

Path Planning of Aerial Robots with Reinforcement Learning

Design Document

Team Name

sdmay24-08

Client

Amir Niaraki

Adviser

Ali Jannesari

Team

Andrew Sailer, Akash Setti, Cody Draper, Jesse Gillingham

Email | Website

sdmay24-08@iastate.edu | <https://sdmay24-08.sd.ece.iastate.edu/>

Revised

12/03/2023

Executive Summary

Project in Brief

The goal of this project is to train a drone using reinforced learning to search a simulated environment such anomalies found are maximized and energy usage is minimized.

Development Standards & Software Practices Used

- Python [1]
- Python Pep 8 Style Guide [8]
- OpenAI Gymnasium [3]
- StableBaselines3 [4]
- Version Control with Github [15]

Summary of Requirements

Simulate drone search of a specified area. Should return metrics of interest including battery usage, time taken, anomalies found, and visual representation of path taken.

Applicable Courses from Iowa State University Curriculum

CprE 288, EE 526, EE 322, ComS 474, DS 303, ComS 309, EE 285, ComS 327

New Skills/Knowledge Acquired During Project

- Rudimentary understanding of reinforcement learning in general
- Complete coverage algorithm
- Version control in group setting with Github
- Collaboration and communication for group coding projects
- Python and relevant Python packages

Table of Contents

Executive Summary	2
Project in Brief	2
Development Standards & Software Practices Used	2
Summary of Requirements	2
Applicable Courses from Iowa State University Curriculum	2
New Skills/Knowledge Acquired During Project	2
Table of Contents	3
1 Preamble	5
1.1 Team Members	5
1.2 Required Skill Sets for Project	5
1.3 Skill Sets of Team	5
1.4 Project Management Style	5
1.5 Initial Project Management Roles	6
1.6 Problem Statement	6
1.7 Requirements & Constraints	6
1.7.1 Functional Requirements	6
1.7.2 Resource Requirements	6
1.7.3 Environmental Requirements	7
1.7.4 Performance Requirements	7
1.7.5 Legal Requirements	7
1.7.6 Maintainability Requirements	7
1.8 Engineering Standards	7
1.9 Intended Users and Uses	8
2 Project Plan	9
2.1 Task Decomposition	9
2.1.1 Verify Performance of Existing Drone Simulation Environment	9
2.1.2 Image Preprocessor	9
2.1.3 Improve Complete Coverage Algorithm	10
2.1.4 Reinforcement Learning Research	10
2.1.5 Train Reinforcement Learning Model	10
2.1.6 Compiling Final Results	12
2.2 Project Management/Tracking Procedures	12
2.3 Project Proposed Milestones, Evaluation Criteria	12
2.3.1 Milestones	12
2.3.2 Evaluation Criteria	13
2.4 Project Timeline/Schedule	14
2.4.1 Initial 1st Semester Timeline	14
2.4.2 Actual 1st Semester Timeline	14

2.4.3 Initial 2nd Semester Timeline	15
2.4.4 Actual 2nd Semester Timeline	15
2.5 Risks and Risk Management/Mitigation	16
2.5.1 Verify Environment	16
2.5.2 Improve Complete Coverage Algorithm	16
2.5.3 Image Preprocessor	16
2.5.4 Reinforcement Learning Research	16
2.6 Personnel Hours	17
3 Design	17
3.1 Design Content	17
3.2 Design Complexity	17
3.2.1 Complete Coverage Engineering	17
3.2.2 Reinforcement Learning Engineering	18
3.3 Modern Engineering Tools	18
3.4 Design Context	19
3.5 Prior Work/Solutions	19
3.6 Design Decisions	20
3.6.1 Complete Coverage	20
3.6.2 Image Preprocessor	25
3.6.3 Simulation Environment & Reinforcement Learning	27
3.6.4 Scope of Project Change	28
3.7 Proposed Design and Design Iterations	28
3.7.1 Design 0 - Client's Existing Code	28
3.7.2 Design 1 - Improved Complete Coverage	30
3.7.3 Design 2 - Reinforcement Learning Introduction	30
3.7.4 Design 3 - Reinforcement Learning with Exploration Reward Function	31
3.7.5 Design 4 - Current Environment and Complete Coverage Algorithm	31
3.8 Design Analysis	33
4 Testing	33
4.1 Unit Testing	33
4.2 Interface Testing	34
4.3 Integration Testing	35
4.4 System Testing	35
4.5 Acceptance Testing	35
4.6 Results	35
5 Professionalism	37
5.1 Areas of Responsibility	37
5.2 Project Specific Professional Responsibility Areas	38
5.3 Most Applicable Professional Responsibility Area	40
6 Closing Material	40

6.1 Discussion	40
6.2 Conclusion	41
6.3 References	41
7.4 Appendices	42
7.5 Team Contract	42

1 Preamble

1.1 Team Members

- Jesse Gillingham
- Andrew Sailer
- Akash Setti
- Cody Draper

1.2 Required Skill Sets for Project

- Python Programming
- Problem solving/debugging
- Making design compromises
- Learning and applying complex ideas from online resources

1.3 Skill Sets of Team

- **Jesse Gillingham:** hardware, embedded systems,
- **Andrew Sailer:** hardware, control systems, programming, machine learning
- **Akash Setti:** software development, data science, machine learning
- **Cody Draper:** software development, embedded system, machine learning

1.4 Project Management Style

We are using a hybrid project management style by incorporating pieces from both the Agile and Waterfall management philosophies. For Waterfall, a rough timeline in the form of a Gantt chart is set up with the client and advisor; enumerating milestones to be met, goals to be accomplished, and their respective deadlines. For Agile, a meeting is scheduled with the client weekly, where past individual accomplishments, challenges, and next steps are discussed in detail.

A Discord server with the team members, client and advisor allows for direct and immediate communication at all times. This improves productivity by permitting questions to be opened up to the entire team daily, as opposed to weekly.

1.5 Initial Project Management Roles

- **Jesse Gillingham:** Reinforcement Learning research, document management
- **Andrew Sailer:** Take notes on in person meetings, complete coverage work
- **Akash Setti:** Record online meetings, complete coverage work
- **Cody Draper:** Coding support, complete coverage work

1.6 Problem Statement

The goal of this project is to train a drone using reinforced learning to search a simulated environment such that maximum anomalies are found and energy usage is minimized. A comparison between these metrics will determine which model performs better on a given image.

1.7 Requirements & Constraints

1.7.1 Functional Requirements

- Drone can be controlled using a complete coverage search algorithm.
- Drone can be controlled using a trained reinforcement learning model.
- Both methods of drone control can be simulated on the same image.
- Both algorithms will return steps used, anomalies found, and display the path taken for comparison.
- An image preprocessing script will mask parts of an image and remove all rewards. The image should then be saved and available for use by both search algorithms.
- A script to generate a randomized image, with parameters to control frequency of anomalies and size of anomalies.

1.7.2 Resource Requirements

- Personal computers for all members
- IDE (VSCode)
- Git
- Python and libraries
 - Gymnasium
 - Numpy

- Pandas
- Math
- Random
- Os
- Stablebaselines3
- Cv2 (opencv-python)
- Threading
- Sys
- Time

1.7.3 Environmental Requirements

Simulated search algorithm should attempt to minimize energy usage. Only applies to a physical drone, which is no longer in the scope of this project.

1.7.4 Performance Requirements

- Algorithms should maximize the number of anomalies found, and minimize energy usage, and time taken for a complete search.
- Training of reinforcement learning models should not exceed one day on a personal machine. Simplifications can be made to reduce environment complexity and reduce training time.

1.7.5 Legal Requirements

Testing with a physical drone will conform with US UAV regulations. The drone will not trespass on private property, or search legally unauthorized areas. Only applies to a physical drone, which is no longer in the scope of this project.

1.7.6 Maintainability Requirements

- Code will be commented and formatted according to Pep 8 Style Guide [14]
- Code will be written in a modular fashion
- Simulation environment will conform to OpenAI's Gymnasium API [3]

1.8 Engineering Standards

PEP 8 Style Guide [14]

All Python source code shall follow the PEP 8 Style Guide for Python code for consistency and readability.

OpenAI Gymnasium [3]

OpenAI's Gymnasium API for reinforcement learning. Describes interface for custom reinforcement learning environments.

Pytorch Documentation [2]

Pytorch API for designing custom deep learning models.

StableBaselines3 [4]

Reinforcement learning models. Compatible with OpenAI Gymnasium.

IEEE 1936.1-2021 - Standard for Drone Application Framework [5]

IEEE's general requirements for safe operation of drones in various applications.

Standards and regulations surrounding drones and their applications.

49 USC 44809: Exception for limited recreational operations of unmanned aircraft

[6]

US Law describing requirements for legally operating a drone (pilot's license), and regulations surrounding the operation of drones.

1.9 Intended Users and Uses

Because this project is attempting to improve upon an existing technology, its users and uses will remain the same. Some potential users of this technology are:

- Agricultural Businesses
- Farmers
- Municipal Governments
- State Governments
- Emergency Responders

Potential uses include types of aerial surveillance such as:

- Crop/Field Monitoring
- Infrastructure Monitoring
- Search & Rescue.

2 Project Plan

2.1 Task Decomposition

A discussion with the client lead to the initial list of tasks:

- Verify Drone Environment
- Object Detection
- Research Reinforcement Learning Algorithms
- Reinforcement Learning Experimentation
- Implementation on Physical Drone

After completion of the first task, verifying the drone environment, the client asked us to instead work on improving the complete coverage algorithm, which is intended to provide a baseline performance score for reinforcement learning. The task proved very challenging and time consuming, and resulted in a major detour from our original schedule. See 2.4.1.

A further discussion with the client led to a revised task list seen in sections 2.1.1 - 2.1.5. This included removing any application of the simulation to a physical drone, due to having underestimated the complexity and time requirements of the simulation portion of the project.

2.1.1 Verify Performance of Existing Drone Simulation Environment

Before any experiments with drone search algorithms can be performed, the drone environment simulation needs to be verified. The client requested the following subsystems in particular: anomaly collection, world padding, and frame resizing of the drone.

These were verified by conducting specific tests where the drone was manually controlled and the output was correlated with the expected values. Fixes were made to anomaly collection as well as world padding.

2.1.2 Image Preprocessor

An additional task from the client involved making a Python script to mask parts of a generated world effectively removing the rewards from them and shrinking the searchable area.

The script is also capable of creating multiple polygons where the drone can travel between separate polygons. Of these polygons the drone is able to complete the complete coverage algorithm.

2.1.3 Improve Complete Coverage Algorithm

In order to validate any sort of performance increase from reinforcement learning, we need to represent the performance of existing work fairly. Research and input from our client suggests current solutions for using drones to search areas involve a complete coverage algorithm.

The existing code from the client allows the drone to search a rectangular area. The client requested that the complete coverage algorithm be able to search a polygon (specified by the image masking script) in an optimal/efficient manner.

2.1.4 Reinforcement Learning Research

Learn the fundamentals of reinforcement learning. Research related applications of reinforcement learning, as well as state of the art of drone search algorithms. Identify relevant performance metrics for comparison. Research the most optimal models/strategies that provide the best results in simulation.

2.1.5 Train Reinforcement Learning Model

Using the custom environment, train reinforcement learning models to search the image for anomalies. The first milestone is to train a model on an image with a line of anomalies (figure 1.). This verifies that the environment is suitable for the model to learn how to find anomalies.

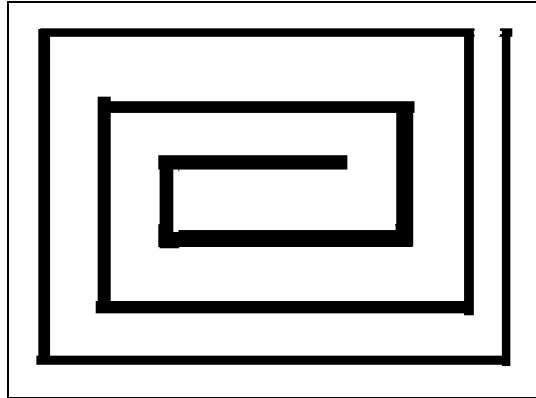


figure 1. Line Image

The next milestone is to train a model on an image of randomly generated anomalies (figure 2.). The image should vaguely resemble the quantity and distribution of anomalies on real images.

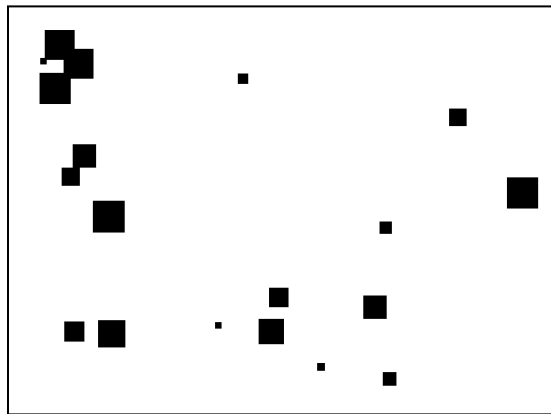


figure 2. Randomized Anomalies Image

The final milestone is to train a model on a real image (figure 3.) given by the client. A new method of calculating rewards from the drone's view will implement a deep learning model also from the client. A new method to avoid double-counting anomalies will need to be conceived.

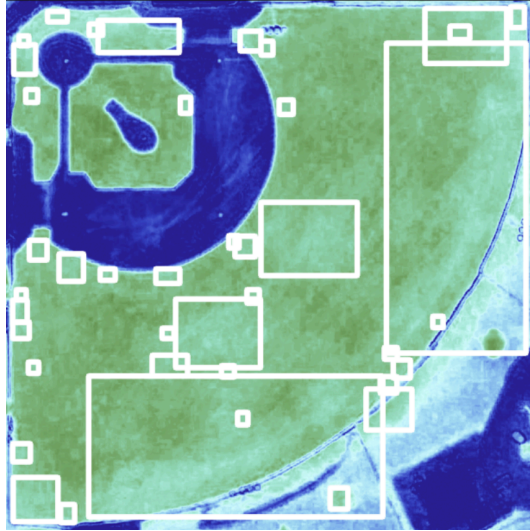


figure 3. Real Image w/ Anomalies Detected

2.1.6 Compiling Final Results

Decisions on best way to compare performance. Advantages and disadvantages of both reinforcement learning and complete coverage. Verifying that chosen metrics fairly represent algorithms. Make sure the algorithms run successfully, and compare them to see what is most optimal in simulation.

2.2 Project Management/Tracking Procedures

Git is used to progress software as a group. The commit log keeps track of individuals' contributions to the project. Separate branches are also created to start new ideas or to debug existing parts of the project.

Google Drive is used to host all files and allow for collaborative work on documents. Revision history can be used to track individuals' contributions.

2.3 Project Proposed Milestones, Evaluation Criteria

2.3.1 Milestones

- Image preprocessor
- Complete coverage algorithm
- Line follower reinforcement learning model
- Randomize image reinforcement learning model
- Real image reinforcement learning model
- Comparison between complete coverage and reinforcement learning on randomized images

2.3.2 Evaluation Criteria

Evaluation of search performance will include the number of steps in the search, the quantity of anomalies found, and the total number of anomalies in the searched image. Percentage of total anomalies found and average anomalies per step will be calculated.

5 simulations of the reinforcement learning model will be used to collect this data since performance varies slightly even on the same image. The average and range of the above data will be recorded. It is expected that the reinforcement learning model will perform better on sparser images, as opposed to complete coverage which will perform better on denser images.

2.4 Project Timeline/Schedule

2.4.1 Initial 1st Semester Timeline

Task \ Date	10/20	10/27	11/3	11/10	11/17	11/24	12/1	12/8	12/15	2nd Sem.
Verify Drone Environment										
Object Detection										
Research RL Algorithms										
RL Experimentation										
Implement on Physical Drone										

figure 4. Initial 1st Semester Timeline

2.4.2 Actual 1st Semester Timeline

Task \ Date	10/20	10/27	11/3	11/10	11/17	11/24	12/1	12/8	12/15	2nd Sem.
Verify Drone Environment										
Improve Complete Coverage Algorithm (Andrew & Cody)										
Image Editor (Cody)										
Research RL Algorithms (Jesse & Akash)										

figure 5. Actual 1st Semester Timeline

2.4.3 Initial 2nd Semester Timeline

Task \ Date	1/19	1/26	2/2	2/9	2/16	2/23	3/1	3/8	3/15	3/22
Research RL Algorithms	█	█	█	█	█	█	█	█		
RL Experimentation			█	█	█	█	█	█	█	█
Compilation of Final Results										

Task \ Date	3/29	4/5	4/12	4/19	4/26	5/3
Research RL Algorithms						
RL Experimentation	█	█	█			
Compilation of Final Results		█	█	█	█	█

figure 6. Initial 2nd Semester Timeline

2.4.4 Actual 2nd Semester Timeline

Task \ Date	1/19	1/26	2/2	2/9	2/16	2/23	3/1	3/8	3/15	3/22
Research RL Algorithms	█	█	█							
RL Implementation			█	█	█	█	█	█	█	█
Complete Coverage & Image Preprocessor	█	█	█	█	█	█	█	█	█	█
Compilation of Final Results										

Task \ Date	3/29	4/5	4/12	4/19	4/26	5/3
Research RL Algorithms						
RL Implementation	█	█	█	█	█	█
Complete Coverage & Image Preprocessor	█	█	█	█	█	█
Compilation of Final Results		█	█	█	█	█

figure 7. Actual 2nd Semester Timeline

2.5 Risks and Risk Management/Mitigation

2.5.1 Verify Environment

This task should not require a lot of hours, unless something is wrong with the drone environment. Should any major bugs be discovered, no other simulation tasks should continue before these issues are resolved. Should issues emerge later in the project, all other work will be paused to resolve any environment issues.

2.5.2 Improve Complete Coverage Algorithm

The overall guidelines and initial implementations were done in the first semester. While different constraints and improvements in the second semester. Design considerations must be made to limit the complexity of this task. See section 3.6.1.

2.5.3 Image Preprocessor

Much like the Complete Coverage Algorithm task, this task's complexity must be constrained to only do what we need. Much of the image preprocessor's functionality depends on the requirements of complete coverage. See section 3.6.2.

2.5.4 Reinforcement Learning Research

Reinforcement learning algorithms and models can be quite complex, and because this project does not necessitate complete understanding of reinforcement learning research should be focused on how to set up the environment to best utilize reinforcement learning.

2.6 Personnel Hours

Task	Estimated Hours	Andrew	Jesse	Cody	Akash
Verify Drone Environment	20	7	7	7	7
Improve Complete Coverage Algorithm	60	20	10	15	10
Image Preprocessor	20	5	0	10	0
Research RL Algorithms	20	3	15	3	10
RL Implementation					
Class Deliverables	40	10	10	10	10
Sum	160	45	42	45	37

figure 8. Personnel Hours

3 Design

3.1 Design Content

The project requires the design, test, and comparison of a complete coverage and reinforcement learning search algorithm. The complete coverage search algorithm acts as the baseline search algorithm. The reinforcement learning search algorithm will attempt to improve upon complete coverage in terms of anomalies found per step.

3.2 Design Complexity

3.2.1 Complete Coverage Engineering

Designing an all encompassing exhaustive search algorithm that covers all edge cases is exceptionally difficult. Therefore, design decisions will need to be made to eliminate

edge-cases and simplify the scope of complete coverage. It is important that complete coverage performs at least semi-optimally so that reinforcement learning has a meaningful baseline to compete against that represents the industry standard solution fairly.

The end product for complete coverage will not generate a perfectly optimized path, but should demonstrate some degree of optimality.

3.2.2 Reinforcement Learning Engineering

Engineering the custom simulation environment such that reinforcement learning models can utilize observations and actions to learn to search images has proven challenging.

Training a model to follow a line was not a milestone in the original project, but was added later to verify that the environment was using the correct observation space, action space, and reward function.

Simplifications to the environment were also performed later on to reduce the training time required for each model. These simplifications include fixing the drone's flight height, effectively removing vertical movement from the action space. Wind was also eliminated from the environment in order to reduce observation space and complexity of reward function.

The reward function and observation space were changed numerous times to learn what inputs the reinforcement model needed to train and perform properly. See 3.6.3.

3.3 Modern Engineering Tools

- **Github:** Industry standard code hosting platform for version control and collaboration
- **VSCode:** Industry IDE used to edit code
- **Discord:** Messaging/chat room application
- **Google Drive:** Cloud based file storage, allows for collaboration of word processing documents, presentation, and spreadsheets
- **Python:** Popular and versatile scripting language
- **Microsoft Teams:** Messaging/chat room application (advisor meetings)

3.4 Design Context

Area	Description	Examples
Public health, safety, and welfare	Our project affects the general well-being of the agricultural industry. Our project affects the general well-being of the people affected by dangerous situations. Our project affects the general well-being of a population suffering from food shortages due to crop decay.	Reduce the exposure to and propagation of plant based diseases in crops. Reduce the time to finding targets in search and rescue missions by automating searching. Increase crop yields by culling infected crops of infection when found.
Global, cultural, and social	Raises ethical concerns over whether employing this technology is net-beneficial to society.	Surveillance jobs normally performed by humans could be replaced with robots. May result in areas being involuntarily surveyed, if used improperly. Could be considered a privacy violation.
Environmental	Is useful for projects spanning large areas which may apply to environmental remediation projects.	Reforestation, chemical clean up, waste management...
Economic	Can replace menial surveillance jobs with automated drones, saving companies money on wages.	Routine surveillance of large-scale infrastructure such as roads, powerlines, buildings...

figure 9. Design Context

3.5 Prior Work/Solutions

The client demonstrated existing software which plans an optimal complete coverage path for a drone based on an outlined image. Our project intends to go a step further and optimize for the number of anomalies found instead of speed of area coverage. There is sufficient research on finding optimal complete coverage paths for different types of robots (roombas, drones, submarines...). These will be useful baselines to compare our final product against. [8, 10, 11, 12]

3.6 Design Decisions

3.6.1 Complete Coverage

The original complete coverage algorithm from the client ran perfectly over the entirety of a rectangular image, but when tasked with searching a polygon of area, many more factors were introduced.

The most optimal way to search a polygon would be to follow the longest edge of the shape to reduce the number of turns made by the drone, saving time and energy. Since the unmasked polygon (to be searched) is described by an array of 2D points, it is difficult to start on the longest edge, follow it in a lawn-mower fashion, and guarantee the entire area of the polygon will be explored. Numerous edge-cases can be trivially thought of.

The most optimal way to search a polygon would be to follow the longest edge of the shape to reduce the number of turns made by the drone, saving time and energy. This in itself is not difficult to achieve if we choose to keep the overall shape of the image rectangular in nature, only search one area, and move with a horizontal search bias as seen in figure 10. Keeping these factors as constant in complete coverage simplifies the search but is poorly optimized.

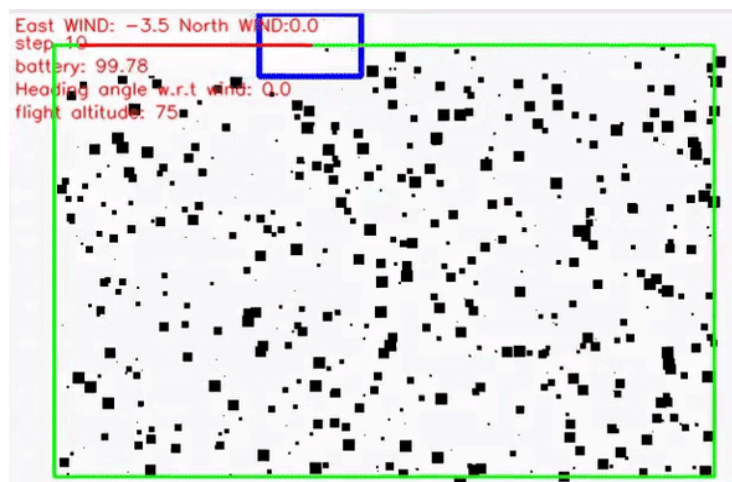
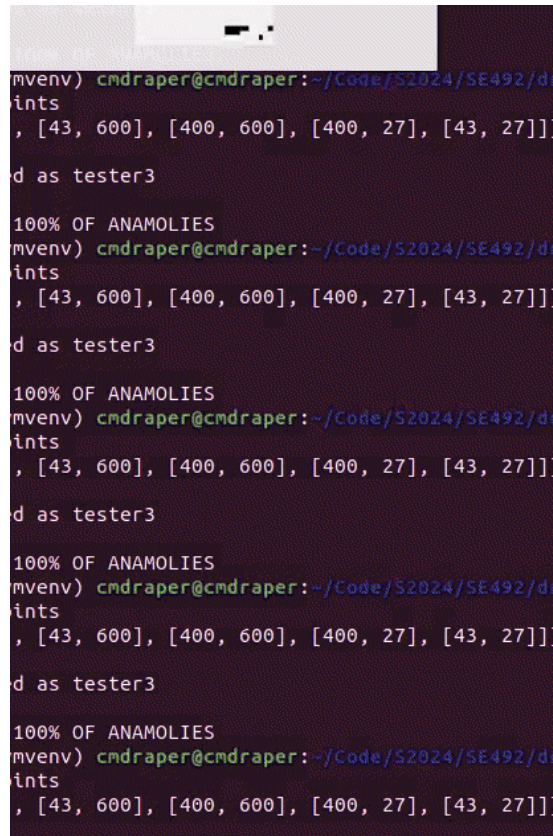


figure 10. Horizontal Search

To optimize the complete coverage algorithm to be as efficient as possible a number of constants need to be allowed to change based on the search space.

The first major design change was to design a vertical search bias component, seen in figure 11, that searches up and down instead of left to right. This allows the

search algorithm to optimize searching a space that is taller than it is wide, limiting the direction changes.



```
mvenv) cmdraper@cmdraper:~/Code/S2024/SE492/d
ints
, [43, 600], [400, 600], [400, 27], [43, 27]]

d as tester3

100% OF ANAMOLIES
mvenv) cmdraper@cmdraper:~/Code/S2024/SE492/d
ints
, [43, 600], [400, 600], [400, 27], [43, 27]]

d as tester3

100% OF ANAMOLIES
mvenv) cmdraper@cmdraper:~/Code/S2024/SE492/d
ints
, [43, 600], [400, 600], [400, 27], [43, 27]]

d as tester3

100% OF ANAMOLIES
mvenv) cmdraper@cmdraper:~/Code/S2024/SE492/d
ints
, [43, 600], [400, 600], [400, 27], [43, 27]]

d as tester3

100% OF ANAMOLIES
mvenv) cmdraper@cmdraper:~/Code/S2024/SE492/d
ints
, [43, 600], [400, 600], [400, 27], [43, 27]]
```

figure 11. Vertical Search

The second major design change was to change from assumed rectangular search area to polygonal search areas seen in figure 12. To do this it requires dynamic bounds definitions of the polygonal shape in the direction of travel to maintain the boundaries of the given area. This is done by finding the intersection of the line in the direction of travel and the line of the shape that comprises the bounds in that direction.

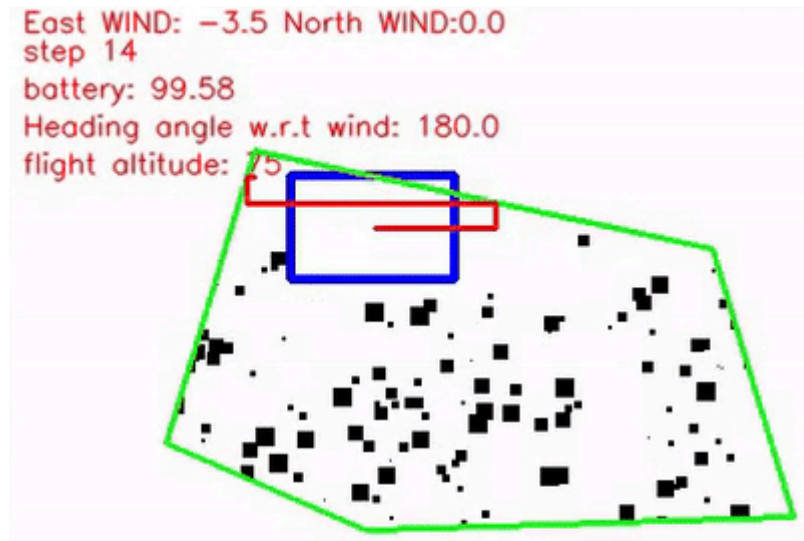


figure 12. Polygonal Search

The third major design change was to change from a singular defined search area to multiple. This allows the user to use the image preprocessor to define multiple search areas in one area, removing areas that are either dead space or unwanted. This would result in multiple areas to search in order and require multiple design changes to implement. The first of these sub-changes is to find the starting and ending location of a polygonal shape within a search area instead of assuming the start at the origin of the shape and the end being when we have found all our anomalies. The second sub-change was to find an efficient way to move from the end of one search area to the beginning of the next. This was done by moving in the delta x and delta y direction from the end point to the start point until at the starting location of the next shape.

The fourth change in design was to draw our path through the shape as we progressed. This was done by reusing a method in our preprocessor image script to draw lines in a cv2 window.

The fifth change in our design was to move the world_generator out of the environment and into its own stand alone script. This allowed us to generate worlds for both complete coverage and reinforced learning as well as give us more control on what variables were used to generate the world.

The sixth change in our design was to make all the systems work together from the command line. Randomly generating a world, processing it, and then searching it took three different command line scripts. If we needed to change any variables in the code we had to change their hardcoded values within one of our four main files. By making each file work together using flags in the complete coverage script, we can change variables and run different sequences of commands with one command. This helped aid in our testing when we changed components.

In both figures, the red box depicts the smallest rectangle containing the masked image, and the red arrows represent the future path of the drone. In figure 13, the drone covers a shape in a horizontal bias search. In figure 14, the drone is midway through its search, displaying its previous pathing. In figure 15, the image shows the full complete path of the drone for a polygonal search area.

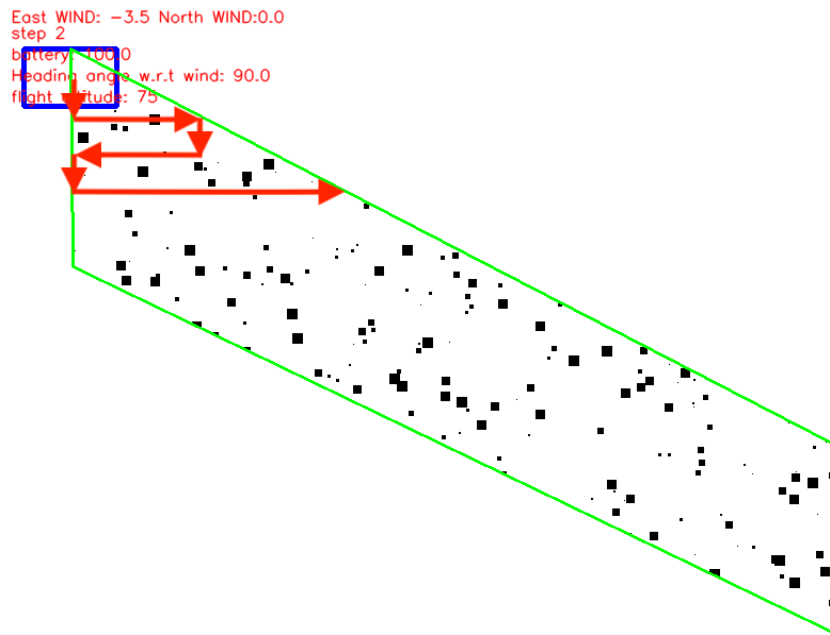


figure 13. Path to be taken by drone

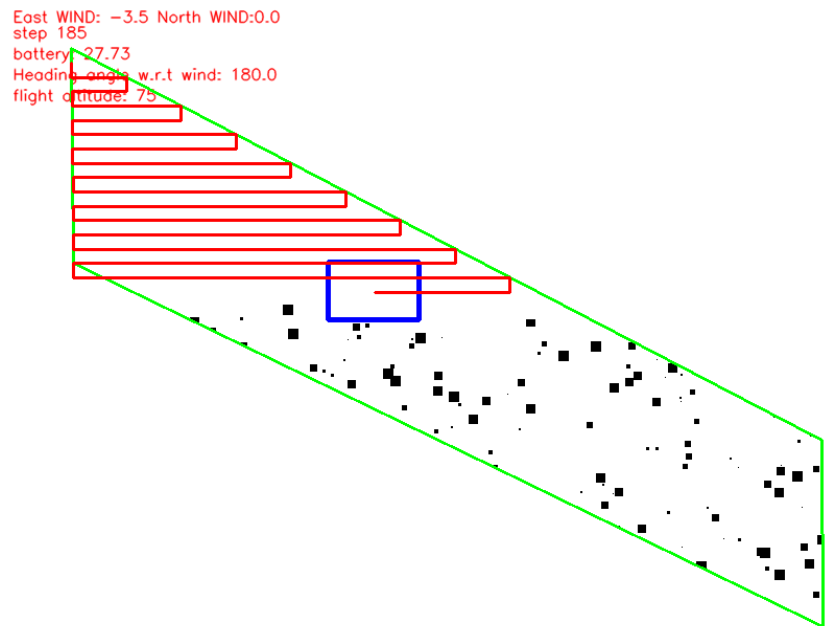


figure 14. Drone's path midway

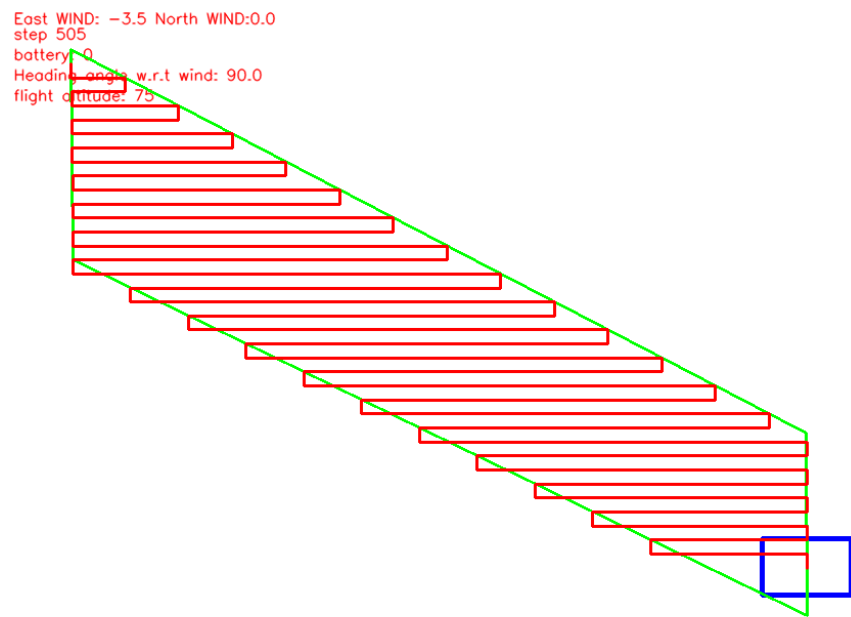


figure 15. Complete path taken

3.6.2 Image Preprocessor

The image preprocessing script started life as a simplified masking program. The user defined a series of points on an image using cv2 to create a singular polygonal shape that contained all the user's desired search area. The script limited the user to shapes with more than 3 points and would only define a singular shape. If a point was made in error by the user the script would need to be re-run.

As we moved from a low fidelity pixelated world to the high fidelity fly over images taken of a soccer field, seen in figure 16, we discovered that the functionality of the image preprocessor was lacking. The preprocessor was changed to implement definition of multiple shapes within the given image, the ability to erase previously defined points was added, and support for larger images were added.

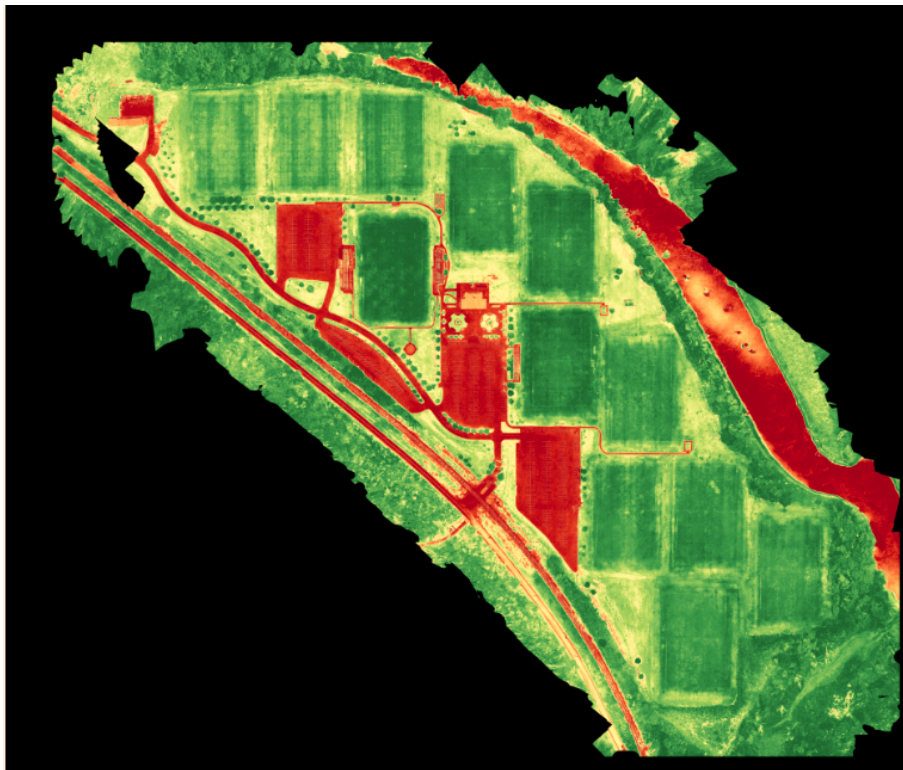


figure 16. Unmasked Image

The largest hurdle for the second version of the preprocessor was the added support for larger images of high resolution. The soccer field image we were given was a few gigabytes in size and required a large amount of RAM to load and process. To overcome this hardware limitation found on most modern day laptops we downscaled a copy of the image given by a factor based on the size of the original image. This

allowed the image to be loaded in a more RAM friendly way while maintaining the original image's granularity.

The cv2 library already supported zoom which allowed the user to zoom into the image to get more granular in their definitions of the shapes. Once the shapes were defined by the user, they were upscaled by the same factor we used to downscale the image and overlaid onto the original and masked.

The preprocessor would then save a masked copy of the image in its original format seen in figure 17, a numpy array of the masked image representing the entire image, and a numpy array that contained a list of shapes composed of the user defined points.

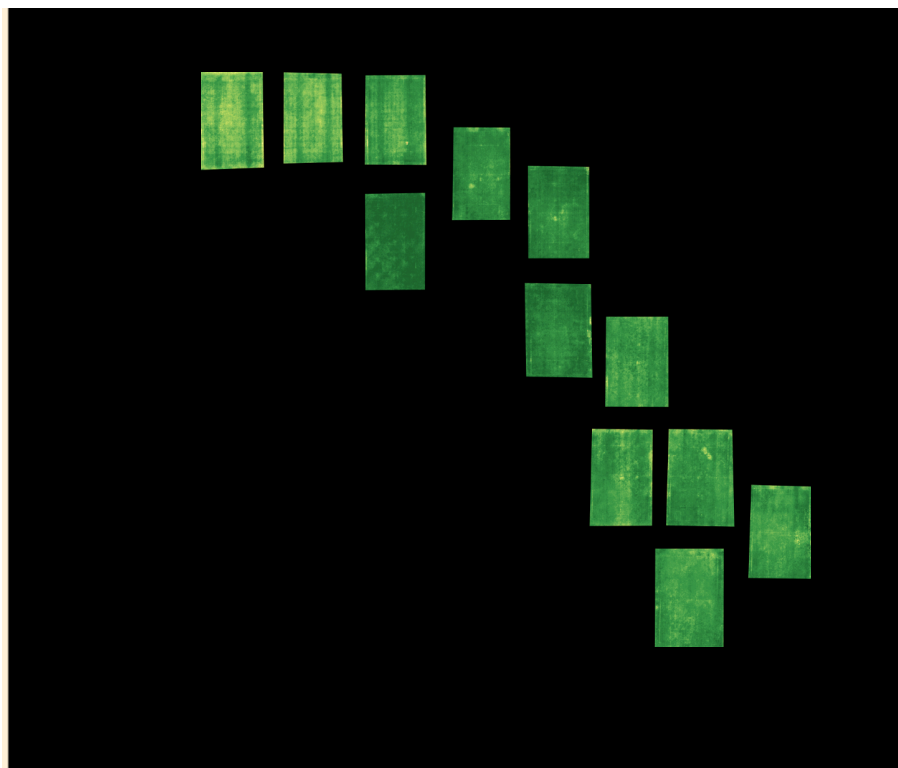


figure 17. Masked Image

3.6.3 Simulation Environment & Reinforcement Learning

As stated in 3.2.2, the aspects of the environment were changed throughout the reinforcement learning implementation process. The different factors changed include:

Action Space:

The action space of the model refers to the range of possible actions that can be carried out by the reinforcement learning model. It is a required variable from Gymnasium.

The action space started as a continuous 3x1 vector, each value in range (-max_velocity, +max_velocity). Each entry in the vector represents the distance to be traveled in each direction in an x, y, z space for a single step. Per advice from the client, the continuous action space introduced unnecessary complexity and inefficiencies to the simulation. The model was required to generate 3 numerical values which were difficult to correlate to received rewards.

The client suggested a discrete action space instead, with 4 actions available: up, down, left, and right. The drone will always use its maximum movement speed which is ideal for exploring worlds. It also eliminates the ability to move vertically, meaning a fixed height must be set at the beginning of the simulation. Reducing the dimensionality of the action space requires less training time for the reinforcement learning model to learn the correlation between actions and observations. This acts as the current action space.

Observation Space:

The observation space refers to information collected from the environment by the reinforcement learning model after each step. Observations act as the inputs to the model, which outputs an action for the agent to perform.

Many different variables were attempted including: the entire view from the drone, current location, x, y wind speeds and battery level. Each value is normalized between 0 and 1. None of these variables resulted in much success with the model.

The current observation space is a result of the current action space. 4 numbers; each an average of the pixel value from each side of the drone's view. The thinking is the model learns to correlate one of its four movements with one of the four averages. This has shown success in the line following images, but not in randomized images.

Reward Function:

After each step the model is given a “reward”, scoring the model’s choice of action. This allows the model to correlate an appropriate action for a given observation.

The calculation of the reward can be engineered to encourage desired behavior of the model. The reward function has been a linear combination of multiple different variables, each weighted by a coefficient. Variables include: anomalies found in last step, cost of movement, penalty for finding no anomalies, penalty for attempting to move out of bounds, and exploring new parts of the map.

The best reward function for learning to follow a line was found to be:

$$\text{reward} = 10(\text{anomalies found}) - 500(\text{if no anomalies found}) - 5(\text{movement cost})$$

3.6.4 Scope of Project Change

Due to unforeseen complexity and time constraints, the scope of the project has been reduced to finalizing the simulations. This means there will not be an implementation on a physical drone.

3.7 Proposed Design and Design Iterations

The final design should include an image preprocessor, complete coverage algorithm, and trained reinforcement learning model.

3.7.1 Design 0 - Client’s Existing Code

When initialized, the drone environment pulls information from a configuration file: world area described in a 3d plane as x, y, and z, a wind speed, a battery life, and a number of rewards. The environment runs a world generator function that sets up the world with an area of x by y and a height of z (start and init function). It then populates the ground with a predefined number of seeds.

The code enters into a loop using functions defined by OpenAI Gymnasium and tailored to our project by us to search out the rewards in a complete coverage algorithm. The algorithm functions by starting in an initial location and sweeping right to left with some predefined overlap searching for rewards (reset is called at the end of each step, step being one movement of the drone). When it has found a reward, it will remove it from the map and continue on until it has reached the end of possible movement in the world or has run out of battery life (stop function). The wind speed increases battery life usage if the drone moves against the wind.

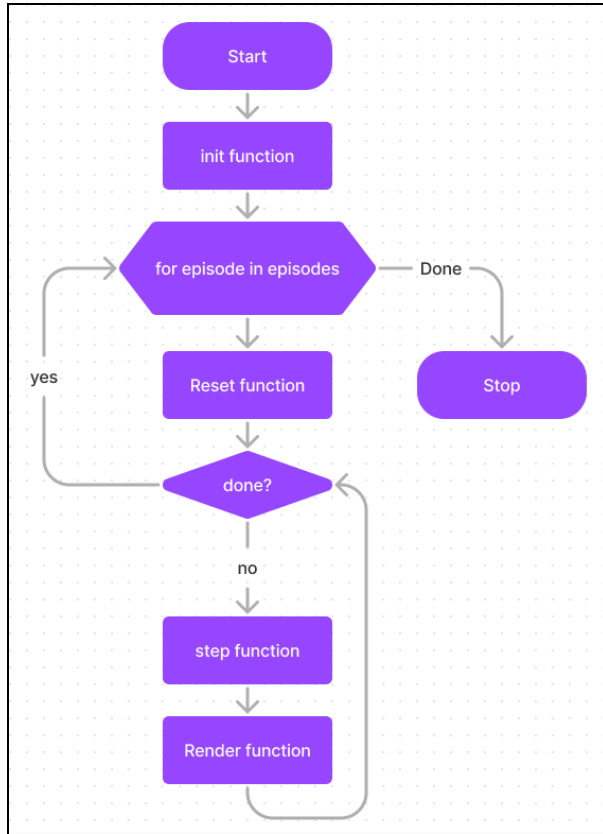


figure 18. Simulation Flowchart

Reinforced Learning was not used at this stage in the design. Its complete coverage algorithm will serve as the baseline for reinforcement learning algorithms in the future. The user is only able to search for objects by having the drone cover a specified area. This works, but is far from optimal.

3.7.2 Design 1 - Improved Complete Coverage

This design has an improved complete coverage algorithm described in figure 17. It runs over masked images in a semi-optimal manner. The polygon shapes come from an image masking script, called “Image Editor Script” in figure 17.

With these two files, the design is now more versatile than before and has the potential to be much more useful in practice by allowing multiple search areas, vertical search, horizontal search, and polygonal search area support that all process and optimize the search path to reduce the size of the path while maintaining complete coverage of the area.

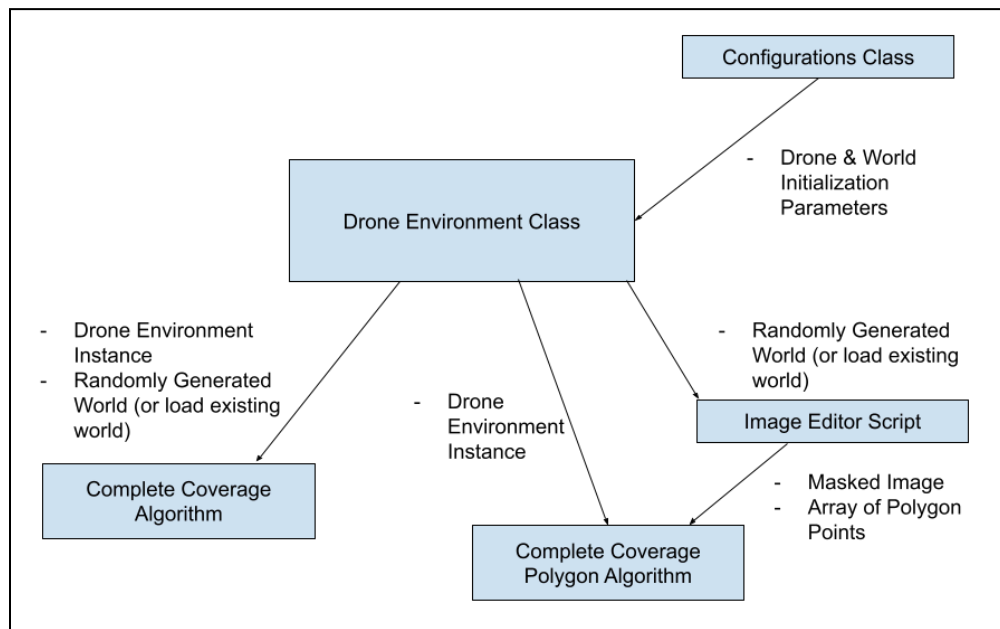


figure 17. Design 1 File Hierarchy

3.7.3 Design 2 - Reinforcement Learning Introduction

The client’s current implementation of reinforcement learning includes the continuous action space. The drone’s view of the image, battery, wind, and location in its observation space, and a reward function consisting of the quantity of anomalies found and a movement penalty. The version of the environment was unable to effectively search random environments and unable to follow lines.



figure 18. Improved Line Follower

3.7.4 Design 3 - Reinforcement Learning with Exploration Reward Function

This version of the reinforcement learning environment attempted to utilize a separate array monitoring which portions of the image had been explored or not. The thinking being that the model would be encouraged to explore unseen parts of the map.

This addition to the reward function didn't change the performance of the learned models significantly.

3.7.5 Design 4 - Current Environment and Complete Coverage Algorithm

Reinforcement Learning Model:

The current reinforcement learning model uses a discrete action space, a simpler observation space and a reward function similar to Design 0 (3.6.3). These changes to the reinforcement learning model's action and observation spaces allows the model to successfully follow lines on images. However, the model still doesn't perform well on randomized images.

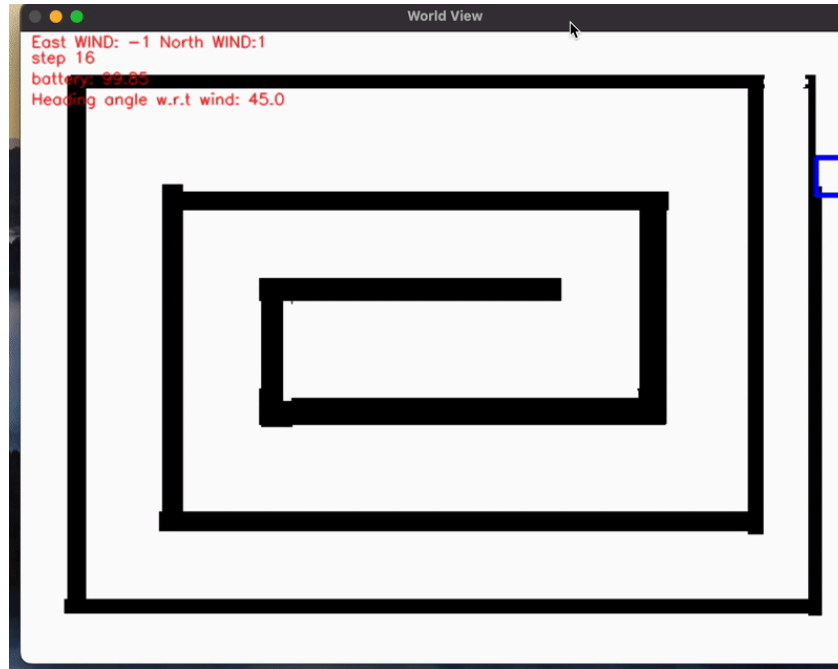


figure 19. Line Follower Model



figure 20. Random Image Model

Complete Coverage:

The current complete coverage model has been improved to efficiently cover multiple polygons when the image has been masked properly. The model can generate a random world of a set size or load a saved world, process and mask that world into

multiple defined search areas, dynamically choose a search bias (vertical or horizontal) dependent on the search area's size, alter it's pathing bounds dependent on the search area's size to stay within the area, and move between search areas all while maintaining complete coverage of all search areas defined.

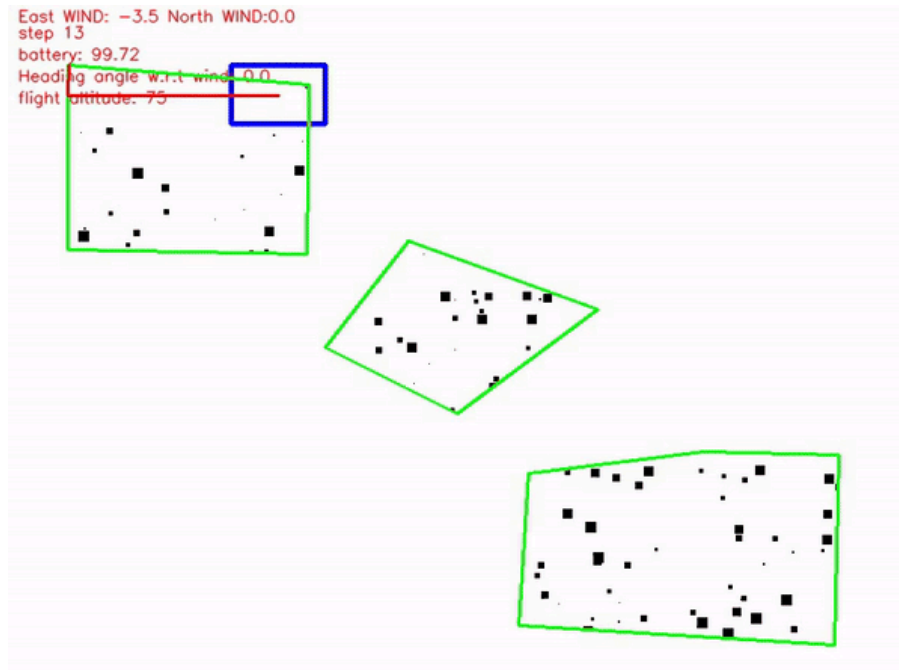


figure 21. Complete Coverage on Multiple Polygons

3.8 Design Analysis

The originally proposed design maintained that there would be a complete coverage algorithm and reinforcement learning model competing, but the search algorithms themselves changed a lot over the course of the semester. The environment also went through simplifications in order to get a reinforcement learning model that worked properly.

4 Testing

4.1 Unit Testing

Systems can be isolated and will be tested using the assert function from the “unittest” library. The program will be broken down and tested to ensure the model runs effectively and the requirements are properly met. An additional testing script file will be written to perform these tests. Systems to be tested are:

- A. Battery
 - Simulation ends when battery = 0
 - Initialized as a 100
- B. Move Cost
 - Equations for calculating cost from drone speed, wind speed, wind angle, drone drag coefficients
- C. Movement
 - Drone adheres to overlap percentage
 - Drone's location does not exceed world boundaries
- D. Frame Capture
 - Correct resolution
 - Scaled correctly based on height and FoV from camera
- E. Reward Collection
 - Does not double count rewards
 - Upscaled rewards not counted more than once
- F. World Padding
 - Sufficient as to not cause simulation error
- G. World Rendering
 - Rendered window is readable and displays useful information
- H. Map Saving/Loading
 - Worlds can be saved and selectively loaded as parameter from command line
 - Saved files are compatible with all algorithms and the image masking script
- I. Drone Environment Initialization
 - Drone and world parameters are loaded from a configuration file, are initialized by hardcoded values or overwritten in the environment file.

4.2 Interface Testing

Readme file which describes how to install necessary python packages and how to run each algorithm from the command line. All commands should be tested on Windows, MacOS, and Linux to ensure all packages are compatible and work as intended.

The generated window displaying the simulation should properly show the area being searched, known rewards, unknown rewards, and the drone's search area. This interface can be tested by running a simulation and verifying that the visuals are being properly displayed.

The image masking interface should properly display the world and allow for easy interaction with the mouse. The interface will allow the user to choose the parameters of the searched area by cropping outside of the defined points placed by mouse interaction. The interface will allow for as many points placed as desired and process all points on the press of the enter key. This will be tested in the same manner as the simulation window.

4.3 Integration Testing

The most critical piece of integration is the drone environment's interaction with OpenAI's Gymnasium API. If this is not done properly RL algorithms from OpenAI will not function properly, if at all. The integration of the image masking script will also need to be integrated and tested.

The first challenge will be to get an imported algorithm to run properly. To do this, OpenAI's Gymnasium API will need to be extended correctly by the drone environment class. Once this is complete, the unit tests from section 4.1 will be used to ensure expected functionality of methods in the new algorithm.

The image masking script needs to be able to load saved worlds from a .npy file, edit them as a .png, and then export the results back into a .npy format. To test this, the file can be loaded by an algorithm to verify if file conversion was performed properly.

4.4 System Testing

All systems need to pass all unit tests and integration tests, and be able to run without errors in different scenarios such as: square or polygon shaped world, many/few rewards, and large/small area. Each algorithm should output relevant metrics, formatted for readability, in the terminal and as a file.

4.5 Acceptance Testing

After the team conducts testing as listed in sections 4.1 - 4.5, testing can be performed with the client to ensure requirements are being met. Additionally, all testing will be uploaded to the shared Git repository owned by the client to be viewed and run at their leisure.

4.6 Results

A test was run to compare performance of the reinforcement learning model and complete coverage algorithm on a randomized image (figure 22.).

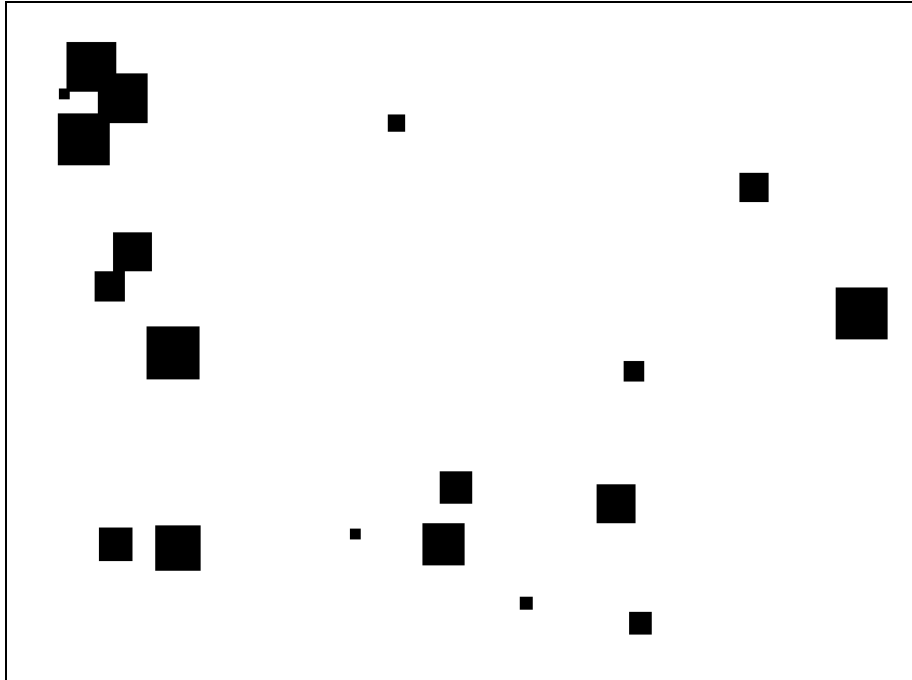


figure 22. Test Image

Testing Conditions:

- 1px = 1m²
- 1step = 1s
- 21475 total anomalies
- 600m x 800m
- 8m/s drone speed
- 50m height
- 40°/60° camera FOV
- 36m x 57m area coverage
- 0 m/s wind
- No battery

Reinforcement Learning Model Performance:

- 20% - 50% Anomaly Collection
- 4300 - 10750 Anomalies Found
- 500 Steps
- 8.6 - 21.5 Anomalies/Step

Complete Coverage Performance:

- 100% Anomaly Collection
- 21497 Anomalies Found

- 1750 Steps
- 12.284 Anomalies/Step

The trained RL model can usually find anomalies faster than CC, but struggles to find a large percentage of the total anomalies. The map most likely favors the reinforcement learning model slightly more because the rewards are distributed rather sparsely.

5 Professionalism

5.1 Areas of Responsibility

Contextualizing Professionalism Paper	IEEE Code of Ethics [13]
Work Competence	<p>“...maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations...”</p> <p>IEEE encourages its members to continually develop one’s skills and to provide the highest quality service possible. It also states that one should know their own limits, and to not take on projects exceeding their capabilities.</p>
Financial Responsibility	Mention of providing products or services at reasonable cost is absent from IEEE’s Code of Ethics
Communication Honesty	<p>“...disclose promptly factors that might endanger the public or the environment...”</p> <p>“...seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others...”</p>

	<p>Working ethically means that all parties should be aware of aspects of a project negatively affecting parties. Provide constructive feedback and gracefully accept criticism.</p>
Health, Safety, Well-Being	<p>“...hold paramount the safety, health, and welfare of the public...”</p> <p>Services for public use should prioritize public users’ safety, health and well-being.</p>
Property Ownership	<p>“...seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others...”</p> <p>Do not use another’s ideas as one’s own and give credit where it is due.</p>
Sustainability	<p>“...strive to comply with ethical design and sustainable development practices...”</p> <p>Where possible, make environmentally conscious decisions to promote sustainability.</p>
Social Responsibility	<p>“...hold paramount the safety, health, and welfare of the public...”</p> <p>Engineers ought to aim to better society and improve the lives of people through their work.</p>

figure 23. Comparison of Contextualizing Professionalism and IEEE Code of Ethics

5.2 Project Specific Professional Responsibility Areas

Contextualizing Professionalism Paper	Application to Project/Our Team's Performance
Work Competence	<p>Our team does its best to decide on the best possible solutions for problems, then implement them in alignment with guidelines outlined in this document.</p> <p>Fair</p>
Financial Responsibility	<p>Our team's work does not contribute to the cost of the project and is not financially compensated. The team may be recognized in a final paper.</p> <p>Not applicable</p>
Communication Honesty	<p>Communicating honestly with our client is important since we are effectively assisting with his research. Each week we honestly discuss our team's accomplishments and shortcomings, intending to promote honest communication between both parties.</p> <p>Good</p>
Health, Safety, Well-Being	<p>Application of this research could be used for maintaining infrastructure or saving lives in search and rescue missions. It could also be used for indirect human harm in military applications. There are applications for this technology to be used for societal good and bad. As junior researchers, we feel there isn't much we can do to influence how this technology will be used.</p>

	Not Applicable.
Property Ownership	When using or citing ideas from sources, our team credits the author with its use. Excellent
Sustainability	One goal of the project is to find a better way to perform aerial searches with drones to minimize energy usage. Our team has done a poor job of this so far, due to the fact that we have not fully tested the battery usage portion of the simulation. Poor
Social Responsibility	There are concerns that the drone may capture images of an area without the property owner’s consent. We have begun to investigate this through the use of the image masking script. Although the complete coverage algorithm still takes pictures of the masked image currently (as of December 2023). Fair

figure 24. Contextualizing Professionalism Applied to the Team

5.3 Most Applicable Professional Responsibility Area

The area our team has the chance to act most professionally in is Work Competence. Our team’s main contribution is to our client’s research [11], which means high quality work should be prioritized. Honest Communication is note-worthy as well, seeing as we are working closely with our client to accomplish tasks.

6 Closing Material

6.1 Discussion

It took a lot of time for the team to start making meaningful progress on the project. In our opinion, this project was very technically challenging and required a lot of research and trial and error before a meaningful understanding of the goal could be realized.

We were able to understand enough aspects of reinforcement learning to make changes that resulted in desired performance. We were able to recreate, to some degree, the industry standard solution of complete coverage in simulation. Unfortunately, we realized these things too late in the project's lifecycle and didn't have enough time to finish our goal of successfully running on real images.

6.2 Conclusion

Although we didn't accomplish our goal, we made meaningful progress on the client's code, and accomplished some of our other milestones. Given another month, we believe we could come close to being able to run on real images.

6.3 References

- [1] Python, "The Python Standard Library — Python 3.8.1 documentation," *Python.org*, 2020. <https://docs.python.org/3/library/index.html>
- [2] "torch — PyTorch 1.12 documentation," *pytorch.org*. <https://pytorch.org/docs/stable/torch.html>
- [3] "Gymnasium Documentation," *gymnasium.farama.org*. <https://gymnasium.farama.org/index.html>
- [4] "Stable-Baselines3 Docs - Reliable Reinforcement Learning Implementations — Stable Baselines3 1.2.0a2 documentation," *stable-baselines3.readthedocs.io*. <https://stable-baselines3.readthedocs.io/en/master/>
- [5] "IEEE SA - IEEE 1936.1-2021," *IEEE Standards Association*. <https://standards.ieee.org/ieee/1936.1/7455/>
- [6] "[USC02] 49 USC 44809: Exception for limited recreational operations of unmanned aircraft," *House.gov*, 2018.

<https://uscode.house.gov/view.xhtml?req=granuleid:USC-prelim-title49-section44809&num=0&edition=prelim>

- [7] *Oreilly.com*, 2023.
<https://www.oreilly.com/radar/wp-content/uploads/sites/3/2019/06/image3-5f8cbb1fb6fb9132fef76b13b8687bfc.png> (accessed Dec. 03, 2023).
- [8] T. M. Cabreira, L. B. Brisolará, and P. R. Ferreira Jr. , “Survey on Coverage Path Planning with Unmanned Aerial Vehicles,” *Drones*, vol. 3, no. 1, p. 4, Mar. 2019, doi: <https://doi.org/10.3390/drones3010004>.
- [9] “Overlap & Flight Pattern | Drone Data Processing,” *Aerotas: Drone Data Processing for Surveyors*. <https://www.aerotas.com/overlap-flight-pattern>
- [10] E. Galceran and M. Carreras, “A survey on coverage path planning for Robotics,” A survey on coverage path planning for robotics,
<https://www.sciencedirect.com/science/article/abs/pii/S092188901300167X>
(accessed Oct. 19, 2023).
- [11] A. Niaraki, J. Roghair, and A. Jannesari, “Visual exploration and energy-aware path planning via reinforcement learning,” arXiv.org,
<https://arxiv.org/abs/1909.12217> (accessed Oct. 19, 2023).
- [12] J. Shi and M. Zhou, “A data-driven intermittent online coverage path planning method for AUV-based bathymetric mapping,” MDPI,
<https://www.mdpi.com/2076-3417/10/19/6688> (accessed Oct. 19, 2023).
- [13] IEEE, “IEEE Code of Ethics,” *ieee.org*, Jun. 2020.
<https://www.ieee.org/about/corporate/governance/p7-8.html>
- [14] G. van Rossum, B. Warsaw, and N. Coghlan, “PEP 8 – Style Guide for Python Code | *peps.python.org*,” *peps.python.org*, Jul. 05, 2001.
<https://peps.python.org/pep-0008/>
- [15] Amir Niaraki and Jansel Herrar, “drone_gym”, Nov. 8, 2022.
https://github.com/AmirNiaraki/drone_gym

7 Appendices

N/A

8 Team Contract

Team Name: RLPIIUAV

Team Members:

- | | |
|----------------------|-------------------|
| 1). Jesse Gillingham | 2). Andrew Sailer |
| 3). Akash Setti | 4). Cody Draper |

Team Procedures:

1. Meetings: Thursdays 7pm, Face-to-Face in SICTR 4201 if possible. Weekly.
2. Communication: Discord and Email.
3. Decisions: Majority Vote or Compromise.
4. Record Keeping: Andrew Sailer will take meeting notes and post them in #meeting-minutes channel in discord. Other relevant documents will be posted in #resources.

Participation Expectations:

1. Expected weekly meetings with mandatory attendance, along with day to day communication through the discord involving the project.
2. Each member will be assigned work for the period of time set in the meeting. Each team member is to finish their section of work in that assigned time. If the team member cannot finish in the allotted time they must inform the rest of the team.
3. Each team member is expected to be very communicative regarding their work
4. Decisions will be posted in Discord for the entire team, and will be respected afterward.

Leadership:

1. Leadership roles for each team member:

We all hold equal responsibility for keeping each other accountable.

2. Strategies for supporting and guiding the work of all team members:

Agile Project Management - discuss what we worked on in weekly meetings, problems we're having, and what we plan to work on in the future.

3. Strategies for recognizing the contributions of all team members:

GitLab will be involved in the project. Based on the commits, the team will be able to see the work for the specific team member.

Collaboration and Inclusion:

1. Jesse Gillingham: Hardware, Embedded Systems,
Andrew Sailer: Hardware, Control Systems, Some Programming.
Akash Setti: Software Development, Data Science, Machine Learning
Cody Draper: Software Development, Embedded System, Machine Learning
2. Strategies for encouraging contributions and ideas from all team members: We will have group meetings and discussions at our major points.
3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)

Goal-Setting, Planning, and Execution:

1. Team goals for this semester: Get familiarized with the code base. Set up the development environment and train RL models. Experiment with different RL Algorithms using stable baseline3.
2. Strategies for planning and assigning individual and team work: Assign work to team members during team meetings.
3. Strategies for keeping on task: Use Gitlab issues and milestones to keep track of timelines and maintain a steady flow of work.

Consequences for Not Adhering to Team Contract:

1. How will you handle infractions of any of the obligations of this team contract? Any infractions will result in a team vote on the specific consequences.
2. What will your team do if the infractions continue?

We will inform the person who determines the grades.

-
- A. I participated in formulating the standards, roles, and procedures as stated in this contract.
 - B. I understand that I am obligated to abide by these terms and conditions.
 - C. I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1). Jesse Gillingham

Date: 9/7/23

2). Andrew Sailer

Date: 9/7/23

3). Akash Setti

Date: 9/7/23

4). Cody Draper

Date: 9/7/23