**AI Project Proposal**

**Literature Search:**

For this project, the group has decided to focus on implementing reinforcement learning within Unity’s [1] 3D simulation environment. This game development platform was selected because of its available content for implementing custom reinforcement learning algorithms using the ML-Agents package [2]. Additionally, Unity has been compared to other simulation platforms [4] and was found to be an effective open-source tool for developing AI agents. There are several example environments [3] that demonstrate the wide range of reinforcement learning applications that can be constructed.

Having decided on a particular software and selecting the ML-agents Unity asset package as a primary resource, a decision was made about whether to create a reinforcement learning application for a real-world task or for a video game AI. The group preferred the former, and additional research was conducted to identify an area serving this purpose. This was found to be robotics, an application-heavy field that is quickly incorporating reinforcement learning to automate robotic tasks across many industries [5]. Sharing a mutual interest in drone technology, the group members have chosen to adapt a similar variation of the work done in [6] within Unity.

**Problem Statement:** *Clearly state the research problem that is being addressed.*

Similar to the objective described in [6], this project aims to control a quadrotor drone for stabilization and navigation by using reinforcement learning. This will be developed entirely in simulation, however in the final presentation an outline of how this can be applied to a physical system will be included.

**Methods**: *Conduct a quick literature search and list a few methods that will be explored. This does not have to be the final list or a detailed review.*

The example environments in [3] utilize many default reinforcement learning algorithms, including proximal policy optimization (PPO), soft actor-critic (SAC), and Deep Q-learning (DQL). In most modern implementations, a neural network is incorporated to help approximate a reward function that will build an effective policy pi(A|S), as shown in the following figure.

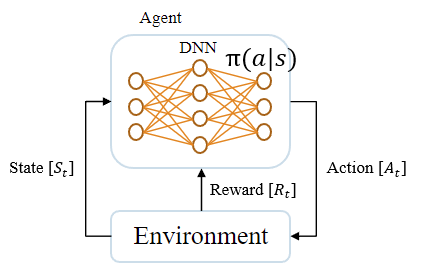


Figure: Deep Neural Networks in Reinforcement Learning

These methods will be tested with the selected application and compared based on their ability to consistently produce a reliable and effective policy after training. Additionally, some parameters will be modified and studied for the best overall method, including the learning rate and target network update rate, as shown in the figure below.

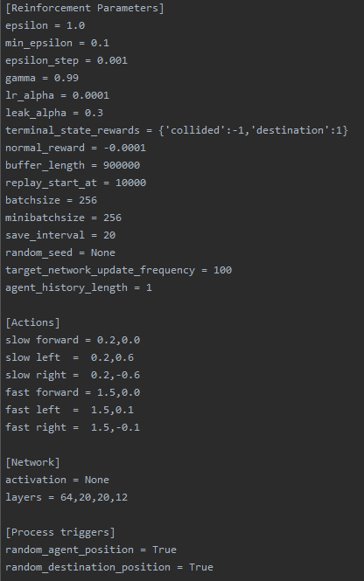


Figure: ML-Agents Reinforcement Learning Configuration Parameters

**Experimental Setup:** *Describe the experimental setup by listing which metrics and datasets will be used for evaluation.*

First and foremost, a custom simulation training environment will be constructed in Unity for implementing reinforcement learning with the chosen application. After configuring the GameObject blocks and necessary components to train a network throughout the learning process, each of the algorithms stated above will be applied to the simulation. Each method will be compared based on the speed of learning an approximated optimal policy as defined by observing the average reward across all episodes, the overall performance of the policy after a fixed number of episodes, and the average number of samples needed for training during each episode. For the particular task being studied in this project, these metrics have been identified in [7], a review of deep reinforcement learning for drone applications.

**(Optional) New Approach**: *If you are planning to come up with a novel idea, provide a rough outline of the research approach.*

If the task can be accomplished successfully using the standard reinforcement learning methods above, and if time permits, a modification / extension of the best method may be applied.

**GITHUB:** *Set up a GitHub repository where all the implemented methods will be hosted. Add me to the repo -* [*https://github.com/kevinpdesai*](https://github.com/kevinpdesai)

**References**

[1] https://docs.unity3d.com/Manual/UnityManual.html

[2] https://github.com/Unity-Technologies/ml-agents

[3] https://github.com/Unity-Technologies/ml-agents/blob/main/docs/Learning-Environment-Examples.md

[4] Juliani, A., Berges, V. P., Teng, E., Cohen, A., Harper, J., Elion, C., ... & Lange, D. (2018). Unity: A general platform for intelligent agents. arXiv preprint arXiv:1809.02627.

[5] Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, *32*(11), 1238-1274.

[6] Hwangbo, J., Sa, I., Siegwart, R., & Hutter, M. (2017). Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters*, *2*(4), 2096-2103.

[7] Azar, A. T., Koubaa, A., Ali Mohamed, N., Ibrahim, H. A., Ibrahim, Z. F., Kazim, M., ... & Casalino, G. (2021). Drone deep reinforcement learning: A review. *Electronics*, *10*(9), 999.