Dynamic Collaborative Multi-Agent Reinforcement Learning Communication for Autonomous Drone Reforestation

Anonymous Author(s)
Affiliation
Address
email

Abstract

We approach autonomous drone-based reforestation with a collaborative multi-agent reinforcement learning (MARL) setup. Agents can communicate as part of a dynamically changing network. We explore collaboration and communication on the back of a high-impact problem. Forests are the main resource to control rising CO2 conditions. Unfortunately, the global forest volume is decreasing at an unprecedented rate. Many areas are too large and hard to traverse to plant new trees. To efficiently cover as much area as possible, here we propose a Graph Neural Network (GNN) based communication mechanism that enables collaboration. Agents can share location information on areas needing reforestation, which increases viewed area and planted tree count. We compare our proposed communication mechanism with a multi-agent baseline without the ability to communicate. Results show how communication enables collaboration and increases collective performance, planting precision and the risk-taking propensity of individual agents.

Keywords: Multi-Agent Reinforcement Learning; Graph Neural Network; Collaboration; Communication; Proximal Policy Optimization; Reforestation;

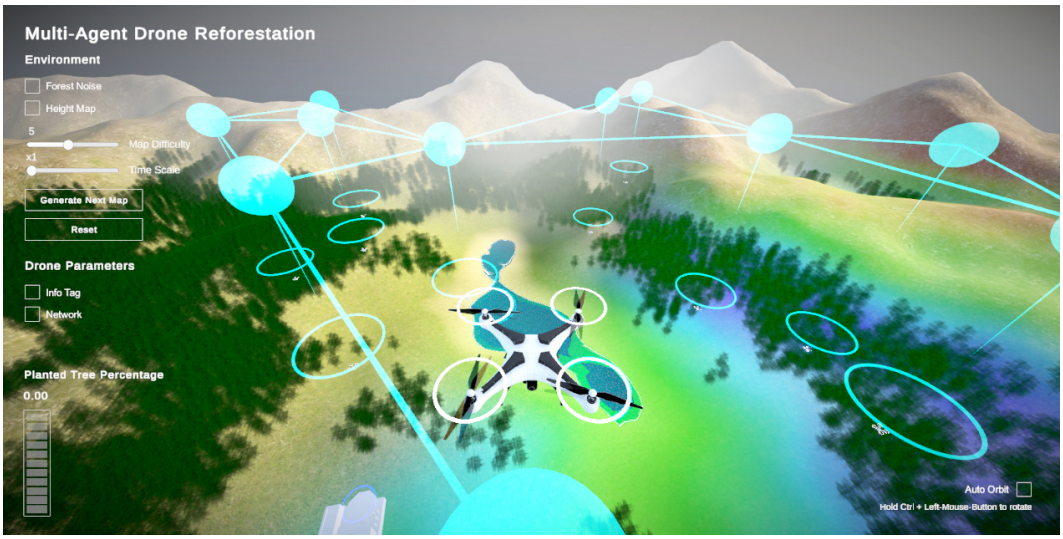


Figure 1: Multi-agent drone reforestation environment with heatmap overlay indicating ideal areas for planting trees (green better, red worse). A web application can be found at: anonymized.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

1 Introduction

1.1 Motivation

The success of our society is based on the human ability to communicate motives. Once intentions overlap amongst concerned parties, they may decide to join forces and collaborate to work towards a common goal that is beneficial to all and might not have been achieved by an individual. A group of people or individuals with specific interests can be seen as an agent, an entity with an agency. Most domains in the real world require collaboration to achieve higher goals and can be described as multi-agent (MA) systems. Human intelligence is the highest observed, but we can draw further inspiration from nature. Species like bees only survive because of their ability to communicate and collaborate effectively. If a beehive successfully fends off predators, assigns tasks to collect nectar from flowers, and building materials, build and repair the hive, their survival is ensured [1].
The last two years have brought climate change to the doorstep of many. Extreme heatwaves, wildfires and floods make life for animals and humans increasingly difficult. While vivid wildfires have destroyed large forest areas [2, 3], cattle pasture is the biggest cause of deforestation [4]. Forests absorb more than a quarter of global human-emitted carbon dioxide [5, 6]. Droughts cause soil erosions and turn pasture and woodland into steppe. The Sahara desert advances irreversibly southward, causing livestock losses, human migration and mortality [7]. The Maradi and Zinder Regions have been regreening five million hectares to prevent further decline of the biosphere and habitability of the Niger [8]. Multiple parameters can make reforestation difficult and must be considered to regreen successfully and sustainably. Trees need to be planted close to existing woodland to shield seedlings from harsh winds and provide them with stable soil. Furthermore, a mixture of native tree species needs to be planted rather than one kind. The additional maintenance, monitoring and transportation challenges, especially in hard-to-reach areas [9] have motivated this work. A wide variety of unresembling landscape scenarios makes this especially hard and requires coordination between multiple actors and information sources.
Core questions we ask: Can agents in a MA system learn the importance of communication and subsequently extend each other's partial observability? Furthermore, can agents learn to use the communicated information to increase individual performance? Can shared information be used to explore further and take actions with higher risks of depleting the battery for the benefit of finding locations with a higher need for reforestation and accordingly higher rewards? Finally, can agents organise themselves and delegate tasks, such that one group of agents become scouts and information collectors to extend the collectives' observable space and another group that carries tree seeds and plants them accordingly in areas they would not have found by themselves? Trying to answer these questions was the challenge of the experiments conducted and presented in this paper.

1.2 Contribution

In this work, we study communication in the context of a highly distributed, dynamic collective of autonomous drones and propose an approach to solve reforestation in hard-to-reach areas. Illegal logging has advanced to untouched areas of the Amazonas [10]. We propose to use autonomous drones to reach areas without infrastructure and roads, plant tree seeds and help monitor reforestation effectively. Controlling a single or collective of drones to fly hardcoded manoeuvres [11] and in formation [12] is a solved problem. However, we are interested in solving the reforestation problem with a decentralised autonomous system consisting of individual agents capable of making ad-hoc decisions in various scenarios. In this work, we utilise Proximal Policy Optimisation (PPO) [13], a state-of-the-art Reinforcement Learning algorithm to train a collective of multiple agents. The task for the MA collective is to pick up tree seeds, search for spots along the perimeter of an existing forest that benefit the most from reforestation and plant the carried tree seeds. Furthermore, the proposed agent collective can communicate to enable collaboration. Agents can capture spots they found in memory while exploring the environment and share the information with neighbouring agents when in reach. Each agent controls a drone and has to execute the planting tasks and search for new spots, but also has to monitor the battery charge status, especially with the additional payload when carying a tree seed. We have developed a simulation environment to train and test various learning mechanisms. Our environment is additionally able to cater for open-ended learning [14] through an infinite procedural generated set of scenarios with possibly increasing difficulty of the terrain topology and forest sparsity. Our contribution is two-fold; we mainly want to demonstrate that a communication mechanism for a MARL system in a partially observable environment can

lead to collaboration and increased collective performance amongst highly dynamic agents with a proximity-based communication network. Furthermore, as a secondary contribution, we want to do so by utilising an environment with a minimal gap to a high-impact real-world problem, namely autonomous drone-based reforestation in hard-to-reach areas [15].

We have designed a task to verify our solution. The task is constrained by a given time frame. Individual agents of the collective spawn at the drone station spawn point. Their initial state is a full battery charge and a loaded tree seed. They then have to search for a spot in urgent need of reforestation to collect a high reward. After agents have dropped their tree seed, they must return to the drone station before depleting their battery. This end-to-end task can be repeated until the episode ends. A MA setup with and without the ability to communicate is trained on a single scenario as well as on multiple scenarios, respectively. The setup without communication ability serves as the baseline. The benefit of communication can be verified by the count of trees planted, but more importantly, the quality and precision of the positioning of the planted tree seed regarding the existing forest volume. Finally, we demonstrate how well agents trained in multiple scenarios perform in an unseen scenario compared to those trained in a single scenario.

1.3 Future Work

The partial observability hinders the agents from keeping track of the big picture. For future work, we would be interested in studying how much of the overall forest in a scenario is explored and if we can observe that agents understand what areas of the forest need reforestation and find a tree seed distribution balance. Once an overall understanding is achieved, agents should be able to weight areas rather than planting tree seeds at the next available good spot greedily. This would require to increase the long-term thinking of agents. There are ideas to have rounds of exploration followed by regrouping and sharing of gathered information before tree seeds are carried and planted according to a plan that has been agreed on collaboratively. Furthermore, studying a larger information memory would be an exciting experiment strand. Questions we might ask are: What if agents can tell each other where a good planting spot is and where enough trees have been planted already, or if no good spots are to be found anymore? Finally, incentivise agents to allocate different tasks to each other, i.e. scout drones that can fly quicker with less energy that gather information, with higher flexibility to explore, and planter drones that pick up, carry and plant tree seeds accordingly.

2 Related Work

Single-Agent Reinforcement Learning (SARL) is a learning paradigm based on an agent taking actions in an environment to maximize cumulative reward. In contrast to other machine learning methods, RL does not require a dataset and learns from collected experiences. Nevertheless, just like a well-curated dataset is crucial for successful supervised learning, the experiences collected by an RL agent heavily depend on the environment design [14]. There is broad interest for RL, from industry [16], but also academia and research. Exemplary, industry applications include optimal control, autonomous vehicles [17], robots [18, 19, 20], cooling management [21], but also social dilemmas [22], economics and finance [23]. Many, if not all, real-world domains and according applications include multiple agents or at least some form of an entity with an agency, active or passive parts of the environment. However, experimenting in the real world can lead to high resource consumption and time spent. After all, an RL system does not only learn from succeeding in a given environment, but a failing experience can be crucial for a robust learning update. This constraint brings us to the need for simulation frameworks, where we can break things and even manipulate time. Furthermore, single- and multi-player games have been popular test-beds for algorithm and learning mechanism development. Board games like GO [24, 25], Chess [26], Shogi [27], Hex [28], Poker [29, 30] and Diplomacy [31, 32], but also computer games i.e. Atari [33], Dota [34], Starcraft [35] and overcooked [36] have shaped the RL field significantly. The combination of simulation and the game domain naturally leads us to game engines, including realistic physics [37], which we will be using for our work.

Collaboration [38, 39, 40] in MA systems can be achieved in various ways and have a rich history of literature [41]. Collaboration does not need active communication [42, 43] and can be achieved through methods such as gradient-based distributed policy search [44], reward function sharing [45], memory sharing [46, 47, 48] and parameter sharing (PS) [19, 48]. Nevertheless, our work focuses on active communication as part of the agents' action space. [49]. The messages each

agent can communicate consist of a three-dimensional vector and represents a location in euclidean space [50]. The agent can decide to save a location as a three-dimensional vector that is worth memorizing and send it to the three nearest neighbouring agents in reach. In contrast, other work proposes communicating more complex information such as intentions or policy gradients [51]. Our communication layer is based on Graph Neural Network (GNN) [52] message passing [53], as part of the agents observation space. This enables the agent to deal with various sized graph-structured data. Recent work of ours has investigated highly decentralized multi-agent communication, with fixed size graph-structured data [54], as well as variable size graph-structured data [55], with communicating as part of the action space. While there has been work on static networks [56], in contrast to our previous work, here we advance static- and investigate dynamically-changing graph-structured data with a maximum size.

There have been multiple single and multi-drone applications in various fields. Previous work has investigated the use of drone networks [57] to fight forest fires by monitoring outbreaks and growth [58]. Furthermore there is work on last-mile delivery based on single autonomous drones [59, 60, 61]. We think it is important to mention two companies, namely Flash Forest [62] and Airseed [63], using Artificial Intelligence (AI) to solve reforestation with drones. Both companies, but also academic work [64] heavily use mapping, pre-planning of flight paths and tree seed drop locations. However, our proposal investigates fully autonomous flight-path planning, drop location scouting and communication amongst all agents, re-charging and battery-life tracking using a single neural network for each drone. Furthermore, we are not trying to replace the proportional integral derivative (PID) controller. The PID is commonly used to make the drone pitch, yaw and roll [65]. While there is work using a neural network to replace the PID and fully control the drone's movement [66], this is out of scope for this work. Instead, we are using a simplified set of commands, actuating the drone to move.

3 Background

3.1 Proximal Policy Optimisation (PPO-Clip)

Proximity Policy Optimisation (PPO), a state-of-the-art, on-policy RL algorithm, has been utilised for training the agents in this work. Our environment's action space consists of discrete and continuous actions, which PPO supports. Two main concepts define the PPO algorithm: 1. PPO performs the largest possible but safe gradient ascent learning step by estimating a trust region, and 2. Advantage estimates how good an action in a specific state is, compared to the average action. Various other RL algorithms, such as Asynchronous Advantage Actor Critic (A3C), use this concept [67]. **Advantage:** Advantage can also be described as the difference of the Q Function and the Value Function: $A(s,a) = Q(s,a) - V(s)$, where s is the state and a the action [68]. The Q Value (Q Function), denoted as $Q(s,a)$, measures the overall expected reward given state s, performing action a. Assuming the agent continues playing until the end of the episode following policy π. The Q is abbreviated from the word Quality, and denoted as: $\mathcal{Q}(s,a) = \mathbb{E}\left[\sum_{n=0}^{N} \gamma^n r_n\right]$. The State Value Function, denoted as $V(s)$, measures, similar to the Q Function, overall expected reward, with the difference that the State Value is calculated after the action has been taken and is denoted as: $\mathcal{V}(s) = \mathbb{E}\left[\sum_{n=0}^{N} \gamma^n r_n\right]$. The Q Value $V(s)$, with $n = 0$, is the expected reward r^0 in state s, before action a was taken, while the Q Value measures the expected reward r^0 after a was taken. **Trust Region:** After some experiences $\pi_{\theta_k}(a_t|s_t)$ have been collected, the trust region can be calculated as the quotient of the current policy to be refined $\pi_\theta(a_t|s_t)$ and the previous policy as follows $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} = \frac{current\ policy}{old\ policy}$. This is a simplified gradient ascent objective function with limited deviation between the current and old policies [69].

$$\mathcal{L}_{\theta_k}^{CLIP}(\theta) = \underset{s,a\sim\theta_k}{\mathbb{E}}\left[\min\left(r_t(\theta)A^{\theta_k}(s,a), g(\epsilon, A^{\theta_k}(s,a))\right)\right],$$

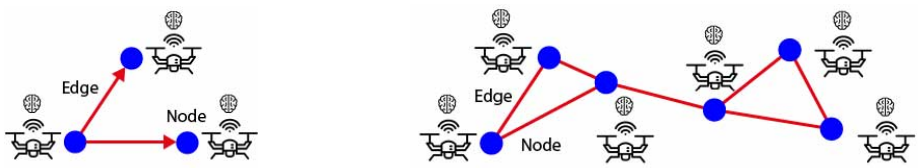
where

$$g(\epsilon, A) = \begin{cases}(1+\epsilon)A, & \text{if } A \geq 0\\ (1-\epsilon)A, & \text{otherwise}\end{cases}$$

The advantage function will be clipped if the probability ratio between the current and the previous policy is outside the range of $(1+\epsilon)$ and $(1-\epsilon)$. This also means that the advantage will never exceed the clipped values and prevents the new policy from getting too far from the old policy. In the original PPO paper by Schulman, et al. (2017) [13] ϵ was set to 0.2. Lastly, the policy that yields the highest sum over all Advantage estimates A_t in range of max time step T of a trajectory $\tau \in \mathbb{D}_k$ will be used to override the old policy θ_{old} [70]: $\theta_{k+1} = \underset{\theta_k}{argmax} \frac{1}{|\mathbb{D}_k|T} \sum_{\tau \in \mathbb{D}_k} \sum_{t=0}^{T} \min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}, g(\epsilon, A^{\theta_k}(s,a))\right)$.

3.2 Graph Neural Network

Scarselli, et al. (2009), developed fundamental work on Graph Neural Networks, leading to many variations, such as Gated Graph Sequence Neural Networks [71], Graph Attention Networks [72] and Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering [73]. A graph is a data structure based on nodes or vertices and edges. The node and edge objects can hold an arbitrary amount of features of any type. An edge represents the relationship between two nodes, but a node can have unlimited relationship edges with other nodes. Edges can be directed from node A to B 2a, or undirected, from node A to B and vice versa 2b.



(a) Directed Graph (b) Undirected Graph

Figure 2: Graph $\mathcal{G}$ consisting of vertices $\mathcal{V}$ (blue dots) and edges $\mathcal{E}$ (red lines): $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

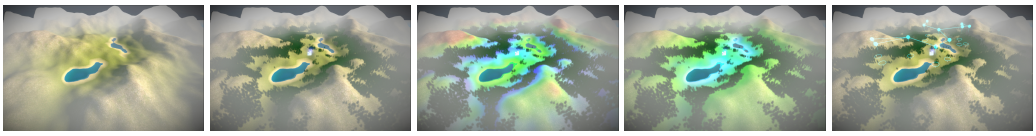
Fundamental functionalities of GNNs are graph, node and edge classification. Features and the existence of nodes and edges can be predicted using, i.e. neighbouring nodes and existing edges. Classification of a graph as a whole can be achieved using node and edge features and the graph's topology as input. However, one of the simplest forms of a GNN is the message passing framework proposed by Gilmer, et al. [53], using the "graph-in, graph-out" network architecture introduced by [74]. Hence the graph topology is not modified but loaded node and edge feature embeddings.

Node states can be denoted as v, edges connected to node v as $x_{co[v]}$. The state of a node h_v may consist of a n-dimensional vector feature. Adjacencies between a node and its neighbours are the mapped transition of the node, denoted as $h_{ne[v]}$, including all neighbouring node features, denoted as $x_{ne[v]}$. The transition function f is used to embed each node on a n-dimensional space [75]: $h_v = f(x_v, x_{co[v]}, h_{ne[v],x_{ne[v]}})$
Breadth-First Search (BFS) [76], Depth-First Search (DFS) [77] and random walk based DeepWalk [78] are popular algorithms to define neighbourhoods in graph structured data, however we define neighbourhoods using n nearest neighbours based on shortest euclidean distances. Furthermore, passing state h_v and feature x_v of nodes and edges, to the GNN outputs the result of function g: $o_v = g(h_v, x_v)$. And finally, applying gradient descent to formulate loss using the ground truth t_v as well as the output o_v of node v: $loss = \sum_{i=1}^{p}(t_i - o_i)$.

4 Method

4.1 Drone Reforestation Environment

We now describe the 3-D Reforestation environment, developed in the game engine unity, used for simulation and online training of the Multi-Agent (MA) baseline and our Multi-Agent communication setup (MAC). The environment is considered solved if an agent picks up a tree seed from the drone station, finds a spot ideal for planting a tree seed, drops the tree seed and returns to the drone station to recharge the battery and pick up the next tree seed. All variations of environment scenarios consist of the following elements: 1. Procedural generated terrain, using octaves, persistence and



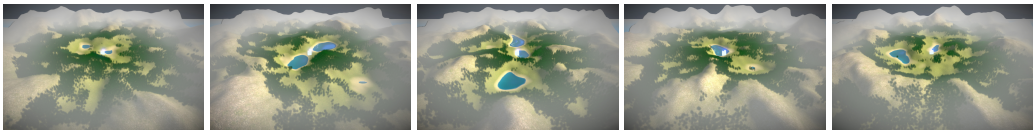
(a) Terrain (b) Forest (c) O. Reforestation (d) Height Map (e) Com. Network

Figure 3: Static environment features. Zoomed-in version in the appendix: D.

lacunarity-based noise 3a, 2. The forest trees, placed in a certain height region, combined with a random noise map, on fertile ground only, here displayed as green grass 3b, 3. The human user interface allows to display the regions that are best for reforestation and yield the highest amount of reward; however, this is not visible to the agent's 3c, 4. A height heat map, also only visible in the human user interface 3d and 5. A network that defines the nearest neighbours dynamically displayed in cyan, visible in the human user interface only 3e. Part of the environment, acting as spawn, tree seed- and battery-charging point, is a drone station marked with (9) on Figure 6. Ten drones controlled by individual agents are spawning at the drone station spawn point in all experiments. The terrain in all scenarios spans 1200 meters in both directions and has a maximum mountain altitude of 100 meters, depending on the difficulty level. Various terrain samples of difficulty levels 1-5 are shown in Figure 4. Furthermore, an additional bowl-shaped height filter adjusts the terrain to be close to a valley when difficulty levels increase. We can generate open-ended scenarios with a random seed input - sample terrains of various random seeds are shown in Figure 5.

(a) Difficulty 1 (b) Difficulty 2 (c) Difficulty 3 (d) Difficulty 4 (e) Difficulty 5

Figure 4: Various difficulty terrain samples. Zoomed-in version in the appendix: F.



(a) Seed 25 (b) Seed 26 (c) Seed 27 (d) Seed 28 (e) Seed 31

Figure 5: Various random seed terrain scenarios. Zoomed-in version in the appendix: E.

4.2 Agent Setup

Goal: Each agent must learn to navigate the drone to the drone station spawn point, where it is automatically serviced, picks up a tree seed and recharges the battery. It then has to fly the drone and scout an ideal spot for dropping the tree seed held while keeping track of battery life, which also needs to cover the way back to the drone station.

Reward Function: The agent reward function consists of multiple parts. If the drone does not hold a tree seed, the agent must navigate back to the drone station. Incremental rewards are yielded for getting closer to the drone station, accumulating to a total of +20, independent of the distance. The agent receives the last increment for arriving at the drone station, picking up a tree seed and recharging the battery. Finally, dropping a tree seed yields a reward of +0 to +30, depending on the location. The ideal tree seed drop area is defined by the distance to existing trees. The reward of +0 to +30 is mapped to a distance as close as 75 to 2.5 meters. I.e., if a tree seed is dropped as close as 2.5 meters from an existing tree, the reward yielded is +30, for a tree seed drop distance of 10 meters, the reward is +26.8, and a tree seed dropped at a distance below 2.5 or above 75 meters yields a reward of +0. The total reward of +30 includes a distance bonus of +10. If the agent is risking running out of battery and finds an excellent spot for a tree seed drop, the distance between the tree seed drop location and the drone station spawn point is factored into the total reward. The battery life ranges

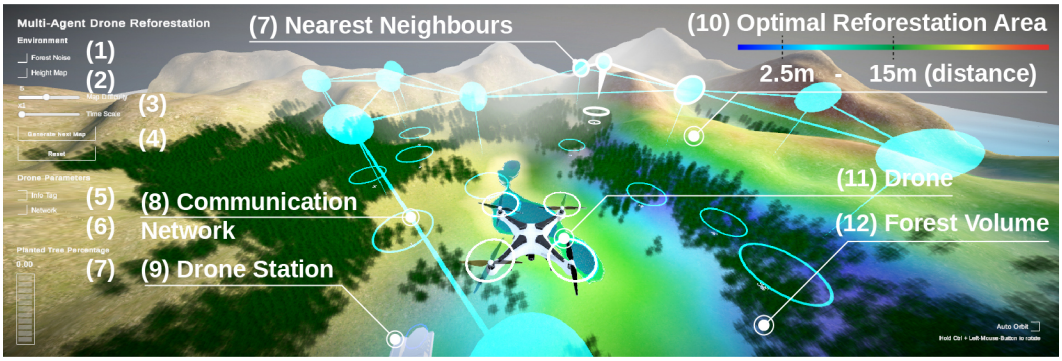


Figure 6: Drone Reforestation Environment: (1) Forest Noise Toggle, (2) Height Map Toggle, (3) Terrain Difficulty Slider, (4) Time Scale Slider, (5) Info Tag Toggle, (6) Network Toggle, (7) Cumulative Performance Bar, (8) Communication Network, (9) Drone Station and Spawn Point, (10) Optimal Reforestation Area Heat Map, (11) Drone, (12) Forest Volume.

from a full charge of 1 and empty at 0. For each step, while holding a tree seed - a higher payload - the battery is being depleted at a rate of -0.001, holding no tree seed at a rate of -0.0005, and yields negative rewards accordingly. This results in 1000 environment steps with a tree seed and 2000 steps without a tree seed until running out of battery life. The gold standard receives a maximum reward of +50 for a single task, excluding negative payload rewards.

Observations: The observation space consist of vector and visual observations.

Vector Observations: All vector observations are normalized and consists of the following: The distance from the drone to the ground as float [0-100], the location of the drone in 3-D space as a vector(x, y, z) [-600-600], the movement direction vector(x, y, z) [0-1], the vector from the drone to the drone station spawn point as a vector(x, y, z) [-600-600], if the drone is holding a tree seed as a bool [true / false] mapped to [0, 1], the battery status as a float [0-1] and lastly three inbox spaces to receive messages from the neighbouring drone's memory, which consist of vector(x, y, z) [-600-600] location information. The final vector observation space size is 30, consisting of 2 stacks of the 15 described observations.

Visual Observations: The visual observation is a grey scale grid of 16x16, cells captured by a down-ward pointing camera attached to each drone with a field of view of 120 - 256 cells each with a float value ranging from [0-1]. This results in a total observation space size of 286. We use a residual neural network (ResNet) architecture consisting of three stacked layers, each with two residual blocks [79] for our visual observations processing.

Actions: The action space consists of a combination of continuous and discrete actions.

Continuous Actions: Each agent has three continuous actions with values in the range of -1 to 1. Continuous actions control the movement of the drone. Continuous action 0 controls the forward and backward movement, action 1 controls the rotation, left and right, and action 2 controls the up and down movement of the drone. The movement speed of the drone is 1-meter per time-step, and the rotation speed is 5-degree per time-step.

Discrete Actions: The discrete action space size is two and can be described as a tree with two branches, each with two possible values [0, 1]. Discrete action 0 drops the tree seed at value 1, and action 1 saves a location to memory at value 1. The location memory can hold a three-dimensional vector representing a location in the environment. Both actions do nothing if the value is at 0.

4.3 Multi-Agent Communication

Our learning mechanism allows the agent to receive graph-structured data. Messages can be exchanged if a drone is within 200 meters of another drone. A total number of three messages can be received, corresponding to three nearest neighbours in the Multi-Agent communication setup (MAC). If there are only two drones close enough, only two messages are received and sent respectively. Sending or receiving a message has no cost and yields no negative or positive reward.

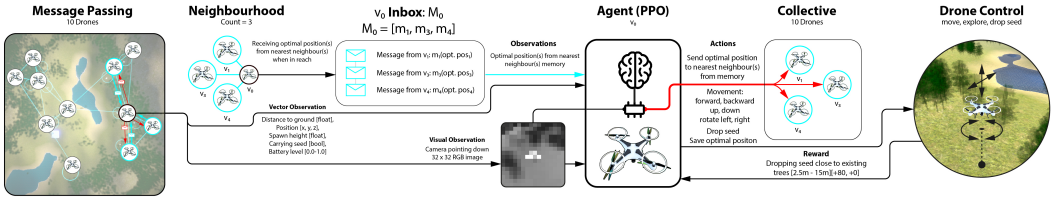


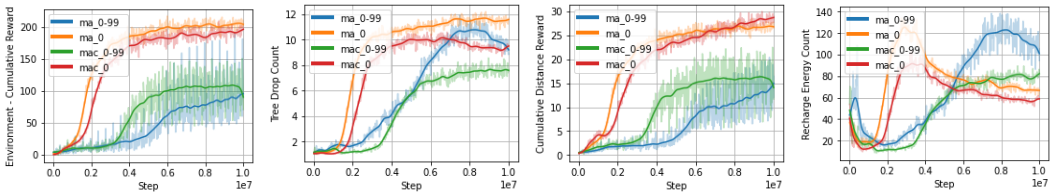
Figure 7: GNN Message Passing Diagram: Message Passing across neighbourhood; Inbox of the individual Drone; Locations for optimal tree seed drop, as part of the RL Agent observation space.

5 Experiments

Table 1: Experiment Setup

Experiment	Agent(s)	Neighbour(s)	Training Seed	Test Seed
1 Multi-Agent (MA)	10	0	0	111
2 Multi-Agent (MA)	10	0	0-99	111
3 Multi-Agent Communication (MAC)	10	3	0	111
4 Multi-Agent Communication (MAC)	10	3	0-99	111

We have trained four different setups for our experiments (Figure 8). **Multi-Agent Setup without Communication as Baseline:** Experiment 1 and 2 have been trained without the ability to communicate: Experiment 1 is trained on the terrain scenario with the random seed 0 and Experiment 2 on the terrain scenarios with random seeds ranging from 0 to 99 in a sequence. **Multi-Agent Setup with Communication:** Experiment 3 and 4 have been trained with the ability to communicate: Experiment 3 is trained on the terrain scenario with the random seed 0 and Experiment 4 on the terrain scenarios with random seeds ranging from 0 to 99 in a sequence. All experiments are then tested on an unseen terrain scenario with the random seed 111.



(a) Cumulative Reward (b) Tree Drop Count (c) Cml. Distance Reward (d) Recharge Energy Count

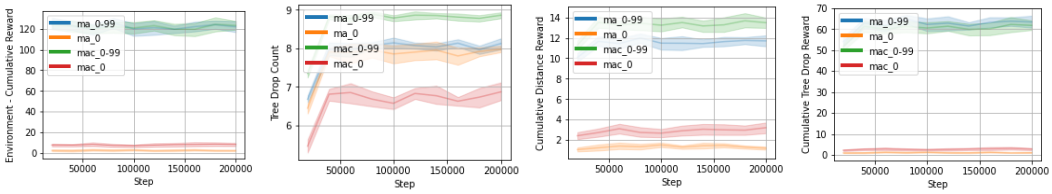
Figure 8: Training Graphs. Zoomed-in version and additional data in the appendix: H.

6 Results and Discussion

Our results (Figure 2) show that Multi-Agent communication outperforms our non-communication Multi-Agent baseline setup. The MA 0 setup that has been trained on the terrain scenario with a random seed 0 performs very poorly on the test terrain scenario with the random seed 111. While the tree drop count is high, the agent drops tree seeds very conservatively without exploring and therefore receives a low cumulative reward. The Multi-Agent setup with the ability to communicate that has also been trained on the terrain scenario with the random seed 0, MAC 0, performs similarly bad, but through communication starts to explore marginally more and therefore receives a higher distance reward. Nevertheless, the cumulative reward is low as well. In contrast, the MA 0-99 setup, that has been trained on terrain scenario with a random seed ranging from 0-99, performs marginally the best in regards to the cumulative reward. Agents trained on multiple scenarios perform better, by a large margin, in comparison to agents that have been trained on a single terrain scenario. While the MA 0-99 setup has the highest cumulative reward marginally, we can observe that the MAC 0-99 setup, achieves the highest tree drop count and travels the furthest to explore (Figure 10), through communication.

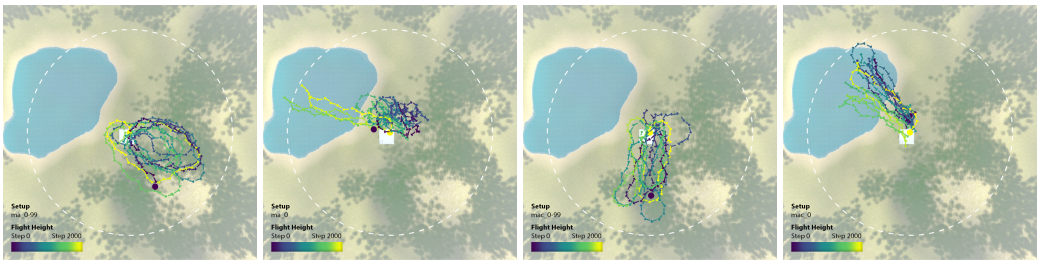
Table 2: Experiment results: Mean Cumulative Reward, Mean Cumulative Tree Drop Count, Cumulative Reward. Testing terrain scenario random seed: 111. Each tested 10 times for 1e6 Steps.

Experiment	Distance Reward (↑ better)	Tree Drop Count (↑ better)	Cml. Reward (↑ better)
1 MA 0	1.29 (±0.40)	7.75 (±0.53)	2.54 (±1.23)
2 MA 0-99	11.45 (±0.95)	7.91 (±0.46)	**122.08 (±8.7)**
3 MAC 0	2.85 (±0.78)	6.61 (±0.51)	7.82 (±2.66)
4 MAC 0-99	**13.18 (±1.14)**	**8.66 (±0.46)**	121.84 (±8.79)



(a) Cumulative Reward (b) Tree Drop Count (c) Cml. Distance Reward (d) Cml. Tree Drop Reward

Figure 9: Test Graphs. Zoomed-in version and additional data in the appendix: I.



(a) MA 0-99 Flight Path (b) MA 0 Flight Path (c) MAC 0-99 Flight Path (d) MAC 0 Flight Path

Figure 10: 2000 Step sample flight path (terrain seed 111). Zoomed-in version in the appendix: I.

7 Conclusion

This work is an approach to solving autonomous drone-based reforestation using Multi-Agent Reinforcement Learning (MARL) with a Graph Neural Network (GNN) communication layer that enables agents to collaborate. We have demonstrated that we can solve this task and generalise well on an unseen terrain scenario. We also show that communication can lead to collaboration and increase the performance of a multi-agent collective, ultimately outperforming the multi-agent setup without the ability to communicate. This is verified by increased tree seed drop counts and the quality and precision of tree seed planting, yielding higher rewards and, subsequently, a more efficient reforestation. Furthermore, we discovered that communication can lead to a higher risk-taking propensity and a larger area of forest explored. If an agent made it to a specific forest area, other agents can. We understand that the simulation to reality gap still exists, but this is a first step toward approaching this high-impact problem. We can see multiple ways to push this work forward. Firstly, improving the environment to be more realistic, with photo-realistic textures, trees, landscape, vegetation and real-world terrain data. The implemented drone control is an abstraction of a drone controller. Secondly, more realistic physics, including winds, and a control system that is closely aligned with standard drone controls, i.e. pitch, yaw and roll. And lastly, the camera vision and sensing can be improved. In this work, we use a 16x16 image as a visual observation input; this could be increased, and subsequently, the network architecture for processing. Furthermore, instead of a grey-scale image, we can add a RGB channel and process the observation image to additionally forward i.e. a depth map. This could lead to higher action precision and improved generalisability. Nevertheless, we believe we contributed value to the multi-agent reinforcement learning, collaboration and communication field on the back of a high-impact problem of relevance and urgency.

References

[1] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Santa Fe Institute Studies on the Sciences of Complexity. Oxford University Press, New York, 1999.

[2] James MacCarthy, Sasha Tyukavina, Mikaela Weisse, and Nancy Harris. New Data Confirms: Forest Fires Are Getting Worse, August 2022.

[3] Alexandra Tyukavina, Peter Potapov, Matthew C. Hansen, Amy H. Pickens, Stephen V. Stehman, Svetlana Turubanova, Diana Parker, Viviana Zalles, André Lima, Indrani Kommareddy, Xiao-Peng Song, Lei Wang, and Nancy Harris. Global Trends of Forest Loss Due to Fire From 2001 to 2019. *Frontiers in Remote Sensing*, 3, 2022.

[4] Elizabeth Dow Goldman, Mikaela Weisse, Nancy Harris, and Martina Schneider. Estimating the Role of Seven Commodities in Agriculture-Linked Deforestation: Oil Palm, Soy, Cattle, Wood Fiber, Cocoa, Coffee, and Rubber. *World Resources Institute*, 2020.

[5] Blanca Bernal, Lara T. Murray, and Timothy R. H. Pearson. Global carbon dioxide removal rates from forest landscape restoration activities. *Carbon Balance and Management*, 13(1):22, November 2018.

[6] Jeff Tollefson. Experiment aims to steep rainforest in carbon dioxide. *Nature*, 496(7446):405–406, April 2013. Number: 7446 Publisher: Nature Publishing Group.

[7] Jan Sendzimir, Chris P. Reij, and Piotr Magnuszewski. Rebuilding Resilience in the Sahel: Regreening in the Maradi and Zinder Regions of Niger. *Ecology and Society*, 16(3), 2011. Publisher: Resilience Alliance Inc.

[8] Francesco S. R. Pausata, Marco Gaetani, Gabriele Messori, Alexis Berg, Danielle Maia de Souza, Rowan F. Sage, and Peter B. deMenocal. The Greening of the Sahara: Past Changes and Future Implications. *One Earth*, 2(3):235–250, March 2020.

[9] Ignacio Amigo. When will the Amazon hit a tipping point? *Nature*, 578(7796):505–507, February 2020. Bandiera_abtest: a Cg_type: News Feature Number: 7796 Publisher: Nature Publishing Group Subject_term: Climate sciences, Climate change.

[10] Juliana Ennes. Illegal logging reaches Amazon's untouched core, 'terrifying' research shows, September 2021. Section: Environmental news.

[11] Zbigniew R. Bogdanowicz. Flying Swarm of Drones Over Circulant Digraph. *IEEE Transactions on Aerospace and Electronic Systems*, 53(6):2662–2670, December 2017. Conference Name: IEEE Transactions on Aerospace and Electronic Systems.

[12] Hyohoon Ahn, Duc-Tai Le, Dung Nguyen, and Hyunseung Choo. Real-Time Drone Formation Control for Group Display. In *Proceedings of the 13th International Conference on Ubiquitous Information Management and Communication (IMCOM) 2019 (pp.778-785)*, pages 778–785. Springer Cham, May 2019.

[13] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv:1707.06347 [cs]*, August 2017. arXiv: 1707.06347.

[14] Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barros, Charlie Deck, Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Michael Mathieu, Nat McAleese, Nathalie Bradley-Schmieg, Nathaniel Wong, Nicolas Porcel, Roberta Raileanu, Steph Hughes-Fitt, Valentin Dalibard, and Wojciech Marian Czarnecki. Open-Ended Learning Leads to Generally Capable Agents, July 2021. arXiv:2107.12808 [cs].

[15] Heinrich Küttler, Nantas Nardelli, Alexander H. Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. The NetHack Learning Environment. *arXiv:2006.13760 [cs, stat]*, December 2020. arXiv: 2006.13760.

[16] Paulo Leitão and Stamatis Karnouskos. *Industrial Agents: Emerging Applications of Software Agents in Industry*. Elsevier, March 2015.

[17] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, Multi-Agent, Reinforcement Learning for Autonomous Driving. *arXiv:1610.03295 [cs, stat]*, October 2016. arXiv: 1610.03295.

[18] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement Learning in Robotics: A Survey. *The International Journal of Robotics Research 32(11):1238-1274*, page 38, 2013.

[19] Jayesh K. Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative Multi-agent Control Using Deep Reinforcement Learning. In Gita Sukthankar and Juan A. Rodriguez-Aguilar, editors, *Autonomous Agents and Multiagent Systems*, volume 10642, pages 66–83. Springer International Publishing, Cham, 2017. Series Title: Lecture Notes in Computer Science.

[20] Zool Hilmi Ismail and Nohaidda Sariff. *A Survey and Analysis of Cooperative Multi-Agent Robot Systems: Challenges and Directions*. IntechOpen, November 2018. Publication Title: Applications of Mobile Robots.

[21] Nevena Lazic, Craig Boutilier, Tyler Lu, Eehern Wong, Binz Roy, MK Ryu, and Greg Imwalle. Data center cooling using model-predictive control. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[22] Joel Z. Leibo, Edgar Duéñez-Guzmán, Alexander Sasha Vezhnevets, John P. Agapiou, Peter Sunehag, Raphael Koster, Jayd Matyas, Charles Beattie, Igor Mordatch, and Thore Graepel. Scalable Evaluation of Multi-Agent Reinforcement Learning with Melting Pot, July 2021. arXiv:2107.06857 [cs].

[23] Arthur Charpentier, Romuald Élie, and Carl Remlinger. Reinforcement Learning in Economics and Finance. *Computational Economics*, April 2021.

[24] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 7587 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Computational science;Computer science;Reward Subject_term_id: computational-science;computer-science;reward.

[25] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, October 2017. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 7676 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Computational science;Computer science;Reward Subject_term_id: computational-science;computer-science;reward.

[26] Murray Campbell, A. Joseph Hoane, and Feng-hsiung Hsu. Deep Blue. *Artificial Intelligence*, 134(1):57–83, January 2002.

[27] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, December 2018. Publisher: American Association for the Advancement of Science.

[28] Thomas Anthony, Zheng Tian, and David Barber. Thinking Fast and Slow with Deep Learning and Tree Search. *arXiv:1705.08439 [cs]*, December 2017. arXiv: 1705.08439.

[29] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. DeepStack: Expert-Level Artificial Intelligence in No-Limit Poker. *Science*, 356(6337):508–513, May 2017. arXiv: 1701.01724.

[30] Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, January 2018. Publisher: American Association for the Advancement of Science.

[31] Thomas Anthony, Tom Eccles, Andrea Tacchetti, János Kramár, Ian Gemp, Thomas C. Hudson, Nicolas Porcel, Marc Lanctot, Julien Pérolat, Richard Everett, Roman Werpachowski, Satinder Singh, Thore Graepel, and Yoram Bachrach. Learning to Play No-Press Diplomacy with Best Response Policy Iteration. *arXiv:2006.04635 [cs, stat]*, January 2022. arXiv: 2006.04635.

[32] AB Calhamer. Diplomacy (game), 1959.

[33] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 7540 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Computer science Subject_term_id: computer-science.

[34] OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with Large Scale Deep Reinforcement Learning, December 2019. arXiv:1912.06680 [cs, stat].

[35] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, November 2019. Number: 7782 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Computer science;Statistics Subject_term_id: computer-science;statistics.

[36] Matthew C. Fontaine, Ya-Chuan Hsu, Yulun Zhang, Bryon Tjanaka, and Stefanos Nikolaidis. On the Importance of Environments in Human-Robot Coordination. *arXiv:2106.10853 [cs]*, June 2021. arXiv: 2106.10853.

[37] Tom Ward, Andrew Bolt, Nik Hemmings, Simon Carter, Manuel Sanchez, Ricardo Barreira, Seb Noury, Keith Anderson, Jay Lemmon, Jonathan Coe, Piotr Trochim, Tom Handley, and Adrian Bolton. Using Unity to Help Solve Intelligence. *arXiv:2011.09294 [cs]*, November 2020. arXiv: 2011.09294.

[38] Philip Cohen, Hector Levesque, and Ira Smith. On Team Formation. In *Contemporary Action Theory. Synthese*, pages 87–114. Kluwer Academic Publishers, 1997.

[39] Keith S. Decker. Distributed problem-solving techniques: A survey. *IEEE Transactions on Systems, Man, & Cybernetics*, 17(5):729–740, 1987. Place: US Publisher: Institute of Electrical & Electronics Engineers Inc.

[40] D. V. Pynadath and M. Tambe. The Communicative Multiagent Team Decision Problem: Analyzing Teamwork Theories and Models. *Journal of Artificial Intelligence Research*, 16:389–423, June 2002. arXiv: 1106.4569.

[41] Yoav Shoham. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

[42] Laëtitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems. *Knowledge Engineering Review*, 27(1):1–31, March 2012. Publisher: Cambridge University Press (CUP).

[43] Liviu Panait and Sean Luke. Cooperative Multi-Agent Learning: The State of the Art. *Autonomous Agents and Multi-Agent Systems*, 11(3):387–434, November 2005.

[44] Leonid Peshkin, Kee-Eung Kim, Nicolas Meuleau, and Leslie Pack Kaelnling. Learning to Cooperate via Policy Search, 2000.

[45] Martin Lauer and Martin Riedmiller. An Algorithm for Distributed Reinforcement Learning in Cooperative Multi-Agent Systems. In *In Proceedings of the Seventeenth International Conference on Machine Learning*, pages 535–542. Morgan Kaufmann, 2000.

[46] Ryan Lowe, YI WU, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[47] Emanuele Pesce and Giovanni Montana. Improving Coordination in Small-Scale Multi-Agent Deep Reinforcement Learning through Memory-driven Communication. *Machine Learning*, 109(9-10):1727–1747, September 2020. arXiv: 1901.03887.

[48] Pablo Hernandez-Leal, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote. A Survey of Learning in Multiagent Environments: Dealing with Non-Stationarity. *arXiv:1707.09183 [cs]*, March 2019. arXiv: 1707.09183.

[49] Ping Xuan, Victor Lesser, and Shlomo Zilberstein. Communication decisions in multi-agent cooperation: model and experiments. In *Proceedings of the fifth international conference on Autonomous agents*, AGENTS '01, pages 616–623, New York, NY, USA, May 2001. Association for Computing Machinery.

[50] MAJA J. MATARIC. Using communication to reduce locality in distributed multiagent learning. *Journal of Experimental & Theoretical Artificial Intelligence*, 10(3):357–369, July 1998. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/095281398146806.

[51] Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. Learning to Communicate with Deep Multi-Agent Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[52] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 2009.

[53] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural Message Passing for Quantum Chemistry. *arXiv:1704.01212 [cs]*, June 2017. arXiv: 1704.01212.

[54] A Anonymous. Collaborative Auto-Curricula Multi-Agent Reinforcement Learning with Graph Neural Network Communication Layer for Open-ended Wildfire-Management Resource Distribution, April 2022. arXiv:2204.11350 [cs].

[55] A Anonymous. The Power of Communication in a Distributed Multi-Agent System, December 2021. arXiv:2111.15611 [cs].

[56] Ravi N. Haksar and Mac Schwager. Distributed Deep Reinforcement Learning for Fighting Forest Fires with a Network of Aerial Robots. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1067–1074, Madrid, October 2018. IEEE.

[57] Kyle D. Julian and Mykel J. Kochenderfer. Image-based Guidance of Autonomous Aircraft for Wildfire Surveillance and Prediction. *arXiv:1810.02455 [cs]*, March 2019. arXiv: 1810.02455.

[58] Fatemeh Afghah, Abolfazl Razi, Jacob Chakareski, and Jonathan Ashdown. Wildfire Monitoring in Remote Areas using Autonomous Unmanned Aerial Vehicles. *arXiv:1905.00492 [cs, eess]*, April 2019. arXiv: 1905.00492.

[59] James F. Campbell. Will drones revolutionize home delivery? Let's get real.... *Patterns*, 3(8):100564, August 2022.

[60] Shahryar Sorooshian, Shila Khademi Sharifabad, Mehrdad Parsaee, and Ali Reza Afshari. Toward a Modern Last-Mile Delivery: Consequences and Obstacles of Intelligent Technology. *Applied System Innovation*, 5(4):82, August 2022. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[61] Ellen Reese and Jake Alimahomed-Wilson. Teamsters Confront Amazon: An Early Assessment - Ellen Reese, Jake Alimahomed-Wilson, 2022. *New Labor Forum*, August 2022.

[62] Flash Forest. Flash Forest.

[63] airseed. airseed.

[64] Gandham Venkata Sai Lohit. Reforestation Using Drones and Deep Learning Techniques. In *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, volume 1, pages 847–852, March 2021. ISSN: 2575-7288.

[65] R. Labayrade and D. Aubert. A single framework for vehicle roll, pitch, yaw estimation and obstacles detection by stereovision. In *IEEE IV2003 Intelligent Vehicles Symposium. Proceedings (Cat. No.03TH8683)*, pages 31–36, June 2003.

[66] G. Anup Venkatesh, P. Sumanth, and K. R. Jansi. Fully Autonomous UAV. In *2017 International Conference on Technical Advancements in Computers and Communications (ICTACC)*, pages 41–44, April 2017.

[67] Udacity-DeepRL. An Introduction to Proximal Policy Optimization (PPO) in Deep Reinforcement Learning, April 2019.

[68] Shaked Zychlinski. The Complete Reinforcement Learning Dictionary, November 2019.

[69] Joshua Achiam. Simplified PPO-Clip Objective, July 2018.

[70] Spinning Up OpenAI. Proximal Policy Optimization — Spinning Up documentation, 2021.

[71] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated Graph Sequence Neural Networks. *arXiv:1511.05493 [cs, stat]*, September 2017. arXiv: 1511.05493.

[72] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *arXiv:1710.10903 [cs, stat]*, February 2018. arXiv: 1710.10903.

[73] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. *arXiv:1606.09375 [cs, stat]*, February 2017. arXiv: 1606.09375.

[74] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *arXiv:1806.01261 [cs, stat]*, October 2018. arXiv: 1806.01261.

[75] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

[76] Paul Burkhardt. Optimal algebraic Breadth-First Search for sparse graphs. *ACM Transactions on Knowledge Discovery from Data*, 15(5):1–19, June 2021. arXiv: 1906.03113.

[77] N. Kaur and D. Garg. Analysis of the Depth First Search Algorithms. *undefined*, 2012.

[78] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. DeepWalk: Online Learning of Social Representations. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, August 2014. arXiv: 1403.6652.

[79] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures, June 2018. arXiv:1802.01561 [cs].

A Appendix

B Pseudocode

PPO-CLIP pseudocode [70, 13]:

Algorithm 1 PPO-Clip

Input: initial policy parameters θ_0, initial value function parameters ϕ_0
for $k = 0, 1, 2, \ldots$ **do**
    Collect set of trajectories $\mathcal{D}_k$ = $\{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.
    Compute rewards-to-go $\hat{R}_t$.
    Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the
    current value function V_{ϕ_k}
    Update the policy by maximizing the PPO-Clip objective:
    $\theta_{k+1} = \underset{\theta}{argmax} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)) \right),$
    typically via stochastic gradient ascent with Adam.
    Fit value function by regression on mean-squared error:
    $\phi_{k+1} = \underset{\phi}{argmin} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left((V_\phi(s_t) - \hat{R}_t \right)$
    typically via some gradient descent algorithm.
end for

Simple Multi-Agent PPO pseudocode:

Algorithm 2 Multi-Agent PPO

for $iteration = 1, 2, \ldots$ **do**
    for $actor = 1, 2, \ldots, N$ **do**
        Run policy $\pi_{\theta_{old}}$ in environment for T time steps
        Compute advantage estimates $\hat{A}_1, \ldots, \hat{A}_T$
    end for
    Optimize surrogate L wrt. θ, with K epochs and minibatch size $M \leq NT$
    $\theta_{old} \leftarrow \theta$
end for

C Hyperparameters

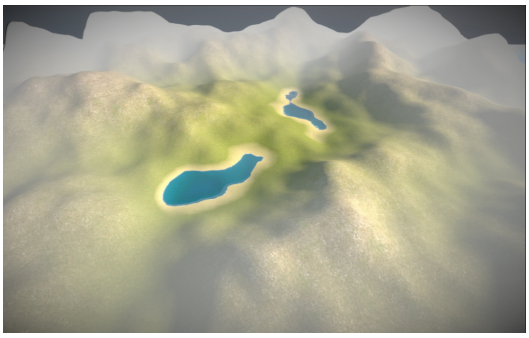
C.1 Multi Agent Training Hyperparameters


```
behaviors:
  MA_Drone:
    trainer_type: ppo
    hyperparameters:
      batch_size: 1024
      buffer_size: 10240
      learning_rate: 0.0003
      beta: 0.005
      epsilon: 0.2
      lambd: 0.95
      num_epoch: 3
      learning_rate_schedule: linear
    network_settings:
      normalize: false
      hidden_units: 128
      num_layers: 2
      vis_encode_type: resnet
    reward_signals:
      extrinsic:
        gamma: 0.99
        strength: 0.9
        network_settings:
          vis_encode_type: resnet
      curiosity:
        gamma: 0.99
        strength: 0.1
        encoding_size: 256
        learning_rate: 0.0003
        network_settings:
          vis_encode_type: resnet
    keep_checkpoints: 5
    max_steps: 10000000
    time_horizon: 100
    summary_freq: 20000
    threaded: true
```

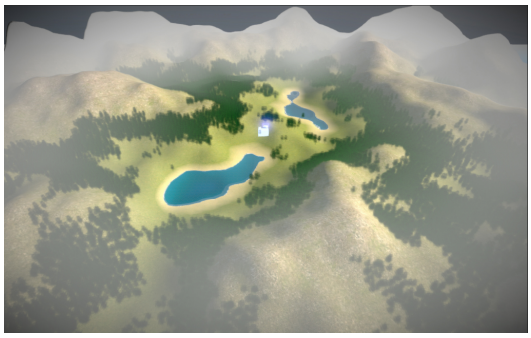
C.2 Hyperparameter Description

Hyperparameter	Typical Range	Description
Gamma	$0.8 - 0.995$	discount factor for future rewards
Lambda	$0.9 - 0.95$	used when calculating the Generalized Advantage Estimate (GAE)
Buffer Size	$2048 - 409600$	how many experiences should be collected before updating the model
Batch Size	$512 - 5120$ (continuous), $32 - 512$ (discrete)	number of experiences used for one iteration of a gradient descent update.
Number of Epochs	$3 - 10$	number of passes through the experience buffer during gradient descent
Learning Rate	$1e-5 - 1e-3$	strength of each gradient descent update step
Time Horizon	$32 - 2048$	number of steps of experience to collect per-agent before adding it to the experience buffer
Max Steps	$5e5 - 1e7$	number of steps of the simulation (multiplied by frame-skip) during the training process
Beta	$1e-4 - 1e-2$	strength of the entropy regularization, which makes the policy "more random"
Epsilon	$0.1 - 0.3$	acceptable threshold of divergence between the old and new policies during gradient descent updating
Normalize	$true/false$	weather normalization is applied to the vector observation inputs
Number of Layers	$1 - 3$	number of hidden layers present after the observation input
Hidden Units	$32 - 512$	number of units in each fully connected layer of the neural network
Intrinsic Curiosity Module		
Curiosity Encoding Size	$64 - 256$	size of hidden layer used to encode the observations within the intrinsic curiosity module
Curiosity Strength	$0.1 - 0.001$	magnitude of the intrinsic reward generated by the intrinsic curiosity module

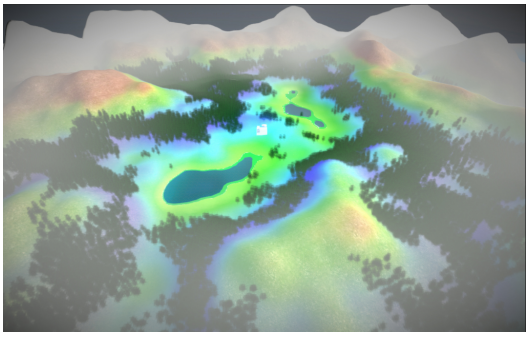
D Environment Features



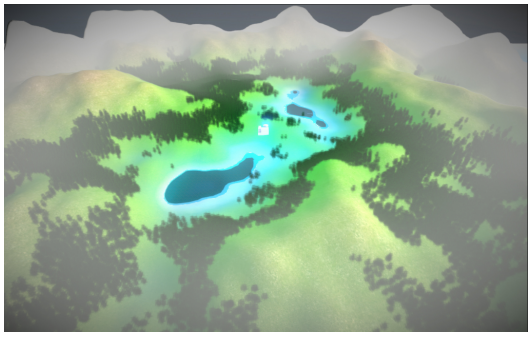
(a) Terrain, Seed 23, Difficulty 5



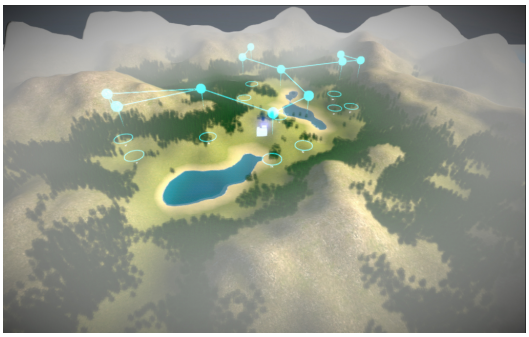
(b) Forest, Seed 23, Difficulty 5



(c) Optimal Reforestation Map, Seed 23, Difficulty 5

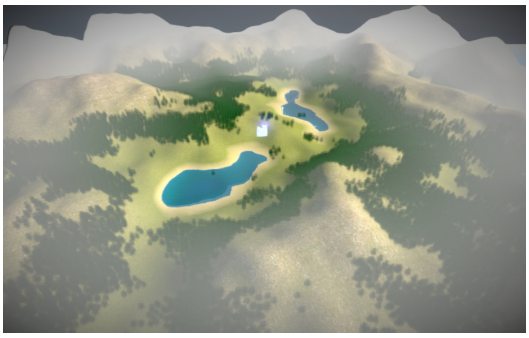


(d) Terrain Height Map, Seed 23, Difficulty 5

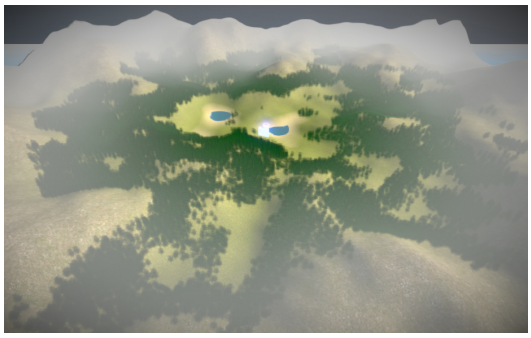


(e) Dynamic Drone Communication Network, Seed 23, Difficulty 5

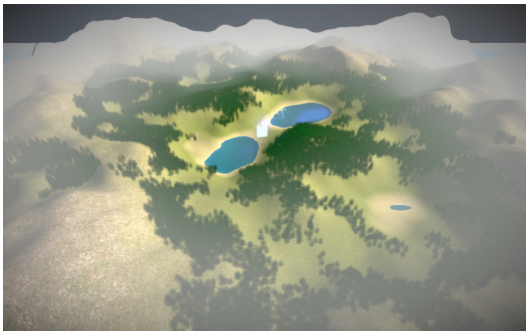
E Environment Scenarios



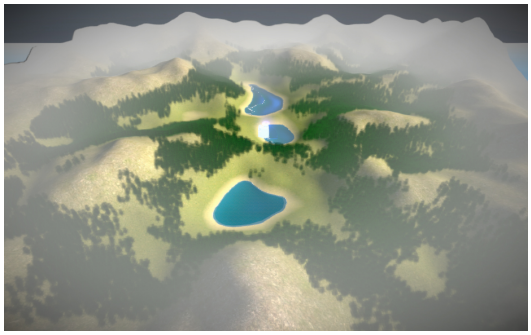
(a) Seed 23, Difficulty 5



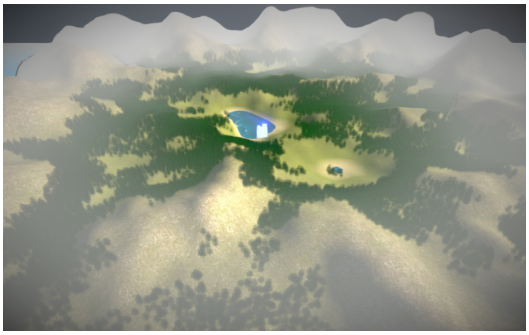
(b) Seed 25, Difficulty 5



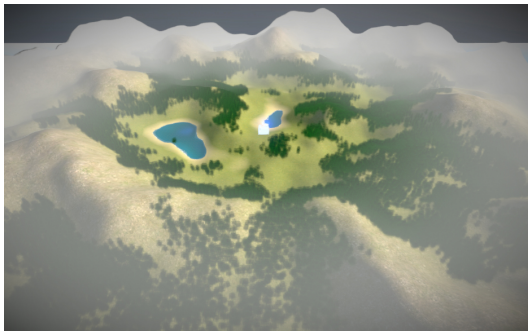
(c) Seed 26, Difficulty 5



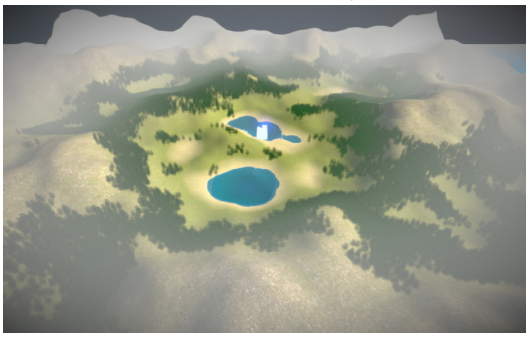
(d) Seed 27, Difficulty 5



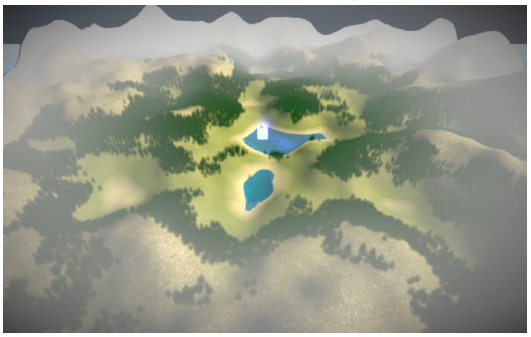
(e) Seed 28, Difficulty 5



(f) Seed 31, Difficulty 5

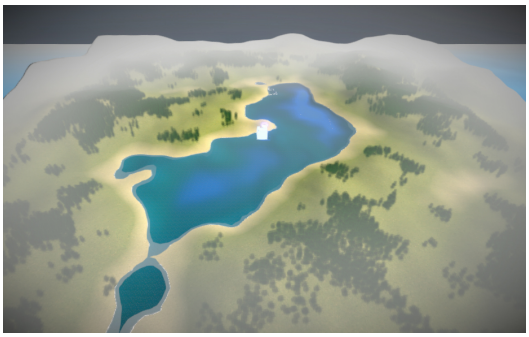


(g) Seed 32, Difficulty 5

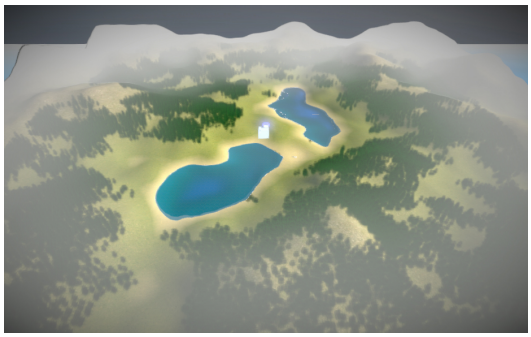


(h) Seed 33, Difficulty 5

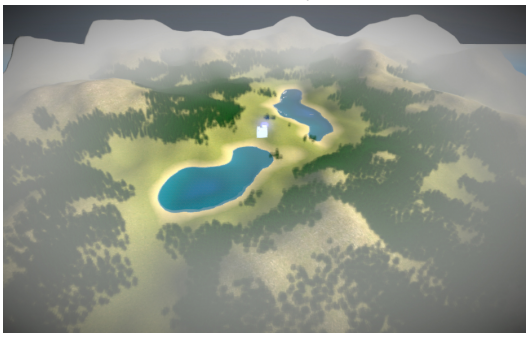
F Environment Difficulty



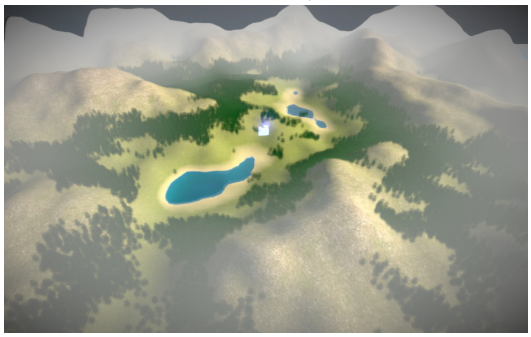
(a) Difficulty 1



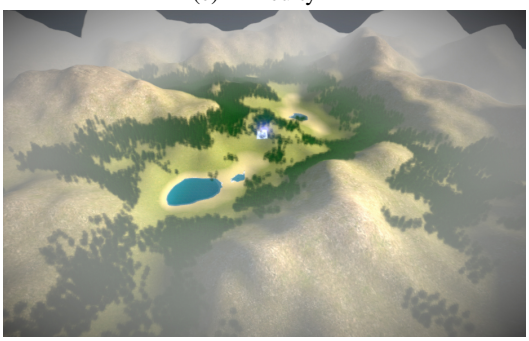
(b) Difficulty 3



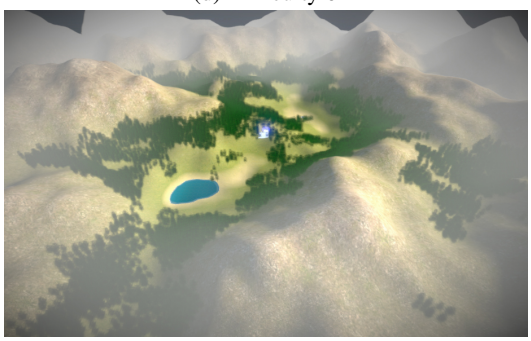
(c) Difficulty 4



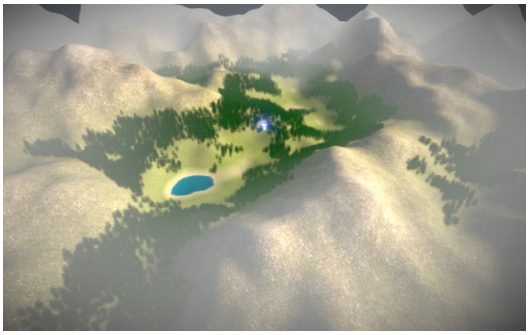
(d) Difficulty 6



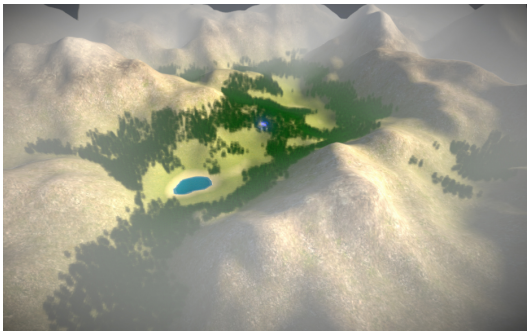
(e) Difficulty 7



(f) Difficulty 8



(g) Difficulty 9



(h) Difficulty 10

G Environment Scenario Samples: Difficulty VS. Seed Matrix

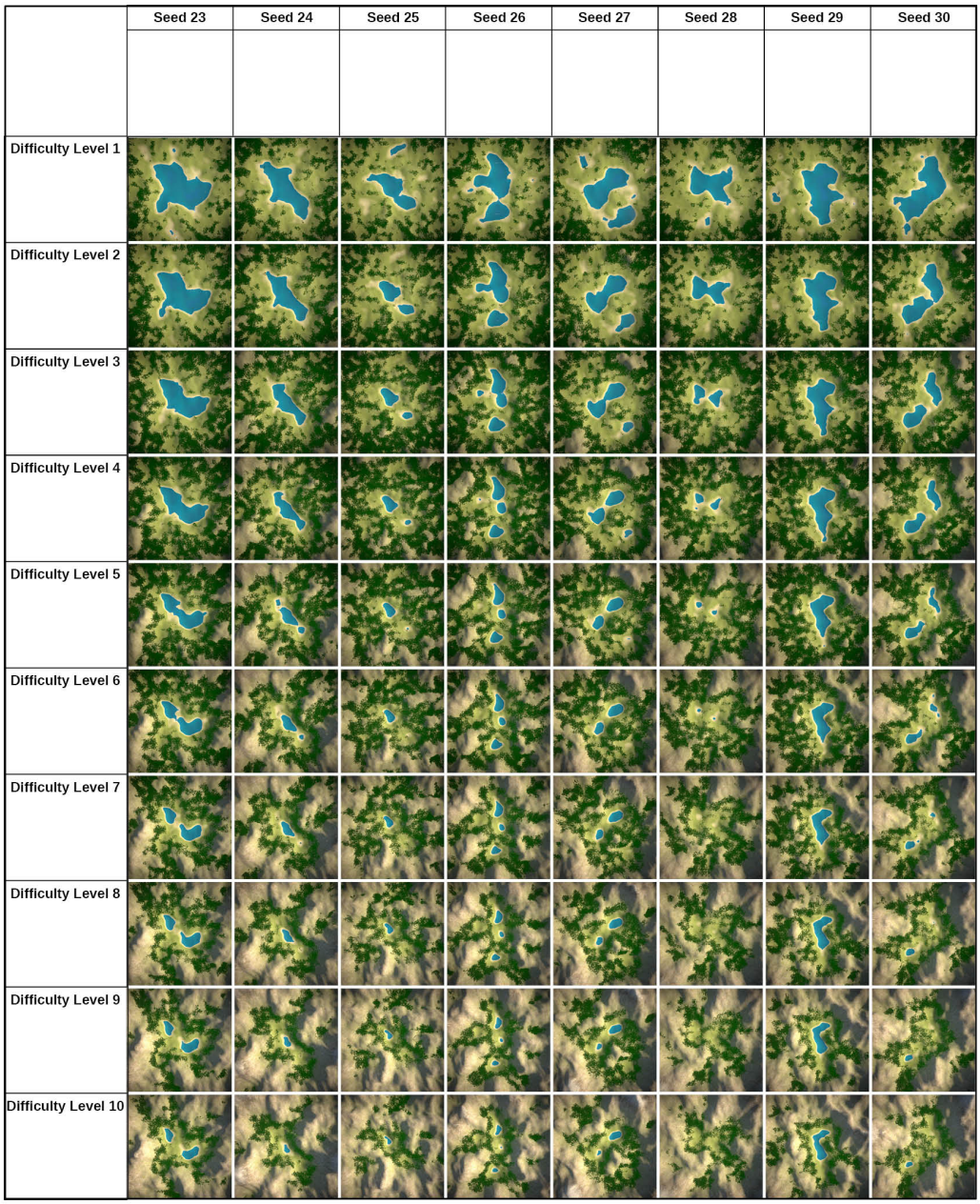
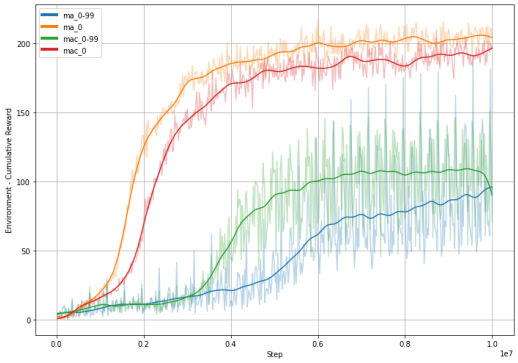
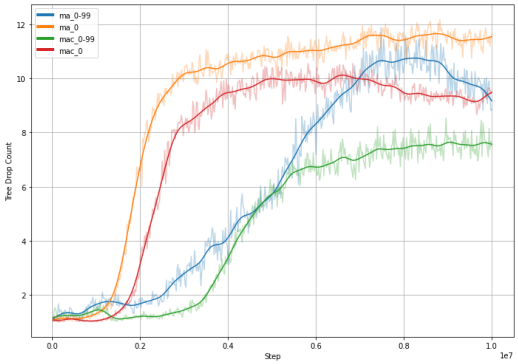


Figure 14: Difficulty Seed Matrix

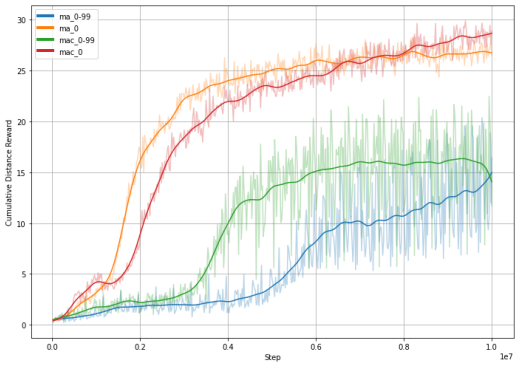
H Training Data



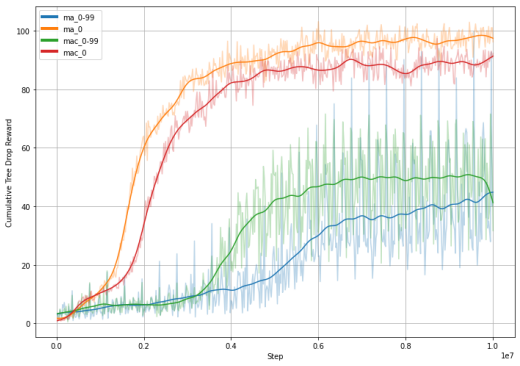
(a) Environment - Cumulative Reward



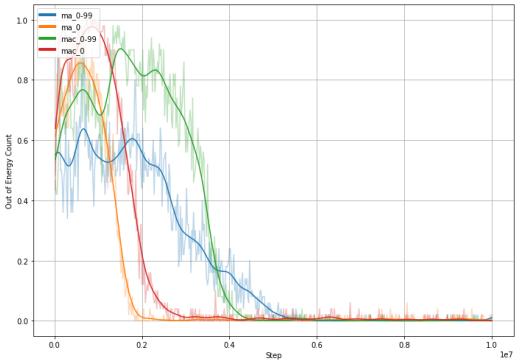
(b) Tree Drop Count



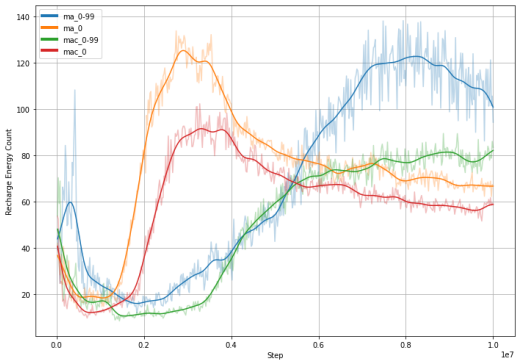
(c) Cumulative Distance Reward



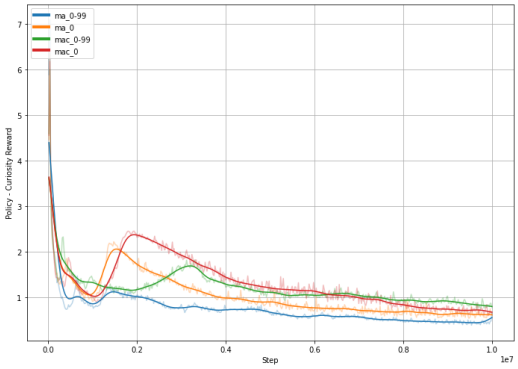
(d) Cumulative Tree Drop Reward



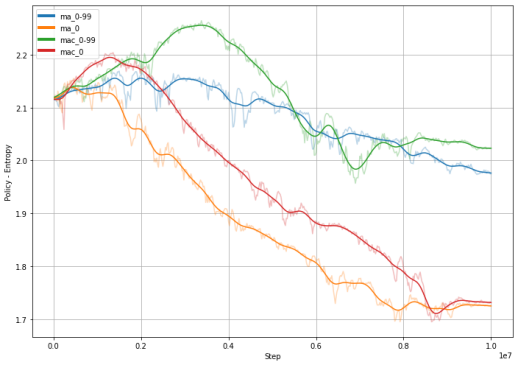
(e) Out of Energy Count



(f) Recharge Energy Count

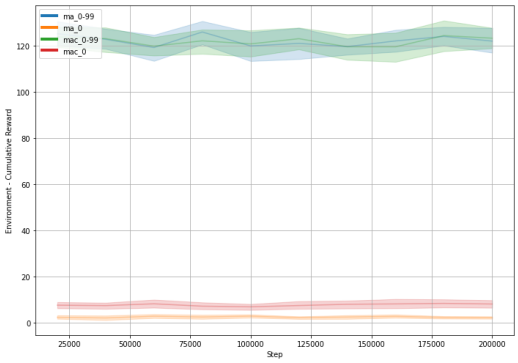


(g) Policy - Curiosity Reward

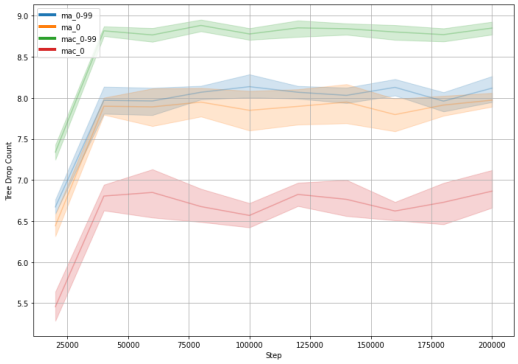


(h) Policy - Entropy

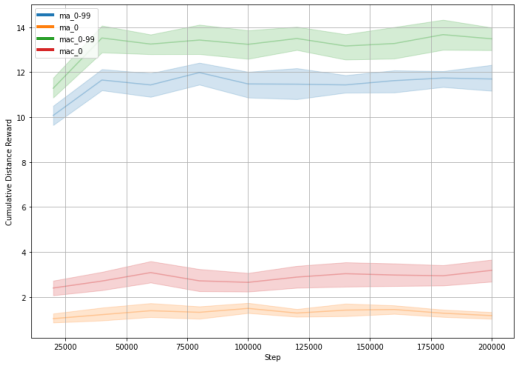
I Test Data: Terrain Scenario Seed 111



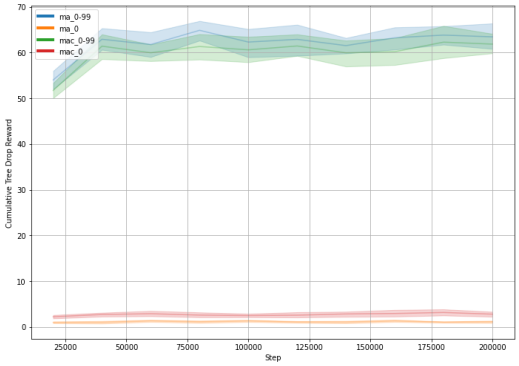
(a) Environment - Cumulative Reward



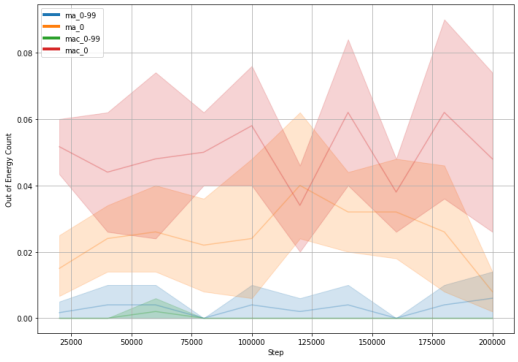
(b) Tree Drop Count



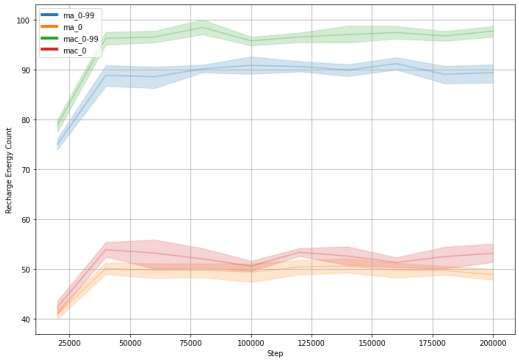
(c) Cumulative Distance Reward



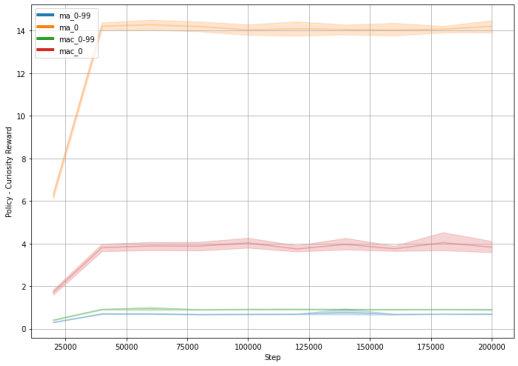
(d) Cumulative Tree Drop Reward



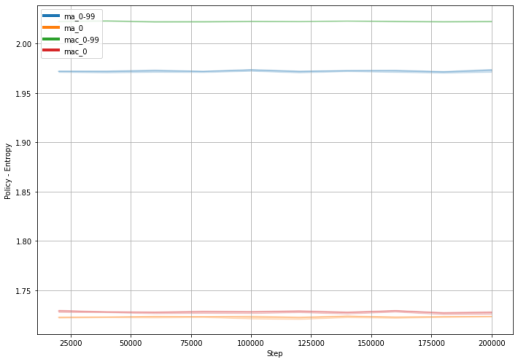
(e) Out of Energy Count



(f) Recharge Energy Count

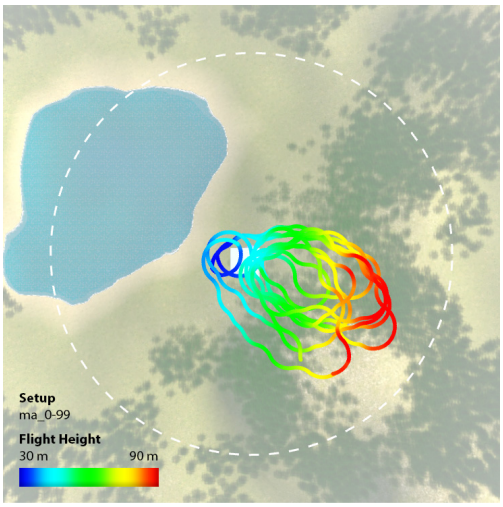


(g) Policy - Curiosity Reward

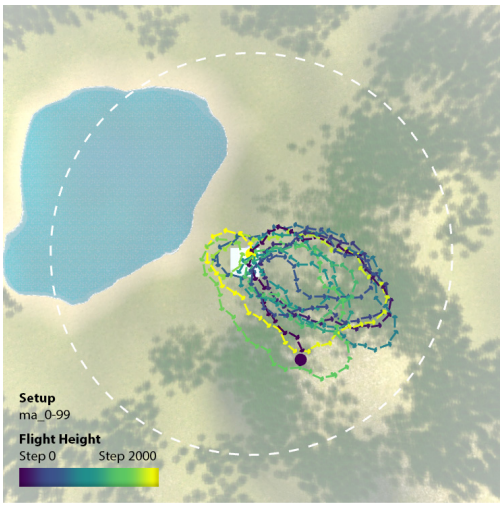


(h) Policy - Entropy

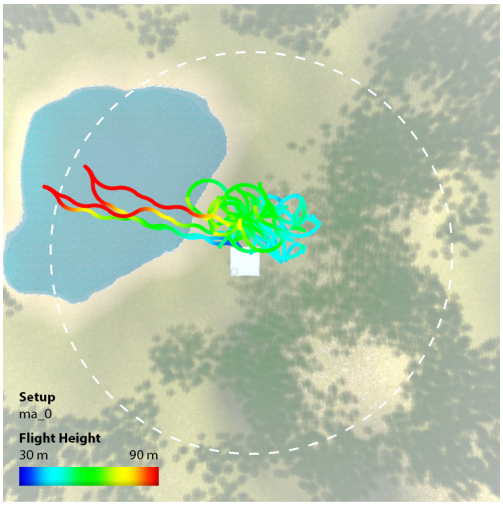
J Flight Path Plot: Flight Height and Path Sequence



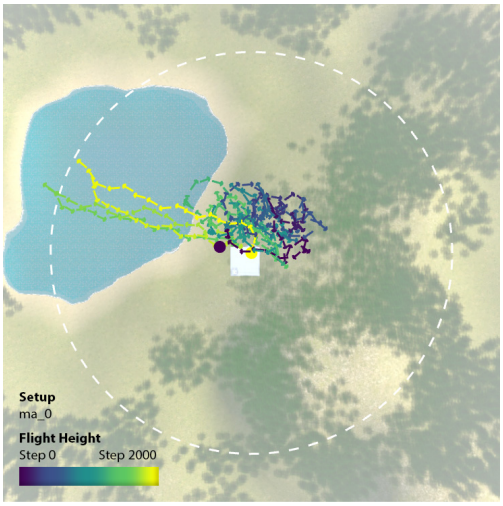
(a) Flight Height: MA 0-99 on Terrain Seed 111



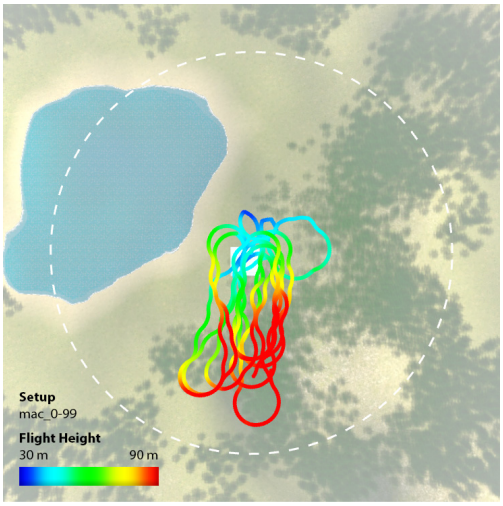
(b) Path Sequence: MA 0-99 on Terrain Seed 111



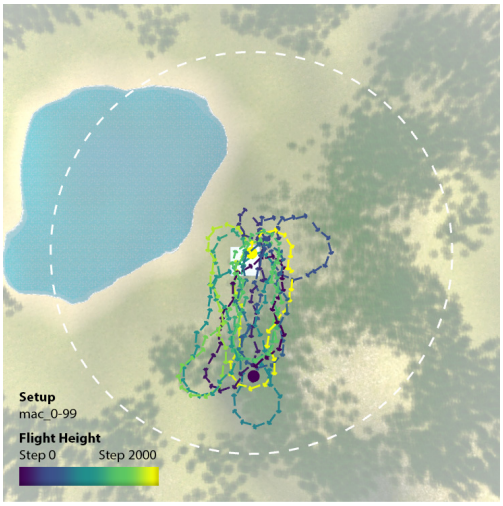
(c) Flight Height: MA 0 on Terrain Seed 111



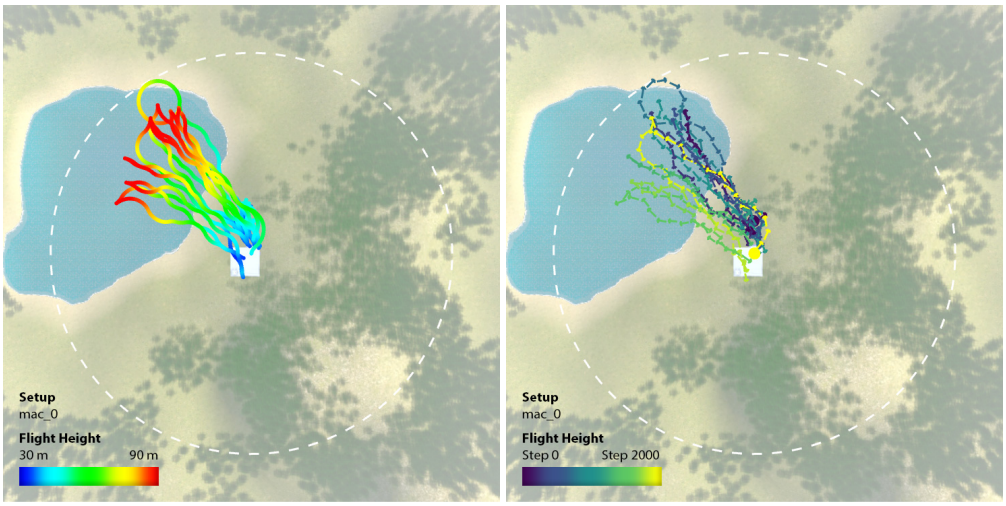
(d) Path Sequence: MA 0 on Terrain Seed 111



(e) Flight Height: MAC 0-99 on Terrain Seed 111

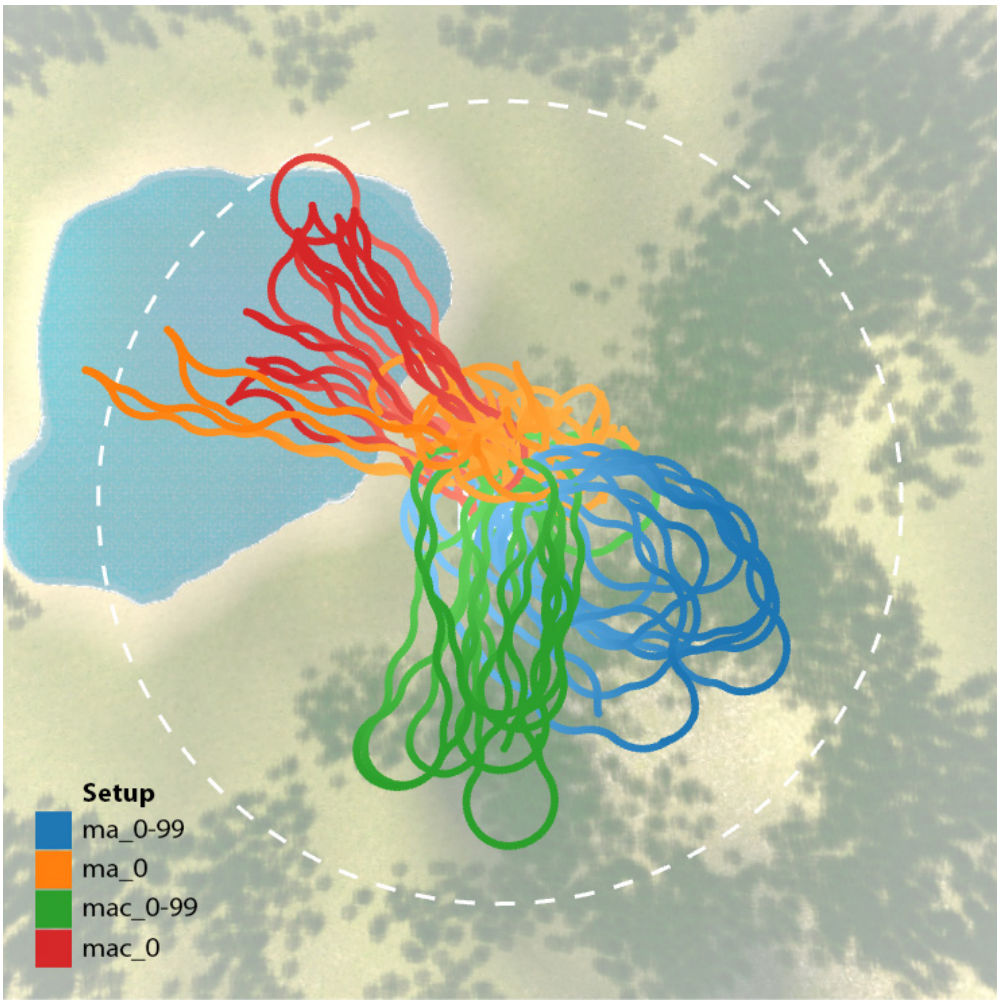


(f) Path Sequence: MAC 0-99 on Terrain Seed 111



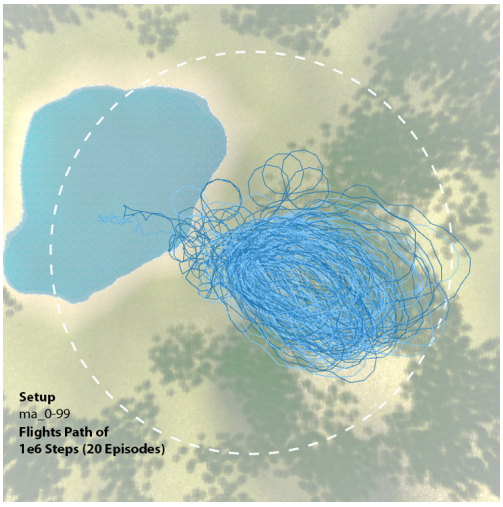
(a) Flight Height: MAC 0 on Terrain Seed 111 (b) Path Sequence: MAC 0 on Terrain Seed 111

J.1 2000 Time Step Inference on Terrain Scenario Seed 111: Setup Comparison

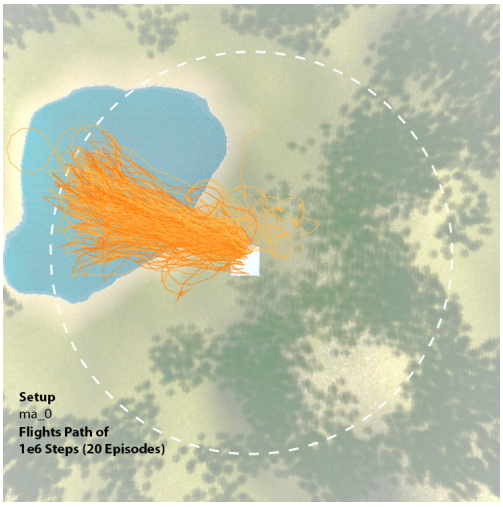


(a) MA 0-99, MA 0, MAC 0-99, MAC 0 - 2000 Steps Inference

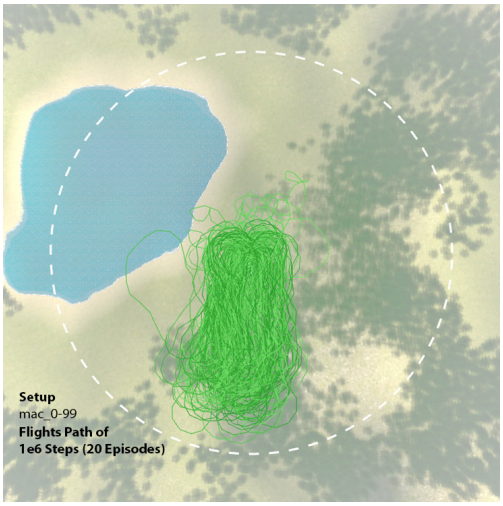
J.2 1e6 Time Step Inference on Terrain Scenario Seed 111



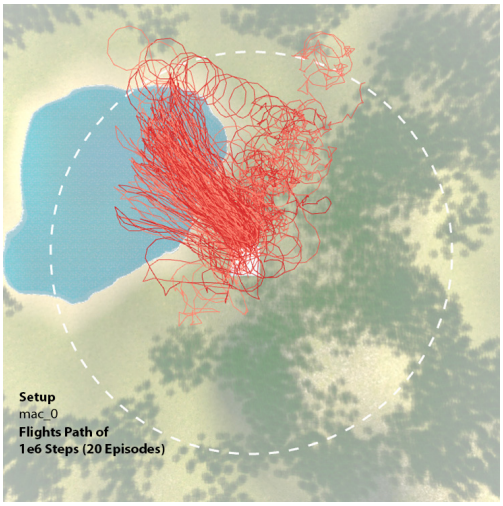
(a) MA 0-99 - 1e6 Steps Inference



(b) MA 0 - 1e6 Steps Inference



(c) MAC 0-99 - 1e6 Steps Inference



(d) MAC 0 - 1e6 Steps Inference