the University of Bristol and the
University of the West of England,
Bristol

Dissertation Report for MSc Robotics

Deep Reinforcement Learning: Curriculum Design as Applied to Quadrupedal Locomotion

Ahmed Khalil

September 18, 2019

supervised by

Dr. Mark Hansen and Yiheng Zhu

Word Count: 13,656

Abstract

A sequence of chosen tasks, curriculum, can facilitate learning on difficult problems and improve performance. A curriculum may be manually or automatically designed. This project aims to explore manual and automatic curriculum design through applying them to the challenging problem of quadrupedal locomotion. Past works formulated the design problem as a Markov Decision Process (MDP) to be solved using Reinforcement Learning methodology. In this project Proximal Policy Optimization (PPO), the current state-of-the-art Deep RL algorithm, is used for the quadrupedal locomotion. Manually drafted curricula are tested for the modes of forwards and backwards curricula. Automatic curriculum generation, as inspired by the MDP problem formulation, is also explored using PPO. The experiments show that curriculum learning is best for improving performance on the easy tasks rather than for their harder ones (easy or hard refer to the tasks comprising a given curriculum). This justified the failure of automatic curriculum generation for designing a meaningful tasks trajectory, under the placed limitations. Moreover, for improving performance on the easy tasks, closely similar tasks deviate towards forwards mode (easy to hard), while less similar tasks opt to backwards mode (hard to easy) for improvements in performance.

1 Acknowledgments

I would like to thank both of my supervisors, Mark Hansen and Yiheng Zhu, for their incredible support and insight. I would also like to thank Kez Smithson Whitehead for his technical help and useful feedback. The PPO code used in this project is from the Stable Baseline RL library, a very well written package I should add. The Mujoco simulation environment is by the Mujoco company. Finally, I want to thank the Reinforcement Learning Reading Group at the University of Bristol for vastly cultivating my knowledge and interest in this field.

Contents

2 Introduction

2.1 Section Overview

The introduction commences by explaining Reinforcement Learning (RL). It then goes on to explain further conceptual details regarding curriculum design. Afterwards, justification for the choice of quadrupedal locomotion as the underlying problem is presented. The section also highlights the aims and objectives of this report. Detail is provided to the report outline sub-section to facilitate the reading of this report.

2.2 Reinforcement Learning

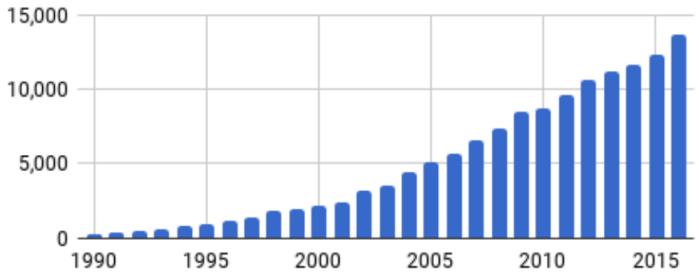


Figure 1: Rapid growth of the RL field. Publication volume vs years. (Plot and caption from Henderson *et al.* (2017))

Generally speaking, RL tries to embed the psychology of learning what to do in a computer. Therefore, it can be applied to almost any optimization problem and used to solve goal-driven tasks to amazing results, as shown by AlphaGo beating the world Go champion $4-1$ (Silver *et al.* (2016)). This has serious implications for the robotic industry as shown in Levine *et al.* (2018). Here a 7 degree-of-freedom robotic arm used Deep Reinforcement Learning (Deep RL) to learn hand-eye coordination with no spatial awareness. Nonetheless, there are yet many technical hurdles to overcome such as reward design and sample inefficiency. Thereby inspiring a rapid growth in research, as as demonstrated by Figure 1. Prior to moving forward, it is important to define a few key terms:

- **Agent** - the entity/network interacting with environment, learning, and taking actions.
- **Environment** - what the agent interacts with to generate experience.
- **State** - a representation of the environment at a certain time instant.
- **Action** - the move exacted by the agent on the environment, affecting it. It is a function of the learned policy.
- **Policy** - an agent's behavior, which determines the action taken based on a state observation.
- **Task** - the goal of an agent, or the problem it is attempting to solve.
- **Reward** - the incentive used to direct the agent's learning.

RL is a subcategory of Machine Learning (ML) revolving around agents that learn from experience generated by interacting with some sort of environment. RL is goal-driven, meaning an agent will have a distinct aim that it attempts to reach, whether that is playing Go or picking up an object using a robotic arm. This goal-driven nature allow agents to be incentivized using a reward/punishment system. Thus, when the agent observes a given state of the environment, it would use its learned policy to determine the best action and take it to maximize reward.

RL problems are modeled as Markov Decision Processes (MDPs), relying on the Markov property, which states that a future state is independent of the past given the present state. Figure 2 depicts the RL process modeled as an Markov Decision Process (MDP).

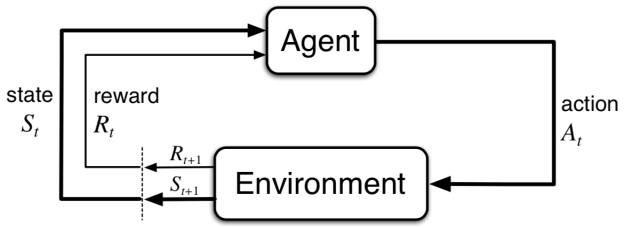


Figure 2: A schematic showing the agent interacting with the environment in a MDP (Sutton & Barto (2018)).

2.3 Curriculum Markov Decision Process

A curriculum of tasks may be designed for an agent to train upon to improve their performance on a desired final task. Curriculum design can be done either manually or automatically. Manual design usually requires domain specific knowledge to create a trajectory of tasks that guarantee improvement for the agent. However, automatic design is generally domain agnostic. The version of automatic design used in this report is inspired from Curriculum Markov Decision Process (CMDP), in which the problem is formulated as an MDP (Narvekar & Stone (2018)). A CMDP is just like an MDP, but for a higher level agent, for the curriculum itself. The version of CMDP in this report differs from that in Narvekar & Stone (2018) in two main aspects. First, in the paper the state space is the set of all possible policies the lower-level agent can showcase. However, in this report a state observation is a history of the past n leg lengths and higher-level actions. Second, more limitations are placed on the higher-level agent in this project, such as the limited discrete actions and the hard performance threshold for task transition. These are elaborated on further in the methodology section. Please note that the term CMDP as used in this report, takes on these differences to the one in Narvekar & Stone (2018).

The CMDP is better elaborated in Figure 3. To a higher-level agent, an action A_n creates a new task for the lower-level agent to train on. The state and reward reported to the higher-level agent are based on the performance of the lower-level. Note that when the lower-level agent interacts with a task, large white arrow, it undergoes its own MDP such as depicted in Figure 2. Then the higher-level agent takes another action to create a new task for the lower-level agent to train on. Since in this project transfer occurs of the value and policy networks of the actor-critic agent (explained in greater detail in the literature review), the agent remains essentially the same. This is reflected in Figure 3.

2.4 Problem Choice: Quadrupedal Locomotion

There are several for choosing quadrupedal locomotion as the underlying task for this report. First, it represents an area in robotics with numerous potential applications. Quadrupedal robots are able to traverse multiple types of environments ranging from

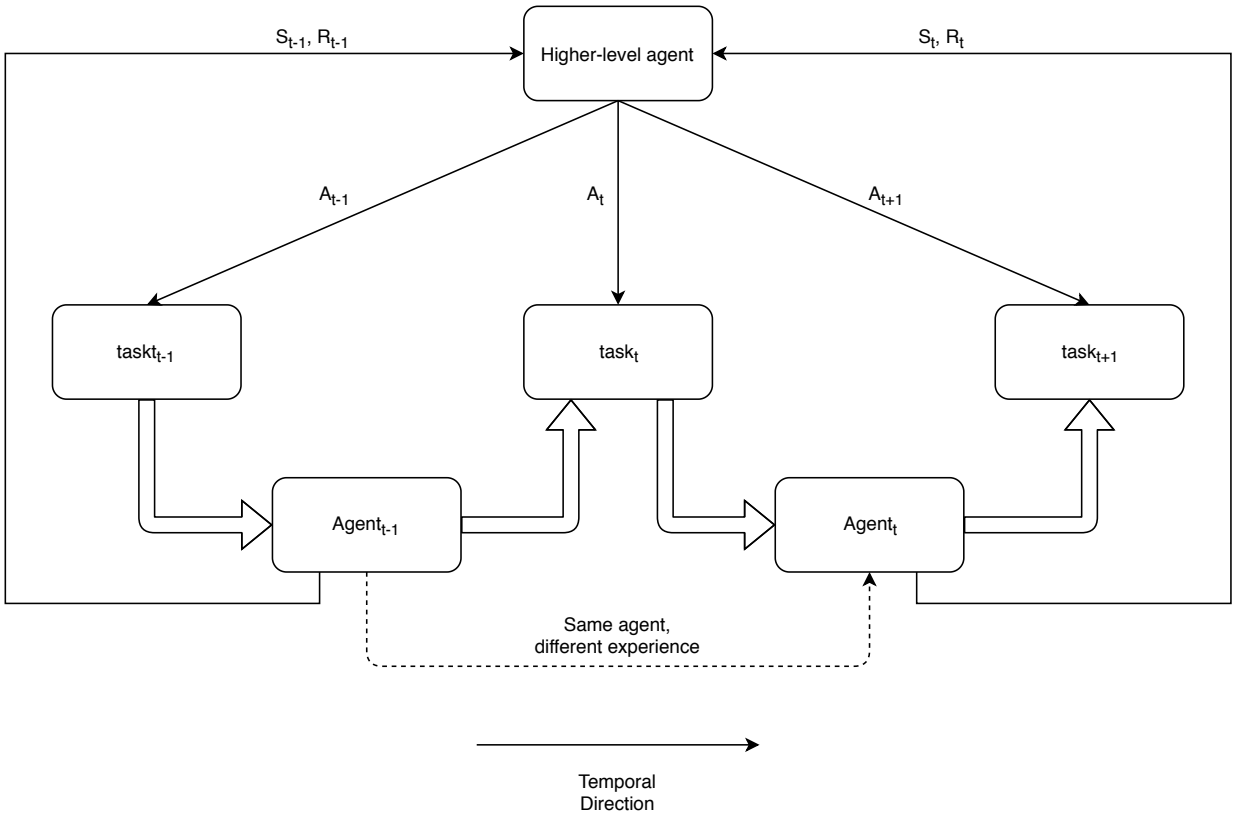


Figure 3: A schematic of a CMDP. The higher-level agent takes an action by creating a new task for the lower-level agent to train on. Then the lower-level agent returns performance metrics for the higher-level agent to obtain a current state and reward observation. This allows it to build the next appropriate task for the lower-level agent to be transfered to and learned on.

woodlands to rocky terrain (Schaeffer & Lindstedt (2013)). They are also able to climb up and down stairs, as opposed to wheeled robots, which makes them very lucrative to the assisted living robotics market. Yet the most likely reason wheeled-robots are the most common, is the current poor locomotion control of quadrupedal robots (Campion & Chung (2008)). Although, it was demonstrated that Deep RL can assist in that domain, much more improvement is still needed (Whitehead (2018)).

Second, this is a highly-dimensional, tough problem to solve. Therefore, it is prime for testing manual and automatic curriculum design. Finally, the nature of this problem allows for ease of curriculum creation. Meaning that slight changes are required to create a new task, in this case merely changing the leg length. For other simpler

problems such as Grid World, or gaming environments, it is quite difficult to create a meaningful array of tasks for experimentation. Figure 4 shows the adapted version of Figure 3 to quadrupedal locomotion.

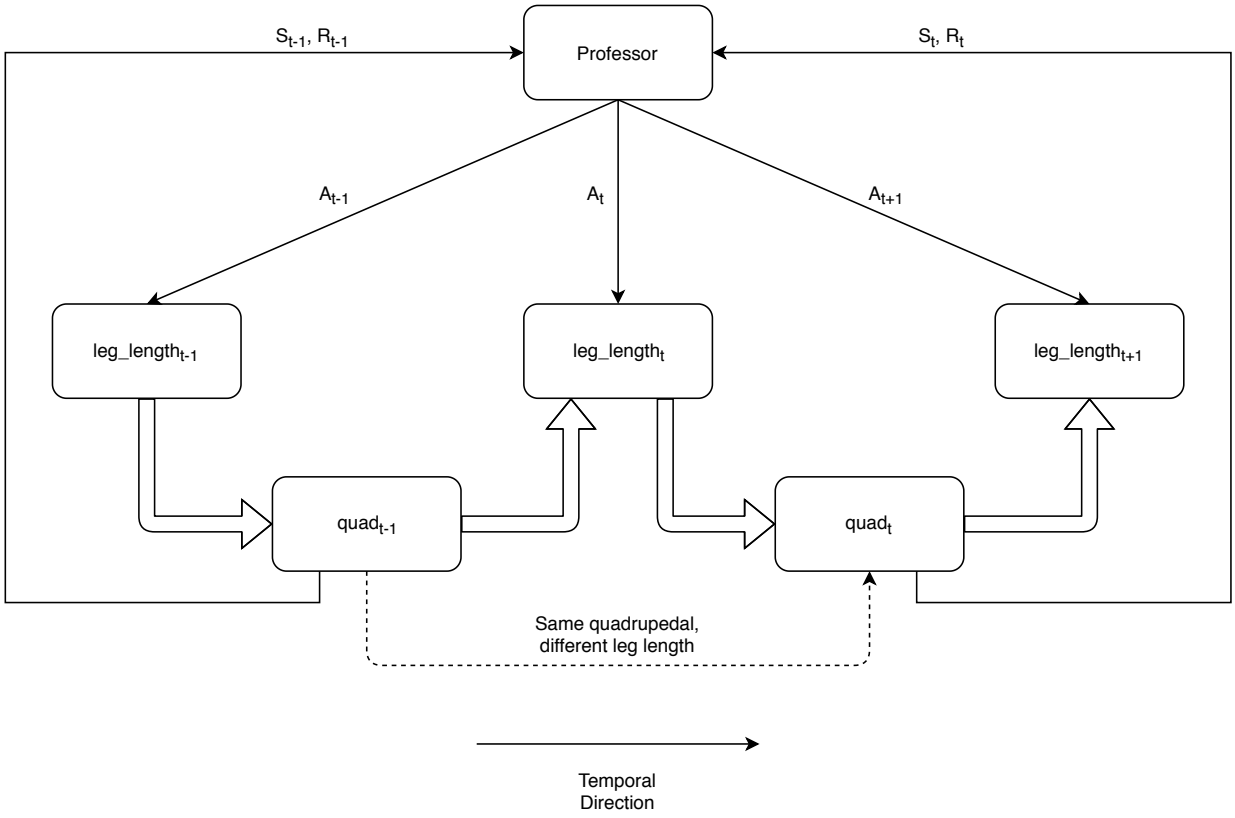


Figure 4: A schematic of a the automatic curriculum generation adapted to the quadrupedal CMDP.

2.5 Aims & Objectives

This project aimed to explore Curriculum Learning (CL) design in a challenging problem setting using state-of-the-art Deep RL techniques. First, a two custom environments are built: one for the quadrupedal, as the lower-level agent, and another for Automated Curriculum Generation (ACG) as a CMDP. Second, the ACG environment is deployed using Proximal Policy Optimization (PPO) for learning and multi-processed PPO on the quadrupedal learning. Upon obtaining findings for those experiments, further experimentation was necessary on manual curriculum design. This was done by manually specifying the task leg lengths and running the quadrupedal learning environment.

2.6 Report Outline

The report is segmented into 6 more sections summarized below.

Section 2 is the literature review performed for this project. Particular emphasis is place on Deep RL and its various techniques. It also explores Transfer Learning (TL) and CL, specifically with regards to manual and automatic curriculum design. Moreover, quadrupedal locomotion in literature is evaluated.

Section 3 details the methodology used for this project. There are 3 main parts here:

- The first section examines the environments built for experimentation. There are two types, the lower-level quadrupedal Mujoco environment, and the CMDP formulated ACG. Particular attention is payed to the logic behind the experiment setting for the ACG problem.
- Afterwards the logic of the manually designed curricula is evaluated. This takes a look at the hypotheses created for testing, the curricula used, and the reasoning behind them.
- Finally, the means of training and logging are probed. ACG was trained using a single environment PPO implementation, while the quadrupedal itself was trained using a multi-threaded PPO implementation.

Section 4 deeply analyzes the results for the manual design experiments. Note that no quantitative results for ACG are presented in this report due to reasons presented

in the discussion. The results section is segmented into 4 main parts:

- Experiments for the no curriculum cases, so each leg length is held constant for the training time of a full curriculum. It was realized that longer lengths obtain higher cumulative rewards, but higher variation in performance.
- Curriculum experiments on leg lengths that are closely spaced in magnitude. This showed that forward mode curricula achieved the best performances, but only when compared to tabula rasa (no curriculum, constant leg length) of the lower leg length of the curriculum.
- Curriculum experiments on leg lengths that are not closely spaced in magnitude. Here backwards mode curricula performed best, but similar to the previous subsection only when compared to tabula rasa (no curriculum, constant leg length) on the easier task in the curriculum.
- Examination of the overall trends in both manual design experiments.

Section 5 discusses the results and provides a potential model for the knowledge space of the quadrupedal locomotion problem studied here. It also takes a keen look at the reasons behind the failure of the ACG experiments for designing meaningful curricula. Afterwards, it discusses the potential of CL in solving the harder problems, and what can be done regarding that.

Section 6 summarizes the future works planned after this project, which are chosen to tackle further confirm or deny the results presented in this report in addition to the proposed knowledge space model. Works include more experimentation with the manual curriculum design in terms of training time, curriculum length, curriculum modes and configurations, and problem setting.

Section 7 is the final section of this report. It concludes that curriculum design is still not powerful enough to tackle difficult problems. This finding is mainly due to the manual experiments and their almost exclusive success on the easy tasks rather than the difficult ones.

2.7 Section Summary

RL deals with learning what to do given a task and an environment with which to interact. It relies on modeling problems as anMDP, which is based on the Markov property. ACG is modeled in this report as a CMDP, which is presented in this section. A Cur-

riculum Markov Decision Processes (CMDPs) is merely the extension of a MDP for a higher level-agent. The quadrupedal locomotion problem is chosen as the underlying problem for this project as it is a difficult one, has many potential applications, and allows for easy task creation. The introduction also presents the aims and objectives to study CL and designing curricula, using PPO, on quadrupedal locomotion. The report outline is showcased in great detail.

3 Literature Review

3.1 Section Overview

This section initiates by exploring Deep RL literature regarding its various algorithms types, namely Value-based Methods (VBMs), Policy-based Methods (PBMs), and Actor-Critic Methods. Afterwards, it examines TL which is a vital concept to this project as it provides the basis for CL. In the CL subsection, emphasis is placed on manual and automatic curriculum design. Finally, the problem of choice in this project, quadrupedal locomotion, is reviewed. Please note that this literature review is heavily adapted from the Dissertation Research Plan submitted for the Robotic Research Plan module at the University of Bristol (Khalil (2019)).

3.2 Deep Reinforcement Learning

Deep RL is the extension of RL with Deep Learning (DL). As a branch of ML, RL deals mainly with learning from experience via an agent, or many, interacting with an environment. The aim of RL techniques is to solve intricate tasks by learning optimal action policies through trial and error. One of the larger issues that faced RL is the dimensionality curse, where complex or continuous problems become near impossible to solve due to their enormous state spaces. This is overcome when arming RL with the function approximation capabilities of Supervised DL. Model-free Deep RL methods can be segmented into Value-based Methods VBMs and Policy-based Methods PBMs, both of which form actor-critic structures when fused.

3.2.1 Value Based Methods

VBMs possess certain advantageous traits over PBMs, they are mostly simpler in their application, and are more stable. Furthermore, they heavily employ Temporal Difference (TD) techniques, which allow for fast, online learning. Mnih *et al.* (2013) is a high impact paper in the Deep RL community, widely accredited to igniting the current rise in research. In this paper, a Deep Q-Learning Network (DQN) maintained the same network architecture and parameters and still learned to play 7 different Atari 2600 games. This was the first paper to extend model-free, VBMs with DL by fixing the

data mismatch problem, since RL and DL are different in nature. A feat made possible through frame concatenation and experience replay buffers. This work was further fleshed out in Mnih *et al.* (2015) by comparing DQN against the best RL algorithms in literature as well as human performance. Against the algorithms, DQN achieved higher performance on 43 of the 49 games. However, against humans DQN was only able to achieve 75% their performance, on most games. These experiments showed strong improvement in performance for Deep RL over RL, almost matching human-level prowess, and most importantly the ability to generalize across multiple tasks.

3.2.2 Policy Based Methods

VBMs are powerful methods but not without weaknesses. For instance, they lack the capacity to tackle highly-dimensional, continuous problem. Primarily due to the curse of dimensionality as such problems must be discretized and stored in tabular form. Moreover, VBMs are only capable of outputting deterministic policies. Thereby preventing meaningful analysis of equivalent policies or states that appear to be identical but are inherently different. Deterministic policies also cannot present the degrees of their action severity, which is important for complex problems. PBMs do not face these weaknesses and are able to learn stochastic policies, enabling them to produce elaborate behaviors. Furthermore, PBMs are able to appropriately balance exploration and exploitation by asymptotically advancing towards deterministic policies. Most importantly, PBMs do not suffer from the same dimensionality curse as VBMs, enabling them to take on continuous and highly-dimensional problems (Sutton & Barto (2018)).

PBMs are based on the REINFORCE rule, the performance of an agent is directly correlated to its policy gradient (Williams (1992)). The initial intuition was to use vanilla gradient descent to find the optimal policy. Nonetheless, this proved to be sample inefficient due to the effects of slight network parameter shifts on the policy space. Given the severity of this effect on gradient descent, large steps are risky. As an analogy, picture climbing a hill with one side being a cliff. A small step forward may cause the climber to reach a slightly higher or lower altitude, but a step in the wrong direction (cliff-side) will cause a sudden drastic drop in altitude. To overcome this issue, Trust Region Policy Optimization (TRPO), an on-policy gradient method, uses KL-Divergence to limit gradient shifts to within a trust region. Simply put, TRPO

creates a guarantee of policy improvement as long as the shift stays within the trust region. This allows for large steps to be taken, accelerating learning and increasing sample efficiency as a consequence (Schulman *et al.* (2015)). That being said, TRPO is difficult to implement and incompatible with networks utilizing parameter sharing or noise augmentation (Schulman *et al.* (2017)). PPO does not posses these limitations, as it is based on first-order optimization. It is also similar in performance to TRPO. Moreover, PPO only differs from vanilla policy gradient optimization in a few lines of code, significantly facilitating its ease of implementation (Schulman *et al.* (2017)).

3.2.3 Actor-Critic Methods

PPO is an actor-critic method, unlike TRPO which is purely policy based. Actor-critics benefit from the combined advantages of VBMs and PBMs. Asynchronous Advantage Actor Critic (A3C) accelerates learning by generating multiple environments with their own agents, effectively collecting experience at a higher rate (Mnih *et al.* (2016)). The synchronous form of A3C is Advantage Actor Critic (A2C). Despite their differences, they both generally perform similarly (Schulman *et al.* (2017)) (Wu *et al.* (2017a)). PPO showcased better performances when compared with A2C (Schulman *et al.* (2017)).

Actor-critic methods also include Actor Critic using Kronecker-factored Trust Region (ACKTR) and Actor-Critic with Experience Replay (ACER). Similar to TRPO, ACKTR leverages trust region optimization, but it also combines it with distributed Kronecker factorization. This provides ACKTR with the same advantages of TRPO in addition to higher sample efficiency and increased scalability (Wu *et al.* (2017b)). Wang *et al.* (2016) states that ACER integrates 'truncated importance sampling with bias correction, stochastic dueling networks and an efficient trust region policy optimization method.' This allows ACER the advantages of stability, sample efficiency, and high performance on continuous problems, and discrete ones. To provide a benchmark for the aforementioned actor-critic methods, Wu *et al.* (2017a) goes on to apply PPO, ACER, ACKTR, and A2C to 49 Atari games. The results demonstrate that A2C, and A3C given their similarity, possessed the worst performance. ACER appeared to surpass ACKTR and PPO in a small number of games, but in the remaining majority the best performance were interchanging between ACKTR and PPO. It is important

to note that in 6 games PPO was the only method to demonstrate learning.

3.3 Transfer Learning

TL pertains to the passing of knowledge between two agents to accelerate learning (Lazaric (2012)). If two tasks are similar and an agent has trained on one, then it may be beneficial to use that learned policy as the initial policy for the second task instead of a random initial policy. This concept is especially beneficial if the second task is much more difficult than the first. In most literature the first task is named the *source* task, while the second is the *target* task. Note that to maintain similarity with other work, value-function transfer is the mode of TL used here (Taylor *et al.* (2007)). Given that PPO is an actor-critic method, the transfer will occur for its policy and value functions.

Often when learning a task, an agent would reject knowledge if it is not directly correlated to the task at hand. An example would be a robotic-arm trying to move a puck to a marked location. Learning to move the puck to another, entirely different, location is certainly not the goal, but there is clearly some overlap between both tasks. Rather than forgetting the experience, Hindsight Experience Replay (HER) tricks the agent into believing this was the goal all along, to learn from it (Andrychowicz *et al.* (2017)). This form of TL is intrinsic as it occurs within the same task rather than across tasks. It is worth mentioning given its effects in enhancing performance and sample efficiency for sparse, goal-driven tasks.

By placing many intermediate tasks between the *source* and *target* tasks, a curriculum can be created. Essentially, CL is the elongated concept of TL as discussed in the upcoming sub-section.

3.4 Curriculum Learning

Taking inspiration from the school educational system, CL is based on the incremental passing, and accumulation, of knowledge to learn a complex task/concept. A student must first learn simple arithmetic, and many other concepts that build on it, prior to eventually tackling multi-variable calculus. Skinner (1958) took this concept and applied it to animal training. In the majority of cases where CL was applied to Supervised Learning (SL), the networks converged lower local minima in addition to demonstrating

higher generalization capabilities (Bengio *et al.* (2009)). It is important to note that the goal of CL in SL is to enhance performance on all tasks, but in RL the main task of concern is that of the *target*. Narvekar *et al.* (2016) were the first to apply CL to RL, demonstrating significant betterment in performance.

3.4.1 Manual Curriculum Design

Curriculum design is highly challenging and remains an open problem. This is further emphasized when the designers are not experts on the task (Peng *et al.* (2016)). There are two main hurdles a designer must overcome. First, ensure that the curriculum actually promotes policy improvement with regards to the *target* task. Second, not fall within *task space canyons*, where seemingly similar tasks actually possess drastically different policies. Thus upon transfer, catastrophic forgetting may occur, or lead policy divergence. It is for those reasons that ACG is becoming an attractive alternative to manual curriculum design. In this case an agent takes on the meta task of designing the curriculum itself.

3.4.2 Automatic Curriculum Generation

Built on an already difficult problem, ACG follows in complexity. This led to many works taking certain measures in simplifying the problem for their research. For instance, Florensa *et al.* (2017) limits the task spectrum by expanding out from the *target* task to create a sequence of tasks n states away. Thus creating tasks identical to the *target* task, but of diverse initial states. Quite differently, Held *et al.* (2017) generated tasks by altering the final task itself. This is due to the nature of their research, which is studying the range of tasks a trained agent can tackle. A further example is Riedmiller *et al.* (2018) and Sukhbaatar *et al.* (2017), who simply the problem by only altering the reward function.

Other works in literature simplified ACG by utilizing domain knowledge to generate their tasks. Svetlik *et al.* (2017) achieve this by using their own heuristic, which tests the relevancy of any task to the *target*, in combination with task descriptors (domain knowledge) to create directed acyclic graphs. Silva & Costa (2018) utilize object-oriented task descriptors to create curricula from the simplified versions of the *target*. This method relies on less domain knowledge than Svetlik *et al.* (2017). Narvekar

et al. (2017) base their work on Narvekar *et al.* (2016) and formulated the problems as MDPs. This is of distinct interest, since now ACG can be treated as a standard RL problem. Similarly, Matiisen *et al.* (2017) frame CL as a Partially Observable Markov Decision Process (POMDP).

3.4.3 Automatic Curriculum Generation as CMDP

Although Narvekar *et al.* (2017) and Matiisen *et al.* (2017) created formulations for learning the ACG problem, neither used them for learning by RL methods. Instead, they leveraged heuristics to create a single curriculum. It was Narvekar & Stone (2018) who showcased, through empirical analysis, that ACG can be formatted as a CMDP and learned upon. This new formulation had two distinct advantages over past literature. First, it is not reliant on the *source* tasks, but is indifferent to them. Second, it is agnostic to the underlying problem the agent is attempting to solve. In other words, it does not rely on nor need domain specific knowledge. Moreover, the experiments in this paper demonstrated that this method is is also invariant to the transfer learning method used, problem action space, and CMDP representation. Nevertheless, Narvekar & Stone (2018) concluded that solving a CMDP generally far exceeds the training time of direct learning on the *target* task. It is imperative to recognize that this work relied heftily on VBMs for the RL methodology. It also relied on tile coding for the state representation, which suffers greatly of the curse of dimensionality (Shannon & Grzes (2018)) as opposed to other methods such as Multilayer Perceptron (MLP).

3.5 Quadrupedal Locomotion

Quadrupedal locomotion is a difficult, goal-based, highly-dimensional problem. It was demonstrated that through modeling the problem as an MDP driven by simple rewards, complex behaviors can emerge (Heess *et al.* (2017)). In this paper, various PBMs and actor-critic methods, such as TRPO, A3C, and PPO, were applied to quadruped locomotion, amongst other tasks. PPO was specifically used on the quadrupedal on multiple easy and hard courses. The difference in difficulty here being the hurdle heights. Some agents were trained through handcrafted curricula, that got harder in hurdle difficulty. Those demonstrated quicker learning.

Expounding on this work, Whitehead (2018) used PPO to teach a quadrupedal how to walk from scratch in Mujoco, motivated by visual stimulation alone. The agent was then transferred onto a real robot for testing, successfully traveling 20% of the target distance. There was also a clear simulation-reality gap in performance as the robot could not steadily maintain a straight line when traveling. The demonstration can be found at this link: `https://www.youtube.com/watch?v=JJlCODV9s7I`

3.6 Section Summary

From the reviewed papers it is clear that actor-critic methods are the current state-of-the-art Deep RL techniques. Most specifically PPO, which appears to outperform most algorithms compared against it. Furthermore, this section took a close look at CL, and the problems concerning its design. Whilst manual design is the simplest path to take, it usually requires heavy reliance on expert domain knowledge. On the other hand, the more domain agnostic path of ACG is ridden with high complexity that requires many simplifications to get it working. A promising new technique is the MDP formulation of ACG. Literature shows its success in learning curricula using VBMs and code tiling state representation. Although, it did require far more samples than direct learning. Quadrupedal locomotion was also reviewed in this section, revealing its success in curriculum formatted settings through the application of PPO in the Mujoco simulation environment.

In this report, manual and automatic curriculum design are explored. This is done by focusing on quadrupedal locomotion as the underlying problem to be learned and solved. All learning is done using PPO, as it is the current state-of-the-art Deep RL algorithm. Manual curriculum design is done without much reliance on domain knowledge. Automatic curriculum design is performed through formulating the design problem as CMDP, as inspired from Narvekar & Stone (2018).

4 Methodology

4.1 Section Overview

This section details the logic and decisions behind the experiments conducted in this project. It also describes the environments created as well as the training and logging data structure. The methodology section is split into:

- **Environments** - the environments PPO was applied to. The first is the quadrupedal Mujoco simulation and the second is an environment specific for ACG.
- **Manual Curriculum Design** - the hypotheses and logic behind the curricula designed for experimentation.
- **Training & Logging** - the training settings and save files data structure.

It is imperative to note that the results section is solely based on the Manual Curriculum Design sub-section. Whilst the Automatic Curriculum Generation sub-section is described to highlight certain elements in the Discussion. No numerical results for ACG are presented in this report, as it failed to created meaningful curricula. The reasons are explored further in the discussion. All the code written for this project can be found at this link: `https://github.com/AGKhalil/diss_quad/tree/master`

4.2 Environments

The environments in this project were custom made and written to fit the OpenAI *gym* environments. This facilitated experimenting with different Deep RL packages such as $stable-baselines$, which is the chosen package for all experiments detailed in this report. All *gym* environments must contain 5 methods, those are:

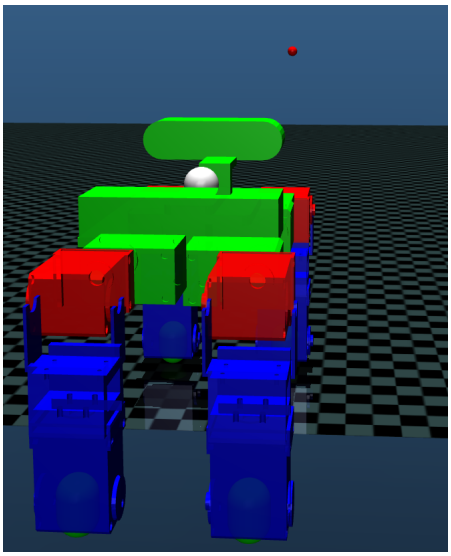
- *init* - initializes the environment and its action space, observation space, and initial observation.
- *step* - takes an action as an input and accordingly modifies the environment. It then returns a new observation of the environment state, reward obtained from taking the action, whether or not a terminal state has been reached, and additional info (for reference).
- *reset* - resets the environment to the initial state and returns the initial observation.

- *render* - renders a visual representation of the environment.
- *close* - closes the environment.

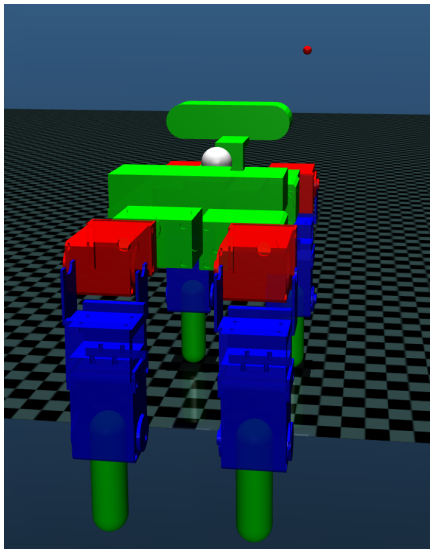
For further information regarding OpenAI Gym please refer to this link: `https://github.com/openai/gym`. For more details on building custom *gym* environments please refer to this link: `https://github.com/openai/gym/blob/master/docs/creating-environments.md`

4.2.1 Quadrupedal - Mujoco

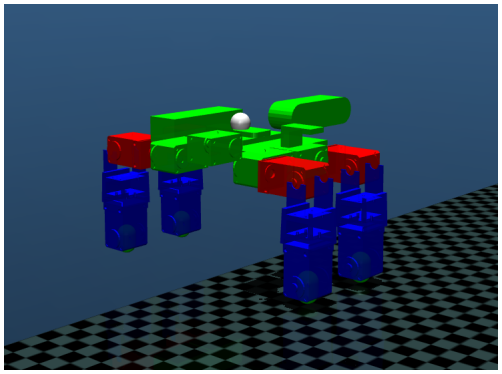
This environment, named QuadEnv consists of a quadrupedal that is initialized standing on its four legs up right, facing the goal (red ball as in Figures 5a and 5b). The agent (quadrupedal) is positively rewarded for its velocity towards the red ball, by making the reward equal to its x-velocity towards the goal. An action consists of the torques for each of the 12 joints on the agent (3 per leg). An episode terminates/resets when the agent either falls or if the agent takes 1000 steps. An observation consists of 12 joint speeds, one per joint, in rad/s. The agent is placed an arbitrarily long distance away from the goal to give it room for experimentation. The sub-figures in Figure 4.2.1 show images of the simulation. The code for this environment can be found at: `https://github.com/AGKhalil/gym-real`



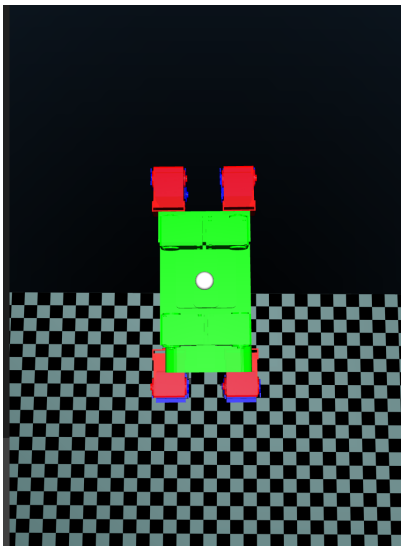
(a) Quadrupedal on the leg length 0.1, facing the goal.



(b) Quadrupedal on the leg length 0.5, facing the goal.



(c) View showing the front of the quadrupedal.



(d) Top view of quadrupedal.

Figure 5: Images of the quadrupedal Mujoco environment.

4.2.2 Automatic Curriculum Generation

In order to attempt ACG, the CLEnv environment is created. Here, ACG is formulated as a CMDP to which PPO (with an MLP for state representation) is applied for learning. This can be thought of as a higher-level agent (professor) designing a learning curriculum for the quadrupedal (student) to accelerate learning from short leg lengths (simple tasks) to longer ones (difficult tasks). Essentially, the professor is tasked with designing a sequence of leg lengths to train the student. Thus every action taken by the professor results in the modification of the quadrupedal leg length, re-setting QuadEnv, and then training the student. Whenever a new step is taken, the student does not start training from scratch, but it uses the trained policy from the previous instance in the curriculum as an initial policy. To a professor, an episode always initiates with the shortest leg length and always terminates with the longest leg length. Fixing the initial and final states provides a tangible goal for the professor.

When the professor changes the leg length, the student starts training for n time-steps, which is a small number of time-steps. After n steps, the student cumulative reward is evaluated. If it is below a certain threshold, the student trains for n more steps. This keeps on going until either performance surpasses the threshold or n_{max} is reached. At this point the professor can evaluate the state observation and decide

on the next action. There is a cumulative counter that keeps track of the amount of n steps per leg length. The professor reward is its inverse. For example, at some point in the curriculum the leg length is 0.5 and the cumulative reward threshold is 300. After n steps, the student has a cumulative reward of 125, thus it continues training for n more steps. After multiple occurrences of this, the student reaches the 300 threshold in $5n$ time-steps. Therefore the professor reward is $\frac{1}{5n}$. This incentivizes the professor to design a better curriculum on the next episode so when leg length 0.5 is reached the student reaches the threshold faster, thereby maximizing the professor reward. The environment is designed that way to bypass the fixed training time restriction of the stable−baselines library, and to provide a relevant reward criteria for the curriculum design.

When a professor takes an action it can only increment, decrement, or maintain the current leg length by a fixed delta. This limitation in the action space severely reduces its size. Furthermore, an episode always terminates after 10 steps, with the final instance being the longest leg length. This is done to try and nudge the professor into modifying the curriculum for the improving training on the longest length. It was also to accommodate the limited resources of this project. Simply allowing the professor to create an arbitrarily long curriculum until reaching the longest leg length took tremendous training time. The observation space for this environment comprised of the past x actions and their respective leg lengths.

The code for this environment can be found at: `https://github.com/AGKhalil/gym-cl`

4.3 Manual Curriculum Design

Suppose a highly challenging task A, which requires certain knowledge K_A to solve, is given. Furthermore, suppose a simpler version of task A named task B, with its own knowledge K_B, exists. If an agent learns to solve task B by obtaining knowledge K_B, then using K_B as an initial starting point for acquiring K_A should result in better performance on task A than when using a random initial policy. This is the hypothesis used in the experiments done in this dissertation report. To explore the effect of CL in complex, continuous, highly-dimensional problems, manual CL is implemented on quadrupedal locomotion within the Mujoco environment. Learning to walk is a complex

task in itself, but it gets progressively more difficult as height increases. An increase in leg length results in a higher Center-of-Mass (CoM) for the quadrupedal, which means an increased sense of balance is required to avoid falling over. This is the motivation behind the curricula used in these experiments. It is hypothesized in this report that any curriculum containing a given task, will improve performance for that given task as compared to its no curriculum case. This takes the form of 4 proposed hypotheses as explained below:

1. $Hyp - A$ - knowledge of how to solve a complex task, with policy initiated from a simpler one, will lead to better performance (cumulative reward) on the complex task than if the complex task is learned from scratch
2. $Hyp - B$ - knowledge of how to solve a complex task, with policy initiated from a simpler one, will lead to better performance (cumulative reward) on the simpler task than if the simple task is learned from scratch
3. $Hyp - C$ - knowledge of how to solve a simple task, with policy initiated from a more complex one, will lead to better performance (cumulative reward) on the simpler task than if the simple task is learned from scratch
4. $Hyp - D$ - knowledge of how to solve a simple task, with policy initiated from a more complex one, will lead to better performance (cumulative reward) on the complex task than if the complex task is learned from scratch

To put the 4 hypotheses in a more concise format refer to Table 1.

Name	**Hypothesis**
$Hyp - A$	Transferring from K_n to train K_{n+1} results in better reward accumulation than K_{n+1} from scratch
$Hyp - B$	Transferring from K_n to train K_{n+1} results in better reward accumulation than K_n from scratch
$Hyp - C$	Transferring from K_{n+1} to train K_n results in better reward accumulation than K_n from scratch
$Hyp - D$	Transferring from K_{n+1} to train K_n results in better reward accumulation than K_{n+1} from scratch

Table 1: All 4 hypotheses tested in this project.

To test all hypotheses, multiple curricula are manually designed and trained. A single curriculum consists of two training instances with a specific leg length assigned to each instance. Note that all 4 legs have the same lengths in all experiments. Two lists of possible leg lengths are used. The first is list $A-[0.1, 0.2, 0.3, 0.4]$ and the second is list $B-[0.1, 0.3, 0.5, 0.7]$. The main difference between both lists is the spread of leg lengths. List A has a fixed increment of 0.1 and list B 0.2. This is done to test the effect of larger gaps in difficulty between the curriculum instances and to have a wider range of difficulties for experimentation. There are 4 possible leg lengths in each list which allows for 12 possible curriculum permutations per list, amounting to 24 curricula as demonstrated in the Results section. For proper comparisons baseline curricula were also generated, which had constant leg lengths throughout the 2 training instances. PPO is used for learning on all agents.

4.4 Training & Logging

The ACG experiments were conducted using PPO, without multiple environment vectorization, as the lower level agent used multiple vectorization of PPO. The remainder of this section pertains to the Manual Curriculum Design experiments, which relied only on the lower-level agent, quadrupedal locomotion in Mujoco. Each instance of the agent run for 1000 save checkpoints. Between any two consecutive checkpoints is 2560 time-steps; therefore, each agent trains for 2560000 time steps. Training is done using the $stable-baselines$ library, which leverages multi-processing to train on multiple CPUs, 20 in this case. All training was performed using PPO, PPO2 in $stable-baselines$. For further details on $stable-baselines$, please refer to the documentation at: `https://stable-baselines.readthedocs.io/en/master/index.html`.

Figure 4.4 shows the logging file structure. Under tf_{save} every model instance would be saved in its own folder, where the folder name is the model name, which is the time-stamp of its creation. Within those folders 2 files get created every checkpoint, marked by their checkpoint time-step. Those 2 save the model hyper-parameters and network architecture as a pickle file for easy loading, and the training cumulative reward vs time in a csv file. There is also a *checkpoints.csv* file which keeps record of all the checkpoint save names. This file is what allows for easy access to loading the models or creating their plots.

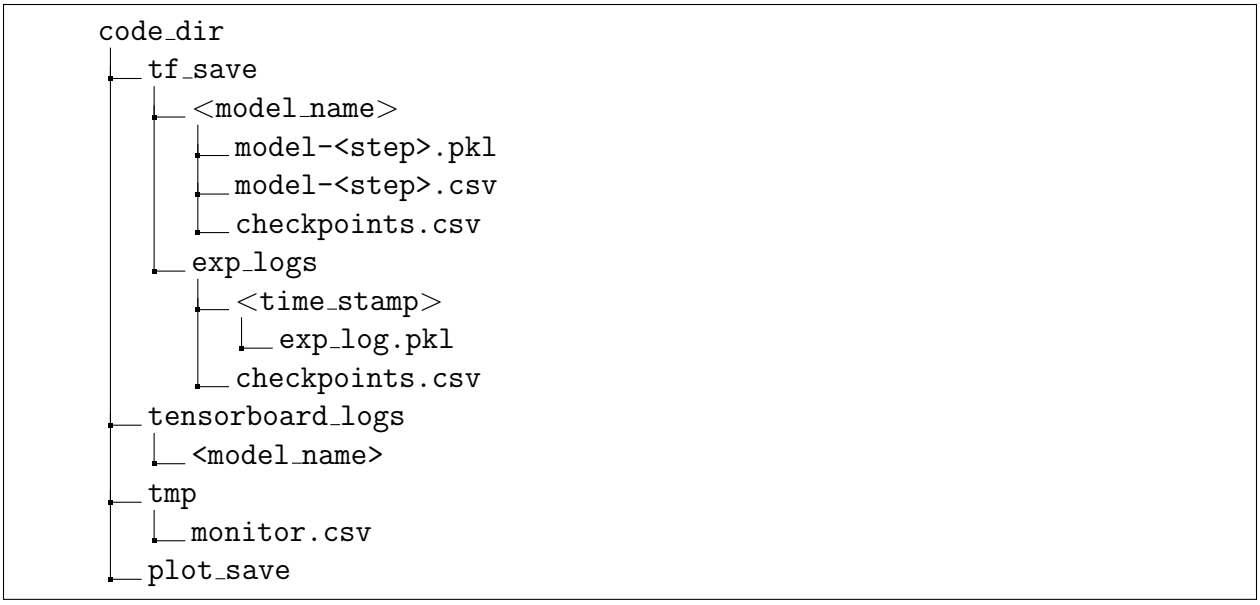


Figure 6: File structure of logging files.

The *exp_log* sub-folder is where a detailed account of the experiments is kept. Every checkpoint a sub-folder of the current time-stamp, including *exp_log.pkl*. This pickle file includes a dictionary detailing the experiment number, leg lengths curriculum sequence, and their respective model names. Each dictionary entry has all its respective seeds.

The *Tensorboard* logs folder is where training details and plots are saved for online visualization. The *tmp* folder is used for temporary saves of the data as they get permanently logged. Finally, *plot_save* is where all the plots created, by a different script, are saved.

4.5 Section Summary

Two custom Python OpenAI *gym* environments are written for the experiments. One for applying PPO to a quadrupedal learning to walk, the other for ACG as formulated as a CMDP. The manually designed curricula were outlined in such a way as to test the 4 hypotheses presented in Table 1. Training was performed using $stable-baselines$ with a detailed logging structure. All experiments, manual and automatic, utilized PPO for learning.

5 Results

5.1 Section Overview

All results here are only for the Manual Curriculum Design experiments. The ACG failed to build any meaningful curricula, but the Manual experiments shed keen insight as to why. This is discussed in detail in the Discussion section.

All curricula have been limited to two leg lengths, meaning two training instances. 24 leg length permutations were tested, with each permutation run for 5 random seeds. Each leg length instance was trained for roughly $2,500,00$ time steps. All episodes, for the quadrupedal locomotion, have a fixed maximum length of 1000 time steps. Training was completed on 20 CPUs, utilizing the multiprocessing capabilities of the $stable-baselines$ library, and took roughly 2 days and 6 hours. Checkpoint saves were performed every 5000 training time steps for the learning curves and model instances. $Tensorboard$ logs were also kept for all models.

The 24 curricula were created through permuting the following list of leg lengths: list $A-[0.1, 0.2, 0.3, 0.4]$ and list $B-[0.1, 0.3, 0.5, 0.7]$. Please note that for ease of comparison, all the figures of the same type have the same fixed axes limits. Furthermore, all plots have two dashed gray vertical lines which represent the end of each leg length training instance. Each performance (cumulative reward) curve represents the mean of the 5 seed moving averages per curriculum, enveloped in their maximum and minimum values. Please note that a curve envelope, which is a term used throughout the report, is the max and min edges encapsulating the curve itself. Each moving average is calculated with a window of 30 data points for smoothing.

The results for each experiment are analyzed in of themselves as well as against the proposed 4 hypotheses. A curve passes its corresponding hypothesis if it out performs its respective tabula rasa. The term tabula rasa refers to runs where no changes were made (blank slate), a constant leg length in this project. The final sub-section of the results takes a closer look at the results for each hypothesis.

The results structure are as follows:

- Baseline Learning Curves (tabula rasa) - cumulative reward and variation plots of the tabula rasa (no curriculum) runs.
- List A Experiments, Small Leg Length Spread - this includes plots for cumulative

reward and variation for each leg length in list A.

- List B Experiments, Large Leg Length Spread - this includes plots for cumulative reward and variation for each leg length in list B.
- Forwards and Backwards Curriculum Learning - analyzing the hypotheses acceptance/rejection percentages and overall patterns.

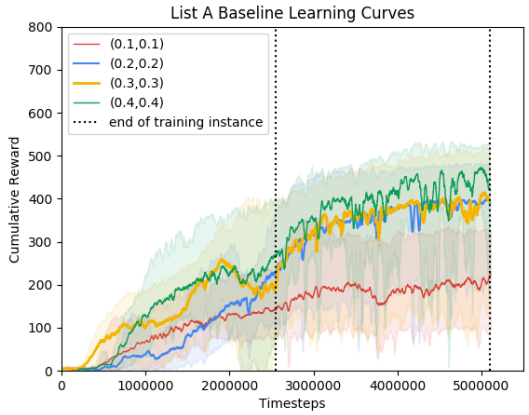
5.2 Baseline Learning Curves (tabula rasa)

Since all curricula permutations were of length 2, baseline runs were performed by fixing the leg length of an agent for two training instances. This created 6 learning curve plots (8 in total including repetitions), one for each distinct leg length in lists A and B (Figures 7a and 7b respectively).

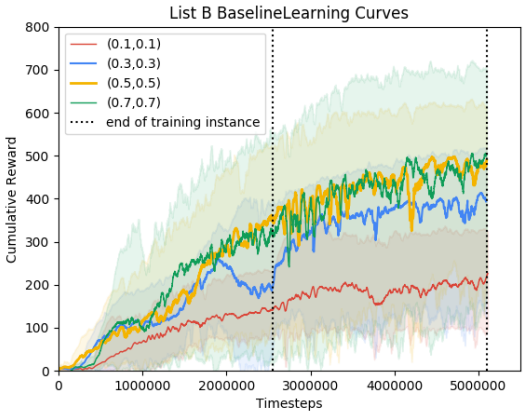
Both figures show that the shortest leg length (0.1) has the lowest cumulative reward over time and that, generally, the leg length has a positive correlation with quadrupedal performance since longer legs tend to accumulate higher rewards, as longer legs mean higher traveling velocities are possible. It is important to note that the largest performance gap is between 0.1 and the longer lengths. Whilst performance increases with leg lengths, the effects are not as pronounced between all the leg lengths as in between 0.1 and 0.2 in Figure 7a and between 0.1 and 0.3 in Figure 7b (almost twice the cumulative reward). For instance, in Figure 7a $(0.2, 0.2)$ and $(0.3, 0.3)$ have pretty much identical final curves. The same goes for $(0.5, 0.5)$ and $(0.7, 0.7)$ in Figure 7b. This is certainly true when considering the curves themselves rather than their envelopes. The envelopes on the other hand show that higher leg lengths can lead to much higher performance values. For example, in Figure 7b $(0.7, 0.7)$ has an envelope that plateaus at a range of ≈ 100 to ≈ 700 while $(0.1, 0.1)$ plateaus at a range of ≈ 100 to ≈ 300. Even for curves that look almost identical, $(0.5, 0.5)$ and $(0.7, 0.7)$ in Figure 7b, there appears to be a difference of ≈ 100 in the max envelope cumulative reward. However, this effect is not as pronounced between lower leg lengths. This is demonstrated by how similar the curves and envelopes are between $(0.2, 0.2)$ and $(0.3, 0.3)$ in Figure 7a.

To investigate this variation further, the running Standard Deviations (S.D.) for baseline lists A and B are plotted in Figures 7c and 7d. Figure 7c shows how close the running S.D. curves are for all baseline curves within list A. The variation still increases with leg length, but not always as showcased by the curves of $(0.3, 0.3)$ and $(0.4, 0.4)$.

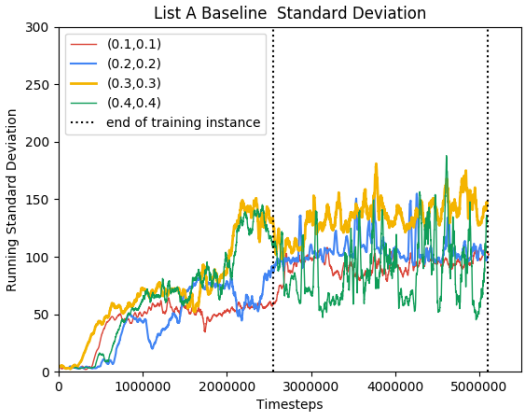
However, Figure 7d shows a greater variation in S.D. curves. Moreover, it is clear here that there is a positive correlation between the performance curves variation and the leg length.



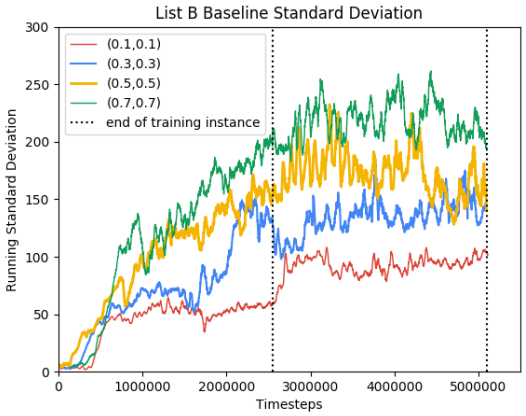
(a) Baseline reward curves for list A.



(b) Baseline reward curves for list B.



(c) Running S.D. of baseline curves for list A.



(d) Running S.D. of baseline curves for list B.

Figure 7: (a) and (b) show the baseline curves plotted as the mean of the running average ($window = 30$) of 5 random seeds for each iteration. Each curve is enveloped with its respective max and min mean values for all seeds. (c) and (d) show the running average ($window = 30$) of the S.D. for list A and B baseline curves. These are obtained from the results of the 5 random seeds for each iteration.

5.3 List A, Small Leg Length Spread

The list A leg lengths are $[0.1, 0.2, 0.3, 0.4]$. These are the values from which the curricula in this section are created. Table 2 shows the list A 12 curricula and their respective Figures for reference.

Curriculum	**Figure**
$(0.1, 0.2)$	Figures 9, 8
$(0.1, 0.3)$	Figures 10, 8
$(0.1, 0.4)$	Figures 11, 8
$(0.2, 0.1)$	Figures 8, 9
$(0.2, 0.3)$	Figures 10, 9
$(0.2, 0.4)$	Figures 11, 9
$(0.3, 0.1)$	Figures 8, 10
$(0.3, 0.2)$	Figures 9, 10
$(0.3, 0.4)$	Figures 11, 10
$(0.4, 0.1)$	Figures 8, 11
$(0.4, 0.2)$	Figures 9, 11
$(0.4, 0.3)$	Figures 10, 11

Table 2: List A curriculum/figure reference table

5.3.1 Leg length 0.1

Figures 8a and 8c show the relevant plots to all curricula ending with leg length 0.1 from list A. As can be seen in Figure 8a, the tabula rasa (no curriculum) case demonstrates the worst performance amongst all curricula. Here, the higher is the leg length of the source task, the better the final performance. This is clear in Figure 8a as $(0.4, 0.1)$ has the highest cumulative reward followed by $(0.3, 0.1)$ and then $(0.2, 0.1)$. In Figure 8c, the curves keep intertwining until the $4e6$ time-step mark. From there it becomes clear that the $(0.1, 0.1)$ and $(0.2, 0.1)$ curves have the higher final S.D. values, although the $(0.1, 0.1)$ curve shows higher stability. They are followed by the $(0.3, 0.1)$ curve which shows lower S.D. values and then the $(0.4, 0.1)$ curve with even lower values. It is worth noting that towards the end the $(0.4, 0.1)$ demonstrates the lowest noisiness in its S.D. curve.

Figures 8b and 8d show the relevant plots to all curricula starting with leg length 0.1 from list A. The 4 curves appear to converge to within the same cumulative reward range ($\approx 200- \approx 240$). Although upon closer inspection around the final half a million time-steps, it becomes clear that the $(0.1, 0.2)$ and $(0.1, 0.4)$ curves converge around similar performance at a roughly higher value than that of the $(0.1, 0.3)$ and tabula rasa curves, which also converge around the same values. The figure also shows that curves $(0.2, 0.1)$ and $(0.2, 0.4)$ are continuing to increase while the $(0.1, 0.3)$ and tabula rasa curves are stagnating in performance. The S.D. Figure 8d shows a similar trend between all its curves; all of them increase notably at the start of the final training instance. All but the $(0.1, 0.2)$ curve converge to the same running S.D. value of (≈ 100). Furthermore, all curves appear to converge around the same time-step ($\approx 3e6$).

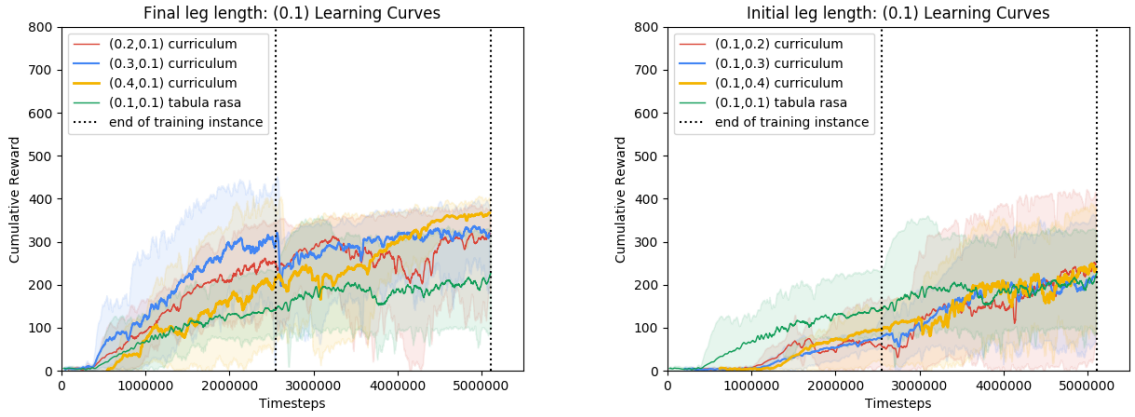
When comparing Figures 8b and 8a (excluding tabula rasa which is on both graphs), two points stand out. First, the performance curves for the final-case are higher than those of the initial-case. Second, although there are differences between the convergence values in the initial-case they are generally very similar, but the same is not true for the final-case. The curves in the final-case exhibit a slightly wider spread especially concerning the tabula rasa. i.e., in Figure 8b all 4 curves converge to within a small range, but in Figure 8a the three curriculum curves converge to a slightly wider range that is much further higher than the tabula rasa. The S.D. Figures 8c and 8d show that the initial-case curves generally converge to higher S.D. values than the final-case.

Furthermore, in the final-case 2 curves have decreasing plots ($(0.3, 0.1)$ and $(0.4, 0.1)$), but in the initial-case none of the curves decrease.

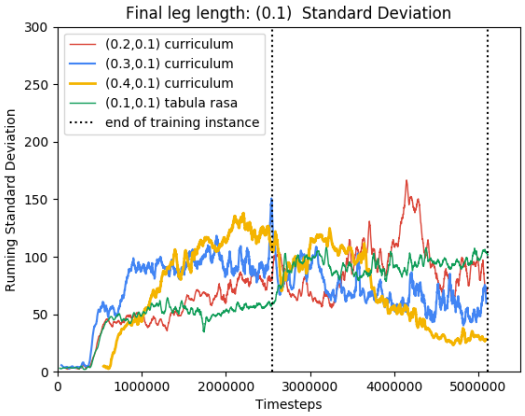
The hypothesis results can be found in Table 3. The first half of the table pertains to the final-case plots, while the second for the initial-case ones.

Curriculum	**Hypothesis tested**	**Result**
$(0.2, 0.1)$	$Hyp - C$	Accepted
$(0.3, 0.1)$	$Hyp - C$	Accepted
$(0.4, 0.1)$	$Hyp - C$	Accepted
$(0.1, 0.2)$	$Hyp - B$	Accepted
$(0.1, 0.3)$	$Hyp - B$	Rejected
$(0.1, 0.4)$	$Hyp - B$	Accepted

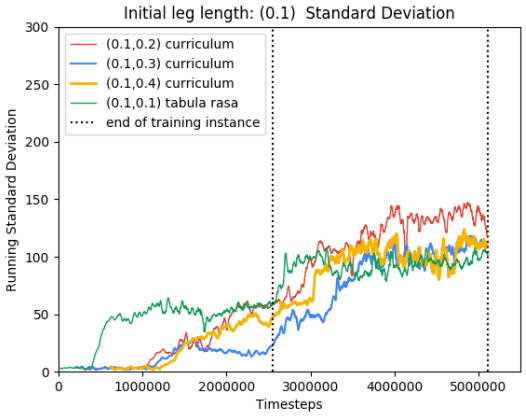
Table 3: The 6 performance plots tested against their respective hypotheses for leg length 0.1.



(a) Reward curves for the final 0.1 curricula. (b) Reward curves for the initial 0.1 curricula.



(c) Running S.D. for the final 0.1 curricula.



(d) Running S.D. for the initial 0.1 curricula.

Figure 8: (a) and (b) showcase the running averages ($window = 30$), and max min envelopes, of all list A curricula of final, and initial, leg length 0.1 respectively, including the tabula rasa (no curriculum) case. Plots are the aggregate of 5 random seeds. (c) and (d) show the running averages ($window = 30$) of the S.D. curves for the curricula in (a) and (b) respectively.

5.3.2 Leg length 0.2

Figures 9a and 9c show the relevant plots to all curricula ending with leg length 0.2 from list A. In Figure 9a, the lowest performance curve is that of $(0.1, 0.2)$. This is followed by $(0.4, 0.2)$, $(0.3, 0.2)$, and the tabula rasa (no curriculum) respectively. $(0.4, 0.2)$ shows a sudden slight drop in performance upon entering the second training instance not apparent in the other curves. It should be noted though that the $(0.4, 0.2)$ performance curve appears to stagnate by the $5e6$ time-step, the other curricula, specifically those of $(0.1, 0.2)$ and $(0.3, 0.2)$, are continuing to rise. It is clear from Figure 9c that the $(0.4, 0.2)$ variation in performance is quite high, although the S.D. drops a little and stops increasing once the agent transitions to leg length (0.2) from (0.4). The second highest running S.D. is that of the $(0.1, 0.2)$ curriculum, which kept increasing steadily even after transitioning to (0.2). The $(0.3, 0.2)$ and tabula rasa curricula possessed deviation plots of even lower values. Although the $(0.3, 0.2)$ had the lower curve of the two, both appeared to steady off once transitioned into (0.2) just as with the $(0.4, 0.2)$ curriculum.

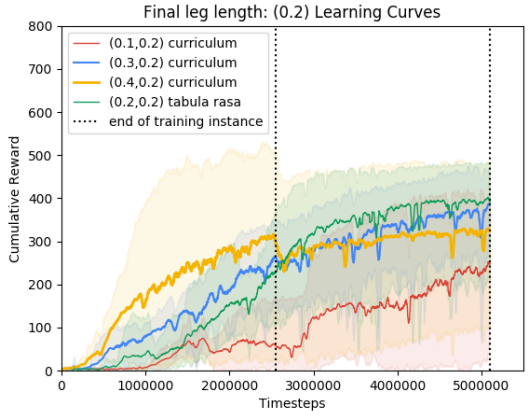
Figures 9b and 9d show the relevant plots to all curricula starting with leg length 0.2 from list A. From Figure 9d, the $(0.2, 0.1)$ curve has the lowest performance plot followed by the $(0.2, 0.4)$ and then the $(0.2, 0.3)$ and tabula rasa curves. Figure 9d shows rises in variation for all curves, particularly those of $(0.2, 0.3)$ and $(0.2, 0.4)$. However, unlike $(0.2, 0.4)$, which maintains its high values, $(0.2, 0.3)$ shows significant reduction, matching its own values from the first training instance. The S.D. curves of $(0.2, 0.1)$, $(0.2, 0.3)$, and tabula rasa appear to converge to the same values.

The performance results between the initial and final cases in Figures 9a and 9b show high similarity, specifically in the second training instance. For instance, all curves except for $(0.2, 0.1)$ and $(0.1, 0.2)$ exhibit similar shapes and final value convergence. The main difference occurs in the first training instance where the final-case has clearly higher performance values. The S.D. curves for $(0.1, 0.2)$ converges to higher values than that of its counterpart. $(0.4, 0.2)$ has much higher variance in the first instance compared to $(0.2, 0.4)$, but they both tend towards the same values in the second instance. $(0.2, 0.3)$ shows an increase in variance in the first half of the second instance, compared to $(0.3, 0.2)$, but similarly they both converge towards the same running S.D..

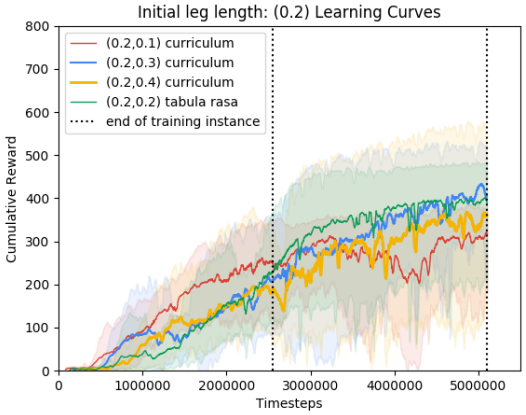
The hypothesis results can be found in Table 4. The first half of the table pertains to the final-case plots, while the second for the initial-case ones.

Curriculum	**Hypothesis tested**	**Result**
$(0.1, 0.2)$	$Hyp - A$	Rejected
$(0.3, 0.2)$	$Hyp - C$	Rejected
$(0.4, 0.2)$	$Hyp - C$	Rejected
$(0.2, 0.1)$	$Hyp - D$	Rejected
$(0.2, 0.3)$	$Hyp - B$	Accepted
$(0.2, 0.4)$	$Hyp - B$	Rejected

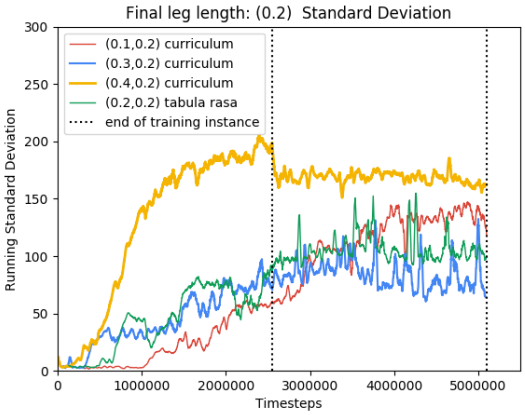
Table 4: The 6 performance plots tested against their respective hypotheses for leg length 0.2.



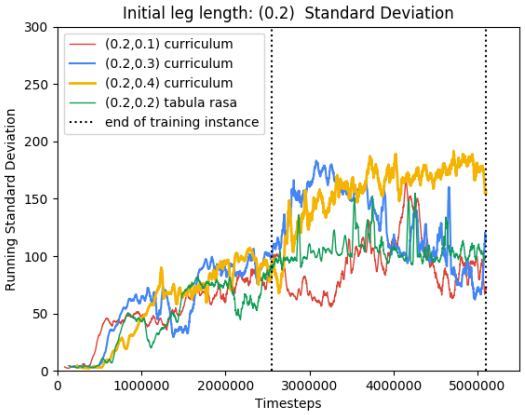
(a) Reward curves for the final 0.2 curricula.



(b) Reward curves for the initial 0.2 curricula.



(c) Running S.D. for the final 0.2 curricula.



(d) Running S.D. for the initial 0.2 curricula.

Figure 9: (a) and (b) showcase the running averages ($window = 30$), and max min envelopes, of all list A curricula of final, and initial, leg length 0.2 respectively, including the tabula rasa (no curriculum) case. Plots are the aggregate of 5 random seeds. (c) and (d) show the running averages ($window = 30$) of the S.D. curves for the curricula in (a) and (b) respectively.

5.3.3 Leg length 0.3

Figures 10a and 10c show the relevant plots to all curricula ending with leg length 0.3 from list A. In Figure 10a, the lowest performance curve is that of the $(0.1, 0.3)$ curriculum. It stagnates to two thirds of the the next higher curriculum, $(0.4, 0.3)$. The highest cumulative reward curve here is that of the tabula rasa up until the final half a million time-steps, where it is surpassed in performance by the $(0.2, 0.3)$ curriculum. Closely following them in magnitudes is the $(0.4, 0.3)$ curve. In Figure 10c, the $(0.1, 0.3)$ posses the least noisy S.D. curve, which stagnated around the last one million time-steps. It was also the lowest curve in magnitude until it was overtaken (in terms of lower variation) by the $(0.2, 0.3)$ curriculum around the last half a million time-steps mark. The tabula rasa curve stagnates rather quickly in comparison to the other curves (around the end of the first training instance). The $(0.4, 0.3)$ starts to steady-off a little towards the end of the second training session; however, it appears to continue increasing.

Figures 10b and 10d show the relevant plots to all curricula starting with leg length 0.3 from list A. The highest performance curve here is that of $(0.3, 0.4)$, followed by tabula rasa, $(0.3, 0.2)$, and then $(0.3, 0.1)$, in that order. Curves $(0.3, 0.4)$ and tabula rasa show sudden jumps in performance upon transitioning to the second training instance, but $(0.3, 0.1)$ shows a sudden drop before stagnating. $(0.3, 0.2)$ improves steadily. The variance demonstrated in Figure 10d is particularly noisy, especially the $(0.3, 0.4)$ curve. The $(0.3, 0.1)$ running S.D. steadily drops from the $1e6$ time-step mark while the $(0.3, 0.2)$ is pretty constant throughout the second training instance.

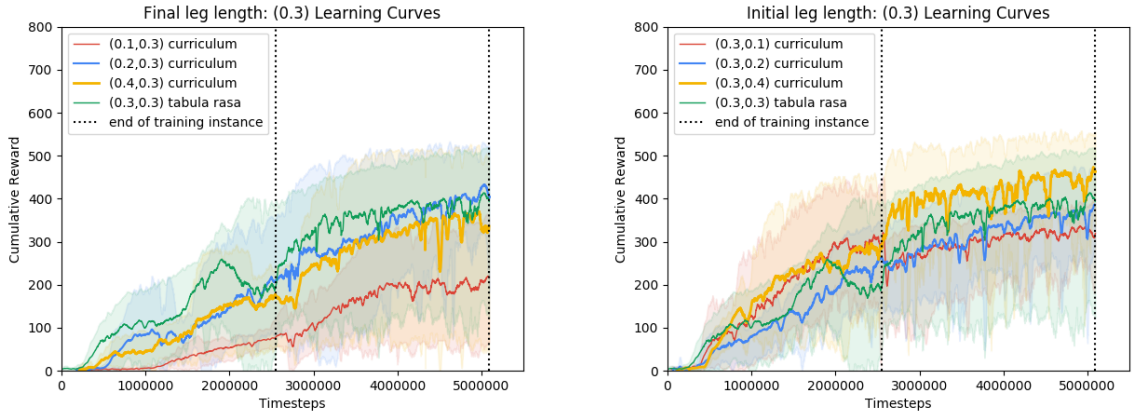
When comparing the Figures 10a and 10b, it appears that the $(0.3, 0.4)$ performance curve is higher than tabula rasa while that of $(0.4, 0.3)$ is lower. $(0.3, 0.2)$ and its counter part are very similar, with $(0.2, 0.3)$ being slightly higher towards the end. The most drastic difference in performance is between $(0.3, 0.1)$ and its counterpart, where $(0.3, 0.1)$ is much higher from the first training instance. In terms of S.D. between Figures 10c and 10d, there are a few differences. $(0.1, 0.3)$ increases in variance throughout unlike its counterpart which has the lowest variance in the 0.3 list A experiments. $(0.4, 0.3)$ also increases while $(0.3, 0.4)$ maintains a central, lower, value of ≈ 100 starting from the second training instance. Unlike the 0.1 and 0.4 curricula, $(0.2, 0.3)$ decreases notably in the second instance while $(0.2, 0.3)$ maintains a similar

trend to $(0.3, 0.4)$. In terms of noisiness, the final-case curricula are much smoother than those of the initial-case.

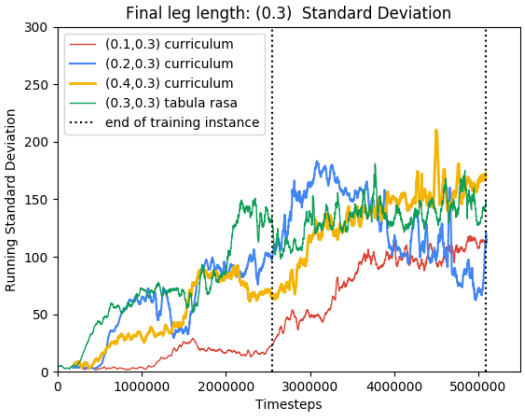
The hypothesis results can be found in Table 5. The first half of the table pertains to the final-case plots, while the second for the initial-case ones.

Curriculum	**Hypothesis tested**	**Result**
$(0.1, 0.3)$	$Hyp - A$	Rejected
$(0.2, 0.3)$	$Hyp - A$	Accepted
$(0.4, 0.3)$	$Hyp - C$	Rejected
$(0.3, 0.1)$	$Hyp - D$	Rejected
$(0.3, 0.2)$	$Hyp - D$	Rejected
$(0.3, 0.4)$	$Hyp - B$	Accepted

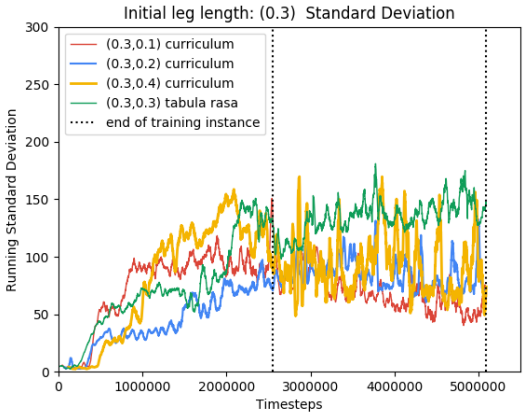
Table 5: The 6 performance plots tested against their respective hypotheses for leg length 0.3.



(a) Reward curves for the final 0.3 curricula. (b) Reward curves for the initial 0.3 curricula.



(c) Running S.D. for the final 0.3 curricula.



(d) Running S.D. for the initial 0.3 curricula.

Figure 10: (a) and (b) showcase the running averages ($window = 30$), and max min envelopes, of all list A curricula of final, and initial, leg length 0.3 respectively, including the tabula rasa (no curriculum) case. Plots are the aggregate of 5 random seeds. (c) and (d) show the running averages ($window = 30$) of the S.D. curves for the curricula in (a) and (b) respectively.

5.3.4 Leg length 0.4

Figures 11a and 11c show the relevant plots to all curricula ending with leg length 0.4 from list A. Figure 11a shows that the lowest performance curve is that of the $(0.1, 0.4)$ curriculum. It is followed by the $(0.2, 0.4)$, which is almost double in final magnitude. The $(0.3, 0.4)$ and tabula rasa cumulative reward curves are almost identical in shape and magnitude, but tabula rasa appears to possess higher values. In Figure 11c, the $(0.3, 0.4)$ and tabula rasa S.D. curves are very noisy in comparison to the $(0.1, 0.4)$ and $(0.2, 0.4)$ curves. The $(0.2, 0.4)$ has the highest variance curve. It is difficult to definitively distinguish the curve with the lowest variance due to the high noisiness of the $(0.3, 0.4)$ and tabula rasa curves. However, it appears they both oscillate about the $(0.1, 0.4)$ curve. It is worth noting that the $(0.1, 0.4)$ and $(0.2, 0.4)$ continued to rise after transitioning to the final training instance, but both the $(0.3, 0.4)$ and tabula rasa curves stagnated ($(0.3, 0.4)$ even dropped).

Figures 11b and 11d show the relevant plots to all curricula starting with leg length 0.4 from list A. The highest performance curve in Figure 11b is that of tabula rasa

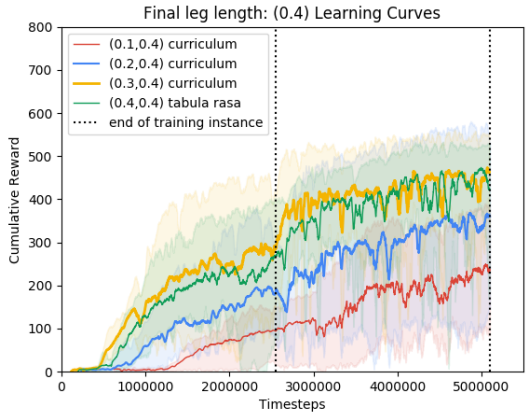
followed by $(0.4, 0.3)$ and $(0.4, 0.1)$ which were very similar throughout training. Closely following them is the $(0.4, 0.2)$ curve. In Figure 11d it is clear that 0.4 can vary greatly in its S.D. as demonstrated by the first training instance for all curricula. $(0.4, 0.2)$ drops a little when transitioning to the second training iteration then maintains its high variation. $(0.4, 0.3)$ rises steadily to match the high variation of $(0.4, 0.2)$. Tabula rasa had a lower variation than the aforementioned curricula, but it was very noisy. The lowest and smoothest S.D. curve goes to the $(0.4, 0.1)$ curriculum.

In comparing the initial and final cases, it is clear $(0.3, 0.4)$ (final-case) had notably higher performance than its counterpart in the initial-case. The reverse is true for the final-case of 0.1, which was much lower in performance than its initial-case plot. The initial and final plots for 0.2 are very similar in curve and overall performance. The only major difference is the dip in performance on instance transition in plot $(0.4, 0.2)$, which is not present in $(0.2, 0.4)$. The S.D. curves exhibit differences as well. The $(0.1, 0.4)$ rises in variation while its counterpart steadily drops. Similar to their performance curves, the 0.2 cases are similar in variation, stagnating to the same value of ≈ 175. The final-case 0.3 curve is very noisy but maintains low variation in comparison to its counterpart which steadily rises to match the $(0.4, 0.2)$ curve in final value.

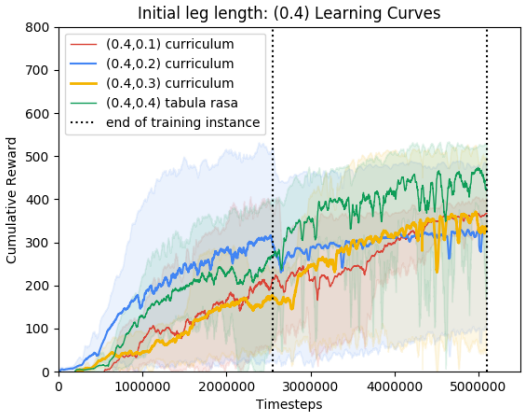
The hypothesis results can be found in Table 6. The first half of the table pertains to the final-case plots, while the second for the initial-case ones.

Curriculum	**Hypothesis tested**	**Result**
$(0.1, 0.4)$	$Hyp - A$	Rejected
$(0.2, 0.4)$	$Hyp - A$	Rejected
$(0.3, 0.4)$	$Hyp - A$	Rejected
$(0.4, 0.1)$	$Hyp - D$	Rejected
$(0.4, 0.2)$	$Hyp - D$	Rejected
$(0.4, 0.3)$	$Hyp - D$	Rejected

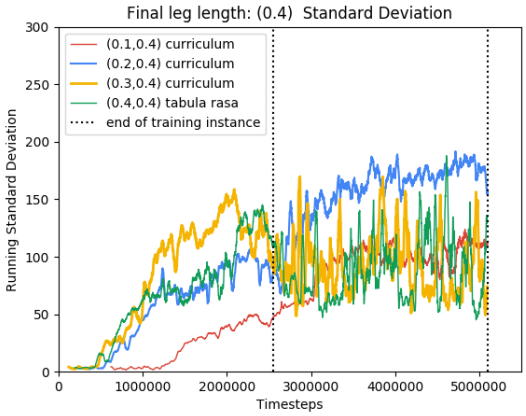
Table 6: The 6 performance plots tested against their respective hypotheses for leg length 0.4.



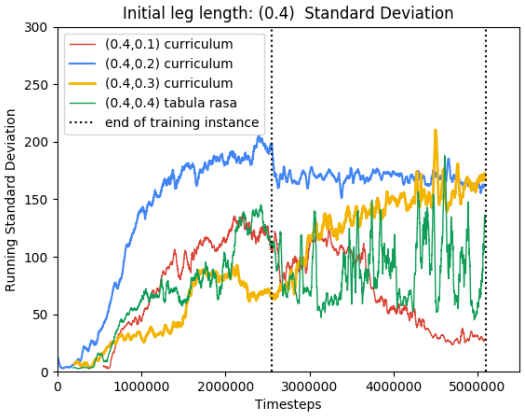
(a) Reward curves for the final 0.4 curricula.



(b) Reward curves for the initial 0.4 curricula.



(c) Running S.D. for the final 0.4 curricula.



(d) Running S.D. for the initial 0.4 curricula.

Figure 11: (a) and (b) showcase the running averages ($window = 30$), and max min envelopes, of all list A curricula of final, and initial, leg length 0.4 respectively, including the tabula rasa (no curriculum) case. Plots are the aggregate of 5 random seeds. (c) and (d) show the running averages ($window = 30$) of the S.D. curves for the curricula in (a) and (b) respectively.

5.3.5 Summary of List A Results

Table 7 summarizes the highest performing curricula as well as the lowest varying curricula for list A. It is apparent that the majority of highest performing curricula are of forward mode. The majority of lowest variation curricula are of backwards mode. It should be noted that leg length 0.2 has two very closely matched curricula in terms of

performance and variance.

Leg length	**Highest Reward**	**CL Mode**	**Lowest Running S.D.**	**CL Mode**
0.1	$(0.4, 0.1)$	Backwards	$(0.4, 0.1)$	Backwards
0.2	$(0.2, 0.3)$ & $(0.3, 0.2)$	Both	$(0.2, 0.3)$ & $(0.3, 0.2)$	Both
0.3	$(0.3, 0.4)$	Forwards	$(0.3, 0.1)$	Backwards
0.4	$(0.3, 0.4)$	Forwards	$(0.4, 0.1)$	Backwards

Table 7: List A curriculum mode table detailing the highest performing curriculum per leg length and its corresponding curriculum mode. It also shows the curricula with the lowest running S.D.

Table 8 demonstrates the hypotheses results for list A. Here $Hyp-B$ and $Hyp-C$ are matched in performance, although $Hyp-B$ is higher. This means that forwards and backwards curricula tested on tabula rasa of the simpler task caused performance improvements of 66.7% and 50.0% respectively. $Hyp-A$ had a low 16.7% acceptance percentage followed by $Hyp-D$, which had no improvements in performance at all when compared to tabula rasa.

Name	**Pass % in list A**	**CL Mode**
$Hyp-A$	16.7	Forwards
$Hyp-B$	66.7	Forwards
$Hyp-C$	50.0	Backwards
$Hyp-D$	0.0	Backwards

Table 8: Hypotheses acceptance percentage for the list A experiments. Each hypothesis is tested on 6 curricula per list.

5.4 List B, Large Leg Length Spread

The list B leg lengths are $[0.1, 0.3, 0.5, 0.7]$, testing a wider spread. These are the values from which the curricula in this section are created. Table 9 shows the list B 12 curricula and their respective Figures for reference.

Curriculum	**Figure**
$(0.1, 0.3)$	Figures 13, 12
$(0.1, 0.5)$	Figures 14, 12
$(0.1, 0.7)$	Figures 15, 12
$(0.3, 0.1)$	Figures 12, 13
$(0.3, 0.5)$	Figures 14, 13
$(0.3, 0.7)$	Figures 15, 13
$(0.5, 0.1)$	Figures 12, 14
$(0.5, 0.3)$	Figures 13, 14
$(0.5, 0.7)$	Figures 15, 14
$(0.7, 0.1)$	Figures 12, 15
$(0.7, 0.3)$	Figures 13, 15
$(0.7, 0.5)$	Figures 14, 15

Table 9: List B curriculum/figure reference table

5.4.1 Leg length 0.1

Figures 12a and 12c show the relevant plots to all curricula ending with leg length 0.1 from list B. The performance curves in Figure 12a, excluding tabula rasa, all maintain similar trends and values. They all drop once transitioned into the second training instance and they all stagnate to a cumulative reward of ≈ 320. Furthermore, they are all notably higher in performance than tabula rasa. Their S.D. plots are not as similar to one another, as demonstrated in Figure 12c. They all (excluding tabula rasa) have high variations in the first instance. 0.3 is the lowest, closely followed by 0.5. The 0.7 has much higher variation than the rest. In the second instance, the curves as ordered from lowest to highest variation are: $(0.5, 0.1)$, $(0.3, 0.1)$, tabula rasa, and $(0.7, 0.1)$.

The 0.5 and 0.3 curves drop when crossed into the second instance. The 0.7 does so as well, but suddenly spikes midway through training.

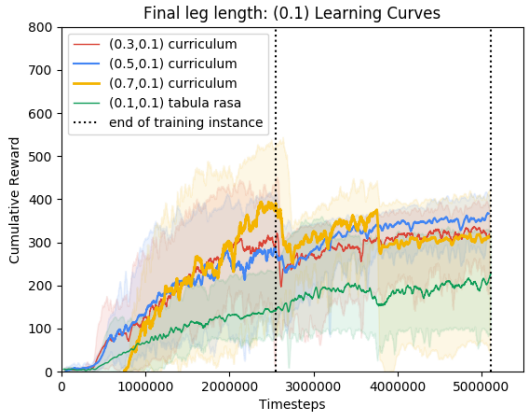
Figures 12b and 12d show the relevant plots to all curricula starting with leg length 0.1 from list B. Figure 12b shows that the performance curves are all very similar to tabula rasa, except for $(0.1, 0.5)$ which keeps on steadily improving past the $4e6$ time-step mark. The 3 curves also demonstrate slight dips in cumulative reward closely after transitioning to the second training instance. In terms of variation, 12d shows that all the curves increase in variation once transitioning into the second training instance. Only $(0.1, 0.5)$ demonstrates any reduction in variation, a little past the $4e6$ time-step mark. Both $(0.1, 0.5)$ and $(0.1, 0.3)$ exhibit similar final S.D. values to tabula rasa.

The plots in Figure 12a are all higher than tabula rasa, but in Figure 12b they are mostly not (with the exception of $(0.1, 0.5)$). In both figures there are drops in performance once transitioned into the second training instance. The variation curves in Figure 12c show lower values as opposed to their counterparts in Figure 12d (with the exception of 0.7).

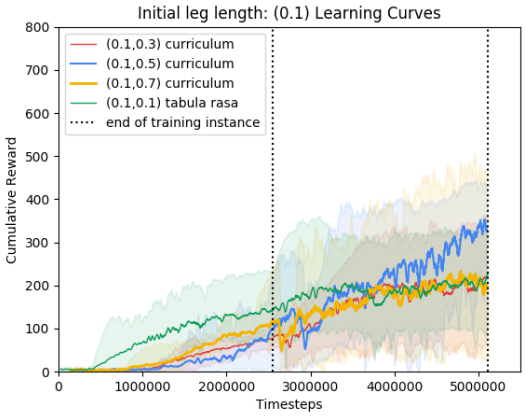
The hypothesis results can be found in Table 10. The first half of the table pertains to the final-case plots, while the second for the initial-case ones.

Curriculum	**Hypothesis tested**	**Result**
$(0.3, 0.1)$	$Hyp - C$	Accepted
$(0.5, 0.1)$	$Hyp - C$	Accepted
$(0.7, 0.1)$	$Hyp - C$	Accepted
$(0.1, 0.3)$	$Hyp - B$	Rejected
$(0.1, 0.5)$	$Hyp - B$	Accepted
$(0.1, 0.7)$	$Hyp - B$	Rejected

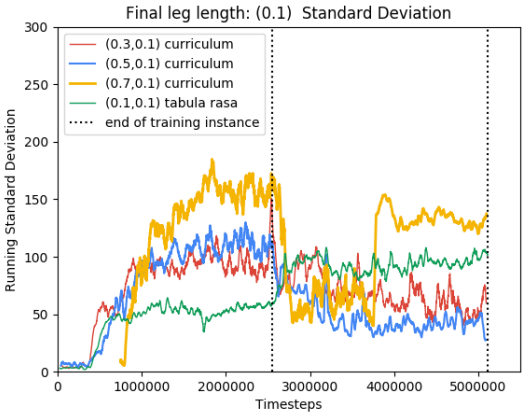
Table 10: The 6 performance plots tested against their respective hypotheses for leg length 0.1.



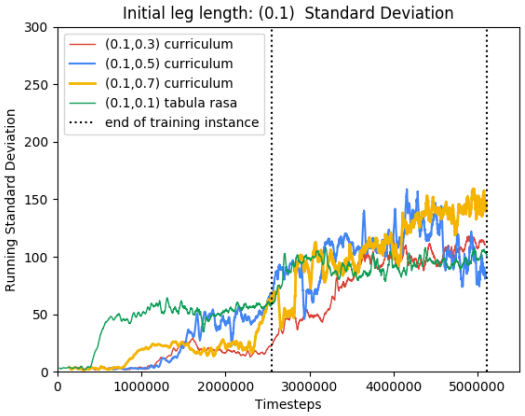
(a) Reward curves for the final 0.1 curricula.



(b) Reward curves for the initial 0.1 curricula.



(c) Running S.D. for the final 0.1 curricula.



(d) Running S.D. for the initial 0.1 curricula.

Figure 12: (a) and (b) showcase the running averages ($window = 30$), and max min envelopes, of all list B curricula of final, and initial, leg length 0.1 respectively, including the tabula rasa (no curriculum) case. Plots are the aggregate of 5 random seeds. (c) and (d) show the running averages ($window = 30$) of the S.D. curves for the curricula in (a) and (b) respectively.

5.4.2 Leg length 0.3

Figures 13a and 13c show the relevant plots to all curricula ending with leg length 0.3 from list B. The $(0.5, 0.3)$ and $(0.7, 0.3)$ performance curves in Figure 13a are very similar and surpass tabula rasa. The $(0.1, 0.3)$ is much lower than all other curves. None of the curves show a decrease in performance, although they all slow down once into

the second training instance. The S.D. curves in Figure 13c show strong reduction in variation for the $(0.7, 0.3)$ curve and moderate reduction for the already low $(0.5, 0.3)$ curve. The $(0.1, 0.3)$ plot is the only one besides tabula rasa to show increase in variation. $(0.1, 0.3)$ is also the smoothest S.D. curve, followed by tabula rasa, $(0.7, 0.3)$, and then $(0.5, 0.3)$.

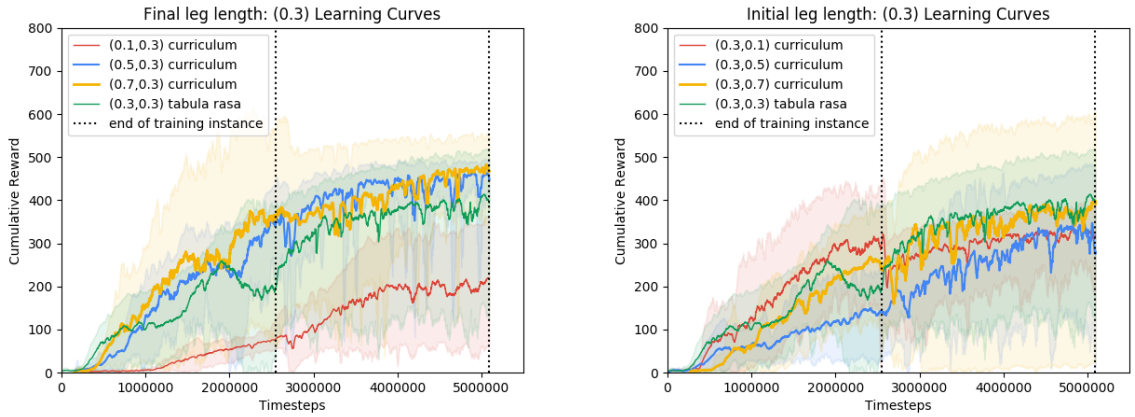
Figures 13b and 13d show the relevant plots to all curricula starting with leg length 0.3 from list B. Figure 13b showcases that non of the curves surpass tabula rasa, although $(0.3, 0.7)$ manages to match it in performance. $(0.3, 0.5)$ and $(0.3, 0.1)$ closely follow it in cumulative reward values. In fact, both those curves stagnate to the same value of ≈ 300. All the curves here, excluding tabula rasa, exhibit sudden drops in performance once entering the second training instance prior to increasing again. Figure 13d shows that $(0.3, 0.7)$ has a significantly larger variation curve in the second training instance in comparison to all other plots. Tabula rasa and $(0.3, 0.5)$ are closely matched and at about half their values is the $(0.3, 0.1)$ S.D. curve, which is the inly one to show a decrease in variation rather than an increase.

The final-cases 0.1 plot has lower performance than its initial-case. The final-case variance curve increases, while the initial-case decreases. The opposite is true for 0.3 and 0.7. For both cases performance is higher in the final-case and their variance curves exhibit opposite behavior. It is important to note that the running S.D. plots for all the final-case curves are lower than tabula rasa, but this is not true for the initial case even though both cases are the result of the same set of leg lengths.

The hypothesis results can be found in Table 11. The first half of the table pertains to the final-case plots, while the second for the initial-case ones.

Curriculum	Hypothesis tested	Result
$(0.1, 0.3)$	$Hyp - A$	Rejected
$(0.5, 0.3)$	$Hyp - C$	Accepted
$(0.7, 0.3)$	$Hyp - C$	Accepted
$(0.3, 0.1)$	$Hyp - D$	Rejected
$(0.3, 0.5)$	$Hyp - B$	Rejected
$(0.3, 0.7)$	$Hyp - B$	Rejected

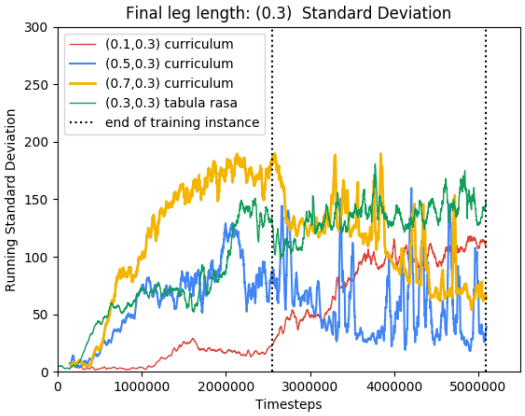
Table 11: The 6 performance plots tested against their respective hypotheses for leg length 0.3.



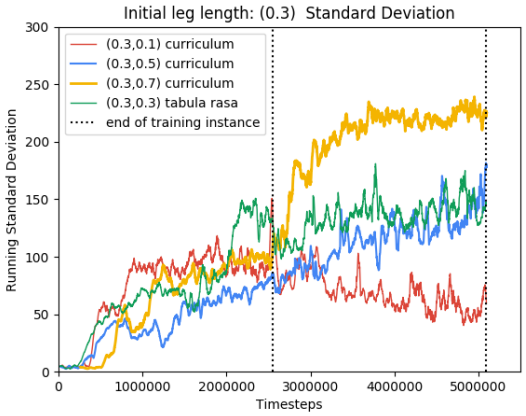
(a) Reward curves for the final 0.3 curricula. (b) Reward curves for the initial 0.3 curricula.

5.4.3 Leg length 0.5

Figures 14a and 14c show the relevant plots to all curricula ending with leg length 0.5 from list B. Figure 14a demonstrates than non of the curves exceed tabula rasa in performance except for $(0.7, 0.5)$. The $(0.1, 0.5)$ and $(0.3, 0.5)$ plots are quite similar with $(0.1, 0.5)$ catching up to $(0.3, 0.5)$ towards the end. Note that both plots score lower performance values than tabula rasa. Figure 14c demonstrates that $(0.7, 0.5)$ starts out with the highest variation but drops during training. $(0.1, 0.5)$ and $(0.3, 0.5)$ however steadily rise throughout training. They are also much less noisy than the $(0.7, 0.5)$ S.D. curve. $(0.1, 0.5)$ drops again around the last half a million time-step mark. By the $5e6$ time-step mark, it becomes clear $(0.7, 0.5)$ has the lowest variation followed closely by



(c) Running S.D. for the final 0.3 curricula.



(d) Running S.D. for the initial 0.3 curricula.

Figure 13: (a) and (b) showcase the running averages ($window = 30$), and max min envelopes, of all list B curricula of final, and initial, leg length 0.3 respectively, including the tabula rasa (no curriculum) case. Plots are the aggregate of 5 random seeds. (c) and (d) show the running averages ($window = 30$) of the S.D. curves for the curricula in (a) and (b) respectively.

$(0.1, 0.5)$. $(0.3, 0.5)$ closely matches the tabula rasa values.

Figures 14b and 14d show the relevant plots to all curricula starting with leg length 0.5 from list B. The cumulative reward plots in Figure 14b closely match tabula rasa, except for the $(0.5, 0.1)$ curve which scores lower values once transitioned into the second training instance. Figure 14d shows that tabula rasa has the highest variation, but it was starting to get matched by the slowly rising $(0.5, 0.7)$ curve towards the end of training. Both $(0.5, 0.1)$ and $(0.5, 0.3)$ decrease once into the second training instance and stagnate around a very low value of ≈ 50. $(0.5, 0.7)$ is the noisiest S.D. curve in this figure followed by $(0.5, 0.3)$, tabula rasa, and then $(0.5, 0.1)$.

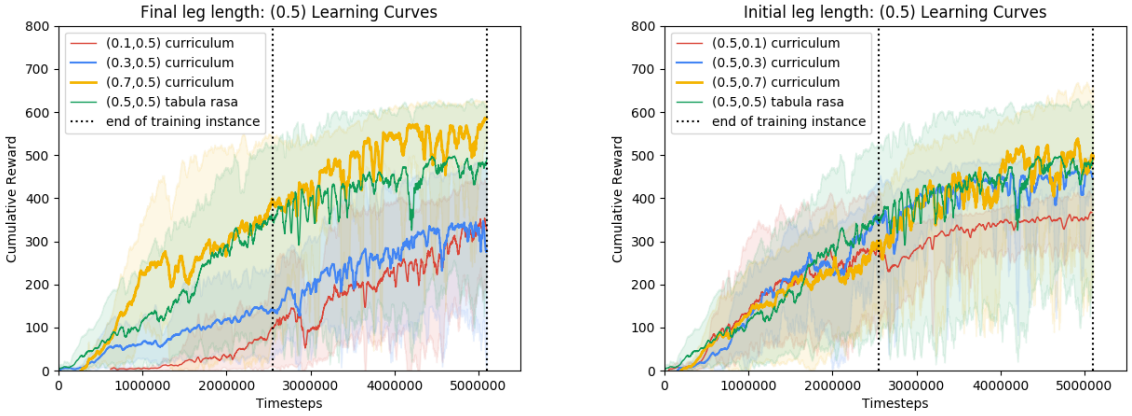
Between the final and initial cases there are a few differences. final-case 0.1 and 0.3 possess lower performance curves than their counterparts in the initial-case. Final-case 0.7 though has a higher curve than the initial-case version. The initial-case curves are much closer in value to tabula rasa than the final-case curves. The variation curves for final-case 0.1 and 0.3 show increase, while their counterparts decrease. The opposite is true for the 0.7 S.D. curves.

The hypothesis results can be found in Table 12. The first half of the table pertains

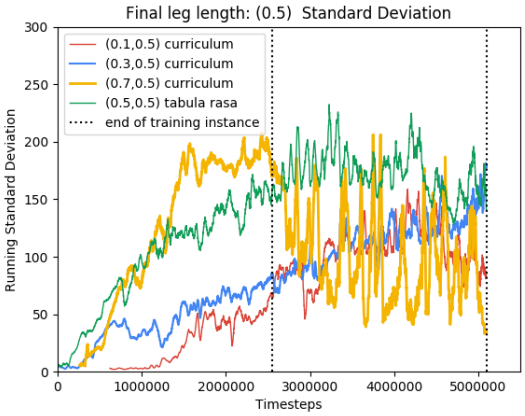
to the final-case plots, while the second for the initial-case ones.

Curriculum	Hypothesis tested	Result
$(0.1, 0.5)$	$Hyp - A$	Rejected
$(0.3, 0.5)$	$Hyp - A$	Rejected
$(0.7, 0.5)$	$Hyp - C$	Accepted
$(0.5, 0.1)$	$Hyp - D$	Rejected
$(0.5, 0.3)$	$Hyp - D$	Rejected
$(0.5, 0.7)$	$Hyp - B$	Accepted

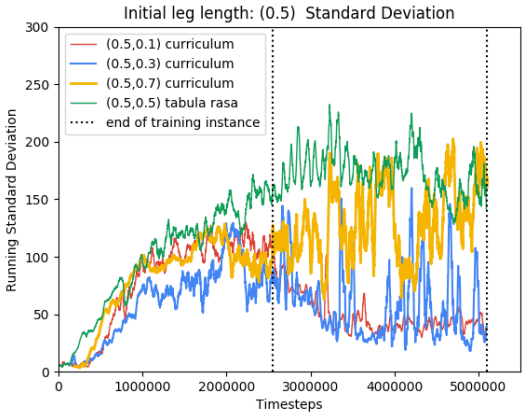
Table 12: The 6 performance plots tested against their respective hypotheses for leg length 0.5.



(a) Reward curves for the final 0.5 curricula. (b) Reward curves for the initial 0.5 curricula.



(c) Running S.D. for the final 0.5 curricula.



(d) Running S.D. for the initial 0.5 curricula.

Figure 14: (a) and (b) showcase the running averages ($window = 30$), and max min envelopes, of all list B curricula of final, and initial, leg length 0.5 respectively, including the tabula rasa (no curriculum) case. Plots are the aggregate of 5 random seeds. (c) and (d) show the running averages ($window = 30$) of the S.D. curves for the curricula in (a) and (b) respectively.

5.4.4 Leg length 0.7

Figures 15a and 15c show the relevant plots to all curricula ending with leg length 0.7 from list B. Figures 15a showcases the $(0.5, 0.7)$ performance curve to be the only one matching tabula rasa in performance. The other two curves, $(0.3, 0.7)$ and $(0.1, 0.7)$, are lower, respectively in that order. Noticeably, Both curves have drops in performance upon entering the second training instance, but $(0.5, 0.7)$ does not. The variation curves in Figure 15c are quite high as opposed to other experiments. The $(0.3, 0.7)$ curve possesses the highest variation, closely matching tabula rasa. The sudden rise only occurs in the second training instance. $(0.5, 0.7)$ maintains similar variations in both instances, although a little higher in the second one and much more turbulent. $(0.1, 0.7)$ also rises much in variation similar to the $(0.3, 0.7)$ curve. Both it and $(0.5, 0.7)$ appear to reach similar final S.D. values (at about ≈ 150).

Figures 15b and 15d show the relevant plots to all curricula starting with leg length 0.7 from list B. In Figure 15b the $(0.7, 0.5)$ curve appears to have the highest cumulative reward over time. This is followed by tabula rasa and $(0.7, 0.3)$, which are very closely

matched. The $(0.7, 0.1)$ has the lowest reward at ≈ 300. None of the curves show sudden drops in performance at entering the second training instance except for the $(0.7, 0.1)$ curve. Figure 15d shows general drop in variation for the $(0.7, 0.3)$ and $(0.7, 0.5)$ curves once in the second training instance. However, both curves increase in noisiness. The $(0.7, 0.1)$ plot drops suddenly at entering the second instance, but rises abruptly around the $4e6$ time-step, which is where a second drop in performance can be seen in Figure 15b.

The performance curve for final-case 0.5 is lower than its initial-case. The variation curve for the final-case is higher on the other hand. The same is true for the 0.3 cumulative reward and S.D. plots. The final-case 0.1 performance curve is also lower than its counterpart; however, their variation plots similar end results since the initial-case curve rose abruptly after dropping at the start of the second training instance.

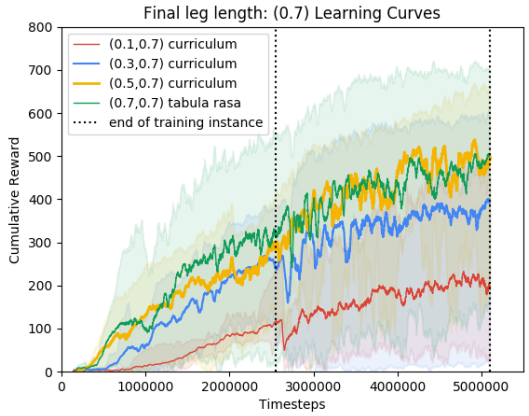
The hypothesis results can be found in Table 13. The first half of the table pertains to the final-case plots, while the second for the initial-case ones.

Curriculum	**Hypothesis tested**	**Result**
$(0.1, 0.7)$	$Hyp - A$	Rejected
$(0.3, 0.7)$	$Hyp - A$	Rejected
$(0.5, 0.7)$	$Hyp - A$	Rejected
$(0.7, 0.1)$	$Hyp - D$	Rejected
$(0.7, 0.3)$	$Hyp - D$	Rejected
$(0.7, 0.5)$	$Hyp - D$	Accepted

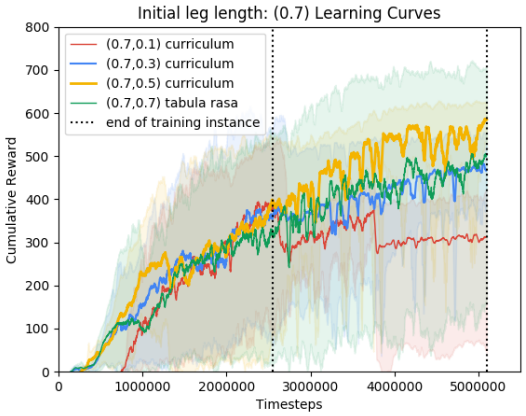
Table 13: The 6 performance plots tested against their respective hypotheses for leg length 0.7.

5.4.5 Summary of List B Results

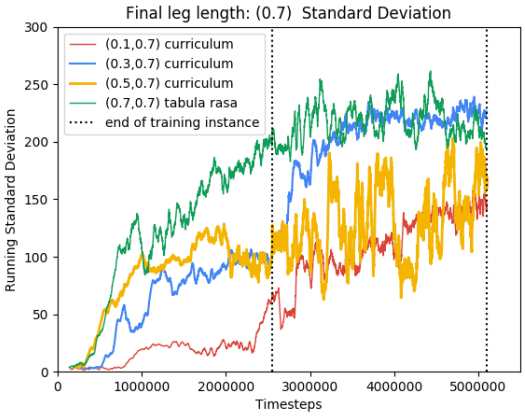
Table 14 shows the highest performing curricula as well as the lowest varying curricula for list B. Clearly all of highest performing curricula are of backwards mode. Similarly, all of lowest variation curricula are of backwards mode. Furthermore, in table 14 for leg length 0.1 the highest performing curriculum is detailed as $(0.5, 0.1)$, however $(0.1, 0.5)$



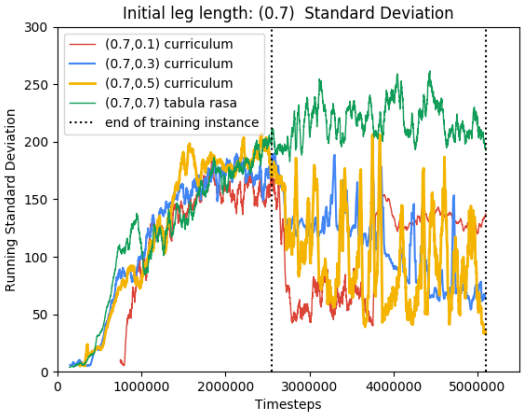
(a) Reward curves for the final 0.7 curricula.



(b) Reward curves for the initial 0.7 curricula.



(c) Running S.D. for the final 0.7 curricula.



(d) Running S.D. for the initial 0.7 curricula.

Figure 15: (a) and (b) showcase the running averages ($window = 30$), and max min envelopes, of all list B curricula of final, and initial, leg length 0.7 respectively, including the tabula rasa (no curriculum) case. Plots are the aggregate of 5 random seeds. (c) and (d) show the running averages ($window = 30$) of the S.D. curves for the curricula in (a) and (b) respectively.

is actually a close contester. In fact, if results were to be extrapolated, $(0.1, 0.5)$ would surpass $(0.5, 0.1)$ in performance. Nevertheless, the tables detail the curricula behaviors only within the fixed training time used for all experiments.

The hypotheses results in Table 15 shows a stunning 100% acceptance percentage to $Hyp - C$. This means all backwards curricula compared to tabula rasa on the simpler

Leg length	**Highest Reward**	**CL Mode**	**Lowest Running S.D.**	**CL Mode**
0.1	$(0.5, 0.1)$	Backwards	$(0.5, 0.1)$	Backwards
0.3	$(0.7, 0.3)$	Backwards	$(0.5, 0.3)$	Backwards
0.5	$(0.7, 0.5)$	Backwards	$(0.5, 0.3)$	Backwards
0.7	$(0.7, 0.5)$	Backwards	$(0.7, 0.5)$	Backwards

Table 14: List B curriculum mode table detailing the highest performing curriculum per leg length and its corresponding curriculum mode. It also shows the curricula with the lowest running S.D.

task had better cumulative reward. A far second in acceptance percentage is $Hyp - B$ at 33.3%, followed by the low 16.7% of $Hyp - D$. $Hyp - A$ had no improvements in performance at all when compared to tabula rasa.

Name	**Pass % in list B**	**CL Mode**
$Hyp - A$	0.0	Forwards
$Hyp - B$	33.3	Forwards
$Hyp - C$	100.0	Backwards
$Hyp - D$	16.7	Backwards

Table 15: Hypotheses acceptance percentage for the list B experiments. Each hypothesis is tested on 6 curricula per list.

5.5 Forwards and Backwards Curriculum Learning

This sub-section takes a closer look at the patterns shown in the lists A and B hypotheses experiments. First, results of lists A and B are compared against each other. Then, the overarching pattern across all experiments is analyzed.

5.5.1 Hypotheses Tests on A vs. B

The results for the hypotheses tests on lists A and B are in summed in Table 16, respectively. Following the same trend in Table 17, the hypotheses test on all experiments, $Hyp - B$ and $Hyp - C$ have the highest acceptance percentage here too. However, note that $Hyp - B$ and $Hyp - C$ have similar percentages in list A (66.7 and 50.0 respectively), but drastically different ones in list B (33.3 and 100.0 respectively). Furthermore, in list A $Hyp - B$ has the highest success rate, but in list B that goes to $Hyp - C$.

Name	**Pass % in list A**	**Pass % in list B**	**CL Mode**
$Hyp - A$	16.7	0.0	Forwards
$Hyp - B$	66.7	33.3	Forwards
$Hyp - C$	50.0	100.0	Backwards
$Hyp - D$	0.0	16.7	Backwards

Table 16: Hypotheses acceptance percentage for lists A and B experiments. Each hypothesis is tested on 6 curricula per list.

5.5.2 Hypotheses Tests on All

Table 17 shows the percentage of curricula (out of 12) passing each hypothesis. As can be seen, $Hyp-A$ has the lowest acceptance percentage of only 8.3%. This means that a forward curriculum used to transfer knowledge on a more complex task rarely succeeds. However, a forward curriculum used to improve performance on the same simple task has a high percentage of succeeding as proven by the 75% acceptance rate of $Hyp-C$. Similar to $Hyp-A$, only 1 curriculum of 12 (8.3%) resulted in performance improvement for $Hyp-D$. This means a backwards curriculum designed to improve performance on a complex task has a low chance of success. $Hyp-C$ possesses the highest acceptance percentage of 75%, meaning a backwards curricula designed to improve performance on the same simple task is highly likely. It is clear upon second inspection that the highest performing curricula in Tables 7 and 14 are those conforming to either $Hyp-B$ or $Hyp-C$.

It is imperative to note that $Hyp-B$ and $Hyp-C$ both transfer their curricula to K_n (the simpler task), while $Hyp-A$ and $Hyp-D$ transfer to K_{n+1} (the more complex task).

Name	**Pass %**	**CL Mode**
$Hyp-A$	8.3	Forwards
$Hyp-B$	50.0	Forwards
$Hyp-C$	75.0	Backwards
$Hyp-D$	8.3	Backwards

Table 17: Hypotheses acceptance percentage for all experiments. Each hypothesis is tested on 12 curricula.

5.6 Section Summary

From the baseline graphs it is clear longer leg lengths have higher cumulative reward curves. They are also prone to higher variations in performance. This is strongly evident in all the experiments where forward curricula tend to increase in variation while backward curricula tend to the opposite.

The remaining results strongly indicate a few points. First, CL is not best for improving improving performance on hard tasks, but is best for doing so on easy tasks. Easy and hard here refer to the comparative difficulty between the tasks comprising the curriculum. This is strongly backed by the results in Tables 17 and 16. Second, a backwards mode curriculum is best suited for tasks with a wider spread in difficulty, while a forwards mode curriculum is best suited for tasks close in difficulty. This is backed by the results in Tables 7 and 14. Third, a backwards mode curriculum is the most ideal for reducing variance in performance as demonstrated in Tables 7 and 14. Finally, backwards mode curricula tend to generally be of better results than forwards mode. This is shown by their domination in lowest variance curves, how close they are in acceptance % between $Hyp - B$ and $Hyp - C$ for list A, and how dominating they are in acceptance % for list B with a 100.0%.

6 Discussion

6.1 Section Overview

The results detailed in the Results section provide interesting insights and pose important questions to CL. The experiments clearly show the potency of both modes of CL to improve performance on the simple task, although favoring backwards mode. An attempt to explain this phenomena is made by proposing a model for the knowledge space of the tasks of this project. Furthermore, a deeper look is taken at the reasons behind the failure of the ACG experiments, which became apparent after the manual design experiments. Moreover, the potential of CL in solving harder problems is discussed.

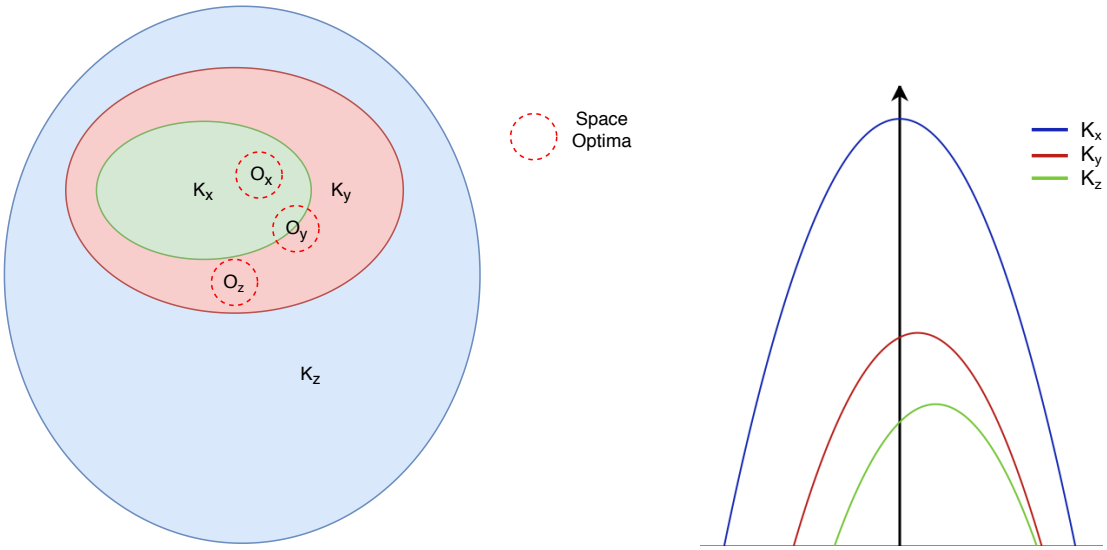
6.2 Limitations Placed

Similar to many of the papers in literature, limitations have been placed on both the manual and automatic experiments to facilitate learning. While this proved successful in the manual case, it did not in the automatic case as discussed further on.

6.3 Knowledge Space Model

The results section strongly back that CL is best suited for improving performance on the simple task. The simple task here means the simplest task in the curriculum itself. This certainly makes sense in the contexts of our daily lives. For instance, a gamer who practices on high difficulty maps will fare much better on the easier ones. Or, an acrobat who can walk on high stilts has better walking balance than the average Joe. Interestingly, this means that practice on a single task for a longer time (tabula rasa) does not equate drastic improvements in performance. Rather, it is the variation in practice, on increasing difficulty, that does the job.

This implies that for a given task, say learning to walk as a quadrupedal, the knowledge space of the simple versions of this task live within the space of the more complex ones. It also implies that their optimal policies coexist in the same neighborhood of the space projection. To flesh out this idea, please refer to Figure 16 and the upcoming setup. If the knowledge space is assumed to be 3-dimensional, then Figure 16a is its 2D projection. Assume then that this space also possesses height proportional to the



(a) 2D knowledge spaces of x, y and z.

(b) Third dimension of the knowledge space.

Figure 16: A simplified version of the knowledge spaces for 3 versions of the same task, increasing in complexity from x, y to z. (a) shows the 2D space and (b) the 3^{rd} dimension which is proportional to the optimality of a policy.

optimality of a given policy, as in Figure 16b. Therefore, the knowledge spaces of tasks x, y and z are 3 hills encapsulated within one another. Assume again that like in the figures, the optimal spaces of the 3 tasks, O_x, O_y, and O_z are close to one another in the projection plane, but are quite far height-wise. Now take a curriculum where an agent learns O_y by training in the y knowledge space (K_y) and then is transferred to the z knowledge space (K_z). It is clear that although the two optima are within the same vicinity in the projection, the drastic height difference prevents the agent from reaching O_z on transfer. This is why if this curriculum is compared against tabula rasa z, it will most likely perform worse. Now reverse the curriculum order, meaning the agent initially trains in K_z prior to transfer to K_y. If it drops from O_z, or anywhere close to it, it will land very close to O_y both in the projection and height dimensions. This is why such a curriculum would fare better in performance as compared to tabula rasa y.

This model also explains the forwards mode and backwards mode phenomena exhibited by the results, where list A favors the forwards mode and list B favors the backwards mode. For instance, if the two spaces are as close in height and size as be-

tween K_x and K_y then a forward curriculum from K_x to K_y will outperform tabula rasa x. Whereas a forward curriculum from K_x to K_z will cause a drop in performance as compared to tabula rasa x. However, a backwards curriculum from K_z to K_x will result in better performance. In other words, if the agent gets close to O_z in the first training instance and then drops in space to K_x, it will land close to O_x. This is especially reinforced by the drops in S.D. plots in the results section from backwards curricula. They mean that range of successful policies gets narrower in the second training instance. The fact that it does not get higher means the two optima are, in some sense, close to one another and the transfer hones in on the easier task optimal policy.

It is imperative to note that the proposed model here is used to reason with, and try to provide justification for, the results of this report. This does not imply that is the definitive knowledge space model for quadrupedal locomotion, or similar tasks. Prior to declaring such a statement, further work must be conducted as detailed in the Further Works section later on in this report.

6.4 Failure of Automatic Curriculum Generation

There is a sub-section in the Methodology section pertaining to ACG. It is mentioned that the Results section does not include any product of these experiments. That is to say, no distinct policy was reached by the professor agent for building viable curricula to improve performance on the longer legs for the quadrupedal. Initially this was hypothesized to be due to the sheer size of the problem space. That the problem was simply too big to solve given the provided computing resources. This was why some limits were placed on the CLEnv environment, such as constraining the professor's action-space. However, after conducting the manually designed experiments and analyzing the results, it appears the reason for failure is in the conditions placed upon the professor themselves. It was forced to design a forwards curriculum from short legs to long legs in order to exceed tabula rasa of the long legs. Yet, as seen in the results section, CL is best for improving performance on the simple task not for learning the more complex ones. The professor was simply given a Herculean task where all the evidence points against its success. This surprising conclusion sheds light on a larger issue in CL in Deep RL. As researchers, the overarching purpose of furthering Deep RL is to enable it to solve the more difficult problems.

6.5 Solving the Harder Problems

CL is based on sound logic and certainly demonstrates its success in our daily lives. The entire educational system is a series of long forwards mode curricula. Backwards mode curricula are also used in the real-world by athletes for instance. It is very common practice to train runners with weight belts and wight bands prior to the main event, where they run weight-free. However, for most engineering applications, and certainly for robotics, the main goal for CL in Deep RL is to improve performance on the harder tasks using forwards mode. Because in most cases, the harder task is too hard to solve under the provided time and computing resources. Therefore the question posses itself, why is forwards mode curricula for learning harder tasks successful in the real-world but not in the experiments conducted in this project? This may be due to the nature of the quadrupedal locomotion problem itself; yet the results show the tasks are close enough in knowledge that transfer helps. It may be due to the design of the experiments; a 2 instance curriculum could simply be too short to gain tangible improvements. It may also be due to the training time per instance; too short or too long. Or it may be due to a limitation in our application of CL, that it should not merely be transferring policy from one task to another. For example, usually a teacher reviews last semester's material prior to moving on to the new. Or they visit them intermittently during the new semester. These are ideas worth future exploration as detailed in the Future Work section.

6.6 Section Summary

The results of the manual design experiments allow for the building of a knowledge space model closely justifying them. The model entails that harder versions of a task encapsulate the knowledge spaces of the simpler versions within theirs. The space is, for the sake of simplification, akin to a 3-dimensional hill, where the 2D projection pertains to the policy and the height to the nuances that come with task complexity resulting in higher optimal actions and performances. The ACG experiments failed due to the lack of CL in improving performance on harder tasks through forwards mode curriculum design. In order to solve harder problems with CL further look must be taken at our experiments, the nature of CL, and our application of CL for such problems.

7 Future Work

The findings of this project put CL under a different light than when initially approached, that of unfeasibility. It would appear CL is not the path to go for tackling more difficult problems; however, this is an immense statement to claim. It is certainly one that requires much more experimentation prior to full confirmation. The curricula in this project consisted of only 2 training instances. Experiments with $3, 4$ and 5 curricula should be conducted to gain a more complete view of CL and its effects. Also, curricula of a much wider array of leg length spreads should be tested rather than only lists A and B. Training, on a single instance, was done for an arbitrary amount of time-steps that "seemed" appropriate. This was done to accommodate the time and computational resources of this project, as training till convergence can take incredibly long time periods to complete. Nevertheless, this minor tweak can have very major effects on the results. Moreover, experiments should be conducted to test different combinations of curricula. This project only explored forwards and backwards modes. It is certainly worth exploring different combinations. For example, a curriculum can be 4 instances long, the first 2 can be forwards, then backwards on the third, then forwards again on the fourth. Furthermore, it is important to note that we believe strongly in our choice of this problem, quadrupedal walking, to study CL. Nonetheless, it is of imperative importance to conduct similar experiments on a similarly complex, but completely different set of tasks. Such tasks can be robotic object pick and place and robotic finger dexterity. This is to further validate, or deny, the findings of this report. In all future work, it is imperative to keep testing the results against the knowledge space model proposed in the Discussion section.

8 Conclusion

Quadrupedal locomotion is a difficult problem, but one with many potential applications varying from the military to assisted living. This project aimed to use this difficult problem as a platform for exploring CL. This is because not only is quadrupedal locomotion a complex problem, but it also allows for the easy building of curricula by merely changing the leg length. CL is an attractive sub-field of Deep RL on robotics as it can allow for the deployment of simple policies on robots and then incrementally teaching them more complex tasks. This is in comparison to tackling highly difficult tasks from scratch, which usually results in hardware fatigue or is simply too expensive in terms of time and computation.

Manually designed curricula of different leg lengths were used for experimentation against 4 different hypotheses. The results demonstrated that CL is best suited for improving performance on the simpler task in the curriculum, rather than the more difficult one. The results also showed that backwards mode curricula are the most likely to show such improvements, as well as cause drops in performance variation. These results justify the failure of the ACG experiments conducted in this project. Moreover, the results allowed for a model of the tasks' knowledge space to be built. Although this certainly needs further experimentation for strong validation.

The results posed important questions regarding CL and its potential for paving the way for solving complex tasks. This needs to be deeply explored with further experimentation such as with: changing the curricula lengths, training until convergence, testing wider spreads of leg lengths, trying new combinations of curricula modes, experimenting with a completely different but difficult task, and always comparing the new results against the proposed knowledge space model.

References

Andrychowicz, Marcin, Wolski, Filip, Ray, Alex, Schneider, Jonas, Fong, Rachel, Welinder, Peter, McGrew, Bob, Tobin, Josh, Abbeel, OpenAI Pieter, & Zaremba, Wojciech. 2017. Hindsight experience replay. *Pages 5048–5058 of: Advances in Neural Information Processing Systems*.

Bengio, Yoshua, Louradour, Jérôme, Collobert, Ronan, & Weston, Jason. 2009. Curriculum learning. *Pages 41–48 of: Proceedings of the 26th annual international conference on machine learning*. ACM.

Campion, Guy, & Chung, Woojin. 2008. Wheeled robots. *Springer handbook of robotics*, 391–410.

Florensa, Carlos, Held, David, Wulfmeier, Markus, Zhang, Michael, & Abbeel, Pieter. 2017. Reverse curriculum generation for reinforcement learning. *arXiv preprint arXiv:1707.05300*.

Heess, Nicolas, Sriram, Srinivasan, Lemmon, Jay, Merel, Josh, Wayne, Greg, Tassa, Yuval, Erez, Tom, Wang, Ziyu, Eslami, Ali, Riedmiller, Martin, *et al.* . 2017. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*.

Held, David, Geng, Xinyang, Florensa, Carlos, & Abbeel, Pieter. 2017. Automatic goal generation for reinforcement learning agents. *arXiv preprint arXiv:1705.06366*.

Henderson, Peter, Islam, Riashat, Bachman, Philip, Pineau, Joelle, Precup, Doina, & Meger, David. 2017. Deep reinforcement learning that matters. *arXiv preprint arXiv:1709.06560*.

Khalil, Ahmed. 2019. Dissertation Research Plan. Unpublished.

Lazaric, Alessandro. 2012. Transfer in reinforcement learning: a framework and a survey. *Pages 143–173 of: Reinforcement Learning*. Springer.

Levine, Sergey, Pastor, Peter, Krizhevsky, Alex, Ibarz, Julian, & Quillen, Deirdre. 2018. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, **37**(4-5), 421–436.

Matiisen, Tambet, Oliver, Avital, Cohen, Taco, & Schulman, John. 2017. Teacher-student curriculum learning. *arXiv preprint arXiv:1707.00183*.

Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David, Graves, Alex, Antonoglou, Ioannis, Wierstra, Daan, & Riedmiller, Martin. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David, Rusu, Andrei A, Veness, Joel, Bellemare, Marc G, Graves, Alex, Riedmiller, Martin, Fidjeland, Andreas K, Ostrovski, Georg, *et al.* . 2015. Human-level control through deep reinforcement learning. *Nature*, **518**(7540), 529.

Mnih, Volodymyr, Badia, Adria Puigdomenech, Mirza, Mehdi, Graves, Alex, Lillicrap, Timothy, Harley, Tim, Silver, David, & Kavukcuoglu, Koray. 2016. Asynchronous methods for deep reinforcement learning. *Pages 1928–1937 of: International conference on machine learning*.

Narvekar, Sanmit, & Stone, Peter. 2018. Learning Curriculum Policies for Reinforcement Learning. *arXiv preprint arXiv:1812.00285*.

Narvekar, Sanmit, Sinapov, Jivko, Leonetti, Matteo, & Stone, Peter. 2016. Source task creation for curriculum learning. *Pages 566–574 of: Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems.

Narvekar, Sanmit, Sinapov, Jivko, & Stone, Peter. 2017. Autonomous Task Sequencing for Customized Curriculum Design in Reinforcement Learning. *Pages 2536–2542 of: IJCAI*.

Peng, Bei, MacGlashan, James, Loftin, Robert, Littman, Michael L, Roberts, David L, & Taylor, Matthew E. 2016. An empirical study of non-expert curriculum design for machine learners. *In: Proceedings of the IJCAI Interactive Machine Learning Workshop*.

Riedmiller, Martin, Hafner, Roland, Lampe, Thomas, Neunert, Michael, Degrave, Jonas, Van de Wiele, Tom, Mnih, Volodymyr, Heess, Nicolas, & Springenberg,

Jost Tobias. 2018. Learning by playing-solving sparse reward tasks from scratch. *arXiv preprint arXiv:1802.10567*.

Schaeffer, Paul J, & Lindstedt, Stan L. 2013. How animals move: comparative lessons on animal locomotion. *Comprehensive Physiology*, **3**(1), 289–314.

Schulman, John, Levine, Sergey, Abbeel, Pieter, Jordan, Michael I, & Moritz, Philipp. 2015. Trust Region Policy Optimization. *Pages 1889–1897 of: Icml*, vol. 37.

Schulman, John, Wolski, Filip, Dhariwal, Prafulla, Radford, Alec, & Klimov, Oleg. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shannon, Jack, & Grzes, Marek. 2018. Reinforcement Learning using Augmented Neural Networks. *CoRR*, **abs/1806.07692**.

Silva, Felipe Leno Da, & Costa, Anna Helena Reali. 2018. Object-Oriented Curriculum Generation for Reinforcement Learning. *Pages 1026–1034 of: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems.

Silver, David, Huang, Aja, Maddison, Chris J, Guez, Arthur, Sifre, Laurent, Van Den Driessche, George, Schrittwieser, Julian, Antonoglou, Ioannis, Panneershelvam, Veda, Lanctot, Marc, *et al.* . 2016. Mastering the game of Go with deep neural networks and tree search. *nature*, **529**(7587), 484.

Skinner, Burrhus F. 1958. Reinforcement today. *American Psychologist*, **13**(3), 94.

Sukhbaatar, Sainbayar, Lin, Zeming, Kostrikov, Ilya, Synnaeve, Gabriel, Szlam, Arthur, & Fergus, Rob. 2017. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407*.

Sutton, Richard S, & Barto, Andrew G. 2018. *Reinforcement learning: An introduction*. MIT press.

Svetlik, Maxwell, Leonetti, Matteo, Sinapov, Jivko, Shah, Rishi, Walker, Nick, & Stone, Peter. 2017. Automatic curriculum graph generation for reinforcement learning agents. *In: Thirty-First AAAI Conference on Artificial Intelligence*.

Taylor, Matthew E, Stone, Peter, & Liu, Yaxin. 2007. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, **8**(Sep), 2125–2167.

Wang, Ziyu, Bapst, Victor, Heess, Nicolas, Mnih, Volodymyr, Munos, Remi, Kavukcuoglu, Koray, & de Freitas, Nando. 2016. Sample efficient actor-critic with experience replay. *arXiv preprint arXiv:1611.01224*.

Whitehead, Kez Smithson. 2018. From Simulation to Reality: Deep Reinforcement Learning for Quadrupedal Robot Locomotion. September. Unpublished.

Williams, Ronald J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, **8**(3-4), 229–256.

Wu, Yuhuai, Mansimov, Elman, Liao, Shun, Radford, Alec, & John, Schluman. 2017a (18 August). *OpenAI Baselines: ACKTR & A2C*. `https://openai.com/blog/baselines-acktr-a2c/`.

Wu, Yuhuai, Mansimov, Elman, Grosse, Roger B, Liao, Shun, & Ba, Jimmy. 2017b. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *Pages 5279–5288 of: Advances in neural information processing systems*.

A Project Code

For the entire project, please refer to `https://github.com/AGKhalil/diss_quad/tree/master`

For the quadrupedal simulation, please refer to `https://github.com/AGKhalil/gym-cl`

For the automatic generation environment, please refer to `https://github.com/AGKhalil/gym-real`

B Ethical Form

University of the West of England
BRISTOL

Faculty of Environment & Technology
Faculty Research Ethics Committee (FET FREC)

ETHICAL REVIEW CHECKLIST FOR UNDERGRADUATE AND POSTGRADUATE MODULES
Please provide project details and complete the checklist below.

Project Details:

Module name	Dissertation
Module code	UFMED4-60-M
Module leader	Maryam Atoofi
Project Supervisor	Mark Hansen
Proposed project title	Investigating Curriculum Learning in Deep Reinforcement Learning

Applicant Details:

Name of Student	Ahmed Khalil
Student Number	18044457
Student's email address	ahmed2.khalil@live.uwe.ac.uk

	CHECKLIST QUESTIONS	Y/N	Explanation
1.	Does the proposed project involve **human tissue, human participants, environmental damage, the NHS, or data gathered outside the UK**?	N	*If the answer to this is 'N' then no further checks in the list need to be considered.*
2.	Will participants be clearly asked to give consent to take part in the research and informed about how data collected in the research will be used?		
3.	If they choose, can a participant withdraw at any time (prior to a point of "no return" in the use of their data)? Are they told this?		
4.	Are measures in place to provide confidentiality for participants and ensure secure management and disposal of data collected from them?		
5.	Does the study involve people who are particularly vulnerable or unable to give informed consent (eg, children or people with learning difficulties)?		

	CHECKLIST QUESTIONS	Y/N	Explanation
6.	Could your research cause stress, physical or psychological harm to anyone, or environmental damage?		
7.	Could any aspects of the research lead to unethical behaviour by participants or researchers (eg, invasion of privacy, deceit, coercion, fraud, abuse)?		
8.	Does the research involve the NHS or collection or storage of human tissue (includes anything containing human cells, such as saliva and urine)?		

Your explanations should indicate briefly for Qs 2-4 how these requirements will be met, and for Qs 5-8 what the pertinent concerns are.

- If Qs 2-4 are answered Yes (Y) and Qs 5-8 are answered No (N), no further reference to the Research Ethics Committee will be required, unless the research plan changes significantly.
- If any of Qs 5-8 are answered Yes (Y), then approval from the Faculty Research Ethics Committee is required *before* the project can start. Approval can take over a month. Please consult with your supervisor about the process.

Your supervisor must check your responses above *before* you submit this form.

Submit this completed form via the *Assignments* area in Blackboard (or elsewhere if so directed by the module leader or your supervisor).

After you have uploaded, your supervisor will confirm that the checklist above has been correctly completed by marking this form as Passed/100% via the *My Grades* link on the Blackboard Welcome tab.

If your submitted answers indicate that further ethical approval is indeed required, then you must also send this completed form to ResearchEthics@uwe.ac.uk

Guidance is available at http://www1.uwe.ac.uk/research/researchethics.
Further guidance can be obtained via the module leader, in the first instance, or the Department's Faculty Research Ethics Committee representatives, including your department's *AHoD for Research*.

Figure 17: Report ethical form showing no ethical clearance necessary for conducting the project.