Analysis of Robotic Control Policies Learned by Deep Reinforcement Learning Networks

Yuchun (Peggy) Wang*, Chuanyu Yang†, Kai Yuan†, Zhibin Li†
*Department of Computer Science, Stanford University
{peggyyuchunwang}@cs.stanford.edu
†School of Informatics, University of Edinburgh
{chuanyu.yang, kai.yuan, zhibin.li}@ed.ac.uk

***Abstract*—Deep reinforcement learning policies have the potential to be applied to real world robotic systems. However, because of the "black box" nature of deep neural networks, researchers do not understand how the control policies are generated. We conducted a systematic analysis on a 2D humanoid robot agent that was trained to balance using a deep reinforcement learning policy. We show that the learned control policy generates an adaptable control policy for the foot tilting height, the policy switches between different control policies based the current phase, and that the policy is able to be modeled as affine functions with different parameters. We can then use these results to reverse engineer how the neural network was able to generate these control policies, allowing for future applications of deep reinforcement learning on real world systems.**

I. Introduction

A. Background

Advances in the area of deep reinforcement learning (DRL) have led to artificial intelligent agents achieving or surpassing human level performance, especially in the areas of game playing, such as Atari [1] and Go [2]. Moreover, DRL has been applied to the area of continuous action spaces, such as finding control policies for simulated physics tasks, including legged locomotion [3] [4]. However, there are very few examples of transferring control policies learned from DRL into the physical world outside of simulation environments. Applying DRL control policies to real world robotic controls is particularly useful, since it will be more efficient than manually programming and tuning classical controls.

Before implementing DRL control policies on real world robots, it is imperative to understand how the deep neural networks (NN) actually compute the control policies. Historically, NN are considered "black box" methods because it is not immediately clear how the trained NN generates the correct (or incorrect) output from an input [5].

We are unable to understand and improve the system if the policy and actions of the agent are not explainable. Preventable dangerous situations, such as when humans are injured by an autonomous robot based on its policy, would occur. Research which attempts to explain how NN thinks, such as Olah et al.'s visualization technique, has been done in the area for image processing [6] [7]. However, to the best of our knowledge, no research has been done in the area of understanding how NN in DRL generates robotic control policies. Therefore, it is of great interest to analyze what control policies the NN generates, how the NN generate control policies, and how the performance of these policies compare to classical control policies.

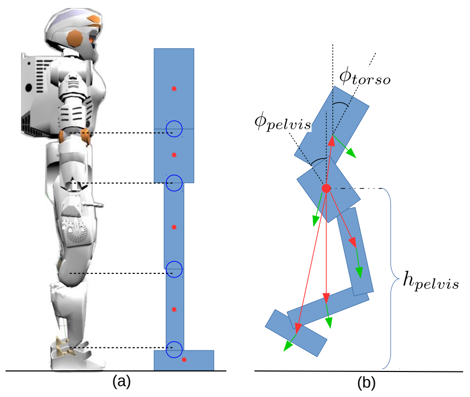


Fig. 1: Depiction of the humanoid character. (a) Side view of 2D humanoid and the Valkyrie robot. (b) State features. Figure courtesy of Yang, Komura, and Li [8].

B. Related Work and Motivation

Yang, Komura, and Li [8] have used the Deep Deterministic Policy Gradient (DDPG) algorithm [9] to train a bipedal humanoid robot to balance in an OpenAI 2D physics simulation environment. The humanoid was modeled on the Valkyrie robot (shown in Fig. 1). State features with data from the kinematics and dynamics of humanoid were logged. Features include the pelvis angle and angular velocity, torso angle, and center of mass (COM) position and velocity. The reward function is then given as a linear combination of state features.

The robot was pushed with a horizontal force for 0.1s from the pelvis. A control policy learned by DDPG was then applied on the joint angles for the humanoid push recovery balance. The optimal joint angles for balancing is learned by the high-level NN controller, and a low-level Proportional Derivative (PD) controller is used to translate the desired joint angles from the the high-level controller into the joint torque.

Human-like behaviors such as foot tilting, knee lock, and heel/toe tipping behaviors naturally occurred as a result of the policy that the deep NN learned [8], even though the reward function and code did not explicitly encourage these behaviors. We are interested in finding out how these behaviors emerge from the NN control policy, how the NN implements these policies, and the comparison of the NN control policy to classical control policies, human control policies, and optimal control policies.

C. *Organization*

This work presents a systematic analysis of a control policy learned by a DRL humanoid robotic agent [8]. This paper is organized as follows. Introduction, background, and related work is described in Section I. Methodology of the experiment is described in Section II. Data analysis of the results is described in Section III, followed by the future work in Section IV and conclusion in Section V.

II. Methodology

A. *Simulation Data Collection*

This work builds on the OpenAI physics simulation environment that Yang, Komura, and Li [8] developed. The simulation is deterministic for each force unit and lasts for 10 seconds. Starting from five seconds, the humanoid is pushed at a constant given horizontal force at the pelvis for 0.1s. The humanoid then attempts to rebalance itself based on the learned control policy using the DDPG algorithm. We made modifications to Yang, Komura, and Li's code that includes logging the force in each simulation and the ability to change the force applied in the command line before the simulation. We then wrote scripts to log and consolidate the data, which is saved in a .mat file. Afterwards, we ran trials where the humanoid is pushed from forces of -670N to 700N, in increments of 10N. Positive forces meant the robot was pushed horizontally from the back of the pelvis, and negative forces meant that the robot was pushed horizontally from the front of the pelvis. If the force is lower than -670N or higher than 700N, the robot falls.

Examples of data collected included the amount of force pushed, center of mass (COM) position and velocity, and foot position and velocity for each time step. The foot position and velocity was based on the COM of the foot joint. The data was sampled at a rate of 500Hz. Since the foot was 0.111m from the ankle to the back and 0.189m from the ankle to the front, we normalized the data by analyzing the force from -400N to 700N based on the foot front and back ratios. We then calculated the COM acceleration by deriving the velocities with respect to time. Lastly, we applied a moving average filter with a window size of 15Hz to the acceleration.

B. *Systematic Data Analysis*

We then performed a systematic data analysis on the learned control policy using Matlab. Results for the analysis can be found in Section III. We defined foot tilting behavior as occurring if the maximum foot height difference is above 0.005m. We categorized the data into positive and negative forces, as well as foot tilting and non-foot tilting behaviors. Five different phases best described the plot: initial push, foot tilting up, foot tilt maximum height, foot tilt down, and settling. The separation of the phases were based on four points: time when push stopped, time of foot maximum height minus 0.002m, time of foot maximum height plus 0.002m, and time of foot reaching starting height after maximum height.

Each separate phase can be described as a different 2D affine function of the form

$$\ddot{x}_{COM} = Ax_{COM} + B\dot{x}_{COM} + C \qquad (1)$$

where $\ddot{x}_{COM}$ is the desired COM horizontal acceleration and where A, B, and C are parameters describing the best fit affine function based on the amount of force applied and phase. x_{COM} is the horizontal position of the center of mass, and $\dot{x}_{COM}$ is the horizontal velocity of the center of mass. We compared the predicted COM horizontal acceleration of the affine function with the actual COM horizontal acceleration from data.

Scripts for the data collection and analysis may be found at https://github.com/PeggyYuchunWang/ControlsAnalyzationDeepRL.

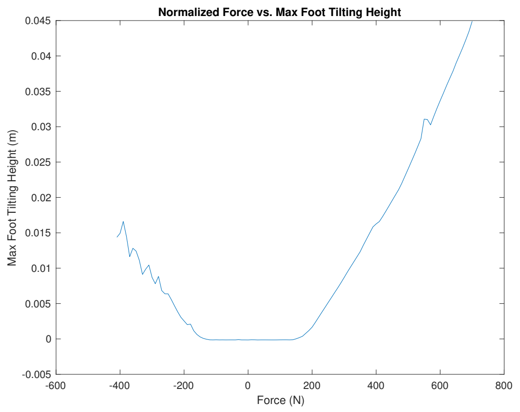


Fig. 2: Force vs. Max Foot Tilt Height

III. Results

A. *Overview*

After analysis, we have determined that the humanoid uses a variety of strategies for its balancing control policy. This control policy can be described as:

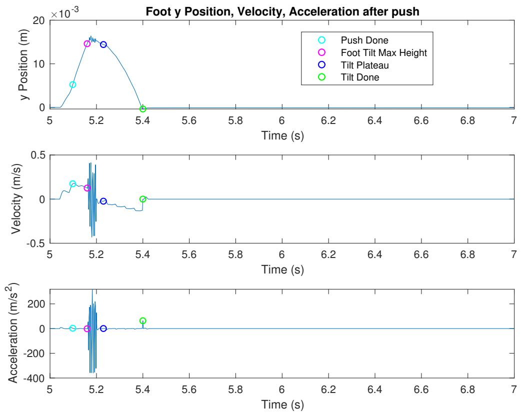
$$\ddot{x}_{COM} = \alpha f(p) + \beta g(\mu) + \gamma h(\eta) \tag{2}$$

where $\ddot{x}_{COM}$ is the desired COM horizontal acceleration, p is the center of pressure of the foot, μ is angular momentum of the robot, and η is the foot tilt height. Functions $f, g,$ and h output the desired COM acceleration based on the variable inputs. $\alpha, \beta,$ and γ are weighted parameters that describe proportionally how much each function contributes to the desired COM acceleration.

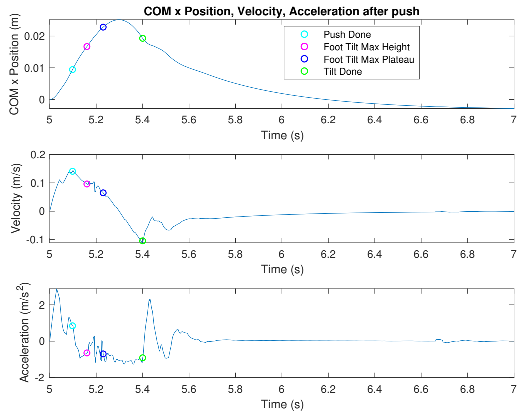
We found that positive and negative forces of the same magnitude exhibited similar behavior. Forces that result in foot tilting behavior had different control policies than forces that do not result in foot tilting behavior. We show a full systematic analysis for the 400N scenario, which includes foot tilting behavior, but these results can be generalized to any other force.

B. *Adaptable Policy for Maximum Foot Tilt Height*

We have also determined that the humanoid applies some control policy to determine the maximum foot tilting height in order to dissipate the force of the push (Fig. 2). The robot varies the maximum foot tilting height based on the amount of force applied and does not always deterministically perform the same maximum foot tilting height for each force. This means that the deep neural network is able to generalize an adaptive control policy based on different dynamical states.



(a) Foot Height vs. Time



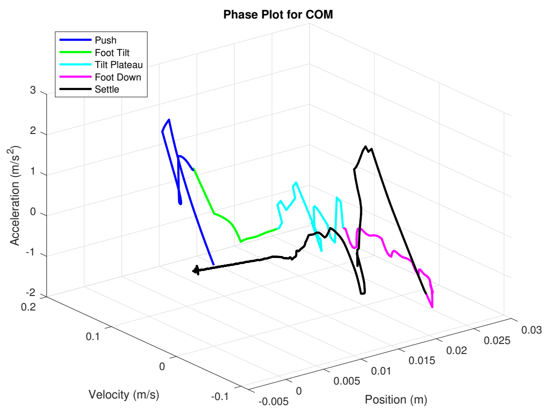
(b) COM Horizontal Position, Velocity, Acceleration

Fig. 3: 400N foot height and COM position, with defined points showing five different phases

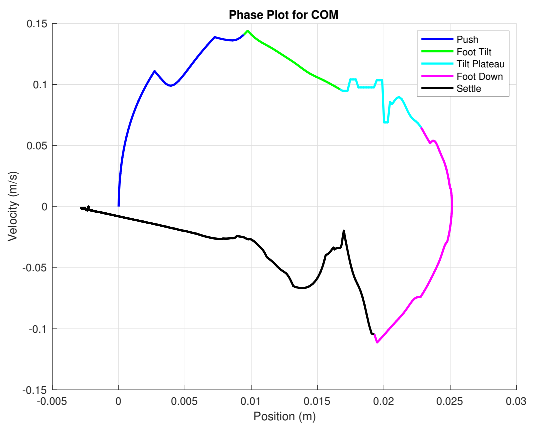
C. *COM Acceleration Policy*

We constructed a model of the COM horizontal acceleration as follows for forces that caused foot tilting behavior. We segmented the control policy in five different phases respectively: initial push, foot tilt up, foot tilt maximum height, foot tilt down, and settling. If no foot tilting behavior occurred, the control policy is modeled as two different phases respectively: initial push and settling phase.

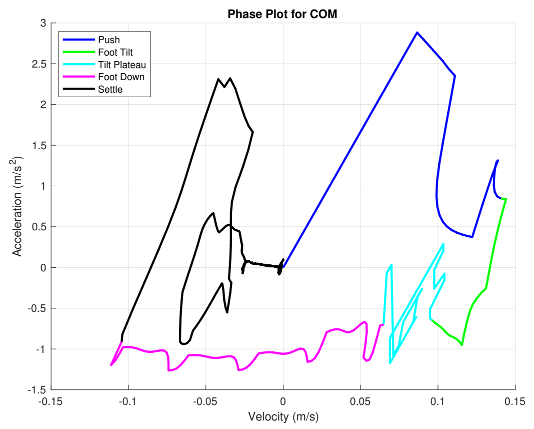
After analysis, we found that the different phases of the COM acceleration control likely depends on the stage of the foot tilting heights. As seen in Fig. 3, points described in Section II corresponding to the state of the foot heights were used. Therefore, the humanoid uses



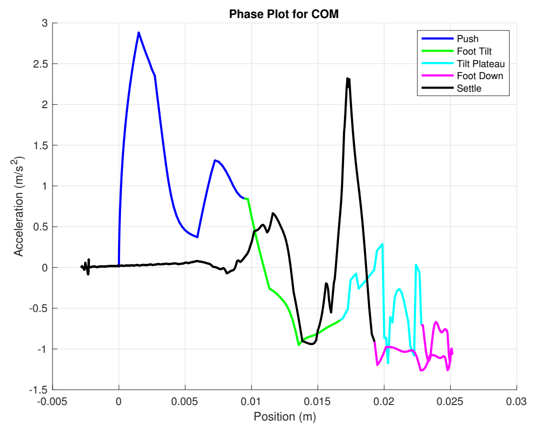
(a) COM Horizontal Phase Plot



(b) COM Horizontal Position vs. Velocity



(c) COM Horizontal Velocity vs. Acceleration



(d) COM Horizontal Position vs. Acceleration

Fig. 4: 400N phase plots showing COM acceleration given COM position and velocity

some mechanism to switch between different control policies based on the realization of the phase it is in (Fig. 4).

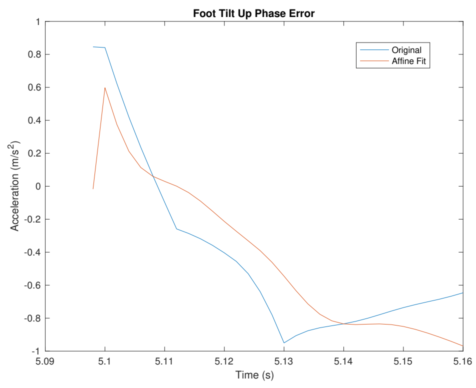
D. *Affine Function Fits*

The horizontal COM acceleration was able to be predicted by the affine function closely, with an average root mean squared error of 2.934. We have shown comparisons of the original horizontal COM acceleration calculated from the data collected with the predicted acceleration from the affine functions in Fig. 5. Since we are able to model the control policy generated by the DRL, we are able reverse engineer how the NN thinks based on the policy.

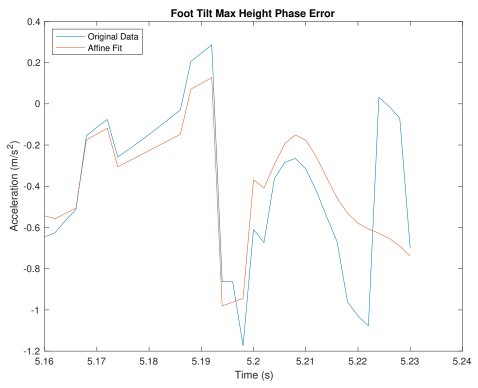
IV. Future Work

We have successfully shown that the robot applies explainable adaptable control policies to its COM acceleration. It would yield more insights to isolate the COM acceleration of the control policy from the gravitational acceleration to show only the acceleration that the controller generates. If we model the humanoid as a linear inverted pendulum or another model of underactuated control, we could subtract the gravitational acceleration of the model from the total COM acceleration.

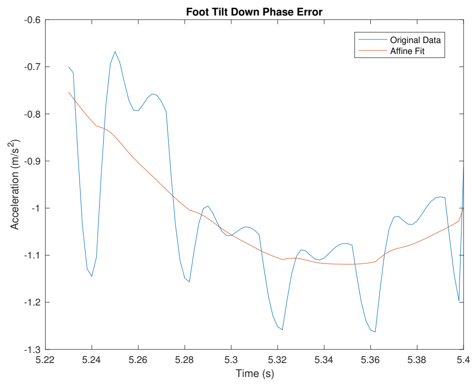
Additionally, it would be interesting to do additional research on how the robot determines its state and what mechanism it uses to switch between control policies. We could also compare the performance of the deep NN control policy with alternative control policies, including



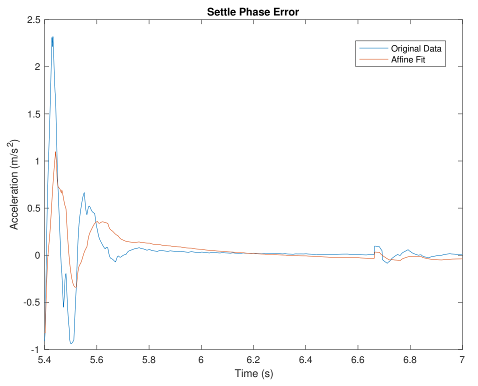
(a) Foot Tilt Up Phase Affine Fit



(b) Foot Tilt Down Phase Affine Fit



(c) Settle Phase Affine Fit



(d) Foot Tilt Down Phase Affine Fit

Fig. 5: Different Phases of 400N Affine Function Fits Compared to Original Data

control policies that humans use for push recovery balance, additional classical control policies, and the optimal control policy. Eventually, we hope to test policies learned by neural networks on physical robots, including the Valkyrie robot.

V. Conclusion

By using machine learning methods such as DRL, we are able to generate control policies without requiring a control engineer to manually tune the controller. As shown in this work, these policies can be explained by modeling them as adaptive control policies for different phases.

We can see that using machine learning for developing control policies has many applications in the robotics industry, especially since it is very time-intensive to manually design control policies for many different areas of the robot. With machine learning methods, we could become much more efficient at building controllers while keeping the same level of performance, if not surpassing them. Therefore, we may soon be seeing humanoid robots which would operate biomechanically similar to actual humans.

Acknowledgment

Many thanks to Dr. Zhibin Li and the Advanced Robotics Lab of the University of Edinburgh for advising and hosting us throughout this project. Additionally, thanks to graduate student mentors Chuanyu Yang and Kai Yuan for their help and insight. Additionally, thanks to Dr. Mykel Kochenderfer and Rachael Tompa of Stanford University for their advising and help. Thanks also to the Stanford University Bing Overseas Program (BOSP) for organization of this overseas seminar and Research Project.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

[3] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[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 (TOG)*, vol. 36, no. 4, p. 41, 2017.

[5] D. Castelvecchi, "Can we open the black box of ai?" *Nature News*, vol. 538, no. 7623, p. 20, 2016.

[6] C. Olah, A. Mordvintsev, and L. Schubert, "Feature visualization," *Distill*, 2017, https://distill.pub/2017/feature-visualization. DOI: 10.23915/distill.00007.

[7] C. Olah, A. Satyanarayan, I. Johnson, S. Carter, L. Schubert, K. Ye, and A. Mordvintsev, "The building blocks of interpretability," *Distill*, 2018, https://distill.pub/2018/building-blocks. DOI: 10.23915/distill.00010.

[8] C. Yang, T. Komura, and Z. Li, "Emergence of human-comparable balancing behaviours by deep reinforcement learning," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, Nov. 2017, pp. 372–377. DOI: 10.1109/HUMANOIDS.2017.8246900.

[9] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International Conference on Machine Learning (ICML)*, 2014.