Proceedings of IMECE International Mechanical Engineering Congress & Exposition
IMECE 2021
November 01-November 05, 2021, Virtual Conference, Online, USA

IMECE2021-70012

SPECIFYING THE SPRING CONSTANT OF A MONOPODE JUMPING SYSTEM WITH REINFORCEMENT LEARNING

Andrew Albright
Department of Mechanical Engineering
University of Louisiana at Lafayette
Lafayette, Louisiana 70504
andrew.albright1@louisiana.edu

Joshua Vaughan
Department of Mechanical Engineering
University of Louisiana at Lafayette
Lafayette, Louisiana, 70504
joshua.vaughan@louisiana.edu

ABSTRACT

Legged systems have many advantages when compared to their wheeled counterparts. For example, they can more easily navigate extreme, uneven terrain. However, there are disadvantages as well, including difficulties seen in modeling the nonlinearities of the system. Research has shown that using flexible components is advantageous in terms of both efficiency and legged locomotive performance. Because of the difficulties encountered in modeling flexible systems, control methods such as reinforcement learning can be used to define control strategies. Furthermore, reinforcement learning can be tasked with learning mechanical parameters of a system to match a control input. It is shown in this work that by deploying reinforcement learning to solve for the optimal spring constant for a pogo-stick jumping system, designs which the agent defines are shown to be higher performing in terms of jumping height.

1 INTRODUCTION

The use of flexible components within legged locomotive systems has proved useful for both reducing power consumption and increasing performance [1–3]. However, designing controllers for these systems is difficult as the flexibility of the system generates nonlinear models. As such, employing series-elastic-actuators (SEA) instead of flexible links is an attractive and popular solution, since the models of the systems become more manageable [2,4,5]. Still, the use of SEAs do not represent the full capability of flexible systems. As a result, other methods that use flexible tendon-like materials meant to emulate more organic designs have been proposed [6]. These however are still not representative of fully flexibly links which have been shown to drastically increase locomotive performance measures such as running speed [7].

Control methods have been developed that work well for flexible systems like the ones mentioned [8, 9]. However, as the systems increase in dimensionality, effects such as dynamic coupling between members make such methods challenging to implement. As such, work has been done which uses neural networks and methods such as reinforcement learning (RL) to develop controllers for flexible systems [10, 11]. For example, RL has been used to create faster running control strategies for flexible-legged locomotive systems that also are robust to different design parameters [12].

In addition to the work done using RL to develop controllers for flexible systems, work has been completed which shows that this technique can be used to concurrently design the mechanical aspects of a system and a controller to match said system [13]. These techniques have even been used to define mechanical parameters and control strategies where the resulting controller and hardware were deployed in a sim-to-real process, validating the usability of the technique [14]. Using this technique for legged-locomotion has also been studied, but thus far has been limited to the case of rigid systems [15].

As such, this paper starts the discovery of using RL for concurrent design of flexible-legged locomotive systems. A simplified flexible jumping system was used where, for the initial work, the control input was held fixed so that the RL algorithm was tasked with only learning optimized mechanical parameters.

The rest of the paper is organized such that in the next section, similar work will be discussed. In Section 3, the pogo-stick environment details will be defined. Then, in Sections 4, the RL algorithm used will be discussed. Following that, in sections 5 and 6, the experimental details and the results will be explained. Finally, in Section 7, the work completed will be concluded.

2 Related Work

2.1 Flexible Locomotive Systems

The use of flexible components within robotics systems has shown improvements in performance measures specifically ones which are locomotive [3]. In particular, advantages have been seen in locomotion applications where crawling and jumping are employed [1]. Previous work has shown that the use of flexible components in the legs of legged locomotion systems increase performance while decreasing power consumption [7]. Related to work on robotics systems employing flexible links, work has been done showing the uses of series-elastic-actuators for locomotive systems [5]. In much of this work, human interaction with the robotic systems is considered where rigidity is not ideal [4]. The studies of flexible systems are challenging however, as the models which represent them are often nonlinear and therefore difficult to develop control systems for. As such, there is a need for solutions which can be deployed to develop controllers for these nonlinear systems.

2.2 Controlling Flexile Systems Using RL

Control methods developed for flexible linked systems have been shown to be effective for position control and vibration reduction [8,16]. Because of the challenges seen in scaling the controllers, methods utilizing reinforcement learning are of interest. This method has been used in simple planar cases, where it is compared to a PD control strategy for vibration suppression and proves to be a higher performing method [17]. Additionally, it has also been shown to be effective at defining control strategies for flexible-legged locomotion. The use of actor-critic algorithms such as Deep Deterministic Policy Gradient [18] have been used to train running strategies for a flexible legged quadruped [12]. Much of the research is based in simulation, however, and often the controllers are not deployed in a sim-to-real fashion which leads to the question on whether or not these are practically useful techniques.

2.3 Concurrent Design

Defining an optimal controller for a system can be challenging due to things like mechanical and electrical design limits. This is especially true when the system is flexible and the model is nonlinear. A solution to this challenge is to concurrently design a system with the controllers so that the two are jointly optimized. This is has been researched in previous work as a strategy to develop better performing mechatronics systems [19]. More recent work has been completed which used advanced methods such as evolutionary strategies to define robot design parameters [20]. In addition to evolutionary strategies, reinforcement learning has been shown to be a viable solution for concurrent design of 2D simulated locomotive systems [13]. This is further shown to be a viable method by demonstrating more complex morphology modifications in 3D reaching and locomotive tasks [15]. However, these techniques have not been applied to flexible systems for locomotive tasks.

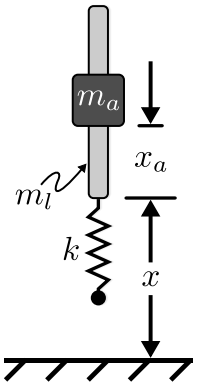

FIGURE 1. Pogo-stick System

TABLE 1. POGO-STICK MODEL PARAMETERS

Model Parameter	Value
Mass of Leg	0.175 kg
Mass of Actuator	1.003 kg
Natural Frequency	11.13 Hz
Spring Constant, k	50000–350000 N/m
Actuator Stroke, $(x_a)_{\max}$	25.0 mm
Actuator Velocity, $(\dot{x}_a)_{\max}$	1.0 m/s
Actuator Acceleration, $(\ddot{x}_a)_{\max}$	10.0 m/s^2

3 Pogo-stick Model Description

The pogo-stick model show in Figure 1 has been shown to be useful as a representation of several different running and jumping gaits [21]. As such, it is used in this work to demonstrate the ability of reinforcement learning for the initial steps of concurrent design. The models parameters are summarized in Table 1.

The variable m_a represents the mass of the actuator, which moves along the rod with mass m_l. A non-linear spring with constant k is used as the representation of flexibility. A damper (not shown in Figure 1), is parallel to the spring. Variables x and x_a represent the systems vertical position with respect to the ground and the actuator's position along the rod, respectively. The system is additionally constrained such that it only moves vertically, so the reinforcement agent is not required to balance the system.

The equations of motion describing the system are:

$$\ddot{x} = \alpha \left(\frac{k}{m_t} x^3 + \frac{c}{m_t} \dot{x} \right) - \frac{m_a}{m_t} \ddot{x}_a - g \tag{1}$$

where x and x_a are position and velocity of the rod respectively, the acceleration of the actuator, x_a, is the control input, and m_t is the mass of the complete system. Ground contact determines the value of α, so that the spring and damper do not supply force while the leg is airborne:

$$\alpha = \begin{cases} -1, & x \leq 0 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

4 Reinforcement Learning

The algorithm used for this work is Twin Delayed Deep Deterministic Policy Gradient (TD3) [22]. This is an actor-critic algorithm wherein there exists two main neural networks and a set of twin trailing networks. The first main network is the actor, which determines the control input. This network takes in the systems state, $\mathcal{S}$, and outputs the action (control input), $\mathcal{A}$, based on the state. The critic is an estimator of the value of being in a state and is used to determine the difference between expected and estimated value to update the actor network during training. It takes in the systems state, $\mathcal{A}$, and outputs the expected future reward, $\mathbb{R}$, from being in that state. The twin trailing networks are used to find the temporal difference error against the critic network which is used to update the critic network.

5 Experiments

5.1 Jumping Types

Two different jump types were used to evaluate if reinforcement learning can generate designs to match the control input from the work in [23]. The first jump type can be seen in Fig 2. Here, the goal of the agent was to learn a spring constant such that given the control input, the system would jump as high as possible. The second jump type can be seen in Fig. 3, where the goal of the agent was the same as the previous jump type but for two jumps given a similar input from [23].

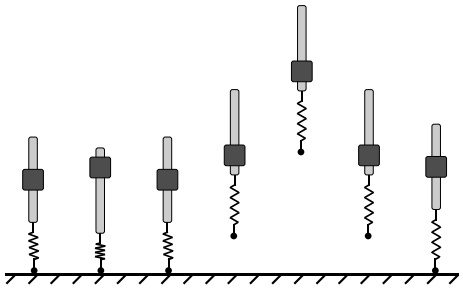

FIGURE 2. Example Single Jump

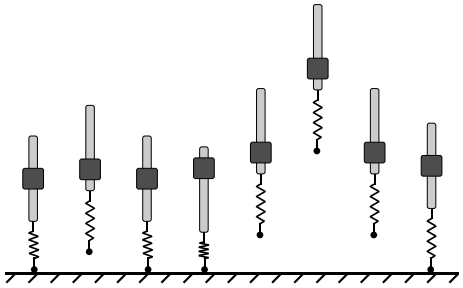

FIGURE 3. Example Stutter Jump

5.2 Control Input

The control input that was used is from the work completed by [23]. This is selected as the control input based on system similarity, and purpose jumping performance. The input shaped control input in the mentioned work, is an attempt at a near optimal jumping height solution. As such, if it is paired with an optimal mechanical design, the systems should perform at near optimal levels.

5.3 Learning Optimal Spring Constant

A reinforcement learning environment was modeled according to an OpenAI Gym structure [24]. With a set control input the agents actions, at each time step during training, were selecting a spring constant. Once selected, the system was simulated and the time series data from the simulation were used as the state transition passed back to the RL algorithm. The reward passed back to the algorithm was the maximum height the system reached so that the agent would learn to maximize jump height.

6 Results

Results.

7 Conclusion

Conclusion.

ACKNOWLEDGMENT

The authors would like to thank the Louisiana Crawfish Promotion and Research Board for their support of this work.

REFERENCES

[1] Sugiyama, Y., and Hirai, S., 2004. "Crawling and jumping of deformable soft robot". *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* ***4***(c), pp. 3276–3281.

[2] Buondonno, G., Carpentier, J., Saurel, G., Mansard, N., De Luca, A., and Laumond, J. P., 2017. "Actuator design of compliant walkers via optimal control". *IEEE International Conference on Intelligent Robots and Systems,* ***2017-Septe***, pp. 705–711.

[3] Hurst, J., 2008. "The Role and Implementation of Compliance in Legged Locomotion". *The International Journal of Robotics Research,* ***25***(4), p. 110.

[4] Zhang, T., and Huang, H., 2019. "Design and Control of a Series Elastic Actuator with Clutch for Hip Exoskeleton for Precise Assistive Magnitude and Timing Control and Improved Mechanical Safety". *IEEE/ASME Transactions on Mechatronics,* ***24***(5), pp. 2215–2226.

[5] Pratt, G. A., and Williamson, M. M., 1995. "Series elastic actuators". *IEEE International Conference on Intelligent Robots and Systems,* ***1***, pp. 399–406.

[6] Iida, F., Gómez, G., and Pfeifer, R., 2005. "Exploiting body dynamics for controlling a running quadruped robot". *2005 International Conference on Advanced Robotics, ICAR '05, Proceedings,* ***2005***, pp. 229–235.

[7] Saranli, U., Buehler, M., and Koditschek, D. E., 2001. "RHex: A Simple and Highly Mobile Robot". *International Journal of Robotics Research,* ***20***(7), pp. 616–631.

[8] Luo, Z.-h., 1993. "Direct Strain Feedback Control of Flexible Robot Arms: New Theoretical and Experimental Results".

[9] Modeling, A. M., 2003. "Gain Adaptive Nonlinear Feedback Control of Flexible SCARA / Cartesian Robots". pp. 1423–1428.

[10] Bhagat, S., Banerjee, H., Tse, Z. T. H., and Ren, H., 2019. "Deep reinforcement learning for soft, flexible robots: Brief review with impending challenges". *Robotics,* ***8***(1), pp. 1–36.

[11] Thuruthel, T. G., Falotico, E., Renda, F., and Laschi, C., 2019. "Model Based Reinforcement Learning for Closed Loop Dynamic Control of Soft Robotic Manipulators". *IEEE Transactions on Robotics,* ***35***, pp. 124–134.

[12] Dwiel, Z., Candadai, M., and Phielipp, M., 2019. "On Training Flexible Robots using Deep Reinforcement Learning". *IEEE International Conference on Intelligent Robots and Systems*, pp. 4666–4671.

[13] Ha, D., 2019. "Reinforcement learning for improving agent design". *Artificial Life,* ***25***(4), pp. 352–365.

[14] Chen, T., He, Z., and Ciocarlie, M., 2020. "Hardware as Policy: Mechanical and computational co-optimization using deep reinforcement learning". *arXiv*(CoRL).

[15] Schaff, C., Yunis, D., Chakrabarti, A., and Walter, M. R., 2019. "Jointly learning to construct and control agents using deep reinforcement learning". *Proceedings - IEEE International Conference on Robotics and Automation,* ***2019-May***, pp. 9798–9805.

[16] Ahmadi, M., and Buehler, M., 1997. "Stable control of a simulated one-legged running robot with hip and leg compliance". *IEEE Transactions on Robotics and Automation,* ***13***(1), pp. 96–104.

[17] He, W., Gao, H., Zhou, C., Yang, C., and Li, Z., 2020. "Reinforcement Learning Control of a Flexible Two-Link Manipulator: An Experimental Investigation". *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–11.

[18] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D., 2016. "Continuous control with deep reinforcement learning". *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*.

[19] Li, Q., Zhang, W. J., and Chen, L., 2001. "Design for control - A concurrent engineering approach for mechatronic systems design". *IEEE/ASME Transactions on Mechatronics,* ***6***(2), pp. 161–169.

[20] Wang, T., Zhou, Y., Fidler, S., and Ba, J., 2019. "Neural graph evolution: Towards efficient automatic robot design". *arXiv*, pp. 1–17.

[21] Blickhan, R., and Full, R. J., 1993. "Similarity in multilegged locomotion: Bouncing like a monopode". *Journal of Comparative Physiology A,* ***173***(5), pp. 509–517.

[22] Fujimoto, S., Van Hoof, H., and Meger, D., 2018. "Addressing Function Approximation Error in Actor-Critic Methods". *35th International Conference on Machine Learning, ICML 2018,* ***4***, pp. 2587–2601.

[23] Vaughan, J., 2013. "Jumping Commands For Flexible-Legged Robots".

[24] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W., 2016. "OpenAI Gym". pp. 1–4.