddavatam for providing guidance throughout the project and useful discussion. We thank K.J Somaiya College of Engineering for assistance throughout the project.

References

[1] M. Focchi, E. Guglielmino, C. Semini, T. Boaventura, Y. Yang, and D. G. Caldwell, "Control of a hydraulically-actuated quadruped robot leg," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 4182–4188.
[2] K. Narioka, A. Rosendo, A. Badri-Spröwitz, and K. Hosoda, "Development of a minimalistic pneumatic quadruped robot for fast locomotion," 12 2012, pp. 307–311.
[3] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. A. Eslami, M. Riedmiller, and D. Silver, "Emergence of locomotion behaviours in rich environments," 2017.
[4] X. B. Peng, G. Berseth, K. Yin, and M. van de Panne, "Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning," *ACM Transactions on Graphics (Proc. SIGGRAPH 2017)*, vol. 36, no. 4, 2017.
[5] 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," 2018.
[6] Z. Xie, G. Berseth, P. Clary, J. Hurst, and M. van de Panne, "Feedback control for cassie with deep reinforcement learning," 2018.
[7] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 3, 2004, pp. 2619–2624 Vol.3.
[8] B. Jang, M. Kim, G. Harerimana, and J. W. Kim, "Q-learning algorithms: A comprehensive classification and applications," *IEEE Access*, vol. 7, pp. 133 653–133 667, 2019.
[9] M. Hutter, C. Gehring, A. Lauber, F. Günther, D. Bellicoso, V. Tsounis, P. Fankhauser, R. Diethelm, S. Bachmann, M. Blösch, H. Kolvenbach, M. Bjelonic, L. Isler, and K. Meyer, "Anymal - toward legged robots for harsh environments," *Advanced Robotics*, vol. 31, pp. 918 – 931, 2017.
[10] 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, Jan 2019. [Online]. Available: http://dx.doi.org/10.1126/scirobotics.aau5872
[11] J. Meng, Y. Li, and B. Li, "A dynamic balancing approach for a quadruped robot supported by diagonal legs," *International Journal of Advanced Robotic Systems*, vol. 12, p. 1, 10 2015.
[12] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," 2016.
[13] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," 2019.
[14] G. Berseth, C. Xie, P. Cernek, and M. V. de Panne, "Progressive reinforcement learning with distillation for multi-skilled motion control," 2018.
[15] X. B. Peng, G. Berseth, and M. van de Panne, "Terrain-adaptive locomotion skills using deep reinforcement learning," *ACM Trans. Graph.*, vol. 35, no. 4, Jul. 2016. [Online]. Available: https://doi.org/10.1145/2897824.2925881
[16] S. E. Levin, D. E, H. Perspective, and S. Of, "Raibert, m. h., "legged robots that balance." cambridge, mass.: Mit press (1986)." 1995.
[17] F. Moro, A. Badri-Spröwitz, A. Tuleu, M. Vespignani, N. Tsagarakis, A. Ijspeert, and D. Caldwell, "Horse-like walking, trotting, and galloping derived from kinematic motion primitives (kmps) and their application to walk/trot transitions in a compliant quadruped robot," *Biological cybernetics*, vol. 107, 03 2013.
[18] N. Hansen, *The CMA Evolution Strategy: A Comparing Review*, 06 2007, vol. 192, pp. 75–102.
[19] A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke, "Policies modulating trajectory generators," 2019.

[20] X. Tran and H. Yanada, "Dynamic friction behaviors of pneumatic cylinders," *Intelligent Control and Automation*, vol. 04, pp. 180–190, 01 2013.

[21] M. Sorli, L. Gastaldi, E. Codina, and S. Heras, "Dynamic analysis of pneumatic actuators," *Simul. Pr. Theory*, vol. 7, pp. 589–602, 12 1999.

[22] W. Sabo and P. Ben-Tzvi, "Maneuverability and heading control of a quadruped robot utilizing tail dynamics," 10 2017, p. V002T21A010.

[23] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 12 2013.

[24] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.

[25] M. Hausknecht and P. Stone, "On-policy vs. off-policy updates for deep reinforcement learning," in *Deep Reinforcement Learning: Frontiers and Challenges, IJCAI Workshop*, New York, July 2016. [Online]. Available: http://www.cs.utexas.edu/users/ai-lab?hausknecht:deeprl16

[26] M. F. Hale, J. L. Du Bois, and P. Iravani, "Agile and adaptive hopping height control for a pneumatic robot," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 5755–5760.

[27] N. Ashraf, R. Mostafa, R. Sakr, and M. Rashad, "Optimizing hyperparameters of deep reinforcement learning for autonomous driving based on whale optimization algorithm," *PLOS ONE*, vol. 16, p. e0252754, 06 2021.