Replication of Multi-Agent Reinforcement Learning for "Hide & Seek" Problem

MCS

by

Muhammad Haider Kamal

Supervisor: Prof. Dr. Muaz Ahmed Khan Niazi

A thesis submitted to the faculty of the Computer Software Engineering Department, Military College of Signals, National University of Sciences and Technology, Rawalpindi in partial fulfillment of the requirements for the degree of MS in Software Engineering

April 2023

Replication of Multi-Agent Reinforcement Learning for “Hide & Seek” Problem

by

Muhammad Haider Kamal

NS 00000328503

Supervisor: Prof. Dr. Muaz Ahmed Khan Niazi

A thesis submitted to the faculty of the Computer Software Engineering Department, Military College of Signals, National University of Sciences and Technology, Rawalpindi in partial fulfillment of the requirements for the degree of MS in Software Engineering

April 2023

Thesis Acceptance Certificate

This is to certify that the final copy of the thesis written by NS **Muhammad Haider Kamal**, Registration no: **00000328503,** Course: **MSSE-27,** on the topic, **"Replication of Multi-Agent Reinforcement Learning for "Hide & Seek" Problem"** has been found complete. The plagiarism report is satisfactory and necessary amendments pointed out by GC members are incorporated.

Signature: ___________________________________

Name of Supervisor: Prof. Dr. Muaz Ahmed Khan Niazi

Date: ____

Signature (HOD): ________________________________

Date: ____

Signature (Dean/Principal): ________________________

Date: _______________________________________

Declaration

I, Muhammad Haider Kamal declare that this thesis titled **“Replication of Multi-Agent Reinforcement Learning for “Hide & Seek” Problem”** and the work presented in it is my own and has been generated by me as a result of my own original research.
I confirm that:

1. This work was done wholly or mainly while in candidature for a Master of Software Engineering degree in MCS NUST.

2. Where any part of this thesis has previously been submitted for a degree or any other qualification at NUST or any other qualification at NUST or any other institution this has been clearly stated.

3. Where I have consulted the published work of others, this is always clearly attributed.

4. Where I have quoted from the published work of others the source is always given. With the exception of such quotations this thesis is entirely my own work.

5. I have acknowledged all main sources of help.

6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Muhammad Haider Kamal.

00000328503

MSSE27

Dedication

“In the name of Allah, the most Beneficent, the most Merciful"

I dedicate this thesis to my parents and teachers.

Who supported me each step of the way.

Acknowledgments

All praise to Allah for the strength and blessings to complete my thesis.

I would like to convey my gratitude to my supervisor, Prof. Dr. Muaz Ahmed Khan Niazi, and Assoc. Prof. Dr. Hammad Afzal for their supervision and constant support. Their continuous help, constructive comments, and suggestions throughout the experimental and thesis works are major contributions to the success of this research. Also, I would thank my committee members; Assoc Prof Dr. Naima Iltaf, and Lt. Col Khawir Mehmood for their support and knowledge regarding this topic.

Last, but not least, I am highly thankful to my parents. They have always stood by my side and have been a great source of inspiration for me. I would like to thank all of them for their support through my times of stress and excitement.

Abstract

Reinforcement learning generates policies based on reward functions and hyperparameters. Slight changes in these can significantly affect results. The lack of documentation and reproducibility in Reinforcement learning research makes it difficult to replicate once-deduced strategies. While previous research has identified strategies using grounded maneuver, there is limited work in the more complex environments. The agents in this study are simulated similarly to Open Al’s hide and seek agents, in addition to a flying mechanism, enhancing their mobility, and expanding their range of possible actions and strategies. This added functionality improves the Hider agents to develop chasing strategy from approximately 2 million steps to 1.6 million steps and hiders shelter strategy from approximately 25 million steps to 2.3 million steps while using a smaller batch size of 3072 instead of 64000. We also discuss the importance of reward functions design and deployment in a curriculum-based environment to encourage agents to learn basic skills along with the challenges in replicating these Reinforcement learning strategies. We demonstrated that the results of the reinforcement agent can be replicated in more complex environment and similar strategies are evolved including” running and chasing” and ”fort building”.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

The word “Artificial” means a replica produced by humans [1]. But replicating AI results and strategies is considered difficult in large part [2]. Overall low replication rates suggest unreliable practices (Christensen and Miguel, 2018). Reinforcement learning (RL) generates policies based on designed reward functions and assigned hyperparameters. The Reward Function is an incentive mechanism that uses reward and punishment to tell the agent what is right and wrong. In RL, the goal of agents is to maximize total rewards. We must sometimes forego immediate gratification to maximize total rewards. Slight changes in these can yield a huge difference in results. Reinforcement learning is also affected by experimental conditions i.e. if the gravity of physics-based simulation or force applied to a working agent changes it can change the behavior pattern of the agent. Thus, replication of once-deduced strategies is difficult and criticized for not being reproducible. RL research tends to be not documented well enough to reproduce the exact reported results as it mostly relies on continuous finetuning and updating of hyperparameters, reward functions, environment variables, sensor types, etc. Inference and result reproducibility should yield enough similar results that can benefit further research and can ease improvements rather than just focusing on reproducing it. But upon minimum info given regarding Agent’s Academy parameters (Environment Variables like gravity, collider conditions, delta time, etc.), Agent’s Behavior parameters (vector space, continuous/discrete actions inference device, collider type, speed, sensors attached, max steps allowed, etc.), calculating similar results are often considered hard to achieve.

The Tool Use from Multi-Agent Interaction paper by OpenAI explores the concept of agents in emergent strategies [3]. In this paper, an agent is defined as a software entity that is capable of perceiving its environment and taking action to achieve a specific goal. Such strategies can either be generated by designing the reward function specified to goal-achieving parameters or waiting for the lucky shot action that the agent performs while moving randomly. This goal of achieving random then becomes the near to best action so far in the agent’s experience and is used to generate optimal policy. The drawback of this approach is that the agent might never learn to reach the optimal solution if the experience that it is gathering is not fruitful. For example, if the agent’s goal is vertically upward and behind a locked door, an agent might take forever to understand a pattern to first move toward a specific door, unlocking it, passing through it, and tagging the target goal. Thus, specially designed reward functions are deployed with a simplistic environment at the beginning to encourage the agent to learn a basic and novice version of the best state of action. The Open AI paper presents a series of experiments in which agents are placed in a simulated environment and tasked with completing a set of objectives. The agents are programmed to learn from their interactions with each other, leading to the development of strategies such as cooperation and tool use. Our agents are simulated with the inclusion of a flying mechanism, enabling them to navigate through three-dimensional space. This feature enhances the agents' mobility and expands their range of possible actions, leading to more diverse and effective behavior. With the ability to fly, our agents can overcome obstacles and traverse complex environments more efficiently, ultimately improving their performance and increasing their chances of success. This added functionality can also allow for the emergence of novel behaviors and strategies, further improving the agents' ability to achieve their goals.

1.2 Motivation and Problem Statement

It has been observed that agents tend to acquire better learning outcomes in a simplified environment [4]. However, such agents may not be well-suited for more complex environments and may exhibit a bias towards them. To address this challenge, agents can benefit from the implementation of curriculum learning. This approach involves gradually increasing the difficulty of the environment once the agent reaches a certain reward threshold, allowing them to better adapt to complex scenarios.

Recent observations of agents in OpenAI's program indicate a preference for 2-dimensional movement, limiting their ability to navigate in three dimensions [3]. However, introducing a third movement direction, such as up and down, can enable agents to fly and discover novel behaviors, leading to more optimal solutions.

Additionally, the limitations of 2-dimensional lidar-like ray cast in detecting opposing agents above or below call for the redesign of observation sensors. This modification can provide agents with a more comprehensive understanding of their surroundings and improve their situational awareness when an ally or opposing team's agent is present.

We are interested in examining the effects of reducing or eliminating negative rewards for behavior that leads to delayed rewards. For instance, seekers are currently penalized with a negative reward if they do not have hiders in sight, even if they are moving in the right direction toward the hiders and are only a few frames away from reaching an optimal position.

We intend to investigate how the elimination or reduction of these negative rewards would affect the behavior of the agents. By doing so, we can better understand how agents perceive and respond to delayed rewards and whether removing negative rewards in these situations leads to more optimal outcomes. This analysis can provide valuable insights into how to incentivize agents to navigate more effectively toward their goals, even in situations where rewards are delayed.

1.3 Objectives

The primary objective of this research is to achieve the following goals:

- We aim to enhance our agents' abilities by introducing drone-like behavior through the addition of flying movements. This feature enables agents to observe and interact with their environment more effectively, leading to improved performance and better results.
- Our objective is to create a competitive multi-agent environment utilizing MA-POCA (MultiAgent POsthumous Credit Assignment) and PPO (Proximal Policy Optimization algorithm) reinforcement learning techniques. The implementation of these methods can facilitate the development of effective strategies among agents, resulting in improved performance and more successful outcomes.
- We aim to fine-tune reward functions and hyperparameters as accurately as possible, based on the existing documentation. This approach can optimize the performance of our agents, enabling them to achieve better results and more efficiently reach their goals.
- Our objective is to showcase the potential of our proposed model by conducting various experiments and comparative studies on reinforcement learning projects using state-of-the-art approaches. This approach can demonstrate the effectiveness and superiority of

our proposed model and enable us to further improve and optimize our agents' performance.

By accomplishing these goals, we can improve our understanding of reinforcement learning agents and contribute to the development of more advanced and smart agents.

1.4 Thesis Contribution

This thesis makes a meaningful contribution to the field of reinforcement learning by proposing new ideas and enhancements that can significantly augment the behavior of multi-agent reinforcement drone bots. The proposed enhancements aim to enable agents to learn tool use in a 3-dimensional environment using a combination of competition and cooperation. In addition, the thesis proposes the implementation of curriculum learning to enhance the agents' understanding of the environment perception and how to take action given the specific state they are in.

The proposed enhancements have the potential to improve the performance and effectiveness of reinforcement learning drone agents, enabling them to better navigate complex environments and achieve more optimal outcomes. The use of competition and cooperation can encourage agents to work together towards a common goal, leading to more efficient and effective decision-making. The implementation of curriculum learning can help agents gradually improve their skills and understanding, leading to better overall performance.

Overall, the contributions are offering new insights and approaches to replicating the behavior of OpenAI’s hide-and-seek agents. The proposed enhancements have the potential to make reinforcement learning more effective and applicable to a wider range of real-world scenarios, making them a valuable addition to the field.

1.5 Background

B. Baker et al proposed a model in which self-play was accomplished using genetic algorithms, intrinsic motivation methods, count-based exploration, and transition-based methods. [3] This is done because, in the previous scenarios, agents were explicitly incentivized to interact with and use tools, whereas, in this scenario, environment agents create this incentive implicitly through multi-agent competition. The methodology is as Hiders are given a reward of 1 if all hiders are hidden and a reward of -1 if any hider is seen by a seeker. Seekers receive the inverse reward: -1 if all hiders are hidden, and +1 otherwise. Agents are penalized with a -10 reward if they venture too far outside of the play area (outside an 18-meter square). A policy network generates an action distribution, and a critic network forecasts discounted future returns. As a result, six emerging strategy phases are introduced. Our approach is via rewarding agents every frame for optimal action and not giving rapid negative rewards. The seekers are given a +0.001 reward if any seeker can have hiders in their field of vision while hiders get a +0.0001 reward every frame they are hidden from the seekers.

E Alonso et al created a navigation system combining the 3D occupancy map, 2D depth map, and absolute goal and agent positions, passing through independent feature extraction layers (3D convolutions, 2D convolutions, and linear layers respectively). [4] The output of each feature extractor is then combined with other state variables, such as relative goal position, speed, acceleration, and previous action. The combined output is fed through several linear layers, followed by an LSTM, to create the final embedding shared by both the policy and critic heads. Our approach was to introduce a frontal vision sensor and spatial grid sensor. The essential functionality needed for visual grid observations is provided by the GridSensorComponent. It offers a GridBuffer that may be used to write normalized float data and encapsulates a GridSensor. I have also provided the shape options. If performing PNG compression on the observation, a ColorGridBuffer is generated (which extends GridBuffer and holds color values). The sensor is based on points rather than verifying collider overlaps for each grid place. This implies that the shape of an item is represented as a collection of local points that have been stored and are subsequently translated into the sensor's frame of reference. This approach has the advantage of scaling well for numerous sensor instances because it needs far fewer overlap checks.

A. Tucker et al deduced a model in which CNN-AIRL is used to reduce the image space and then feed the Adversarial IRL, which is a continuous action space and uses a fully connected policy and reward network [5]. They use Proximal Policy Optimization (PPO) to train the policy network instead of Trust-Region Policy Optimization (TRPO). M. Lukas developed a multi-agent competition environment in which each Bot, using a State Machine, has 6 states. The Security Agents use Proximity Policy Optimization (PPO) reinforcement learning algorithms which observe the Player's positions, rotation, velocity, carrying a box, box positions of your team and rotations, blue team score, red team score, and deduce if a player is a player, neutral or bot (true, neutral or false) and vice versa. 2 tests were performed. Test 1 with the Player on Red Team and Bot on the Blue team, this test failed when the player changed teams because the agent needed a more diverse environment to learn, So, the 2nd Test was performed having the Player and Bot both at each team. In my approach, 2 separate training sessions were held in which once the seekers are trained enough for the "running and chasing" task, the training of hiders begins. This is because hiders needed a much smarter agent to train against in competition. They needed to observe the severity of staying outside the safe room and not locking the doors and windows. Thus the hiders were trained using the Proximity Policy Optimization (PPO) reinforcement learning algorithms while the seekers are trained using the MultiAgent POsthumous Credit Assignment (MA-POCA).

Multi-agents gain skills including scouting for foes in new regions, looking for cartridges after an injury, and delivering ammo to many threatening targets. J. Lai et al utilized the PPO method which determines a conservative lower limit of the objective function using a simple clip, boosting sampling efficiency. [6] Two networks are being trained by ActorCritic. One network estimates or criticizes Q-values, while the other decides the actor's or agent's policy or behaviors. [19] Our method is using the curriculum learning approach in which lowering the complexity of the environment increases the agent's learning speed and makes the agents more dynamic to changing environment.

I. M. A. Nahrendra et al proposed the Retro-RL paper which intends a method for training a deep reinforcement learning algorithm to improve the control of a tilting-rotor drone. [7] The approach involves combining a nominal controller with a deep Q-network (DQN) that learns to adjust the controller's output based on the drone's state and the desired trajectory. The authors evaluate their approach in simulation and on a physical drone, demonstrating improved performance compared to the nominal controller alone. However, the approach requires a large amount of training data and may not be easily generalizable to other drone platforms or control tasks. Our approach uses the 3-axis based locomotion in the x, y, and z axis while having a rotation on the y-axis with the ability to drag the objects as well.

R. Zhang et al paper "Game of Drones: Multi-UAV Pursuit-Evasion Game with Online Motion Planning by Deep Reinforcement Learning" proposes a method for online motion planning and decision-making in a multi-UAV pursuit-evasion game using deep reinforcement learning (RL). The authors design a neural network-based policy that takes as input the positions and velocities of all UAVs and outputs the actions for each UAV. [8] They use an actor-critic algorithm to learn this policy from scratch, without relying on any prior knowledge of the environment or system dynamics. The proposed method is evaluated in a simulated multi-UAV game, where a team of pursuer UAVs tries to capture a single evader UAV. The results show that the proposed method outperforms several baseline methods, including a centralized planner and a non-learning decentralized planner. The authors also demonstrate the scalability of their method by increasing the number of UAVs in the game. The paper contributes to the growing field of using RL for decision-making and control in multi-agent systems. The proposed method has potential applications in surveillance, security, and search and rescue scenarios where multiple UAVs must work together to achieve a common goal. However, the proposed method only considers a simplified pursuit-evasion game, and further research is needed to evaluate its performance in more complex scenarios. Our approach is a multi-agent competition where team A will compete with an already trained and smart team B. This is done as agents learn better in the simpler and more non-complex environment faster rather than against a super brain or very complex environment.

A. Devo et al paper "Autonomous Single-Image Drone Exploration with Deep Reinforcement Learning and Mixed Reality" proposes a method for autonomous exploration with a drone using deep reinforcement learning (RL) and mixed reality. [9] The authors train a neural network-based policy using RL to control the drone's movement and optimize its exploration behavior based on a single RGB image as input. The proposed method is evaluated in both simulation and a real-world environment using a DJI Mavic Pro drone. The results show that the proposed method outperforms several baseline methods, including random exploration and an information-gain-based exploration algorithm. The authors also introduce a mixed reality interface that provides the user with a first-person view of the drone's perspective, enabling real-time feedback and intervention to adjust the exploration trajectory. The paper contributes to the growing field of using RL and mixed reality to enable more autonomous and intuitive control of drones. The proposed method has potential applications in search and rescue, inspection, and surveillance scenarios where a drone needs to autonomously explore and map an unknown environment. However, the proposed method assumes a known environment, and further research is needed to evaluate its performance in more complex and dynamic environments. Additionally, the mixed reality interface requires a human operator, and future work could explore ways to make the exploration process more fully autonomous. Our approach is using mbaske’s 3d grid sensors that help create a frontal sensor that is used to generate shapes and get position data of detectable objects. And a spatial sensor that detects the position of the objects around the agent.

G. Wu et al proposed the "Reinforcement Learning based Truck-and-Drone Coordinated Delivery" paper which intends a method for optimizing the delivery process of a truck-and-drone system using reinforcement learning (RL). [10] The authors use a deep RL algorithm to learn the optimal coordination strategy between a truck and a drone to minimize the overall delivery time. The proposed method is evaluated in a simulated delivery scenario, where the truck and drone must coordinate to deliver packages to different locations. The results show that the proposed method outperforms several baseline methods, including a random coordination strategy and a heuristic-based coordination strategy. The paper contributes to the growing field of using RL for decision-making and control in multi-agent systems. The proposed method has potential applications in logistics and transportation, where a truck-and-drone system can be used to deliver goods more efficiently and cost-effectively. However, the proposed method only considers a simplified delivery scenario, and further research is needed to evaluate its performance in more complex and dynamic environments. Additionally, the proposed method assumes that the locations of the packages are known in advance, and future work could explore ways to incorporate uncertainty and adaptability into the delivery process. Our approach uses a curiosity hyperparameter to increase agents' randomness and promote the explore ways to incorporate vagueness.

C. de Souza et al proposed the "Decentralized Multi-Agent Pursuit Using Deep Reinforcement Learning" paper proposes a method for coordinating a team of agents to pursue a target using deep reinforcement learning (RL). [11] The authors use a deep Q-network (DQN) to learn a decentralized policy that enables each agent to make independent decisions based on its local observations. The proposed method is evaluated in a simulated pursuit scenario, where a team of agents must coordinate to capture a moving target. The results show that the proposed method outperforms several baseline methods, including a centralized planner and a non-learning decentralized planner. The paper contributes to the growing field of using RL for decision-making and control in multi-agent systems. The proposed method has potential applications in surveillance, security, and search and rescue scenarios where multiple agents must work together to achieve a common goal. However, the proposed method only considers a simplified pursuit scenario, and further research is needed to evaluate its performance in more complex and dynamic environments. Additionally, the proposed method assumes that the agents have perfect communication, and future work could explore ways to incorporate communication constraints and failures into the coordination process. Our approach uses PPO and MA-POCA in a competition.

D. Hong et al proposed the "Energy-Efficient Online Path Planning of Multiple Drones Using Reinforcement Learning" paper which suggests a method for optimizing the energy consumption of a fleet of drones during online path planning using reinforcement learning (RL). [12] The authors use a deep RL algorithm to learn a policy that determines the optimal path for each drone to minimize energy consumption, considering the dynamic nature of the environment. The proposed method is evaluated in a simulated scenario, where a fleet of drones must visit a set of target locations while minimizing their energy consumption. The results show that the proposed method outperforms several baseline methods, including a random path planner and a heuristic-based path planner. The paper contributes to the growing field of using RL for decision-making and control in drone systems. The proposed method has potential applications in surveillance, inspection, and search and rescue scenarios where a fleet of drones must operate for extended periods and cover large areas. However, the proposed method only considers a simplified path-planning scenario, and further research is needed to evaluate its performance in more complex and dynamic environments. Additionally, the proposed method assumes that drones have perfect communication and coordination, and future work could explore ways to incorporate communication constraints and failures into the planning process. Our approach introduces curriculum learning where the complexity of the environment is gradually increased in four different stages. Level one has one doorway and one prop. Level two has one doorway, one window, and two props, level three has two doorways, one window, and three props while level four has two doorways two windows, and four props.

1.6 Thesis Organization

The structure of the thesis is as follows:

- Chapter 2 contains the literature reviewed in the thesis. The previous related work regarding reinforcement learning multi-agents. Competing and cooperating techniques and existing case study-based approaches applied to the learning agents are covered in the chapter.

- Chapter 3 covers the method and research methodology, details of the structure of 3-dimensional agents, frontal camera sensor, spatial sensors, and method used in conducting the experiments and environmental setups.

- Chapter 4 contains the reinforcement learning models executed. The simulation results for each model and the attributes used are discussed in detail. The discussion and Analysis of the results are also part of this chapter.

- Chapter 5 is the concluding chapter. The conclusion and future research gaps are described in this chapter.

Chapter 2

Method

2.1 Agent Structure

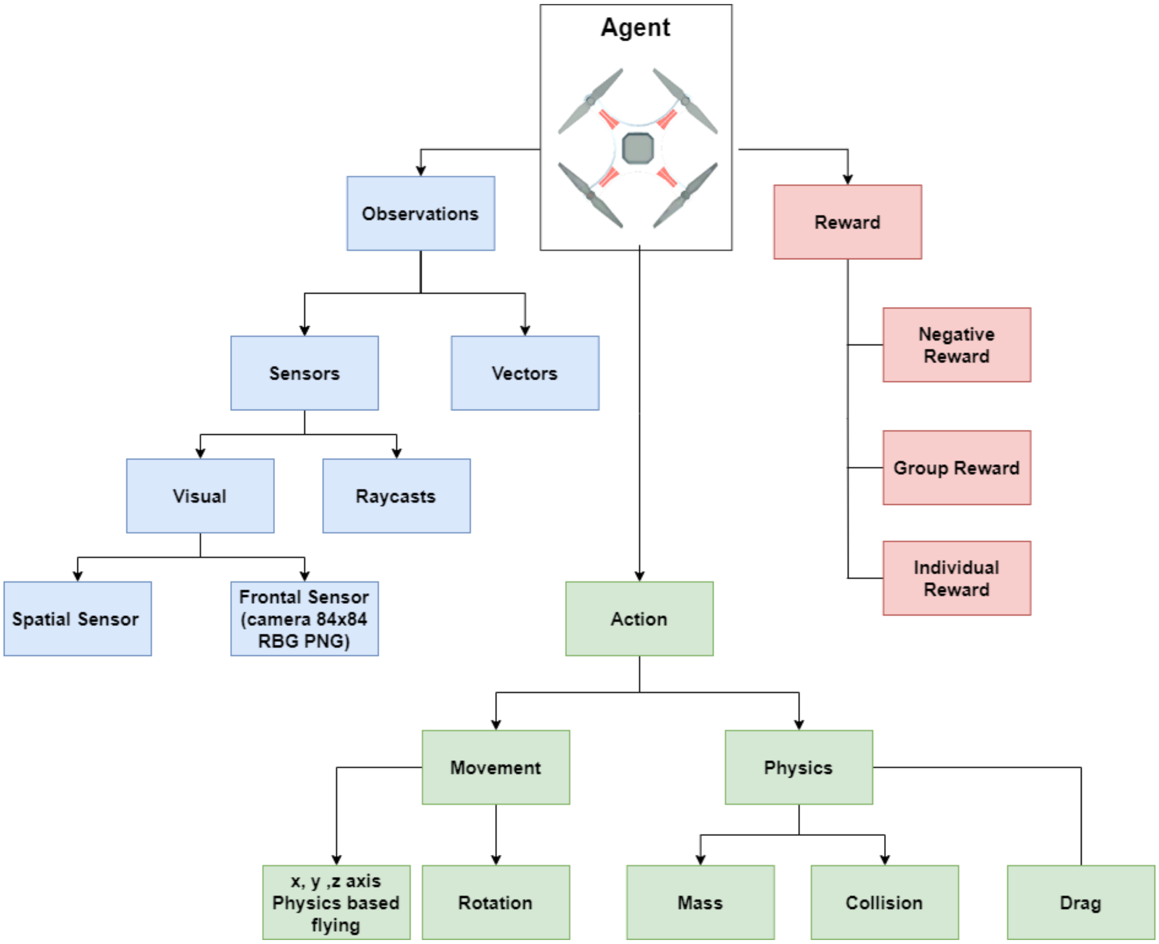


Figure 1 High-level architectural design of Agent

Figure 1 represents the architectural structure of an agent. The types of observations it's receiving, the reward types it's getting, and the pilot parameters that include movement and physics-related types.

To explain further, the upcoming section of this chapter includes descriptions regarding the following:

2.2 Agent:

An agent is characteristically represented as a neural network that takes in Observational inputs from its environment and outputs actions. The agent learns to perform these actions in response to its environment by training on a dataset of observations and rewards. The following are the three observation types our agents are receiving which are thoroughly explained further in this chapter:

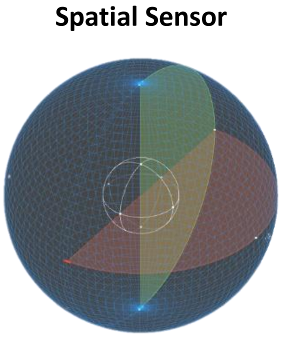


Figure 2 "360°" in 3d space

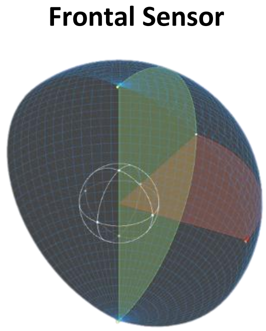


Figure 3 "135°" in 3d space

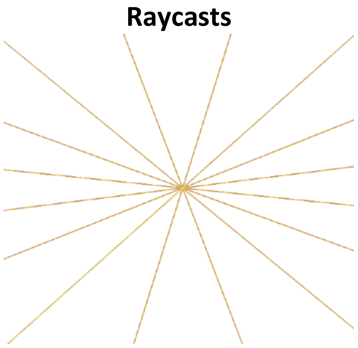


Figure 4 "360°" in 2d space

2.2.1 Hider Agents:

One of our agents, whose objective is to flee from the seeker agents and build a shelter out of the environment's objects. These agents are rewarded when they are successful in hiding from seekers either by staying out of their sight or by building a fort for more security.

2.2.2 Seeker Agents:

The other type of our agents whose objective is to find and tag hider agents. They must learn to navigate toward the hiders and collide with them. Seeker agents get rewards if hider agents are in their sight or they tagged any hider. These seekers are responsible to learn movement in flying like drone maneuvers, identify the target (Hiders), locate that target, adjust it to face directly to the target, and move toward it avoiding obstacles and static walls.

2.2.3 Neural Network structure:

Agents are assigned 256 hidden units along with 2 hidden layers.

2.3 State Abstraction

A special Grid sensor is designed for capturing agents in 3d space. It contains a set of sensors that capture the state of the environment by dividing the space into a 3D grid of cells and recording the occupancy or other features of each cell.

2.3.1 Grid Construction:

To construct the grid, we divide the 3D space into a set of uniformly sized cells. Each cell is defined by its center point, which can be calculated as:

$$C_i = B_i + \left(\frac{1}{2}\right) * S_i$$

Were:
C_i is the center point of the i-th cell.
B_i is the bottom-left corner of the i-th cell.
S_i is the size of the i-th cell.

2.3.2 Occupancy Calculation:

The occupancy of each cell in the grid is calculated by checking if any part of the object or agent is inside the cell. This is done using a binary function $B(x)$ that returns 1 if x is inside the object and 0 otherwise. The occupancy O_i of the i-th cell can be calculated as:

$$o_i = max_{j \in O} B\left(c_i - p_j\right)$$

Where:
O is the set of all objects in the environment.
P_i is a point on the surface of the i-th object.
$B(C_i - P_i)$ is a binary function that returns 1 if x is inside the object and 0 otherwise.

2.3.3 Feature Calculation:

In addition to occupancy, we can also calculate other features of each cell, such as distance to the nearest object or the average color of objects inside the cell. The feature value $f_{i,k}$ of the k-th feature in the i-th cell can be calculated as:

$$f_{i,k} = n \sum_{j \in O} B\left(c_i - p_j\right) f_{j,k}$$

Where:
O is the set of all objects in the environment.
P_i is a point on the surface of the i-th object.
$B(C_i - P_i)$ is a binary function that returns 1 if x is inside the object and 0 otherwise.
n is the number of objects in the environment.

2.3.4 Frontal Shape Observation

The Frontal Shape Sensor is a type of visual sensor that captures the shape and appearance of objects in the environment from a frontal perspective. The sensor works by capturing an image of the environment from the agent's current position and angle and then processing the image to detect and classify objects based on their shape and appearance.

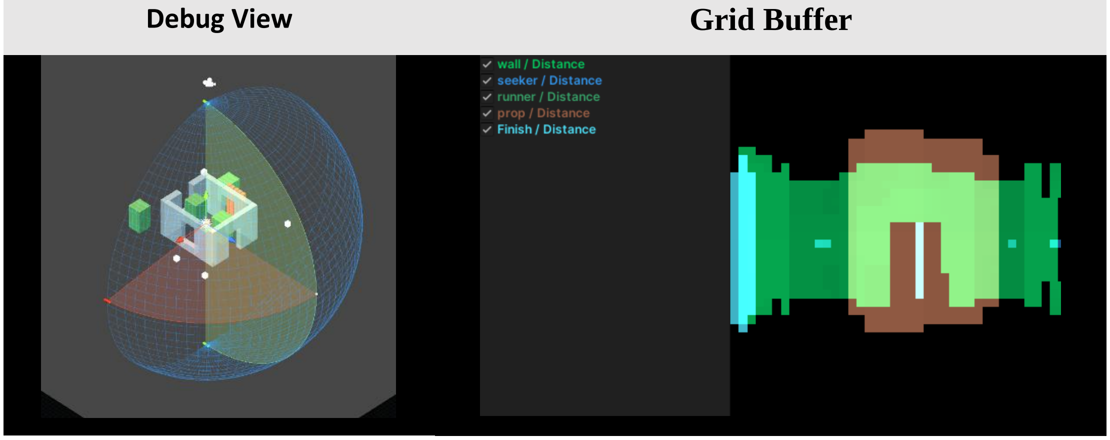


Figure 5 Frontal Shape Sensor for Agent's Observation

The frontal field of vision is responsible for getting the position and shape of detectable objects in Infront of the viewing agent. The initial collider buffer is set to 400 and if in case of agents detect more; the buffer will double itself. The latitude angle north and south are set to 90 degrees each while the longitudinal angle is set to 84 degrees yielding a smaller field of vision. The arc angle of a single FOV grid cell in degrees. Determines the sensor resolution:

$$C_d = \pi * 2 * D \frac{360}{C_d}$$

where Cd is equal to cell size at distance and Ca equals cell arc value.

The following is the terminology used to setup values in the sensors:

Lat Angle North - The FOV's northern latitude (up) angle in degrees.

Lat Angle South - The FOV's southern latitude (down) angle in degrees.

Lon Angle - The FOV's longitude (left & right) angle in degrees.

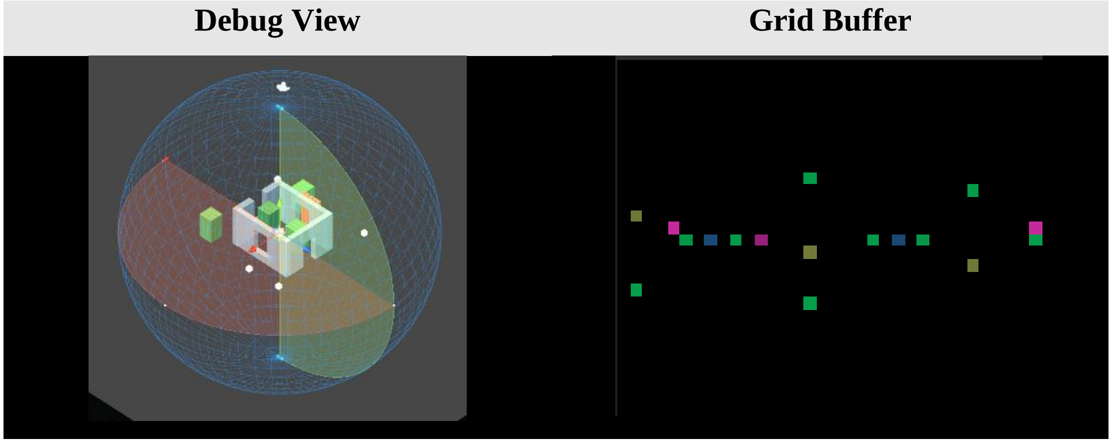
Min Distance - The lowest possible detection distance (near clipping).

Max Distance - The upper limit detection distance (far clipping).

Normalization - How to normalize object distances. 1 for linear normalization. Set the value to < 1 if observing distance changes at close range is more critical to agents than what happens farther away.

2.3.5 Spatial Position Observation

The Spatial Sensor can be used to detect the presence of specific objects in the environment and can provide information about the relative location and orientation of those objects with respect to the agent. The Spatial field of vision which is the surround positioning lidar used to identify the position, as well as the distance to the surrounding detectable objects its initial collider buffer, is set to 32, and if in case of agents detect more the buffer will double itself. The latitude angle north and south are set to 90 degrees each while the longitudinal angle is set to 180 degrees yielding a 360 field of vision. and 2d ray casts pointing outward along the x and z-axis around the agent and agents’ velocity with its facing direction along the z-axis.


Debug View

Grid Buffer

Figure 6 Spatial Sensor for Agent's Observation

The Spatial Sensor represents the environment as a set of spatial features. This sensor captures information about the location and characteristics of objects in the environment, such as their position, orientation, and size.

2.3.6 Ray-casts Observation

Agents have 2d Ray-casts that surround them observing. In ML-Agents, Ray-cast Sensors are a type of sensor that provides information about the environment to the agent. A Ray-cast Sensor works by casting a ray or multiple rays from a point on the agent's body to detect objects in the environment. The sensor returns information about the distance, angle, and type of the detected objects. Ray- cast Sensors are commonly used in robotics and game development to simulate perception and enable agents to interact with the environment. In ML-Agents, Ray-cast Sensors can be used to provide the agent with information about the environment, such as the location of obstacles, the distance to objects, and the presence of other agents.

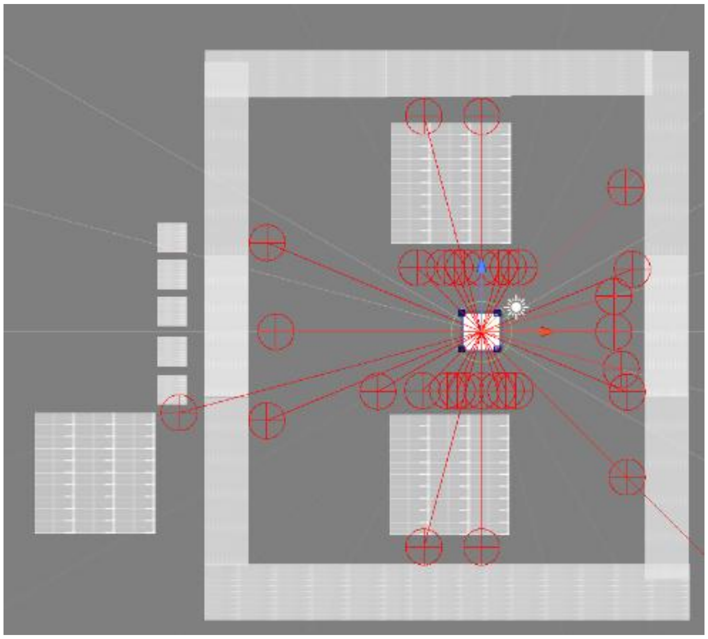


Figure 7 2d Ray casts

The agent has 8 ray casts per direction with a max ray degree of 180 which means by adding both directions we get a 360 view. The sphere ray cast radius is 0.3 while the ray length is set to 20.

2.3.7 Realtime values via script

Values include normalized potion of self, normalized velocity, facing direction vector, normalized rotation, a bool telling if the agent is dragging prop, and a normalized timer that finishes when environmental steps finish.

2.4 State Diagram:

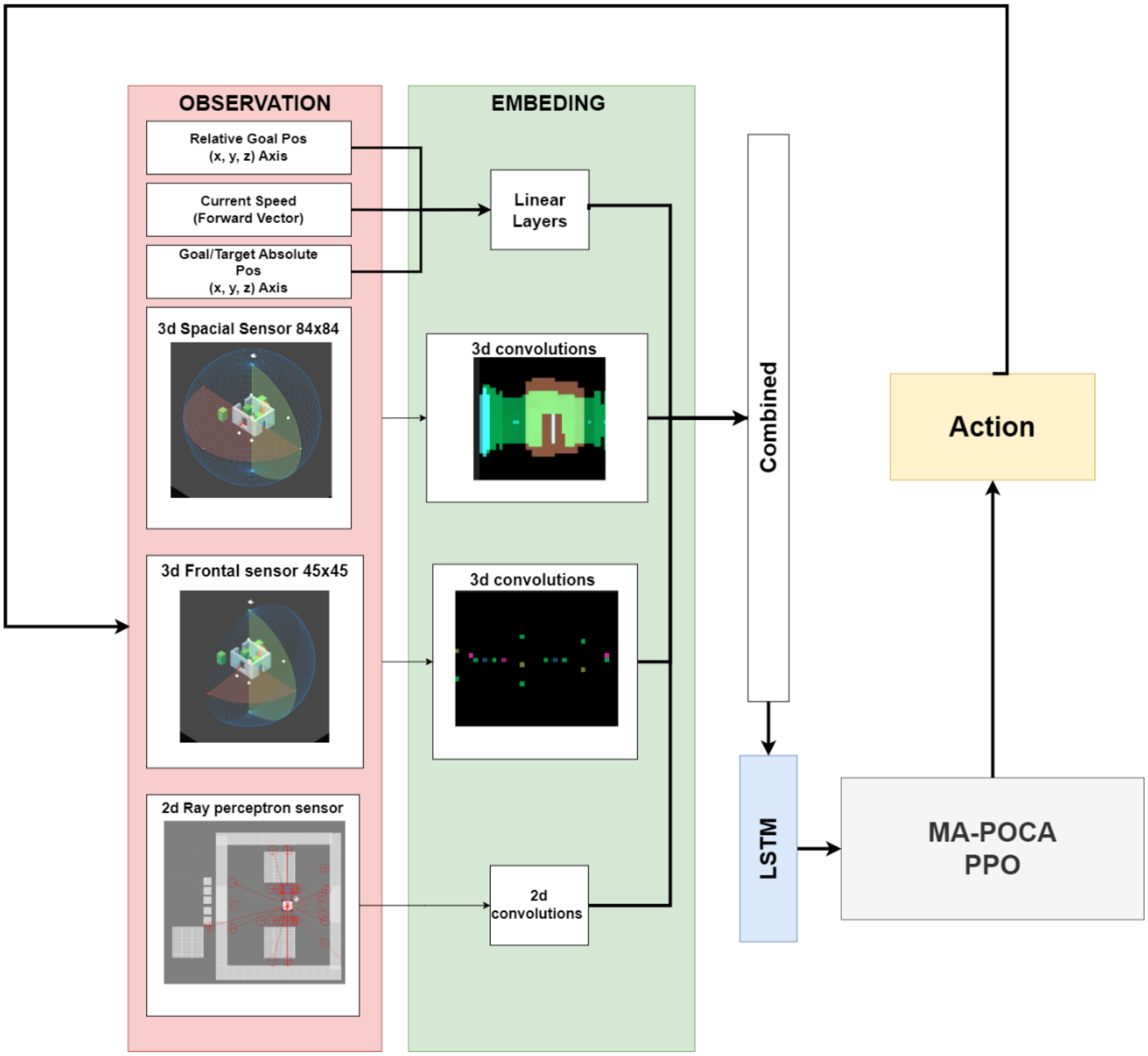


Figure 8 State Diagram of Agents observation to action flow

Figure 8 represents the observation being provided to the agents to generate actions and develop the best policies. The observations are then fed into MA-Poca for Seekers and PPO for Hiders.

Observations including relative position, current velocity, target relative position, 3d spatial sensor, 3d frontal sensor which includes 84x84 RGB camera, and 2d ray-casts are taken from the agent to the observation buffer and sent to generate optimal policy to generate the best result.

2.5 Environmental Design

2.5.1 Pre-Hider and Seeker Experiments

we conducted three different yet progressive experiments including the “hummingbird experiment”, “target drone experiment” and the “eye experiment”.

Hummingbird	Target Drone	The Eye
Info: Training a humming bot to collect nectar [14]	**Info**: Training a drone bot to tag target	**Info**: Training an eye bot to avoid collision with an ally and reach the target
Conditions: Stationary Target Wide boundaries Known Target	**Conditions:** Dynamic Target Normal boundaries Known Target	**Conditions:** Dynamic Target Small boundaries unknown Target

2.5.1.1 Hummingbird Agent - Experiment

Observing Agents Behavior, the Agent seems to wobble around trying different actions to be able to achieve reward and avoid punishment.

Reward = +.01f if Agent is in Nectar

Reward = -0.5 if Agent Collides

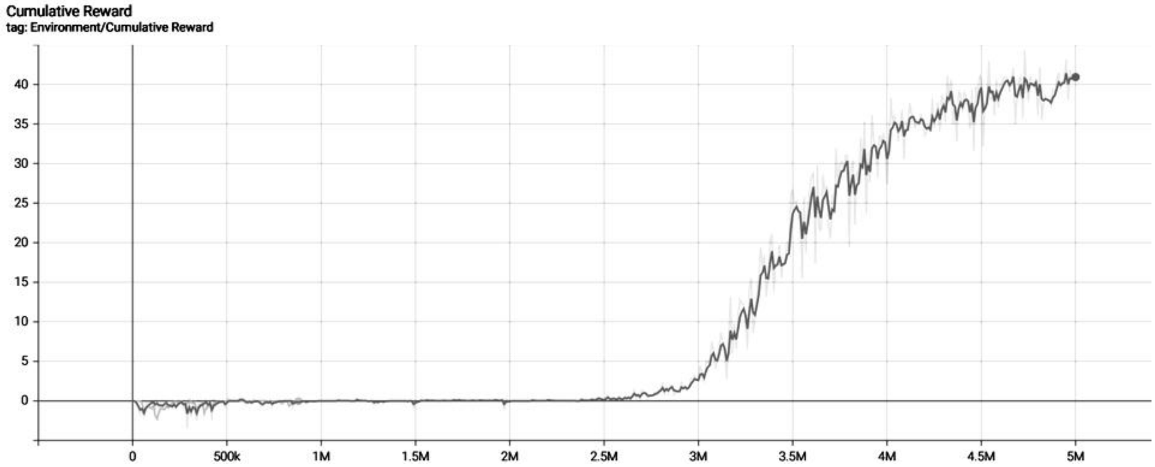


Figure 9 Cumulative Reward for "Hummingbird" agent

After approx. 2.5M Steps, Agent seems to figure out a pattern to successfully navigate to the nectar Collider.

2.5.1.2 Drone Agent Experiment

Observing Agents Behavior, we find Drone Agent seems to wobble around trying different actions to be able to achieve reward and avoid punishment.

Reward = +1f if Agent is in Nectar

Reward = -1 if Agent Collides

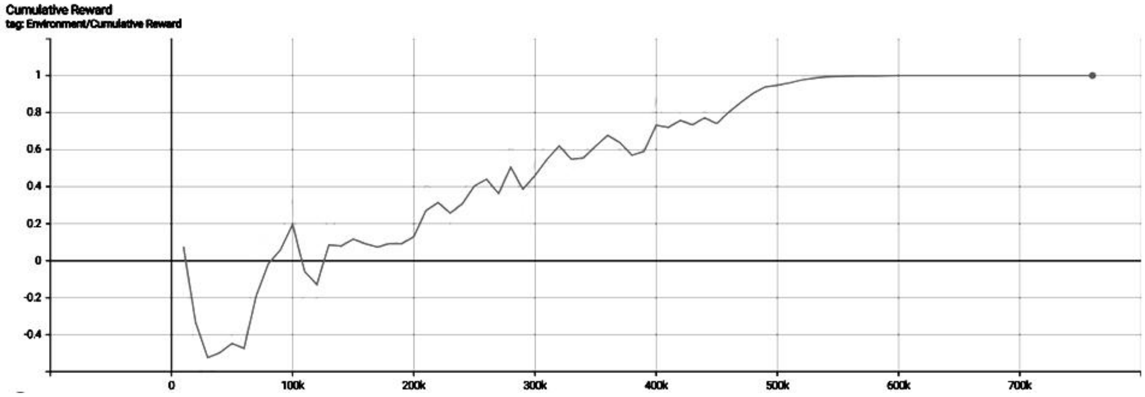


Figure 10 Cumulative Reward for "Drone Target" agent

2.5.1.3 EyeAgent Experiment

Observing Agents Behavior, the Agent seems to wobble around trying different actions to be able to achieve reward and avoid punishment.

Reward = +1f if Agent is in Target

Reward = +0.001 if Agent Looks at Target

Reward = -0.2 if Agent Collides

Reward = -0.5 if Agent Collides

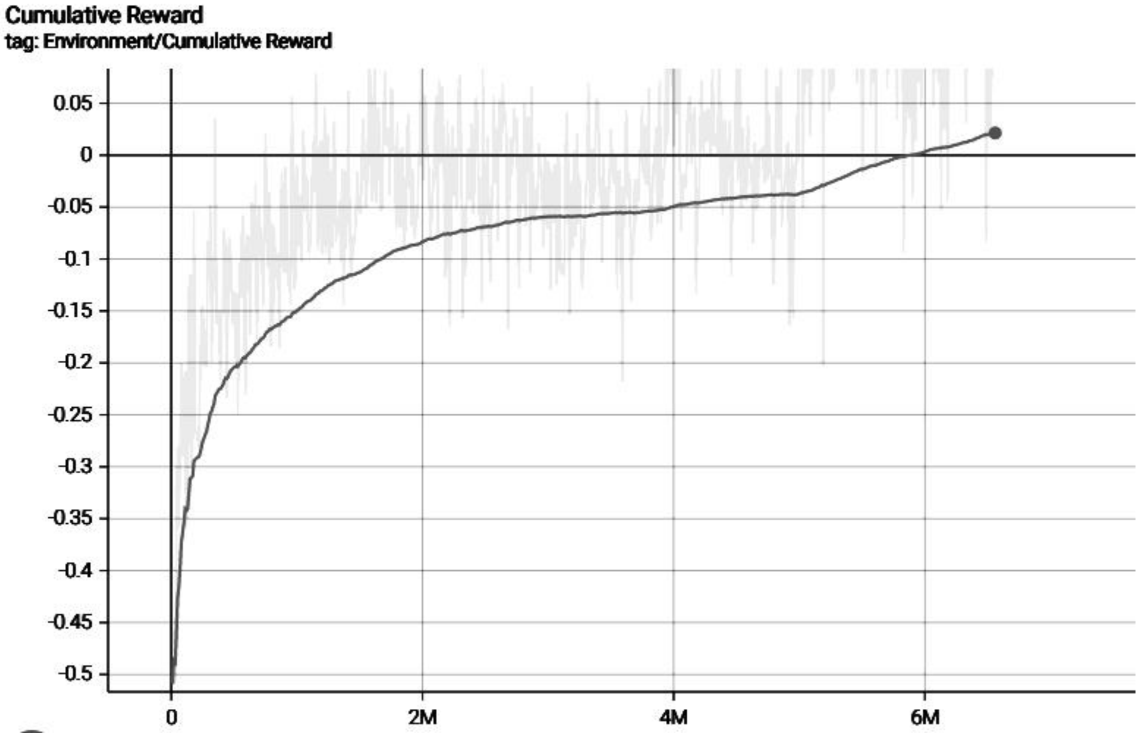


Figure 11 Cumulative Reward for "The Eye" agent

After approx. 6.5M Steps, Agent seems to figure out a pattern to successfully stop colliding with walls and with other agents.

2.6 Instance Setup

For our research, we are conducting two separate training sequentially. Seekers are using MA-POCA, which is used to train a group of seekers (1-4). While the hider agents are using PPO (Proximal Policy Optimization) which is used to train hider agents. They use the props available in the environment to shield themselves for a shorter reward or use it to block the windows and doorways (best policy).

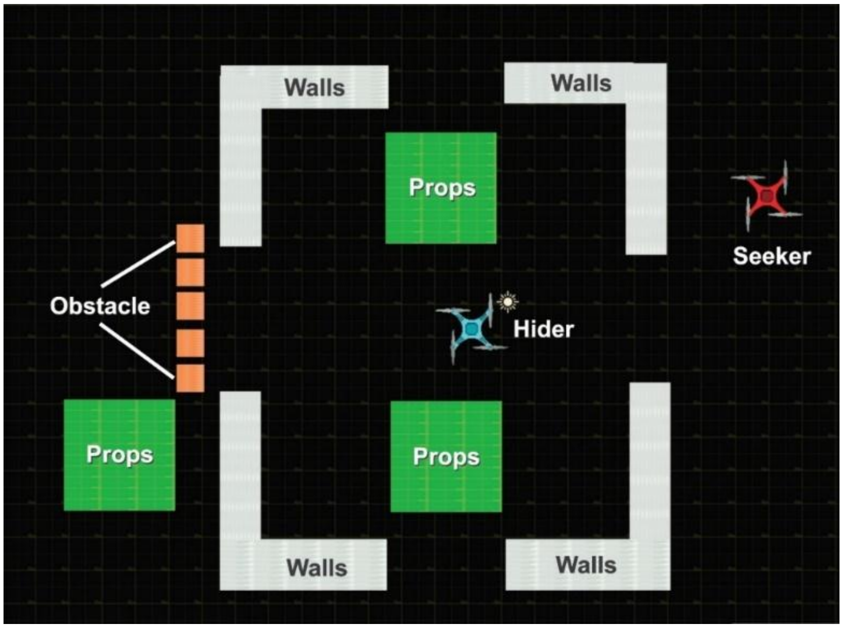


Figure 12 Environment visual (2d view)

Figure 12 demonstrates the training and testing environment that consists of 1-2 Hiders Drones, 1-4 Seeker Drones, 1-4 Props (doors, windows), 5 obstacles, 4 “L-shaped” walls, and 4 boundaries around the whole setup.
Seeker: (MA-POCA) The Hunter Drone Agent tasked to locate and hit Hider to avoid walls and obstacles.
Hider: (PPO) The Escaping Drone Agent tasked to Lock doors and clear obstacles.
Props: Draggable/Lockable entity used by agents for their gain.
Walls: Static non-movable entity used to teach navigation.
Obstacle: Physics-based Rigid bodies blocking agents’ path to the desired goal.

2.6.1 Parallelism

Training is performed using the principle of parallelism. “n” consecutive similar environment prefabs are initialized altogether. Each prefab instance has an identical list of agents and a similar environment structure. Experience can refer to the dataset of observations and rewards used to train an agent's neural network. Let E be the experience in n instances:

$$E_t = \sum_{n=1}^{n} (E_1 + E_2 + E_3 \ldots\ldots E_n)$$

By training multiple agents at once, the overall training time can be reduced, as the agents can learn from their experiences in parallel rather than one at a time. It is achieved using a technique called asynchronous training, where multiple agents are trained concurrently, and their experiences are used to update the neural network at different times.

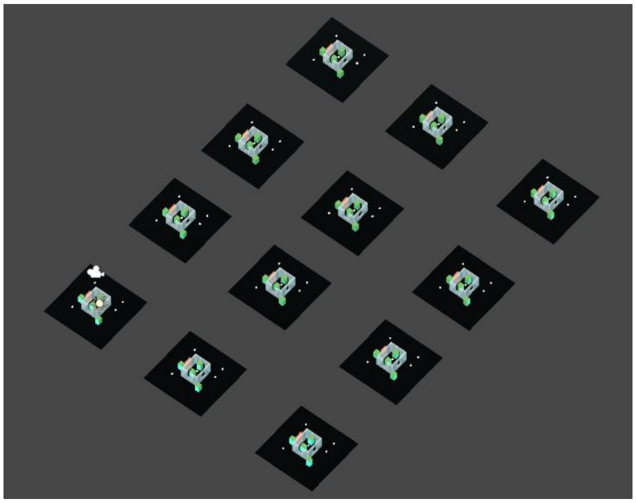


Figure 13 Parallelism - 12 Environments

Figure 13 demonstrates Trajectory-based parallelism which involves running multiple instances of the simulation environment in parallel, with each instance running a different agent. Each agent collects a sequence of experiences, or "trajectory", from interacting with the environment, and these trajectories are used to update the agent's model.

2.6.2 Multi-Agent Prep-Phase

Agents are given a total of 3072 steps in which they can make decisions every third step and update their actions accordingly. Hiders are given a 40% preparation time before the seekers are allowed to move. In this “prep phase” hiders are allowed to move freely while avoiding contact with seekers. Hiders can learn to identify, drag, and lock movable props (i.e., boxes) to use them to their advantage. During this period, seekers are observing the hider’s location and shapes (if hiders accidentally stay in Infront of the seeker’s field of vision) but they are restricted with their movement.

$$S_p = S_m * \left(\frac{40}{100}\right)$$

Where S_p are the Steps for the prep phase and S_m are the Max Steps.

prep_phase_steps	=	Max_env_steps	*	(40/100)
1228.8	~	3072	*	(40/100)

2.6.3 Multi-Agent Test-Phase

The test phase refers to that portion of training in which the seekers are let loose and given control over themselves. That means seekers can now also move along with hider agents and try to hunt them and tag them. After 40% of steps are completed out of max environment steps, for the rest 60% of the steps, seekers are allowed to move and use their learned policy to focus on targets and reach them.

$$S_t = S_m * \left(\frac{60}{100}\right)$$

Where S_t are the Steps for the test phase and S_m are the Max Steps.

test_phase_steps	=	max_env_steps	*	(60 / 100)
1843.2	~	3072	*	(60 / 100)

2.7 PLAYTESTING RL AGENTS

2.7.1 Observing Player Progression via Curriculum learning

Our Hider agents are trained via a form of machine learning method known as curriculum learning which includes progressively raising the level of complexity of the training examples provided to an AI agent. Task Sequencing: The first step in curriculum learning is to define a sequence of tasks with increasing difficulty. This sequence can be represented as a function that maps the current training iteration to the corresponding task. For example, the task at iteration t could be represented as:

$$f(t) = Task_t$$

The plan is to start with simple instances and progressively get more complicated as the agent gets better at the task.

$$C_r > T_r \rightarrow D += 1$$

Where C_r is the current reward, $C_r$is the Cumulative threshold reward. D is the difficulty as in levels. That is the current reward exceeds a given number of episodes, the environment will evolve to complex itself to make it harder for an agent to learn gradually. Agents are introduced with four different levels of environmental setup in Figure 12. Levels are made difficult to complete progressively. For example, level one has the least difficulty while level four has the maximum. Curriculum learning is applied to educate the agents on how to play the game of hide and seek more effectively. The hiders and the seekers are two different teams of agents in this game. The hiders must conceal themselves in their surroundings, while the seekers must track them down and tag them. By gradually escalating the level of difficulty, curriculum learning is integrated into the game of hide and seek with the students. The hiding places could be harder to locate as the hiders get more adept at doing so. Similar to this, as the game goes The curriculum learning in ml-agents hide and seek is intended to assist the agents in learning more quickly and effectively. The agents can learn the game in a more organized and effective manner by starting with simple instances and progressively adding complexity.

2.7.2 Seeker's Assault Horizon

The idea behind this is inspired by the military term “Dog Fight”. A dogfight is a kind of aerial conflict in which two or more aircraft engage in a close-quarters battle. High-speed maneuvers and intricate strategies are frequently used, with each pilot attempting to outwit their rival. Dogfights may be quite dangerous since they frequently take place at great heights and make use of cutting-edge equipment. Our agent’s navigational capabilities are designed by keeping the “running and chasing” ability in mind. Seekers are to keep hiders in their assault horizon to gain rewards and even tag them to get an additional reward.

With the help of a specially designed target field of vision “reward strength” signal, seekers can identify what angle suits best to adjust its rotation and position accordingly, so it is facing and is right in Infront of its target.

Let T = forward vector “z-axis” i.e. (0,0,1) and let a = difference between this agent’s position P_1 and other agent’s position P_2.

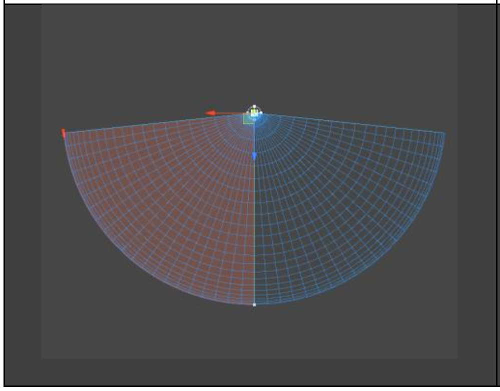
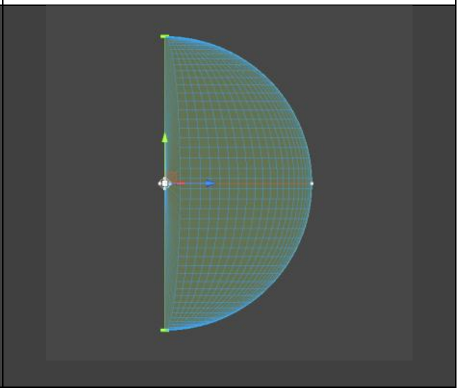
$$a = P_2 - P_1$$

$$R_d = |a|$$

$$d = T.\left(\frac{a}{R_d}\right)$$

$$S_s = \left(\frac{d}{R_d}\right)$$

Where S_s is the Signal Strength, R_d is the magnitude of a, d represents the dot product between the forward vector and $\left(\frac{a}{R_d}\right)$

Top View	Side View
	

2.7.3 Hider's Prop Use

As our agents have flying-like capabilities, introducing a ramp was of no use as agents might learn to just fly over the wall. So, we introduced windows and doorways which when blocked with props, make a safe house for the hiders. There is a single prop assigned for each window or door. Agent must learn to efficiently learn to handle the props to minimize the time required to close that door and also learn which prop is best for quickly closing a specific location door. If the agent is using the farthest prop to close a door It will damage the efficiency as it needs to close the rest of the doors too.

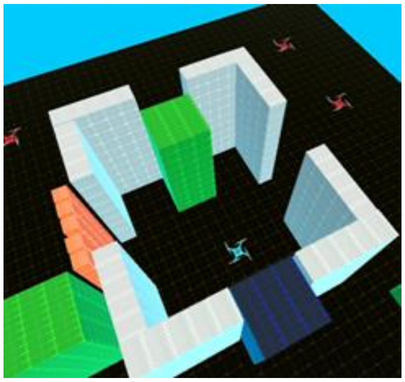


Figure 14 Prop 1 used successfully

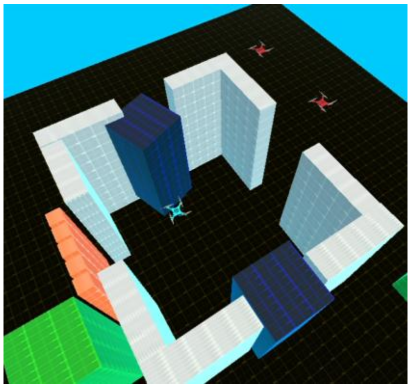


Figure 15 Prop 2 was used successfully.

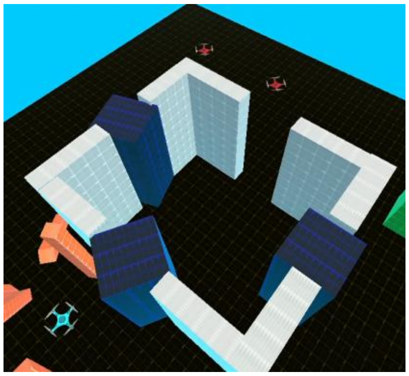


Figure 16 Prop 3 was used successfully while Avoiding obstacles.

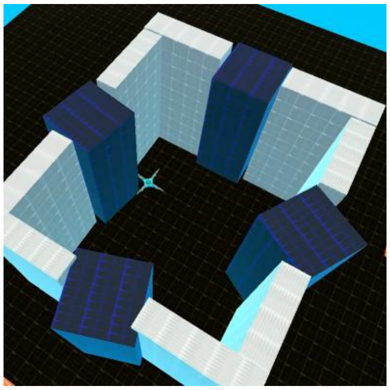


Figure 17 Prop 4 with max distance and avoiding seekers collision used successfully.

2.7.4 Clearing Obstacles

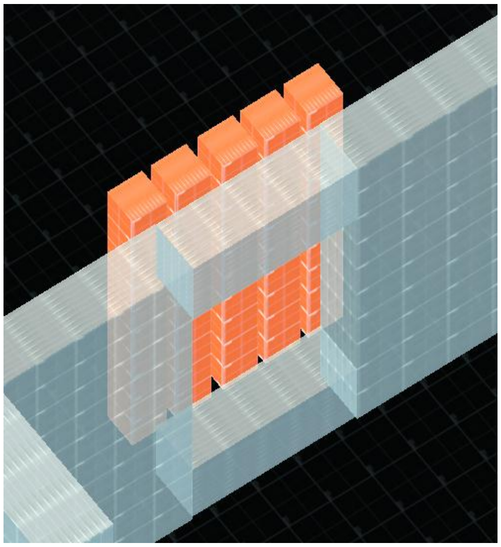


Figure 18 Obstacles are demonstrated with orange color.

Agents have an additional problem of clearing the obstacles to make a path for closing the doorway. These orange blocks have mass and light gravity. This light gravity increases the difficulty as it's more complex to clear them away with one stroke as they start to float in the path of agents. Agents need to constantly move away from the obstacles and clear their path to the optimal solution.

Chapter 3

Results and Analysis

In the previous section, the methodology of our two agents including the algorithms, observations, and their reward distribution has been described. The following section includes the results and analysis of Hiders and Seekers agents. This section will be divided into two major portions each highlighting the resulting behavior and analytical outcome of Seekers and Hiders.

By using the methodology explained in the above section it's been deduced that training agents with a lesser negative reward or lesser penalties yield a quicker learning behavior and can result in faster convergence to the optimal policy. This occurs especially in navigation-related tasks where an agent might be in the right direction to target but is getting a penalty for not looking at it. PPO and MA-Poca tend to increase its reward based on observations so it is beneficial to leave the negative rewards at a minimum. Hiders are given a 0.001 reward for every frame they are hidden while seekers are given a 0.001 reward if hiders are in the field of vision and +1 if they crash hiders.

Seekers are trained in a group of 2-4 multi-agents and assigned cumulative rewards i.e. each will get the reward for good action and each will get a penalty if one does a bad action. It resulted in the occurrence of cooperation among them. Seeker's behavior shows they tend to, with time, explore random paths lesser and use the target's (Hider) location observation to calculate the shortest path and shortest facing direction. This behavior helps the seekers to put hiders easily in their FOV (field of vision). Seekers were successful in emerging "Running and chasing" which includes sub-strategies like locating, navigating, finding Hiders, navigating around walls, and door identification. As Seekers are already flying (drones) they learned to ramp themselves up with an upward thrust through the windows. As windows do not allow the drone to enter without moving vertically up or down ('down', in our case as the agent can be near the roof) thus "ramp use" strategy is also satisfied as they are successful in entering the hider's rooms via learning vertical movement and clearing obstacles Infront of windows.

Hiders are trained solo and as a group of 2 multi-agents. As their main reward is based on isolating themselves in a safe location so their rewards are quite far in the future i.e. agents need to perform more random actions to identify the best state-action situation to push in the buffer to generate an optimal policy. To overcome this barrier, we used curriculum learning which by increasing the complexity of the environment gradually yields an agent understanding of the basic concept of what actions are required to get a max reward in an episode.

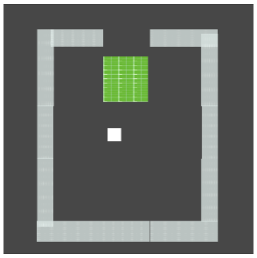


Figure 19 Level One

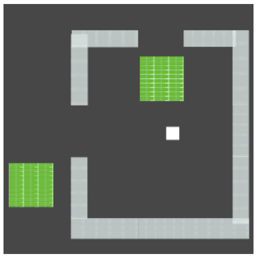


Figure 20 Level Two

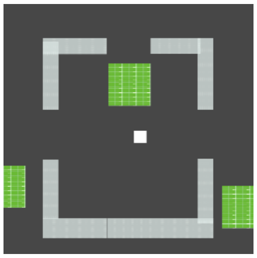


Figure 21 Level Three

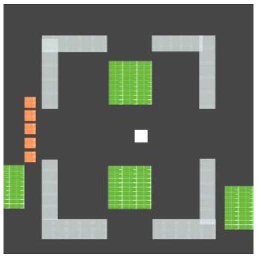


Figure 22 Level Four

Figure 19 represents the easy environmental setup showing a basic training scenario where the Hider Agent (white box) just needs to drag the prop (green box) to the doorway and lock it there so the Seeker Agents are not able to get into the confined safe space of the Hiders. Figure 20 represents a medium environmental setup showing a simple layout with twice the number of doorways and tools (props) that can be used to block the seekers and make a safe place for hiders. Figure 21 represents a moderately hard setup with three doorways and three props. This introduced unidirectional movement for the hider to push and pull props in both the left and right directions. Figure 22 represents the hardest environment, where the hider agent has four doorways and just four props, so it has to use them in the given time (40% of the full episode) and introduces obstacles that hinder the path and cause physical barriers in movement and visual barriers in the field of visual input.

The environment has six key components. Walls, Props, Hiders, Seekers, Boundaries, and obstacles. Walls are static objects designed specifically to gradually increase the difficulty by enabling or disabling a chunk of it as shown in figures 1-4 above. Props are used as a door or a shield that can be dragged into the vacant door/window spaces and locked there so seekers are unable to get inside the hider’s safe hideout. While obstacles are pure physical rigid bodies that can be pushed. They have gravity involved with much less weight which ends in gliding behavior that continues to stay in the path of agents even after being pushed once. By inducing such curriculum learning behavior, hider agents tend to grab the concept of safely dragging the prop to the opening and closing it more quickly and efficiently than directly starting the training at level four. One can reduce or increase the levels just to get optimal results from the agents. This subtle increase in the complexity of the environment rather than forcing the agent to adapt to the more complex one yields beneficial results. We did two experiments by training the hider agents until 3.3M steps in both traditional and curriculum learning. With comparing to traditional non-curriculum learning experiments there was found no increase in reward cumulative rewards as shown in the graph.

3.1 Curriculum Learning vs Traditional Learning

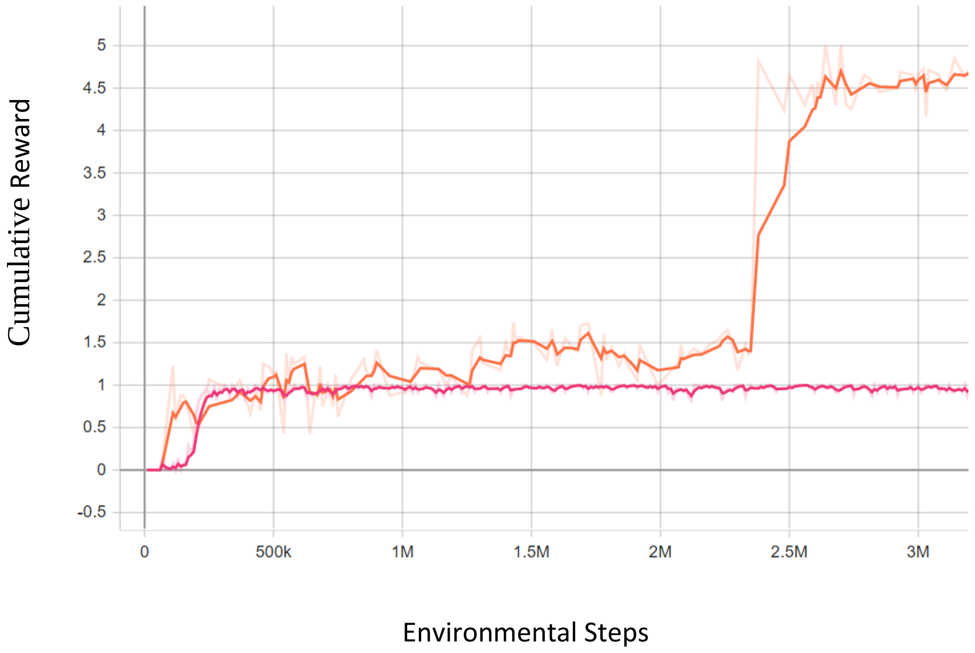


Figure 23 Reward Distribution for Hider Agents comparing curriculum vs traditional environmental setup.

Figure 23 is representing cumulative reward comparison over 3.3M environmental steps between curriculum vs traditional learning environmental setup. The solid magenta-colored line represents the traditional approach where Hider agents were able to just run away from seekers satisfying the first strategy and not yielding to the second. While the solid orange line represents curriculum learning where training was divided in the forms of levels (Figure 19) which yields better continuous reward per frame as hiders were able to hide for more time steps and get more reward every frame.

Reward, in the case of hiders, is given on the bases of how many frames the hiders can hide from the field of vision sensor of seeker agents. More the frames they are hidden the more they can get rewarded. With the curriculum, learning involved hiders were able to learn a basic behavior and adopt a converging policy that results in a better hiding behavior. Therefore the curriculum learning approach was adopted for further experimentation.

3.2 Hiders

3.2.1 Cumulative Reward:

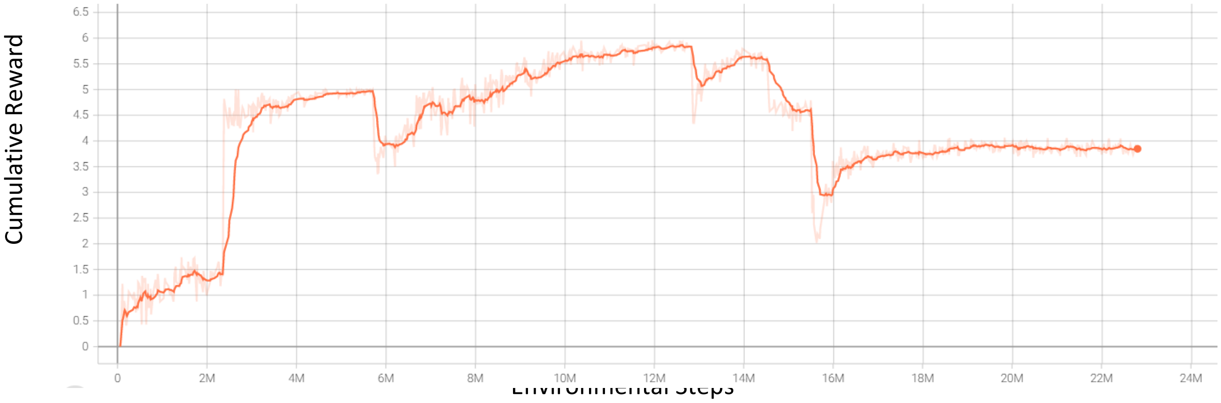


Figure 24 Hiders Cumulative Reward Smoothed 0.99

The solid orange line represents Hider's cumulative reward over the environmental steps. Training is initialized using Level One. Level Two is introduced at 5.7M environmental steps. Level Three at 12.8 steps while Level Four with decreased reward per frame at 15.7M steps.

A spike at 160k environmental steps axis represents Hider Agents were able to learn to get some reward by crudely running and escaping from the seekers. Until 2.3M steps, Hiders were running and escaping from the seekers more efficiently and thus were getting more rewards per episode. From 2.3M to 5.7M a sudden increase in reward was observed as level one (Figure 19) of curriculum learning was finally working and yielding maximum reward. At the 5.7M step, level two was introduced and agents experienced a sharp decrease in rewards as the difficulty was increased. But as they have already a semi-trained brain, so they continued to update the policy and adopt it. At the 12.8M step, the third level was introduced which shows a steep downfall indicating complexity for the agent to learn. And finally, at the 15.2M step level, four was introduced and the reward was also reduced for an agent to learn from the very complex condition, but it still manages to maintain a solid positive reward after 16.5M steps.

3.2.2 Episode Length:

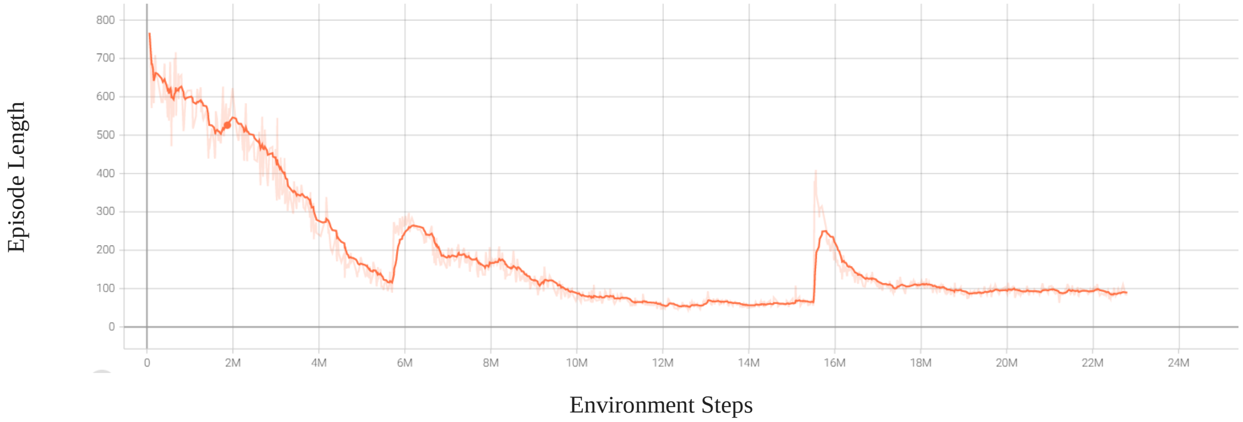


Figure 25 Episode Length of Hiders Corresponds to decisions made per episode.

The graph represents the gradual decrease in episode length as few decisions are required to complete an episode. The solid orange line represents the episode length over the environmental steps.

Training starts with 768 decisions made per episode.

episode_length	=	max_episode_steps	/	decision_requests
768	=	3072	/	4

The training environment is designed in such a way that it will reset the episode as soon as one of the agents is successful in making the next strategy. i.e., if “running/chasing” hider performs “fort building”. , the environment will reset to promote the use of lesser episode length. This behavior encourages the agents to find the optimal next policy and strategy faster. Observing the graph we can see a slight increase in episode length at 5.7M steps where level two of the environment is introduced. This means the agent requires more steps to find a new policy to get more rewards. At the 15.8M step, another bump is observed showing level three is deployed and it affected the current behavior such that it needs more time to converge. After 18M steps, there seems a steady graph representing a steady and constant episode length. This means agents need at least 21 steps to converge to the optimal policy.

Required_episode_steps	=	episode_length	/	decision_requests
21	=	84	/	4

3.2.3 Curiosity Reward:

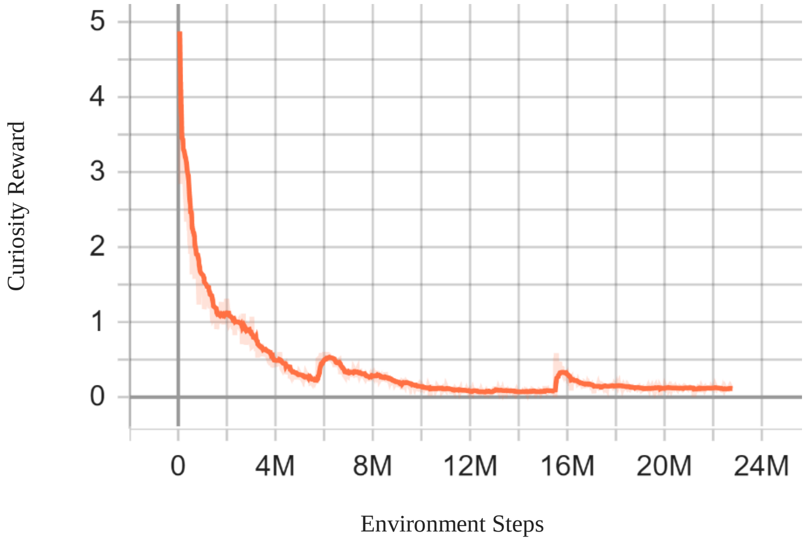


Figure 26 Curiosity reward for hiders based on curriculum learning.

Hider Agents is given a curiosity-based reward based on the exploration it does. The solid orange line represents the curiosity reward given over environmental steps.

The Agent's aim in reinforcement learning is to develop a behavior that maximizes a reward. A reward is often determined by the surroundings and relates to achieving some goal. These are known as "extrinsic" rewards since they are defined outside of the learning process. However, rewards can be specified outside of the environment to encourage the agent to act in specific ways or to help in the learning of the genuine extrinsic reward. These are known as "intrinsic" reward signals. The agent will learn to maximize the overall reward by combining extrinsic and intrinsic reward signals. ML-Agents allow reward signals to be specified in a modular manner. The extrinsic Reward Signal, which is by default activated, symbolizes the rewards established in the surroundings. When extrinsic incentives are scarce, the curiosity reward signal encourages your agent to investigate. As seen from the graph, the curiosity reward is very high at the beginning of the training. This means the agent is performing more random actions and being rewarded to explore the environment more. But with time it decreases it showing it got the rewarding behavior and is now using lesser random and curious actions. At 5.7 with the addition of level two of curriculum learning, we observe a slight bump in the graph representing agent got curious about the newly added walls and props.

3.3 Seekers

3.3.1 Group Cumulative Reward:

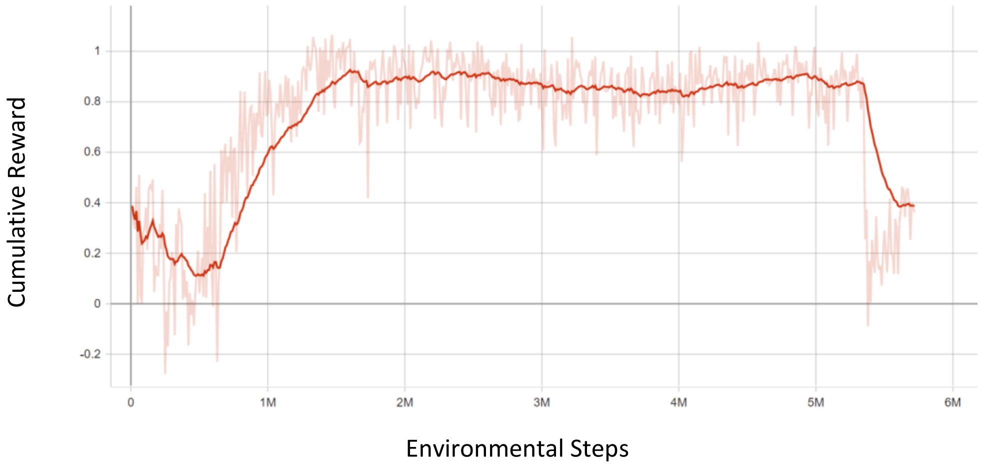


Figure 27 Seeker's Group Cumulative Reward Smoothed 0.95

Training is initialized using Level Four. At 630k steps, seekers learned to rotate toward the hiders and keep them in the field of vision. At 1.6M step seekers were successful in traversing obstacles, navigating through walls, and moving toward hiders. At 5.3M step seekers were trained again just to check how they behave when hiders are also trained for level four environmental setup, and it shows a steep descent in reward as they were not unable to see hiders. The solid red line represents the group cumulative reward given over environmental steps.

Seekers are trained in two phases, first one is from 0-5.3M steps. This phase includes the "running & chasing" hiders as a competition so they will learn this strategy as well. While phase two starts at 5.3M steps which includes training with "fort building" hiders that yields a sharp descent in the graph. This indicates the complexity and difficulty the seekers were facing in locating the hiders once they emerge to the second strategy. Seekers are given a +0.001 reward for every frame they see hiders in their field of vision. And no reward when they do not have them in sight. This is done because if we give a negative reward for such a constant frame rate just because the seeker's field of vision does not contain hiders, we are damaging the path navigation behavior. Such that an agent might be on the right path but is getting penalized just because it is not there yet. Seekers are tested in a group of 2-4 multi-agent setup and solo as well. While training the 2-seeker agent with the same brains are being trained. Dual input is used but the rewards are given on team bases. If one agent is successful both agents will get rewarded and will share the updated best policy while if one is getting panelized both will face this.

3.3.2 Episode Length:

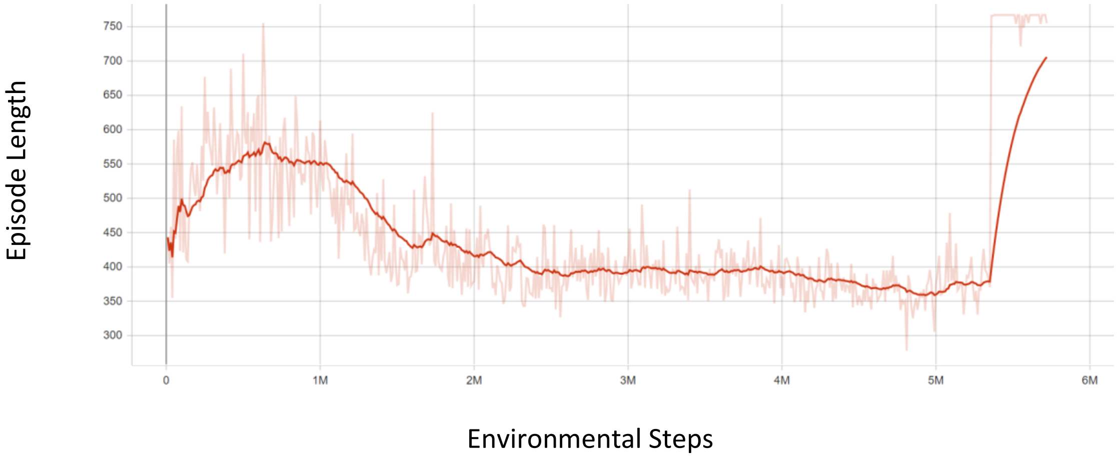


Figure 28 Episode Length for Seeker agents Corresponds to decisions made per episode.

The graph represents the gradual decrease in episode length as few decisions are required to complete an episode and a sudden increase demonstrating agents require more decisions when the opponent is far more difficult. The solid red line represents the episode length over the environmental steps.

Seeker agents are programmed in such a way that if they have hider agents in sight they get 0.001 rewards per frame but if they collide with them the episode ends. Thus, the more the seekers can collide quickly the lesser the episode length gets. This is done to encourage quick navigation and faster resetting of the training environment because if the seeker can collide with the hider, the hiders are not protected well, and there is less benefit in training again in an already won episode.

episode_length	=	(max_episode_steps	-	remaining_steps)	/	decision_requests
443	=	3072	-	1300	/	4

0-630k steps seekers took longer to make an optimal policy. Soon after 630k steps, seekers took lesser decisions to get to the best possible state and achieve a better goal. At 5.35M step seekers were trained again for phase two. In which hiders were successful in fort building. We can observe a rapid growth in episode length which indicates that the current policy was not enough to encounter much smarter competition and is taking much more decisions than before.

3.3.3 Self-Play ELO

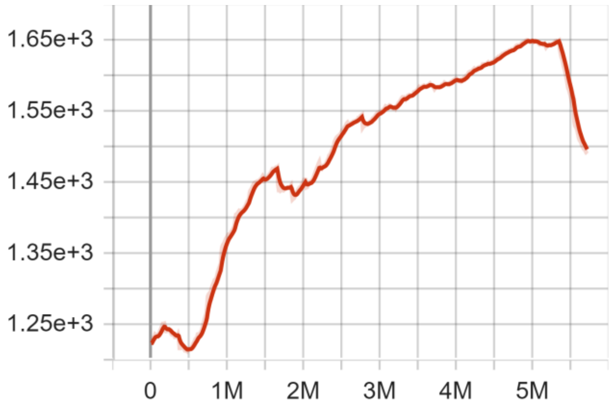


Figure 29 ELO for seeker agents with save steps equal to 61440, team change equal to 184320

Self-play training adds extra complicating elements to the standard problems associated with reinforcement learning. In general, the tradeoff is between the final policy's skill level and generality and the stability of learning. Training against a group of slowly changing or unchanging enemies with low diversity yields a more consistent learning process than training against a set of rapidly changing adversaries with high diversity. This tutorial addresses the disclosed self-play hyperparameters and intuitions for tweaking them in this context.

The graph ascended gradually until the hiders learned to hide successfully. At this point, the learned policy to follow and hit the hiders was not so fruitful as hiders are in a confined safe room and cannot be seen at all, so it descends.

3.4 Key Difference

As Larger batch sizes require more GPU memory and processing power (Rotenberg, 2020), with our approach and methodology, we proved that agents emerged strategies much faster and efficiently with much less processing power. While batch size is equal to number of steps in an episode, buffer size was selected using the following formulae:

$$B = b * E_n * P_n$$

Where:

B = Buffer Size

b = Batch Size

E_n = Environment Count

P_n = Instances Count

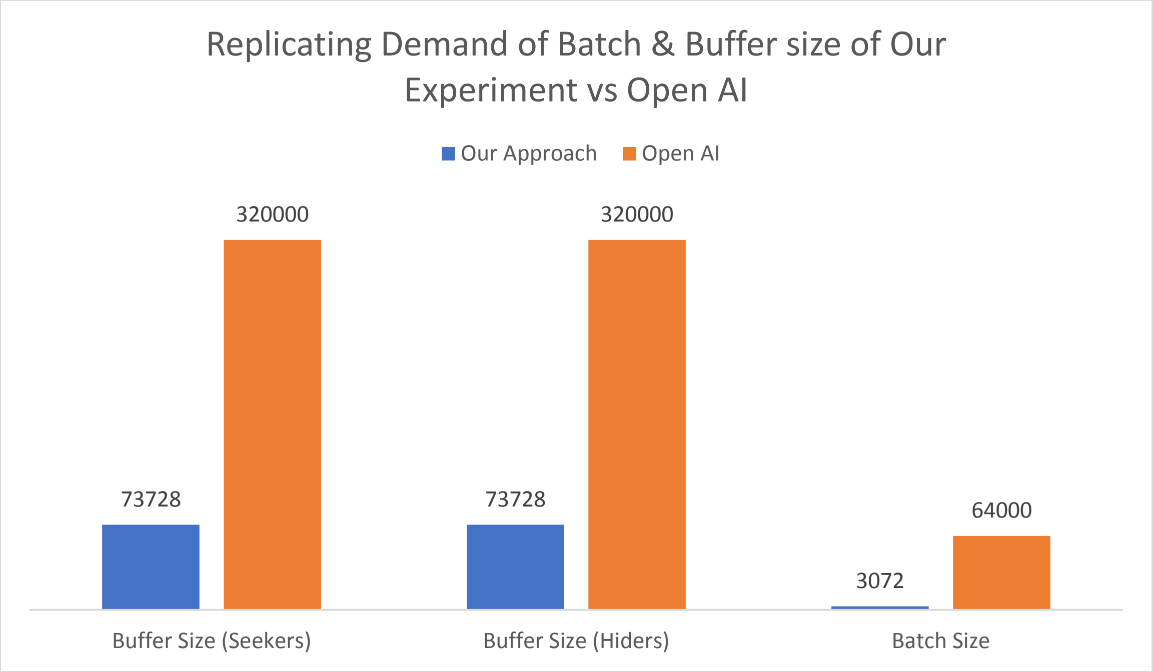


Figure 30 Batch & Buffer size requirements for the experiment’s vs Open AI's requirements

Our method utilizes significantly fewer system resources compared to Open AI's experiment for deriving strategies as the buffer size required by our approach is approximately one-fourth of what was needed in the Open AI experiment. Similarly, the batch size required by our model is approximately one-twentieth of what was required by Open AI. This significant reduction in resource requirements demonstrates the efficiency and effectiveness of our approach.

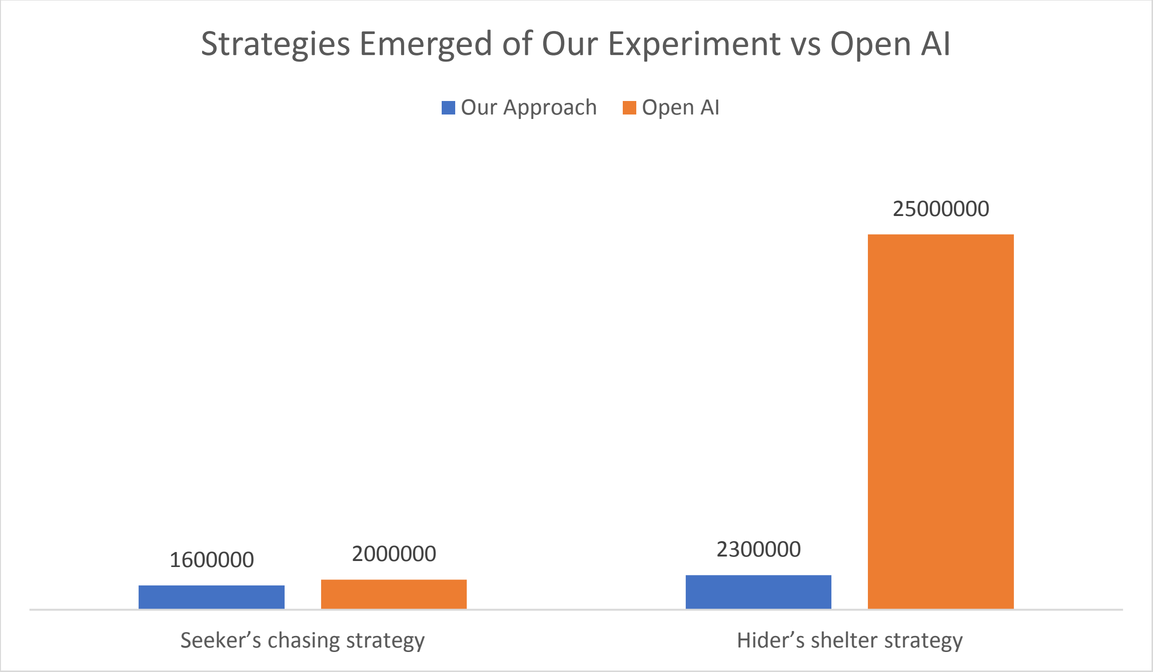


Figure 31 Strategies emerged of our method vs Open AI's method.

In addition to requiring fewer resources, our approach also deduces strategies in a much earlier environmental step count. For instance, the Running and Chasing strategy was identified approximately 0.4 million steps earlier in our approach than in the Open AI experiment. Similarly, our model was able to discover the strategy for creating shelter 22.7 million steps earlier than in the Open AI experiment. This early identification of strategies could prove to be a significant advantage in certain contexts.

Overall, our approach offers a more resource-efficient and time-efficient method for deducing strategies than Open AI's experiment. By requiring significantly fewer resources and identifying strategies earlier in the environmental step count, our model could prove to be an effective tool in various domains. The results of our experiment demonstrate the potential of our approach to advance the field of strategy identification and inform decision-making in a range of practical applications.

3.5 Discussion and Related Work

D. Yang et.al proposes a method for adaptive inner-reward shaping in sparse reward games where the agent receives only occasional feedback. [13] The authors argue that inner-reward shaping can help improve the agent's learning efficiency and speed. They propose an algorithm called AIRS (Adaptive Inner Reward Shaping) that dynamically adjusts the shaping weight during training based on the agent's learning progress. The authors evaluate the algorithm on two Atari games and show that it outperforms existing methods in terms of sample efficiency and final performance. Instead of adaptive rewards we used curriculum learning in which the environment's difficulty level is adaptive in the sense as it increased once met with an assigned reward threshold.

M. Lukas et.al proposed previous cheating in multiplayer video games, and the need for effective anti-cheat systems. The problem statement is that existing anti-cheat systems are often reactive and not effective against new forms of cheating. The method used is reinforcement learning to develop an adaptive and proactive anti-cheat system based on agents that can learn to detect and respond to cheating behaviors in real-time. [14] An outcome is a promising approach to developing effective anti-cheat systems that can adapt and evolve to new forms of cheating. We used multi-agent competition between two teams of agents which enables agents to learn from an experienced brain.

X. Wenwen et.al discusses the application of end-to-end behavior decision-making based on deep reinforcement learning (DRL) in autonomous driving. The problem statement is to develop an agent that can learn to make decisions based on visual inputs from a car's camera. [15] The method used in the paper is a DRL approach that maps inputs to actions through a neural network. The outcome of the paper is the successful training of an agent that can make safe and efficient driving decisions in a simulated environment. We introduced frontal and spatial sensors for the agent's real-time locational and shape structure observation.

R. Dawkins et.al explores the concept of arms races, both between and within species, and how they evolve through natural selection. The problem statement is to understand the dynamics of arms races and how they lead to the evolution of certain traits. The method used is a combination of theoretical models and empirical data from various species. The outcome of the paper is a deeper understanding of the mechanisms behind arms races and their evolutionary implications. [16] We introduced competition between said two races i.e., hiders and seekers.

B.W.Wirtz et.al examines the potential of Artificial Intelligence (AI) to transform the public sector and improve efficiency and effectiveness in areas such as healthcare, education, and governance. [17] The problem statement highlights the ethical and legal considerations, data quality, and workforce implications that arise with AI adoption in the public sector. The paper reviews existing literature and case studies of AI applications in the public sector and provide recommendations for policymakers and practitioners. The result emphasizes the potential benefits of AI in the public sector but also highlights the need for careful planning and management to address the challenges and risks associated with AI adoption. Introducing auto-pilot in drones enables better use of resources and saves the economy by reducing human controllers.

M. Asplund et.al proposes an Artificial Intelligence (AI) based marine autopilot trained using reinforcement learning in the Unity simulation environment. [18] The background highlights the potential benefits of AI-based autopilots in marine navigation, including improved safety and efficiency. The problem statement discusses the challenges of designing an AI-based autopilot and the need for training in a simulation environment. The paper uses a combination of Deep Q-Network (DQN) and Proximal Policy Optimization (PPO) algorithms to train the autopilot, and the result shows promising performance in avoiding obstacles and navigating through complex environments. The paper suggests that AI-based autopilots have the potential to revolutionize marine navigation and reduce the risk of accidents. We introduced a flying-like mechanism that includes the x,y, and z axis for movement.

L. Pinto proposes an Asymmetric Actor Critic (AAC) algorithm for image-based robot learning. [19] The background highlights the potential of image-based robot learning in complex environments where traditional sensor-based methods may not be effective. The problem statement discusses the challenges of designing an effective image-based learning algorithm and the need for efficient feature extraction. The paper uses a combination of convolutional neural networks (CNN) and AAC to learn robotic tasks from raw images, and the result shows promising performance in grasping and manipulation tasks, outperforming existing methods. The paper suggests that AAC has the potential to enable more efficient and effective image-based robot learning in a variety of applications. We not only added a camera for image observation but also added a spatial sensor and 2d ray casts for observations.

P. Reizinger proposes an attention-based curiosity-driven exploration method to tackle the exploration problem in deep reinforcement learning. [20] Traditional exploration strategies are not effective in complex environments with high-dimensional state and action spaces. The proposed approach uses curiosity-driven exploration, guided by an attention mechanism, to explore uncertain or unexpected areas of the state space. Experimental results demonstrate that the proposed approach outperforms traditional exploration strategies and achieves state-of-the-art performance on several benchmark environments, including Atari games and continuous control tasks. The attention mechanism enables the agent to focus on relevant information and learn faster with fewer samples. We used curiosity along with curriculum learning that enabled agents to earn rewards more quickly.

J. Z. Leibo proposes an auto-curricular framework for multi-agent intelligence research that enables agents to learn dynamically from each other using methods such as self-play and population-based training. [21] The framework is evaluated on several benchmark environments, and results show that it can lead to the emergence of innovation and the development of novel solutions to complex problems in multi-agent intelligence. The proposed approach offers a promising direction for future research in this field. We enabled agents to use this along with curiosity and curriculum learning.

A. Devo et.al proposed Autonomous single-image drone exploration with deep reinforcement learning and mixed reality" using the Proximal Policy Optimization (PPO) algorithm, a popular deep reinforcement learning algorithm. The authors also use a variant of PPO called Curiosity-Driven Exploration (CDE) to train the drone agent to explore its environment. [9] CDE incorporates an intrinsic reward signal based on prediction error to encourage the agent to explore areas it has not visited before. The algorithm is trained on a large dataset of simulated and real-world drone images and is used to guide the drone in real-time to explore unknown environments. We introduced a competition for agents and a curiosity element thus forcing agents to explore more states to gain reward.

C. Ofria et al present Avida, a software platform for conducting computational experiments in evolutionary biology. Avida uses digital organisms, which are self-replicating computer programs, to simulate the process of biological evolution [22]. The platform provides a flexible and customizable environment for studying a wide range of evolutionary processes and has been used extensively in evolutionary biology research. The paper demonstrates the utility of Avida by presenting several examples of research studies conducted using the platform. We perform our experimentations in unity3d. Also used parallelism for quick learning of the brain.

A. E. Youssef, et al The paper "Building your kingdom: Imitation Learning for a Custom Gameplay Using Unity ML-agents" proposes an imitation learning method based on convolutional neural networks (CNNs) for training non-player characters (NPCs) in a game environment. The method involves recording the gameplay data of a human player and using it to train a CNN using supervised learning. [23] The trained CNN is then used to control the behavior of NPCs in the game using Unity ML agents. The results show that the method can effectively train NPCs to mimic the behavior of human players, leading to more realistic and engaging gameplay experiences. We used PPO and MA-Poca for training the hiders and seekers respectively.

J. Paredis et al present the concept of coevolutionary computation, which involves evolving two or more populations that interact with each other. The problem statement involves the limitations of traditional genetic algorithms in finding optimal solutions for complex problems. The method proposed is to use coevolutionary computation to evolve the populations simultaneously, with one population representing the problem solutions and the other representing the problem environment. The outcome is a more efficient and effective method of solving complex problems, demonstrated through simulations and experiments. [24] The algorithms used include Genetic Algorithms, Coevolutionary Genetic Algorithms, and Fitness Sharing. We made competition between two different agents that need to coevolve with each other to be able to successfully compete and gain their desired rewards.

R. Miikkulainen et.al proposes a coevolutionary approach to generating complex and adaptive solutions in competitive environments. The problem statement is that traditional evolutionary algorithms may struggle to produce complex solutions in such environments. The method involves multiple populations evolving in a competitive setting, with complexity being the term used to describe the process of generating increasingly complex solutions through the competitive interactions of the populations. [25] Two algorithms, CCEA and HyperNEAT, are introduced and evaluated in the paper. The outcome is a demonstration of the effectiveness of competitive coevolution and complexity in generating solutions to complex problems, with both algorithms showing significant improvements over traditional evolutionary algorithms. We introduced four levels of complication for the environment. The harder one replaces itself with the difficult one once the reward threshold is met.

L. Panait et.al proposes that review of the current state of research on cooperative multi-agent learning. The background is that multi-agent systems are becoming increasingly important in many fields, and there is a need for effective learning methods for these systems. The problem statement is that cooperative multi-agent learning is challenging because the agents must learn to coordinate their actions and communicate effectively. [26] The method involves reviewing and categorizing the existing literature on cooperative multi-agent learning. The term "learning architectures" is introduced to describe the different approaches used in the literature. The outcome is a comprehensive review of the current state of research on cooperative multi-agent learning, including the different learning architectures and their relative strengths and weaknesses. We added competition among two different agents. They did not have similar rewarding behaviors. Instead, they have separate ones.

G. Ostrovski et.al proposes a new exploration method for reinforcement learning. The background is that exploration is critical for effective reinforcement learning, but existing methods can be inefficient in large state spaces. The problem statement is that uncertainty-based exploration methods can struggle in structured state spaces. The method uses a neural density model to estimate state probability density and compute a count-based exploration bonus. [27] The Q-learning algorithm is used in the paper, and the proposed method is named "Count-based Exploration with Neural Density Models (C51)." The outcome is a demonstration that the proposed method can outperform existing exploration strategies in complex environments with large state spaces, with significantly fewer environment interactions. We used PPO vs MA-POCA on two different agents.

J. Foerster et.al proposes a new multi-agent reinforcement learning algorithm. The background is that multi-agent systems are becoming increasingly important, but existing algorithms can struggle with non-stationarity and credit assignment. [28] The problem statement is that current methods for multi-agent policy gradients can suffer from high variance and poor convergence. The method involves using counterfactual reasoning to estimate the value of actions taken by other agents, improving credit assignment. The proposed algorithm is named "Counterfactual Multi-Agent Policy Gradient (COMA)." The outcome is a demonstration that the proposed algorithm can outperform existing methods in several multi-agent environments. We used PPO and MA-POCA.

G. R. Hunt et.al investigates the tool-making ability of New Caledonian crows. The background is that tool use is a rare and impressive ability, and crows have been shown to use tools in the wild. The problem statement is to understand the cognitive processes and mechanisms behind this behavior. The method involves observing and recording the crows' tool-making behavior and analyzing the resulting data. [29] The outcome is a demonstration that New Caledonian crows can create complex tools by modifying branches and twigs, indicating that they possess advanced cognitive abilities. We used props for the agents to use to modify the environment for their reward gains.

M. S. Abdul Hameed et.al proposes a curiosity-driven approach to reinforcement learning for robots in a manufacturing setting. The background is that traditional reinforcement learning can be slow and inefficient in industrial settings, and there is a need for more effective methods. The problem statement is to find a way to improve exploration in these settings, where trial and error can be costly. The method involves using a curiosity-driven approach that rewards the robot for exploring new and interesting states. [30] The outcome is a demonstration that the proposed method can improve the learning speed and efficiency of a robot in a manufacturing cell, leading to better performance and reduced production time. In addition to curiosity, we added a continuous but very small negative reward for the agents that are given to them until they somehow learn to finish the episode. And by finishing the episode we mean either seeker finds and tags the hiders or hiders successfully arrange the props in such a manner that they are well hidden for the entire run.

D. Pathak et al propose a curiosity-driven approach to reinforcement learning. The background is that traditional exploration methods can be inefficient and may not scale to complex environments. The problem statement is to find a way to improve exploration in reinforcement learning algorithms. The method involves using a self-supervised prediction task to train the agent to predict its future states and using the prediction error as a curiosity-driven exploration bonus. The proposed algorithm is called the "Intrinsic Curiosity Module (ICM)." The outcome is a demonstration that the proposed method can outperform existing exploration strategies in several environments, leading to faster learning and better performance. [31] We provided agents with curiosity. curiosity encourages an agent to seek out new experiences and learn from them, rather than simply repeating actions that have been rewarded in the past. This can be especially useful in complex, dynamic environments where the optimal action may be uncertain or constantly changing.

C. Rosser et al propose a curiosity-driven approach to prevent undesired actions in autonomous agents. The background is that traditional reinforcement learning algorithms can lead to agents taking unintended actions in certain situations. The problem statement is to find a way to prevent agents from taking these undesired actions. The method involves using a curiosity-driven approach that rewards the agent for exploring states that are unlikely to lead to undesired actions. [32] The proposed algorithm is called "Curiosity-Driven Undesirable Action Avoidance (CD-UAA)." The outcome is a demonstration that the proposed method can effectively prevent agents from taking undesired actions in several environments, leading to improved safety and performance. We used negative rewards or penalties for certain actions like hitting the boundary. This enables the agent to learn which actions are not to be taken under consideration.

A. Price et.al proposes a curriculum learning approach to improve the training of reinforcement learning agents using the Unity ML-Agents platform. The background is that traditional reinforcement learning methods can be inefficient and slow to learn in complex environments. The problem statement is to find a way to improve the learning speed and efficiency of agents. The method involves gradually increasing the difficulty of the environment and the task, using a curriculum that is designed to guide the agent toward the final goal. [33] The outcome is a demonstration that the proposed curriculum learning approach can significantly improve the learning speed and performance of agents in several Unity-based environments, compared to standard training methods. We used curriculum learning which involves gradually increasing the complexity of the task or environment that an agent is asked to learn. This is done by breaking down the overall task into smaller, easier subtasks and gradually increasing the difficulty level as the agent improves.

C. de Souza et al pursuit using deep reinforcement learning" is a paper that proposes a decentralized deep reinforcement learning approach to enable a team of agents to collaboratively pursue a target in a dynamic environment. The background is that traditional centralized approaches to multi-agent reinforcement learning can become computationally infeasible as the number of agents increases. The problem statement is to develop a decentralized approach that can scale to many agents. The method involves using a deep reinforcement learning approach, where each agent has its local policy that is trained using experience gained from its interactions with the environment and other agents. The proposed algorithm is called "Decentralized Deep Q-Network (DDQN)." [11] The outcome is a demonstration that the proposed approach can effectively enable a team of agents to pursue a target in a dynamic environment, leading to improved performance and scalability. We used MA-POCA for training agents that are up to mark for multi-agents as if they get spawned or deleted during training, MA-POCA supports this behavior and distributes the rewards accordingly.

G. Zuin et al propose techniques for explainable resource scales in collectible card games" this is a paper that explores the use of deep learning techniques to improve the balancing of resources in collectible card games. [34] The background is that balancing resources in these games is critical to ensure fairness and enjoyable gameplay, but it is challenging due to the large number of factors that need to be considered. The problem statement is to develop an approach that can effectively balance resources in a way that is explainable to players. The method involves using a deep learning model to learn the relationship between game features and resource scales and then using this model to provide explanations for the chosen resource scales. The proposed algorithm is called "Explainable Resource Scaling using Deep Learning (XRL)." The outcome is a demonstration that the proposed approach can effectively balance resources in a way that is explainable to players, leading to an improved gameplay experience. We used curriculum learning introducing one prop for the door at a time. This enables agents to identify the concept that agents have to use one prop for one door and have to take the other prop to another door.

T. T. Nguyen et.al provides an overview of the challenges and solutions in applying deep reinforcement learning (DRL) to multiagent systems. The background is that multiagent systems are prevalent in many real-world applications, but traditional methods for coordinating agents can be limited. The problem statement is to explore how DRL can be used to improve the coordination of agents in multiagent systems. [35] The method involves reviewing the challenges and solutions in applying DRL to multiagent systems and examining various applications of DRL in this context. The outcome is a comprehensive review of the challenges, solutions, and applications of DRL in multiagent systems, highlighting the potential of DRL to address coordination problems in complex systems.

E. Alonso et al present a method for training agents to navigate complex environments in AAA video games using deep reinforcement learning (DRL). The background is that navigation in complex video game environments can be challenging for traditional AI techniques. The problem statement is to explore the use of DRL for navigation in video games. The method involves using a DRL algorithm called Deep Q-Network (DQN) to learn navigation policies from raw pixel inputs. [4]The outcome is a successful application of DRL to navigation in complex video game environments, demonstrating the potential of DRL for improving game AI. We used PPO and MA-POCA for agents to learn navigation using rewarding behaviors.

P. Xu et.al presents a method to improve exploration in deep reinforcement learning algorithms for video games by incorporating a part-aware exploration bonus to encourage the agent to interact with all parts of the game environment and evaluates the effectiveness of this method on several Atari games using the DQN algorithm, showing significant performance improvements compared to baseline methods. We used the curiosity element for agents to explore and get the reward.

N. D. Dung et.al presents an algorithm for drone-following models in smart cities to improve traffic flow and reduce congestion. The problem statement is how to effectively and safely use drones in urban environments. The method involves developing a drone-following algorithm that takes into account traffic conditions, vehicle speeds, and the presence of other drones. [36] The algorithm used in the paper is a fuzzy control system. The outcome of the paper is a proposed model for a drone-following algorithm that can be applied in real-world scenarios. We used fontal special sensors and ray casts for agents to explore the environment.

I. Mordatch et.al investigates how compositional language emerges in a population of agents, which can interact with each other and their environment through a shared communication protocol. The problem statement is to understand how a group of agents, each with their own goals and limited abilities, can develop a common language to achieve their objectives in a collaborative setting. The method used in the paper involves training agents with a variant of the Reinforcement Learning algorithm to learn to communicate and collaborate in a simulated environment. [37] The outcome of the paper demonstrates that grounded compositional language can emerge spontaneously in a population of agents without explicit language supervision, paving the way for further research in this area. We used MA-POCA for multi-agents so they can train parallel because MA-POCA supports when an agent from a group of agents de-spawns or gets destroyed.

N. Heess et.al presents a study of artificial intelligence agents learning to walk in various terrains using reinforcement learning. The background discusses the importance of locomotion in robotics and the potential for AI to solve this problem. The problem statement is the difficulty of training agents to walk in complex environments without hand-engineering solutions. The method used is reinforcement learning, where agents receive rewards for successful locomotion and are penalized for falling [38]. The algorithm used is Deep Deterministic Policy Gradient (DDPG). The outcome of the paper is the successful training of agents to walk in various terrains, demonstrating the potential for AI to solve complex locomotion problems. Along with giving positive rewards we also introduced curiosity and curriculum learning that helped agents in successful locomotion.

T. Bansal et.al proposes a method for generating complex behaviors in artificial intelligence agents through competition. The background discusses the importance of emergent complexity in AI and the potential for multi-agent systems to create it. The problem statement is the difficulty of designing agents to exhibit complex behaviors without hand-engineering solutions. The method used is a competition-based approach where agents compete to complete a task. [39] The algorithm used is Multi-Agent Deep Deterministic Policy Gradient (MADDPG). The outcome of the paper is the successful generation of complex behaviors in multi-agent systems, demonstrating the potential for competition-based approaches to generate emergent complexity in AI. we use two different types of agents that were in a competition based on the hide-and-seek principle.

S. Liu et.al presents a method for generating coordinated behavior in artificial intelligence agents through competition. The background discusses the importance of coordination in multi-agent systems and the potential for competition to induce it. The problem statement is the difficulty of designing agents to exhibit coordinated behavior without hand-engineering solutions. The method used is a competition-based approach where agents compete with each other to complete a task that requires coordination. [40] The algorithm used is Multi-Agent Deep Q-Network (MADQN). The outcome of the paper is the successful generation of coordinated behavior in multi-agent systems, demonstrating the potential for competition-based approaches to generate emergent coordination in AI. We introduce two different agents that competed one agent was a hider and the other was seeker secret agents were using the proximal policy optimization algorithm while the hiders were using MAPOCA.

M. Alwateer et.al presents a method for enabling drone services through crowdsourcing and scripting. The background discusses the potential for drones to provide various services and the challenges of designing systems to support these services. The problem statement is the difficulty of creating a platform that allows users to easily request drone services and script drone behavior. The method used is a crowdsourcing and scripting approach where users can request services and specify drone behavior through a web-based platform. [41]

The outcome of the paper is the successful implementation of a platform that enables drone services, demonstrating the potential for crowdsourcing and scripting approaches to enable the use of drones in various applications. We use frontal and spatial sensors for the drones which enabled the drones to identify and observe the environment around them. We also introduced a level of difficulty by adding windows and doors. The agent has to pick up the probes and take them to the windows which were higher than the doors to create a safe surrounding.

D. Hong et.al presents a method for energy-efficient path planning of multiple drones using reinforcement learning. The background discusses the importance of energy efficiency in drone applications and the potential for AI to optimize drone behavior. The problem statement is the difficulty of designing systems to optimize energy consumption for multiple drones flying in a dynamic environment. The method used is a reinforcement learning approach where agents learn to navigate through an environment while minimizing energy consumption. [12] The algorithm used is Deep Deterministic Policy Gradient (DDPG). The outcome of the paper is the successful implementation of an energy-efficient path planning system for multiple drones, demonstrating the potential for reinforcement learning to optimize drone behavior. If you talk about optimization our agents are designed for the shortest earliest path as we were giving negative rewards per frame if the agent is not able to finish the episode earliest.

J. Weng et.al proposes a method for accelerating the training of reinforcement learning agents through parallelization. The background discusses the importance of parallelization in reinforcement learning and the potential for faster training times. The problem statement is the difficulty of training agents with large amounts of data without parallelization. The method used is a parallelization approach where multiple environments are executed simultaneously on a single machine. [42] The algorithm used does not apply to this paper. The outcome of the paper is the successful implementation of a parallelization engine called EnvPool, demonstrating the potential for parallelization to accelerate the training of reinforcement learning agents. Unity’s ML agent provides a methodology known as parallelism in which we introduce 12 similar environments prefabs each with a set of similar numbered agents. during the training phase each prefab was initialized and the agents inside them get to train differently.

T. S. Ray et.al presents a method for studying evolution and ecology in digital organisms through a simulation platform called Avida. The background discusses the importance of understanding evolution and ecology in biological systems and the potential for digital simulations to provide insights into these processes. The problem statement is the difficulty of studying evolution and ecology in biological systems due to the complexity of interactions and the limitations of empirical data. The method used is a simulation-based approach where digital organisms evolve and interact with each other in a controlled environment. [43] The outcome of the paper is the successful implementation of a simulation platform that allows researchers to study evolution and ecology in digital organisms, demonstrating the potential for digital simulations to provide insights into complex biological processes. we introduce the evolution of two different drones one was the seeker and one was the hider they need to evolve to compete with each other and get the reward. we can observe them evolving according to the complexity of the environment and the complexity of other agents' brains.

S. Leonardos et.al proposes a method for balancing exploration and exploitation in multi-agent learning using catastrophe theory and game theory. The background discusses the importance of balancing exploration and exploitation in multi-agent learning and the potential for AI to optimize learning strategies. The problem statement is the difficulty of designing systems that balance exploration and exploitation in multi-agent learning. The method used is a hybrid approach that combines catastrophe theory and game theory to optimize learning strategies [44]. The algorithm used is the Replicator Dynamics algorithm. The outcome of the paper is the successful implementation of a learning system that balances exploration and exploitation in multi-agent learning, demonstrating the potential for hybrid approaches to optimize learning strategies in complex environments. We introduced curiosity for the exploration of the multi-agents but we also added negative divorce for the agents does not deviate from the main rewarding actions.

D. A. Suyikno et.al proposes a method for designing feasible NPC hiding behavior in a hide-and-seek 3D game simulation using goal-oriented action planning. The background discusses the importance of designing realistic NPC behavior in-game simulations and the potential for AI to improve game design. The problem statement is the difficulty of designing feasible NPC hiding behavior in a complex game simulation. The method used is a goal-oriented action-planning approach that generates plans for NPC behavior based on predefined goals. [45] The algorithm used is the Fast Downward planner. The outcome of the paper is the successful implementation of a system that generates feasible NPC hiding behavior in a hide-and-seek 3D game simulation, demonstrating the potential for goal-oriented action planning to improve NPC behavior in game simulations. In this paper, the target was stationary while our experimentation contained both of the agents having a separate brain and separate training. Our experiment contains two different agents that were competing with each other and with time they were getting more experience.

R. Zhang et.al proposes a method for online motion planning in a multi-UAV pursuit-evasion game using deep reinforcement learning. The background discusses the importance of efficient motion planning in UAV applications and the potential for AI to improve performance. The problem statement is the difficulty of designing efficient motion planning strategies for multiple UAVs in complex environments. The method used is a deep reinforcement learning approach that learns to generate motion plans in real-time. The algorithm used is the deep Q-network algorithm. The outcome of the paper is the successful implementation of a system that generates efficient motion plans for multiple UAVs in a pursuit-evasion game, demonstrating the potential for deep reinforcement learning to improve motion planning strategies in complex environments [8]. We used PPO and MA-POCA to train the agents and we also introduced a continuous negative reward which was very small and was given at each frame until the agent learns to end the episode quickly by doing this the agent was able to end the episode as soon as possible to reduce the quantities negative reward.

M. Jaderberg et.al proposes a method for achieving human-level performance in 3D multiplayer games using population-based reinforcement learning. The background discusses the potential for AI to improve game performance and the challenges of achieving human-level performance. The problem statement is the difficulty of designing game AI that can perform at a human level in complex, dynamic environments. The method used is a population-based reinforcement learning approach that trains multiple agents simultaneously using an evolutionary algorithm. The algorithm used is the Covariance Matrix Adaptation Evolution Strategy (CMA-ES). [46] The outcome of the paper is the successful demonstration of a system that achieves human-level performance in a 3D multiplayer game, demonstrating the potential for population-based reinforcement learning to improve game AI performance in complex environments. With the help of curriculum learning, we made our agents this much smarter that they were able to identify and overcome the challenges of the gradually difficult environment end with the help of competition between the two hiders and seekers.

A. T. Bourdillon et.al proposes a method for integrating reinforcement learning in a virtual robotic surgical simulation. The background discusses the potential for using reinforcement learning to improve surgical training and the challenges of designing effective simulation environments. The problem statement is the difficulty of designing realistic surgical simulations that can effectively train surgeons. The method used is a reinforcement learning approach that trains a virtual robot to perform surgical tasks. The algorithm used is the Q-learning algorithm [47]. The outcome of the paper is the successful demonstration of a system that can effectively train a virtual robot to perform surgical tasks, demonstrating the potential for reinforcement learning to improve surgical training in a simulated environment. we used a unities simulation environment in which we introduce two different drones that for training with two different algorithms PPO and MA-POCA.

S. Sukhbaatar et.al proposes a method for automatic curriculum generation in reinforcement learning through asymmetric self-play. The background discusses the challenges of designing effective curricula in reinforcement learning and the potential benefits of using an intrinsic motivation to guide learning. The problem statement is the difficulty of designing effective curricula and the need for a method that can automatically generate curricula tailored to individual agents. The method used is an asymmetric self-play approach that encourages agents to explore novel strategies and tasks. The algorithm used is a combination of Proximal Policy Optimization (PPO) and Curiosity-Driven Exploration (CDE) [48]. The outcome of the paper is the successful demonstration of a system that can automatically generate effective curricula for individual agents, demonstrating the potential for intrinsic motivation and asymmetric self-play to improve reinforcement learning. We used a combination of proximal policy optimization and MA-POCA to create effective curricula using reinforcement learning intrinsic motivation.

S. Forestier et.al proposes a method for intrinsic motivation and automatic curriculum learning for reinforcement learning agents. The problem statement is to enable agents to autonomously explore their environments and learn new skills without explicit external rewards or human intervention. The method involves creating an automatic curriculum of increasingly challenging goals for the agent to pursue based on its current skill level. The algorithm used is called Intrinsically Motivated Goal Exploration Processes (IMGEP) [49]. The outcome is that the proposed method improves exploration and learning efficiency, leading to better performance on a variety of tasks compared to existing methods. we edit curiosity elements in our agents that introduced an exploration bonus and enable the agent to explore more and learn more feasible and suitable actions.

S. Singh talks about intrinsically motivated reinforcement learning. The background of this paper is reinforcement learning, where an agent learns to take actions in an environment to maximize a reward signal. The problem statement is that most RL algorithms require an explicit reward function, which can be difficult or expensive to define in some domains. The method used is intrinsically motivated RL, which generates its reward signal based on the agent's internal states and goals [50]. The name of the algorithm used is Intrinsic Motivation and Procedural Generation (IMAP). The outcome of this paper is that IMAP can achieve better performance and sample efficiency compared to traditional RL methods in some domains. We added the curiosity element that introduced the intrinsically motivated agents Heather was first given a bonus upon exploring the environment and getting such actions that resulted in rewarding them even more.

A. Tucker wrote on inverse reinforcement learning for video games. The background of this paper is the use of Inverse Reinforcement Learning (IRL) in video games, which aims to infer the reward function from demonstrations of expert gameplay. The problem statement is how to apply IRL to learn the reward function in video games where the reward signal is not explicitly defined. The method used is an IRL algorithm that infers the reward function based on observations of the expert's actions. The name of the algorithm used is Maximum Entropy Inverse Reinforcement Learning (MaxEnt IRL) [5]. The outcome of this paper is that the MaxEnt IRL algorithm is effective in learning the reward function in video games and can be used to train agents to perform well in those games.

Y. Burda et.al worked on a Large-scale study of curiosity-driven learning. In his paper, the background of this paper is the use of curiosity-driven learning as a method for artificial agents to acquire skills and knowledge. The problem statement is that there is a need for an effective and scalable approach to curiosity-driven learning. The method used in this paper involves training a large number of agents in a variety of environments and analyzing the resulting data. The algorithm used is a form of intrinsic motivation called the "intrinsic curiosity module" (ICM), which encourages agents to explore their environment and learn new skills. The outcome of the study shows that the use of curiosity-driven learning with ICM can lead to significant improvements in agent performance and skill acquisition, especially in complex and dynamic environments [51]. With the decision to add curiosity-based agents, we also introduce competition among two different types of agents hiders and seekers.

The paper "Layer Normalization" by Ba, Kiros, and Hinton proposes a normalization method for deep neural networks that helps to overcome the degradation problem of deep architectures. The problem statement is that the performance of deep neural networks tends to saturate or even degrade as the network depth increases. The proposed method, named "Layer Normalization," normalizes the inputs to each layer based on the statistics of their activations [52]. This is achieved by computing the mean and standard deviation of the activations across the feature dimensions and normalizing them. The outcome of the paper is a method that improves the training speed and performance of deep neural networks on various tasks such as image classification and language modeling. In the scenario of hard-coded observations, we normalize these values to get the neural network to train much faster as compared to training on larger numerical values.

A. Rajeswaran et.al works on Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. The background of the paper is the development of robotic manipulation skills using deep reinforcement learning and demonstrations. The problem statement is to overcome the challenge of high-dimensional state and action spaces in robotic manipulation tasks. The method used is combining reinforcement learning with demonstrations from human experts. The algorithm used is Deep Deterministic Policy Gradient (DDPG) [53]. The outcome of the paper is that the proposed method can learn complex manipulation tasks with high-dimensional action spaces and improve the sample efficiency of reinforcement learning.

OpenAi et.al proposed work on learning Dexterous In-Hand Manipulation. The background of the paper is the challenge of in-hand manipulation for robotic grasping and manipulation tasks. The problem statement is to learn a policy for dexterous in-hand manipulation from scratch without human demonstrations. The method used is reinforcement learning, specifically the soft actor-critic algorithm, which is a deep reinforcement learning algorithm that allows for continuous control [54]. The outcome of the paper is a trained policy that can perform complex in-hand manipulation tasks, such as rotating an object in the hand or flipping it over, without prior demonstrations or guidance. We used two competing agents one in one the higher second one is the seeker, and also to introduce competition among them. Seekers were using PPO while hiders were using MA-POCA.

S. Sukhbaatar et.al proposes a method for learning communication protocols between multiple agents using backpropagation. The problem statement is to enable multiple agents to learn to communicate with each other in a decentralized way without explicit coordination. The method involves using an autoencoder to learn a shared communication space between agents and using backpropagation to train the agents to map their observations to this shared space [55]. The algorithm used is called Differentiable Inter-Agent Learning (DIAL), which is an extension of the REINFORCE algorithm. The outcome is that DIAL enables agents to learn communication protocols in a decentralized way and outperforms other communication methods in various multi-agent environments. This methodology is a lot similar to imitation learning in which the hiders and seekers can learn from expert human beings' demonstrations but that disables the ideology behind emergent behavior so we'd rather use curriculum learning to train agents gradually.

J. N. Foerster et.al proposed work on Learning to communicate with deep multi-agent reinforcement learning. The background of this paper is that communication is a fundamental aspect of human cooperation and a key component of successful multi-agent systems. The problem statement is to design agents that can learn to communicate with each other to achieve a common goal. The method used in this paper is deep multi-agent reinforcement learning, which employs a centralized critic and decentralized actors. The algorithm used is called MADDPG (Multi-Agent Deep Deterministic Policy Gradient) [49]. The outcome of this paper is that the proposed method achieved successful communication between agents in a cooperative navigation task and outperformed other baseline methods. To achieve cooperation between the agents we introduce multiple agents in a single team they learned to coordinate among themselves to achieve a greater goal

N. Haber et.al proposes a framework for self-aware agents that can play a variety of games without explicit rewards. The problem statement is to enable agents to develop their own goals, rather than relying on predefined objectives, by using curiosity-based intrinsic motivation. The method used is a combination of reinforcement learning and meta-learning. The proposed algorithm is called "Intrinsically Motivated Goal Exploration Processes with automatic curriculum learning" (IMGEP-AC), which involves generating goals using a self-organizing map and learning an exploration policy with intrinsic motivation [56]. The outcome is that the proposed framework can learn to play several games, including some that require long-term planning and coordination between agents, without the need for external rewards or supervision. This experimentation is done without the mentioning of specific goals, we on the other hand introduced reward functions that enable the agent to learn from them more quickly and efficiently rather than exploring all of the environment so that it can learn on a very lucky shot.

S. Hochreiter et.al proposed a new type of artificial neural network called LSTM that is capable of handling long-term dependencies, which was a major limitation of traditional neural networks. The problem statement was that standard recurrent neural networks (RNNs) failed to capture the long-term dependencies in sequences due to the vanishing gradient problem. The LSTM architecture addressed this issue by introducing memory cells with gating mechanisms that selectively forget and update information over time [57]. The algorithm used in this paper is called LSTM, which is a type of RNN with memory cells and gating mechanisms. The outcome of this paper was the development of a new neural network architecture that could overcome the problem of vanishing gradients and handle long-term dependencies in sequences, which has become widely used in various applications such as natural language processing, speech recognition, and image captioning. We also introduce LSCM in our agents with a sequencing length of 64 and a memory size of 256. This enabled the agents to identify what observations were taken and when those observations were fruitful to generate a rewarding action.

J. Z. Leibo proposed work on Malthusian reinforcement learning in which the background of the paper "Malthusian reinforcement learning" is to address the problem of catastrophic forgetting in reinforcement learning models. The paper introduces a novel method to avoid catastrophic forgetting by limiting the capacity of the agent's neural network through a Malthusian growth constraint [58]. The algorithm used in this paper is a modified version of Q-learning called "Malthusian reinforcement learning". The outcome of the paper shows that the proposed approach can effectively avoid catastrophic forgetting and improve the agent's performance on a variety of tasks. The proximal policy optimization algorithm already addresses this problem in which when the new policy drastically changes from the previous policy, it normalizes the change so that the newer and more variants of policy do not affect the existing good policy.

C. D. Rosin et.al propose a method to generate high-quality opponents in competitive games by co-evolving them alongside the player agents. The problem statement is to create challenging and adaptive opponents in games that can provide a stimulating experience for players. The method involves co-evolving a population of players and opponents using a fitness function that incentivizes the players to win against increasingly challenging opponents. The algorithm used is called Competitive Co-evolution. The outcome of the paper is a set of techniques to create adaptive and challenging opponents that can help improve the player experience in competitive games. To create such opponents that are worth trading against we trained the opponents until they were showing very prominent results and then we switched the training to the other agents such that the other agent has now a much stronger brain rather than a random action brain to train against

R. Lowe et.al proposes a multi-agent actor-critic algorithm for learning in mixed cooperative-competitive environments, where agents have to balance between cooperation and competition. The authors highlight the importance of developing algorithms that can handle both types of situations, as many real-world scenarios fall into this category. They use a centralized critic and decentralized actors to ensure scalability and efficiency [59]. The algorithm is tested on a variety of tasks, including a predator-prey game and a traffic intersection control problem, and outperforms other baseline methods. The algorithm's name is Multi-Agent Deep Deterministic Policy Gradient (MADDPG). Our methodology included training seekers first so that hiders can learn the main hiding function Otherwise hiders cannot learn whether they have to confine themselves and use the tools properly so that they can hide in a specific room odyssey place.

J. Z. Leibo et.al proposed work on Multi-agent reinforcement learning in sequential social dilemmas. The background of the paper "Multi-agent reinforcement learning in sequential social dilemmas" is on the challenges of cooperation and competition among multiple agents in complex environments. The problem statement is on finding effective strategies for agents to collaborate and compete in a way that maximizes their collective reward. The method used is multi-agent reinforcement learning, where agents learn from their individual experiences and interactions with other agents to improve their policies [21]. The algorithm used is a variant of deep Q-learning called Deep Recurrent Q-Networks (DRQN). The outcome of the paper is a framework for multi-agent reinforcement learning that addresses social dilemmas and demonstrates the effectiveness of the DRQN algorithm in complex multi-agent environments. We introduced a competition between PPO and MAPOCA two generate new emerging strategies.

J. Perolat et.al aims to develop a computational model to study the dynamics of common-pool resource appropriation in a multi-agent setting. The problem statement focuses on understanding how different agents interact with each other while competing for limited resources, and how these interactions affect the overall system behavior. The method used is a multi-agent reinforcement learning approach, where agents learn from their past experiences to optimize their behavior over time. [60] The specific algorithm used in this paper is Q-learning. The outcome of this study helps to provide insights into the factors that drive common-pool resource appropriation and could inform the design of better policies to manage such resources.

L. Buşoniu proposed an overview of Multi-agent Reinforcement Learning. The background of this paper is the growing interest in multi-agent systems, which can be found in several domains, such as robotics, economics, and social networks. The problem statement is that multi-agent systems are complex and challenging to design due to the interactions and dependencies among the agents. The method used is reinforcement learning, which provides a framework for agents to learn and adapt to their environment based on rewards and punishments. [61] The name of the algorithm is Multi-Agent Reinforcement Learning (MARL). The outcome of this paper is an overview of the state-of-the-art in MARL, including its challenges, applications, and future directions. We not only introduces multi-agent reinforcement learning agents but also introduced computing agents such that their reward function was entirely different from each other.

H. Wafa et.al describes a method for simulating multi-agent hide-and-seek games, where the hiders aim to remain hidden while the seekers try to find them, using trust region policy optimization with a modified policy gradient algorithm. The goal is to demonstrate the effectiveness of reinforcement learning algorithms in solving complex, dynamic problems with multiple agents, as well as to explore the dynamics of competitive and cooperative interactions in multi-agent systems [62]. The outcome shows that the proposed approach can learn effective policies for both the hiders and seekers and that the performance of the agents improves as the number of agents and complexity of the environment increases. We introduced a third dimension which enables the agents to fly like drones and also introduces a complex environment that they have to solve by using the props and dragging the props on a heightened opening or doorway such that hiders can confine themselves in a safe environment.

D. D. Ningombam et.al proposes a novel exploration technique, called "Unexplored Move Pruning" (UMP), for multi-agent reinforcement learning (MARL) in competitive games. The problem statement is that traditional exploration techniques often require a large number of samples to find effective policies in MARL. The method combines UMP with an existing MARL algorithm, Counterfactual Multi-Agent (COMA), to improve exploration efficiency [63]. The outcome of the paper demonstrates that UMP can significantly improve the exploration efficiency in MARL, leading to better performance in competitive games with fewer samples. We enabled the agent to explore the environment and reward them for exploring as curiosity was added to it. This enabled the agent to use the prop to drag it and take it to a farther down-the-lane doorway so that it can close it.

C. Diuk et.al proposed work on an object-oriented representation for efficient reinforcement learning. The background of this paper is on reinforcement learning and its representation in an object-oriented form, which can be used to build efficient models for a range of applications. The problem statement is to represent the world state in a structured way that can be learned and updated by an agent. The method used is object-oriented representation, which defines the state of the world in terms of objects, their attributes, and relationships [64]. The algorithm used in this paper is Q-learning, which is a well-known model-free reinforcement learning algorithm. The outcome of this paper is that the object-oriented approach can be a powerful tool for building efficient models for a range of applications. We not only use obey rented representation but also introduced an environmental master class that was able to control all of the agent's reward functions and reset the environment when the episode was finished this master class was able to identify and sync all of the agent's episode length.

C. Jestel et.al proposes a method to obtain robust control and navigation policies for multi-robot navigation via deep reinforcement learning. The problem statement is to improve the navigation and control policies for multiple robots in dynamic environments, considering the uncertainties and disturbances. The method uses a deep reinforcement learning algorithm called Trust Region Policy Optimization (TRPO) to learn the policies [65]. The outcome shows that the proposed approach can effectively learn robust navigation policies for multiple robots in dynamic environments, outperforming the traditional rule-based method. We use the successor of TRPO which is PPO and MAPOCA. Which resulted in faster and quicker convergence of strategies.

A. Cohen et.al states in multi-agent reinforcement learning" discusses the use of absorbing states, which are states from which the agents cannot escape, in multi-agent reinforcement learning (MARL). The problem statement is that the use of absorbing states can lead to suboptimal policies, and the paper proposes a method to address this issue. The method used in the paper involves modifying the reward function to discourage agents from entering absorbing states [66]. No specific algorithm name is mentioned in the paper. The outcome is that the proposed method improves the performance of MARL agents in tasks with absorbing states. We addressed this issue by introducing a curiosity module in the agent which enables the agent to explore the environment to get more rewarding actions.

Y. Duan proposed work on One-shot imitation learning. the background of this paper is on imitation learning, a popular technique for teaching agents to perform tasks by learning from expert demonstrations. The problem statement is that traditional imitation learning algorithms typically require a large amount of data to learn from, which can be time-consuming and expensive. The authors propose a method called one-shot imitation learning, which allows an agent to learn a new task from a single demonstration [67]. The algorithm used is a variation of Siamese neural networks that use a shared weight structure to compare the input of the demonstration and the current state of the environment. The outcome of the paper shows that their approach is effective and efficient, allowing agents to learn new tasks quickly and with fewer data than traditional imitation learning methods. Ml-Agents just provide imitation learning in which the agent can be trained with an existing demo brain. but we didn't use imitation learning as it will eliminate the emerging behavior of strategies as it will only learn from the demonstrations and build upon those demonstrations.

J. Ren et.al proposes a method to balance the reward function in reinforcement learning to account for orientation-preserving actions. The problem statement involves cases where the optimal policy requires a robot to rotate an object and perform other actions that change the object's orientation, but the reward function only values the final object configuration. The method uses a balancing term to reward orientation-preserving actions, and it applies a projection to ensure that the final reward is non-negative [68]. The outcome of the paper is an approach to improve reinforcement learning in tasks that require orientation preservation, such as manipulation tasks. The proximal policy optimization algorithm already addresses this issue that the agent at the end of the episode needs to increase the reward from the baseline 0 and it will take a lot of episodes to do this.

R. Wang et.al proposes a new approach to generating diverse and complex environments for reinforcement learning, called Paired Open-Ended Trailblazer (POET). The problem statement is to create a learning environment that can continuously generate increasingly complex and diverse challenges for agents to learn from. The method involves evolving pairs of environments and agents, where the fittest agents are paired with the fittest environments and vice versa [69]. The algorithm used is a combination of novelty search and multi-objective optimization. The outcome is a system that can generate an endless stream of diverse and challenging environments for reinforcement learning, resulting in agents that can generalize better and perform well in unseen environments. We adopted this ideology that a trained brain should compete with a non-trained brain we trained the seekers first and then started training on the hiders this enables the hider to completely understand how difficult the task is and randomly moving or staying behind a prop will not solve their permanent issue.

N. Matsumura et.al proposes a novel method for improving the performance of a drone-based multi-input multi-output (MIMO. The problem statement is to find the optimal placement of the drone to maximize the system's capacity and minimize power consumption. The method uses a genetic algorithm to search for the optimal placement of the drone based on various constraints and objectives. The proposed algorithm is named the Genetic Algorithm for Drone Placement (GADP) [70]. The outcome of the paper shows that the proposed algorithm significantly improves the system's performance compared to conventional methods. Reinitialize the drones in a random position so that they can learn to navigate to the goal whether they are near or farther away. This enables the agent to adapt to the localization and to navigate in a complex environment.

M. Jaderberg et.al proposes a method for training neural networks using a population-based approach that searches for optimal hyperparameters such as learning rates and weight initialization schemes. The problem statement is to improve the efficiency and effectiveness of training neural networks. The method used is a combination of evolutionary algorithms and reinforcement learning, where populations of neural networks are trained and evaluated in parallel to optimize the hyperparameters [71]. The algorithm used is called Population Based Training (PBT). The outcome of the paper shows that PBT can achieve state-of-the-art performance on various deep learning benchmarks while being computationally efficient. He used a population of 12 environments running parallel with each other each environment contains a set of agents that are training and learning experiences based on their actions in the given state.

G. Wu et.al proposes a reinforcement learning-based approach to optimize the coordination of trucks and drones for delivery. The problem statement is to find the optimal allocation of delivery tasks between trucks and drones to minimize the delivery cost. The method uses a centralized RL algorithm with a shared value function and a decentralized actor policy [10]. The name of the algorithm is "Multi-Agent Deep Deterministic Policy Gradient (MADDPG)." The outcome of the paper shows that the proposed approach can significantly reduce the delivery cost and improve delivery efficiency compared to the traditional delivery methods.

A. Ahadi et.al focuses on the replication crisis in computing education research, where several studies are non-replicable. The problem statement is the lack of rigor and transparency in research practices, leading to unreliable results. The method used in this paper is a survey of computing education researchers to understand their attitudes and experiences related to replication [2]. The outcome of this paper is the identification of barriers to replication and recommendations for improving research practices in computing education. We adopted the ideology of an open AI hydrogen secret experiment and replicated it in a drone-like fashion in which the agents can fly and they have to learn to use the props and close heightened entry points.

A. Ahadi et.al addresses the lack of replication studies in computing education research (CER) and the potential consequences for the validity and reliability of research findings in this field. The authors conducted an online survey to investigate the attitudes and experiences of CER researchers regarding replication studies. While most respondents viewed replication as important, only a small percentage had conducted replication studies themselves, citing time and resource constraints as barriers [2]. The study highlights the need for more support and incentives for replication studies in CER to ensure the rigor and credibility of the research in this field. I would experiment and manage to generate similar results much faster with very less batch size and resources required during the hide-and-seek experimentation. Thus replicating using our methodology resulted in better behavior.

Y. Liu et.al addresses the issue of replication in AI research. Replication is essential to verify and validate the results of AI research, but it is often overlooked or considered unimportant. The authors propose the use of replication markets to incentivize and facilitate replication studies in AI. Replication markets are prediction markets that allow researchers to bet on the likelihood of a replication study producing the same results as the original study. The authors conducted a replication market for two AI studies and found that the markets provided an effective and efficient way to incentivize replication and encourage transparency in AI research [40]. The study highlights the potential of replication markets to improve the rigor and credibility of AI research and calls for further experimentation and refinement of the approach.

I. M. A. Nahrendra proposed that tilting-rotor drones are a promising technology for autonomous aerial transportation, but they are challenging to control due to their complex dynamics. The existing nominal controllers for tilting-rotor drones are not optimal and do not account for all the complex dynamics of the drone, leading to suboptimal performance. To address this problem, the authors propose a new approach called Retro-RL, which combines a nominal controller with deep reinforcement learning (RL) to learn a better control policy for the drone [7]. The Retro-RL algorithm uses a policy gradient method to train the RL agent, which is then combined with the nominal controller retroactively. The authors evaluate the Retro-RL approach on a simulated tilting-rotor drone and demonstrate that it outperforms the nominal controller in terms of stability, tracking performance, and robustness to disturbances. The study highlights the potential of RL to improve the performance of complex systems and the importance of combining RL with existing control methods to achieve better results.

C. Florens et.al tackle the problem of designing an effective curriculum for reinforcement learning (RL) agents. Traditional RL algorithms often assume a fixed curriculum, which can be suboptimal for complex tasks. The authors propose a new approach called reverse curriculum generation, which uses a generative adversarial network (GAN) to generate a sequence of increasingly difficult tasks for the RL agent. The GAN is trained to generate tasks that are easy for the RL agent to solve but difficult for a fixed heuristic agent. The RL agent is then trained on the generated tasks in reverse order, starting from the most difficult task and working backward. The authors evaluate the approach on several Atari games and demonstrate that it outperforms traditional RL algorithms and other curriculum generation methods. The study highlights the potential of using GANs to generate effective curricula for RL agents and the importance of tailoring the curriculum to the agent's capabilities [72]. Emulation provides the ability to imitate from a demo generated by an experienced player this can also be identified as reverse reinforcement learning In such a manner that the agents are given a demonstration of exact actions that will yield them better rewards. But in our case, the environment was very complex and the agents were spawning randomly thus having a demonstration of every possible state to action was not a very bright idea thus we introduced curriculum learning which enabled the agent to learn gradually with the increase of difficulty in the environment.

M. Seo et.al proposes a novel method to solve the credit assignment problem in RL agents receiving sparse binary rewards, which involves using a neural network to predict the expected future rewards for each action in the current state, and then assigning credit to the actions that contribute to the future rewards. The authors evaluate the proposed method on several sparse reward environments and demonstrate that it outperforms traditional RL algorithms and other credit assignment methods [73]. The study emphasizes the importance of developing new approaches to address the challenges of sparse reward environments and highlights the potential of rewards prediction for solving the credit assignment problem. With the help of curriculum learning and curiosity, items were able to identify the sparse rewards and learned very quickly how to end the episode while on their best behavior.

M. M. M. van Dooren et.al investigates the impact of different types of rewards on play persistence in adolescents with and without substance dependence. The authors note that gaming can be addictive and that understanding how different types of rewards affect play persistence could inform strategies for mitigating problematic gaming behaviors. The study uses a controlled experiment to compare the effects of monetary rewards, virtual points, and social rewards on play persistence in both groups [74]. The authors find that monetary rewards have the strongest effect on play persistence, followed by social rewards, with virtual points having the weakest effect. The study provides insights into the factors that motivate play in adolescents and could inform the development of interventions for problematic gaming behaviors.

N. Jaques et.al proposes a method to improve the performance of multi-agent RL by using social influence as intrinsic motivation for agents. The authors note that in multi-agent environments, agents must learn to cooperate and coordinate their actions, which can be challenging. The study proposes using social influence as a form of intrinsic motivation, where agents are rewarded for influencing the actions of other agents in the environment. The authors evaluate the proposed method on several multi-agent environments and demonstrate that it outperforms traditional RL algorithms and other intrinsic motivation methods. The study provides insights into how social influence can be used to incentivize cooperative behavior in multi-agent environments and highlights the potential of intrinsic motivation for improving RL performance [75]. With the help of curiosity and continuous increase in the difficulty of the environment, we induce the agent with the ability to learn in a gradually difficult environment. The social influence can be demonstrated as the gradual increase in the difference of the environment that makes it much more difficult for the agent to identify and navigate with the newer level.

O. E. Gundersen et.al examines the issue of reproducibility in AI research. The authors note that many AI research results are difficult to reproduce due to the complexity of the models and the lack of standardization in the field. The study proposes several guidelines and best practices for improving reproducibility in AI research, including using open-source software, providing detailed documentation, and making data and code publicly available. The authors also review several reproducibility initiatives in the field and highlight the need for greater collaboration and standardization [76]. The study provides important insights into the challenges of reproducibility in AI research and proposes actionable solutions for improving the reliability and transparency of AI research results.

V. Bapst et.al proposes a structured approach to training agents for physical construction tasks, using a combination of RL and supervised learning. The study highlights the complexity of physical construction tasks and the need for coordinated planning and execution. The proposed method outperforms traditional RL methods and demonstrates the potential for using structured agents in real-world applications [77]. The study provides insights into how structured agents can be designed and trained for complex tasks, such as physical construction. With the help of unity 3D, we introduce physics in the agent's properties including mass drag angular drag clean gravity, etc. this enabled the agent to learn in a real lifelike manner rather than just a physics-less simulation.

J. Achiam et.al proposes a novel approach to incorporating intrinsic motivation in RL by using surprise as a reward signal. The study argues that surprise can be used to encourage exploration and learning more efficiently in complex environments. The proposed method is based on an extension of Q-learning called Surprise Q-learning, which uses a separate module to compute surprise values [78]. The study demonstrates the effectiveness of the proposed method in several benchmark tasks and shows that it outperforms traditional RL methods. The study suggests that surprise-based intrinsic motivation can be a useful tool for RL agents operating in complex and uncertain environments. once the price element in our research was the enabling of curriculum learning on the environmental prefabs once the agent learned to get a certain amount of reward the environment was made difficult test new challenges arises as a surprise for the agent

J. Lehman proposed work on the surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities. The background of this paper is the study of evolutionary computation and artificial life research communities. The problem statement is to examine the surprising creativity of digital evolution. The method used is to collect anecdotes from researchers in these fields. The paper presents a collection of stories that demonstrate the creative capabilities of digital evolution, highlighting the importance of exploring new ideas and approaches in AI research [79]. We use the principle of coevolution in agents. hydrants and seekers were given the competition in cooperation to themselves stop

G. Tesauro et.al describes a reinforcement learning algorithm, TD-Gammon, which uses temporal difference learning to train a neural network to play backgammon. The background of the paper is the field of machine learning, specifically reinforcement learning, and the problem statement is how to train an agent to play backgammon using

TD learning. The method used is a combination of supervised learning and reinforcement learning with a neural network as the function approximator, and the algorithm used is temporal difference learning [80]. The outcome of the paper was a successful demonstration of the TD-Gammon algorithm, which was able to achieve human-level performance in playing backgammon. instead of using temporal difference learning we introduce proximal policy optimization and MA-POCA.

J. Lai et.al present a method for training an AI agent to play a third-person shooter game using Unity's ML-Agents toolkit. The problem statement is how to develop an agent that can navigate the game environment, avoid obstacles, collect items, and engage in combat effectively. The method used involves designing reward functions and using deep reinforcement learning with a neural network architecture [6]. The algorithm used is a variant of deep Q-learning. The outcome of the paper is that the trained agent demonstrates improved performance in the game compared to a baseline model, indicating the effectiveness of the proposed method. Our experimentation introduced the flying leg mechanism along with two different brains that were training concurrent with each other rather than a similar brain with different reward functions pool shop.

T. Singh et.al focuses on the problem of predicting trajectories of moving objects in a 3D virtual environment. The method used involves training a deep reinforcement learning model using Unity's ML-Agents toolkit, which allows for training in a realistic simulated environment [81]. The algorithm used is a deep neural network trained with reinforcement learning. The outcome is a model that can accurately predict trajectories in real-time, making it suitable for applications such as autonomous driving and robotics. We introduce spatial sensors, frontal camera sensors, vectors, and ray casts.

S. Mohamed et.al introduces a novel approach for intrinsically motivated reinforcement learning by maximizing information gain through variational inference. The problem statement is the lack of a general-purpose intrinsic motivation algorithm, which can be used across different tasks without manual tuning. The method utilizes variational autoencoders to learn a representation of the state space and trains the agent to maximize the information gained between the current state and its future representation [82]. The name of the algorithm is Variational Information Maximization (VIM). The outcome of the paper shows that VIM outperforms several state-of-the-art intrinsic motivation algorithms on various tasks. We on the other hand used the curiosity element in the proximal policy optimization for the agents to learn to observe and explore the environment to get such actions that can result in rewards even more.

R. Houthoof et.al present VIME (Variational Information Maximizing Exploration), a method for exploring environments in reinforcement learning. The background is that exploration is crucial for effective learning, but current methods have limitations. The problem statement is that current methods do not effectively balance exploitation and exploration. The method involves using variational inference to approximate the exploration bonus that guides the agent's behavior [83]. The algorithm used is VIME. The outcome of the paper is that VIME outperforms other exploration methods in several benchmarks. Reintroduce the combination of curriculum learning and curiosity, this reducibility to learn the specific environment and then gradually increase the difficulty of the environment while the agent is still curious so that it can observe and re-evaluate the environment to get better action to state results.

Conclusions and Future work

In conclusion, our research has shown that replication of strategies is possible by utilizing a combination of simple game rules, fostering multi-agent competition, and implementing standard reinforcement learning algorithms at scale, and agents can acquire complex strategies and skills. The introduction of curriculum learning has further accelerated the learning process, allowing agents to utilize their environments and props more efficiently. This approach has the potential to significantly reduce costs and improve efficiency in real-world environments, particularly in the field of autonomous systems.

Furthermore, the development of autonomous systems has been a crucial area of research in recent years. The ability to deploy autonomous systems in a range of industries has the potential to improve safety, reduce costs, and increase efficiency. Our research has contributed to this field by providing a framework for the development of intelligent agents capable of learning complex strategies and skills.

Overall, the potential applications of this research are vast, particularly in industries such as transportation, logistics, and agriculture. By utilizing intelligent agents capable of learning complex behaviors, these industries could reduce costs, improve efficiency, and enhance safety. Therefore, our research has the potential to make a significant impact on the development of autonomous systems and has promising implications for a range of industries.

The training of agents can be extended to encompass more complex environments, where additional agent types can be introduced to heighten the level of training difficulty. The incorporation of weather parameters can also enable agents to adapt to challenging weather conditions, thereby enhancing their overall performance. However, it has been observed that agents may exploit the physics of their environment to manifest glitched, cheat-like behaviors. For instance, seekers have been observed to glitch through walls when hit with high velocity, while hiders tend to immobilize themselves in corners of the boundary to avoid incurring a significant collision penalty from seekers. The elimination of such glitched behaviors through the design of an environment that precludes their manifestation can dramatically reduce the time and resources expended in the agent training phase. As such, the creation of an environment that discourages such undesirable behaviors is a crucial aspect of effective agent training.

References

[1] F. E. S. L. a. C. F. A. Xie, "Improvisation through physical understanding: Using novel objects as tools with visual foresight," *arXiv,* 2019.

[2] D. R. A. S. V. B. Y. L. I. B. K. T. D. R. T. L. E. L. M. S. V. L. R. P. M. B. O. V. P. B. Vinicius Zambaldi, "Relational Deep Reinforcement Learning," 2018.

[3] J. C. W. a. C. G. B. W. Wirtz, "artificial intelligence and the public sector—applications and challenges," *Int. J. Publ. Adm,* vol. 42, no. 7, pp. 1-20, 2018.

[4] A. Ahadi, A. Hellas, P. Ihantola, A. Korhonen and A. Petersen, "Replication in computing education research: Researcher attitudes and experiences," *Proceedings of the 16th Koli Calling International Conference on Computing Education Research,* pp. 2-11, 2016.

[5] B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, B. McGrew and I. Mordatch, "Emergent Tool Use From Multi-Agent Autocurricula," *arXiv [cs.LG],* 2019.

[6] R. Miikkulainen and K. O. Stanley, "Competitive Coevolution through Evolutionary Complexification," *arXiv [cs.AI],* 2011.

[7] E. Alonso, M. Peter, D. Goumard and J. Romoff, "Deep Reinforcement Learning for navigation in AAA video," *Proceedings of the Thirtieth International Joint,* 2021.

[8] A. Tucker, A. Gleave, and S. Russell, "Inverse reinforcement learning for video games," 2018.

[9] J. Lai, X.-L. Chen and X.-Z. Zhang, "Training an agent for third-person shooter game using unity," *DEStech trans. Comput. sci. eng.,* no. icaic, 2019.

[10] I. M. A. Nahrendra, C. Tirtawardhana, B. Yu, E. M. Lee and H. Myung, "Retro-RL: Reinforcing nominal controller with deep reinforcement," *IEEE Robot. Autom. Lett.,* vol. 7, no. 4, pp. 9004-9011, 2022.

[11] R. Zhang, Q. Zong, X. Zhang and D. Liqian, "Game of drones: Multi-UAV pursuit-evasion game with online," vol. PP, pp. 1-10, 2022.

[12] A. Devo, J. Mao, G. Costante and G. Loianno, "Autonomous single-image drone exploration with deep reinforcement," *IEEE Robot. Autom. Lett.,* vol. 7, no. 2, pp. 5031-5038, 2022.

[13] G. Wu, M. Fan, J. Shi and Y. Feng, "Reinforcement Learning based Truck-and-Drone Coordinated," *IEEE Trans. Artif. Intell.,* p. 1, 2021.

[14] C. de Souza, R. Newbury, A. Cosgun, P. Castillo, B. Vidolov and D. Kuli, "Decentralized multi-agent pursuit using deep reinforcement," *IEEE Robot. Autom. Lett.,* vol. 6, no. 3, pp. 4552-4559, 2021.

[15] D. Hong, S. Lee, Y. H. Cho, D. Baek, J. Kim and N. Chang, "Energy-efficient online path planning of multiple drones using," *IEEE Trans. Veh. Technol.,* vol. 70, no. 10, pp. 9725-9740, 2021.

[16] "ML-Agents: Hummingbirds," *Unity Learn.*

[17] D. Yang and Y. Tang, "Adaptive inner-reward shaping in sparse reward games," *2020 International Joint Conference on Neural Networks (IJCNN),* pp. 1-8, 2020.

[18] M. Lukas, I. Tomicic and A. Bernik, "Anticheat system based on reinforcement learning agents in Unity," *Information (Basel),* vol. 13, no. 4, p. 173, 2022.

[19] X. Wenwen, "Application Research of end to end behavior decision based on deep reinforcement learning," *Proceedings of the 2021 5th International Conference on Electronic Information Technology and Computer Engineering,* 2021.

[20] R. Dawkins and J. R. Krebs, "Arms races between and within species," *Proc. R. Soc. Lond. B Biol. Sci.,* vol. 205, no. 1161, pp. 489-511, 1979.

[21] B. W. Wirtz, J. C. Weyerer and C. Geyer, "Artificial intelligence and the public sector—applications and challenges," *Int. J. Publ. Adm.,* vol. 42, no. 7, pp. 1-20, 2018.

[22] M. Asplund and D. Näslund, "Artificial intelligence based marine autopilot: Trained using reinforcement learning in the Unity simulation environment," 2022.

[23] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba and P. Abbeel, "Asymmetric actor critic for image-based robot learning," *arXiv [cs.RO],* 2017.

[24] P. Reizinger and M. Szemenyei, "Attention-based curiosity-driven exploration in deep reinforcement learning," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* pp. 3542-3546, 2020.

[25] J. Z. Leibo, E. Hughes, M. Lanctot and T. Graepel, "Autocurricula and the emergence of innovation from social interaction: A manifesto for multi-agent intelligence research," *arXiv [cs.AI],* 2019.

[26] C. Ofria and C. O. Wilke, "Avida: a software platform for research in computational evolutionary biology," *Artif. Life,* vol. 10, no. 2, pp. 191-229, 2004.

[27] A. E. Youssef, S. E. Missiry, I. Nabil El-gaafary, J. S. ElMosalami, K. M. Awad and K. Yasser, "Building your kingdom Imitation Learning for a Custom Gameplay Using Unity ML-agents," *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON),* pp. 509-514, 2019.

[28] J. Paredis, "Coevolutionary computation," *Artif. Life,* vol. 2, no. 4, pp. 355-375, 1995.

[29] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Auton. Agent. Multi. Agent. Syst.,* vol. 11, no. 3, pp. 387-434, 2005.

[30] G. Ostrovski, M. G. Bellemare, A. van den Oord and R. Munos, "Count-based exploration with neural density models," *arXiv [cs.AI],* 2017.

[31] S. Forestier, R. Portelas, Y. Mollard and P.-Y. Oudeyer, "Intrinsically Motivated Goal Exploration Processes with automatic curriculum learning," *arXiv [cs.AI],* 2017.

[32] G. R. Hunt and R. D. Gray, "The crafting of hook tools by wild New Caledonian crows," *Proc. Biol. Sci.,* vol. 271 Suppl 3, no. suppl_3, pp. S88-90, 2004.

[33] M. S. Abdul Hameed, M. M. Khan and A. Schwung, "Curiosity based RL on robot manufacturing cell," *2021 22nd IEEE International Conference on Industrial Technology (ICIT),* vol. 1, pp. 1048-1053, 2021.

[34] D. Pathak, P. Agrawal, A. A. Efros and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," *arXiv [cs.LG],* 2017.

[35] C. Rosser and K. Abed, "Curiosity-driven reinforced learning of undesired actions in autonomous intelligent agents," *2021 IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI),* pp. 39-42, 2021.

[36] A. Price, "Curriculum Learning With Unity ML-Agents," 2020.

[37] G. Zuin, L. Chaimowicz and A. Veloso, "Deep learning techniques for explainable resource scales in collectible card games," *IEEE trans. games,* vol. 14, no. 1, pp. 46-55, 2022.

[38] T. T. Nguyen, N. D. Nguyen and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE Trans. Cybern.,* vol. 50, no. 9, pp. 3826-3839, 2020.

[39] N. D. Dung and J. Rohacs, "The drone-following models in smart cities," *2018 IEEE 59th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON),* pp. 1-6, 2018.

[40] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," *arXiv [cs.AI],* 2017.

[41] N. Heess, D. Tb, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. A. Eslami, M. Riedmiller and D. Silver, "Emergence of locomotion behaviours in rich environments," *arXiv [cs.AI],* 2017.

[42] T. Bansal, J. Pachocki, S. Sidor, I. Sutskever and I. Mordatch, "Emergent Complexity via Multi-Agent Competition," *arXiv [cs.AI],* 2017.

[43] Y. Liu, M. Gordon, J. Wang, M. Bishop, Y. Chen, T. Pfeiffer, C. Twardy and D. Viganola, "Replication markets: Results, lessons, challenges and opportunities in AI replication," *arXiv [cs.CY],* 2020.

[44] M. Alwateer, S. W. Loke and N. Fernando, "Enabling drone services: Drone crowdsourcing and drone scripting," *IEEE Access,* vol. 7, pp. 110035-110049, 2019.

[45] J. Weng, M. Lin, S. Huang, B. Liu, D. Makoviichuk, V. Makoviychuk, Z. Liu, Y. Song, T. Luo, Y. Jiang, Z. Xu and S. Yan, "EnvPool: A highly parallel reinforcement learning environment execution engine," *arXiv [cs.LG],* 2022.

[46] T. S. Ray, "Evolution, ecology and optimization of digital organisms".

[47] S. Leonardos and G. Piliouras, "Exploration-exploitation in multi-agent learning: Catastrophe theory meets game theory," *Artif. Intell.,* vol. 304, no. 103653, p. 103653, 2022.

[48] D. A. Suyikno and A. Setiawan, "Feasible NPC hiding behaviour using goal oriented action planning in case of hide-and-seek 3D game simulation," *2019 Fourth International Conference on Informatics and Computing (ICIC),* pp. 1-6, 2019.

[49] M. Jaderberg, V. Dalibard, S. Osindero, W. M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan, C. Fernando and K. Kavukcuoglu, "Population Based Training of neural networks," *arXiv [cs.LG],* 2017.

[50] A. T. Bourdillon, A. Garg, H. Wang, Y. J. Woo, M. Pavone and J. Boyd, "Integration of reinforcement learning in a virtual robotic surgical simulation," *Surg. Innov.,* p. 15533506221095298, 2022.

[51] S. Sukhbaatar, Z. Lin, I. Kostrikov, G. Synnaeve, A. Szlam and R. Fergus, "Intrinsic motivation and automatic curricula via asymmetric self-play," *arXiv [cs.LG],* 2017.

[52] J. N. Foerster, Y. M. Assael, N. de Freitas and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," *arXiv [cs.AI],* 2016.

[53] S. Singh, A. G. Barto and N. Chentanez, "Intrinsically Motivated Reinforcement Learning," *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada],* 2004.

[54] Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell and A. A. Efros, "Large-scale study of curiosity-driven learning," *arXiv [cs.LG],* 2018.

[55] J. L. Ba, J. R. Kiros and G. E. Hinton, "Layer Normalization," *arXiv [stat.ML],* 2016.

[56] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *arXiv [cs.LG],* 2017.

[57] OpenAi, M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng and W. Zaremba, "Learning Dexterous In-Hand Manipulation," *arXiv [cs.LG],* 2018.

[58] S. Sukhbaatar, A. Szlam and R. Fergus, "Learning Multiagent Communication with Backpropagation," *arXiv [cs.LG],* 2016.

[59] N. Haber, D. Mrowca, L. Fei-Fei and D. L. K. Yamins, "Learning to play with intrinsically-motivated self-aware agents," *arXiv [cs.LG],* 2018.

[60] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.,* vol. 9, no. 8, pp. 1735-1780, 1997.

[61] J. Z. Leibo, J. Perolat, E. Hughes, S. Wheelwright, A. H. Marblestone, E. Duéñez-Guzmán, P. Sunehag, I. Dunning and T. Graepel, "Malthusian reinforcement learning," *arXiv [cs.NE],* 2018.

[62] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *arXiv [cs.LG],* 2017.

[63] J. Perolat, J. Z. Leibo, V. Zambaldi, C. Beattie, K. Tuyls and T. Graepel, "A multi-agent reinforcement learning model of common-pool resource appropriation," *arXiv [cs.MA],* 2017.

[64] L. Buşoniu, R. Babuška and B. De Schutter, "Multi-agent Reinforcement Learning: An Overview," *Innovations in Multi-Agent Systems and Applications - 1,* pp. 183-221, 2010.

[65] H. Wafa and J. Santoso, "Multiagent simulation on hide and seek games using policy gradient trust region policy optimization," *2020 7th International Conference on Advance Informatics: Concepts, Theory and Applications (ICAICTA),* pp. 1-5, 2020.

[66] D. D. Ningombam, "A novel exploration technique for multi-agent reinforcement learning," *2022 IEEE 3rd Global Conference for Advancement in Technology (GCAT),* pp. 1-6, 2022.

[67] C. Diuk, A. Cohen and M. L. Littman, "An object-oriented representation for efficient reinforcement learning," *Proceedings of the 25th international conference on Machine learning - ICML '08,* pp. 240-247, 2008.

[68] C. Jestel, H. Surmann, J. Stenzel, O. Urbann and M. Brehler, "Obtaining robust control and navigation policies for multi-robot navigation via deep reinforcement learning," *arXiv [cs.RO],* 2022.

[69] A. Cohen, E. Teng, V.-P. Berges, R.-P. Dong, H. Henry, M. Mattar, A. Zook and S. Ganguly, "On the use and misuse of absorbing states in multi-agent reinforcement learning," *arXiv [cs.LG],* 2021.

[70] Y. Duan, M. Andrychowicz, B. C. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel and W. Zaremba, "One-shot imitation learning," *arXiv [cs.AI],* 2017.

[71] J. Ren, S. Guo and F. Chen, "Orientation-preserving rewards' balancing in reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.,* vol. 33, no. 11, pp. 6458-6472, 2022.

[72] R. Wang, J. Lehman, J. Clune and K. O. Stanley, "Paired Open-Ended Trailblazer (POET): Endlessly generating increasingly complex and diverse learning environments and their solutions," *arXiv [cs.NE],* 2019.

[73] N. Matsumura, K. Nishimori, R. Taniguchi, T. Mitsui and T. Hiraguri, "Performance improvement of drone MIMO relay station using selection of drone placement," *2018 IEEE International Workshop on Electromagnetics:Applications and Student Innovation Competition (iWEM),* p. 1, 2018.

[74] M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castañeda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, N. Sonnerat, T. Green, L. Deason, J. Z. Leibo, D. Silver, D. Hassabis, K. Kavukcuoglu and T. Graepel, "Human-level performance in 3D multiplayer games with population-based reinforcement learning," *Science,* vol. 364, no. 6443, pp. 859-865, 2019.

[75] C. Florensa, D. Held, M. Wulfmeier, M. Zhang and P. Abbeel, "Reverse curriculum generation for reinforcement learning," *arXiv [cs.AI],* 2017.

[76] M. Seo, L. F. Vecchietti, S. Lee and D. Har, "Rewards prediction-based credit assignment for reinforcement learning with sparse binary rewards," *IEEE Access,* vol. 7, pp. 118776-118791, 2019.

[77] M. M. M. van Dooren, V. T. Visch and R. Spijkerman, "Rewards that make you play: The Distinct effect of monetary rewards, virtual points and social rewards on play persistence in substance dependent and non-dependent adolescents," *2018 IEEE 6th International Conference on Serious Games and Applications for Health (SeGAH),* pp. 1-7, 2018.

[78] N. Jaques, A. Lazaridou, E. Hughes, C. Gulcehre, P. A. Ortega, D. J. Strouse, J. Z. Leibo and N. de Freitas, "Social influence as intrinsic motivation for Multi-agent deep reinforcement learning," *arXiv [cs.LG],* 2018.

[79] O. E. Gundersen and S. Kjensmo, "State of the art: Reproducibility in artificial intelligence," *Proc. Conf. AAAI Artif. Intell.,* vol. 32, no. 1, 2018.

[80] V. Bapst, A. Sanchez-Gonzalez, C. Doersch, K. L. Stachenfeld, P. Kohli, P. W. Battaglia and J. B. Hamrick, "Structured agents for physical construction," *arXiv [cs.LG],* 2019.

[81] J. Achiam and S. Sastry, "Surprise-based intrinsic motivation for deep reinforcement learning," *arXiv [cs.LG],* 2017.

[82] J. Lehman, J. Clune, D. Misevic, C. Adami, L. Altenberg, J. Beaulieu, P. J. Bentley, S. Bernard, G. Beslon, D. M. Bryson, P. Chrabaszcz, N. Cheney, A. Cully, S. Doncieux, F. C. Dyer, K. O. Ellefsen, R. Feldt, S. Fischer, S. Forrest, A. Frénoy, C. Gagné, L. L. Goff, L. M. Grabowski, B. Hodjat, F. Hutter, L. Keller, C. Knibbe, P. Krcah, R. E. Lenski, H. Lipson, R. MacCurdy, C. Maestre, R. Miikkulainen, S. Mitri, D. E. Moriarty, J.-B. Mouret, A. Nguyen, C. Ofria, M. Parizeau, D. Parsons, R. T. Pennock, W. F. Punch, T. S. Ray, M. Schoenauer, E. Shulte, K. Sims, K. O. Stanley, F. Taddei, D. Tarapore, S. Thibault, W. Weimer, R. Watson and J. Yosinski, "The surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities," *arXiv [cs.NE],* 2018.

[83] G. Tesauro, "Temporal difference learning and TD-Gammon," *Commun. ACM,* vol. 38, no. 3, pp. 58-68, 1995.

[84] T. Singh, "Trajectory prediction through deep reinforcement learning in unity," 2021.

[85] S. Mohamed and D. J. Rezende, "Variational information maximisation for intrinsically motivated reinforcement learning," *arXiv [stat.ML],* 2015.

[86] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck and P. Abbeel, "VIME: Variational Information Maximizing Exploration," *arXiv [cs.LG],* 2016.

[87] A. Kelly, "AI Flight with Unity ML-Agents Course —," 2020.

[88] "Concrete Problems in AI Safety".

[89] N. Justesen, P. Bontrager, J. Togelius and S. Risi, "Deep learning for video game playing," *IEEE trans. games,* vol. 12, no. 1, pp. 1-20, 2020.

[90] "grid-sensor: Grid Sensor Components for Unity ML-Agents".

[91] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli and S. Whiteson, "Counterfactual Multi-Agent Policy Gradients," *arXiv [cs.AI],* 2017.

[92] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup and D. Meger, "Deep Reinforcement Learning that Matters," *arXiv [cs.LG],* 2017.

[93] P. Xu, Q. Yin, J. Zhang and K. Huang, "Deep reinforcement learning with part-aware exploration bonus in video games," *IEEE trans. games,* p. 1, 2021.

[94] S. Liu, G. Lever, J. Merel, S. Tunyasuvunakool, N. Heess and T. Graepel, "Emergent coordination through competition," *arXiv [cs.AI],* 2019.

[95] C. D. Rosin and R. K. Belew, "Methods for Competitive Co-Evolution: Finding Opponents Worth Beating," *Proceedings of the 6th International Conference on Genetic Algorithms,* pp. 373-381, 1995.

[96] J. Z. Leibo, V. Zambaldi, M. Lanctot, J. Marecki and T. Graepel, "Multi-agent reinforcement learning in sequential social dilemmas," *arXiv [cs.MA],* 2017.

[97] "Perpetuating evolutionary emergence".

[98] R. Herbrich, T. Minka and T. Graepel, "TrueSkill™: A Bayesian Skill Rating System," *Advances in Neural Information Processing Systems,* vol. 19, 2006.

[99] M. G. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton and R. Munos, "Unifying count-based exploration and intrinsic motivation," *arXiv [cs.AI],* 2016.

[100] C. L. Keefer, "Artificial cloning of domestic animals," *Proc. Natl. Acad. Sci. U. S. A.,,* vol. 112, p. 8874–8878, 2015.