Analyzing reinforcement learning algorithms with a physics simulation

Edward Muschamp

University of Lincoln, School of Computer Science

19694636@students.lincoln.ac.uk

Keywords: Balance Control, Agent, Reinforcement Learning (RL)

1 Introduction

1.1 Overview of Project

Balance control is the process of helping an agent (for example, a robot or a physics-based character in a video game) stay balanced. This is done through sensors fetching data from the environment and then calculating what actions need to be taken to maintain balance. A problem with this comes when the agent faces an overly complex situation and therefore sensors and calculations are needed. These calculations need to be performed and executed instantly to make sure the agent can respond in time before losing balance. If the agent is a physical robot, then the calculations will have to be done wirelessly because an immensely powerful computer is needed which cannot be attached to the robot. Wireless connections can cause latency issues and "errors and delays in communication can have serious consequences for a system and therefore reliability is of utmost importance" (Voigtländer et al., 2017).

A solution to is to use reinforcement learning (RL), a subcategory of machine learning, and is a way to enable the computer to learn through exploration and feedback received in a given environment (Sutton and Barto, 2018). RL can be especially useful when controlling agents whether they are in a simulation or in the world (robotics). It works by rewarding desired behavior and punishing negative behavior. This is achieved by setting certain parameters for the agent to achieve and giving the RL algorithm certain inputs to make the agent achieve the parameters. After a certain number of iterations, the RL algorithm will have learnt what inputs are required to complete the parameters in any given situation, rather than having to do a new calculation for every new situation. Using RL would solve the problem of errors and delays in communication because it makes the agent learn how to balance before it needs to rather than relying on calculations for every situation. This is done by running the RL algorithms in a simulation to teach the agent beforehand, and therefore removing any delays in communication or errors. This means that the agent does not have to rely on the speed of the calculations because it has already learnt how to balance on its own.

In this project I will use multiple RL algorithms in a physics-based balance control simulations and compare each algorithm to see which are better and worse at controlling balance. The simulation will involve a pole with a pivot point on the top which will allow the pole to move in three dimensions. The pole will start facing downwards and the RL algorithms will control the forces put upon the pole to try to get it to face upwards and hold it for as long as possible. The best algorithm will be the one which holds the pole up for the longest.

1.2 Aims and Objectives

Aim: To analyze and compare different reinforcement algorithms using a physics simulation.
Objectives:

- Acquire a publicly available physics simulation software which can collaborate with reinforcement learning algorithms
- Generate the pole-balancing test in the physics simulation
- Obtain three publicly available reinforcement learning algorithms connect them to the physics simulation
- Execute each algorithm and compare the results in a table showing time balanced

2 Time Plan

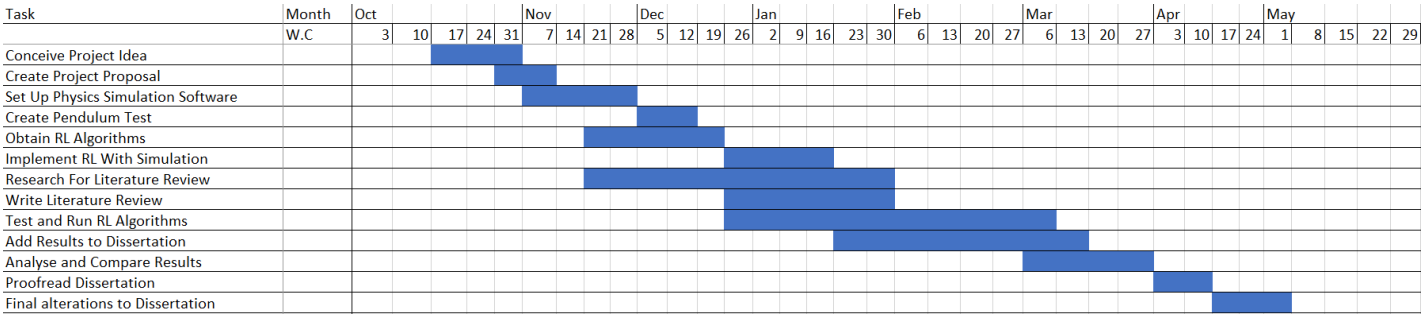

Fig. 1. Gantt chart showing time slots for each main task in the project.

3 Risk Analysis

One issue that may arise is if the physics simulation software I obtain is not able to produce what I want or is not supported anymore, meaning there will not be enough online resources or documentation needed to understand the program. This is quite

likely because many publicly available programs are no longer supported and most of them are only used for specific reasons and may not support the test which I want to do. This will only cause minor time loss as it will not be difficult to obtain another program as there are plenty of free physics software online. I will make sure this does not happen by researching each program before using them to see if they are still supported and are able to do run the pole-balancing test. I will also make sure I have 2 to 3 other programs I can use in case the one I am using is no longer suitable for me to use.

Another risk is that the physics simulation which I use may not be compatible with reinforcement learning and may not allow outside software to control it. This is not too likely as most physics simulators allows for outside software to control it and is usually made purposely for it. This will not cause a significant impact as I will only need to find a new physics simulator or different RL algorithms, all of which there are many online. I can mitigate this from happening by using a physics simulator that is purpose built for RL and encourages RL algorithms to be used on it. This will also help with usability as the documentation of the program will show how to use RL with it.

Another issue I may face is if the pole-balancing test does not produce satisfactory results with the RL algorithms and it is possible none of them will be able to learn how to hold the pole-balancing upright. However, this is not likely because the test only has three inputs and therefore should be simple for the algorithms to understand how to achieve the goal. If it does occur, then it may cause a minor time loss as I will have to create a new test to use which may take time to devise as well as implement. To mitigate this I will create other ideas for tests and implement them early on to see if they work in the simulation. This means that if the pole-balancing test fails then I will have others ready.

References

SUTTON, R. S. & BARTO, A. G. (2018). *Reinforcement learning: An introduction*, MIT press.

VOIGTLÄNDER, F., RAMADAN, A., EICHINGER, J., LENZ, C., PENSKY, D. & KNOLL. (2017), 5G for robotics: Ultra-low latency control of distributed robotic systems. In: *2017 International Symposium on Computer Science and Intelligent Controls (ISCSIC)*. IEEE.