Employing Reinforcement Learning-powered drone swarms to locate shipwrecked individuals

Jorás Custódio Campos de Oliveira[a], **Pedro Henrique Britto Aragão Andrade**[a], **Renato Laffranchi Falcão**[a], **Ricardo Ribeiro Rodrigues**[a]

[a]*Computer Engineering, INSPER*

Prof. Fabrício Barth

Abstract—This project explores the effectiveness of Reinforcement Learning (RL) algorithms in search-and-rescue (SAR) operations, specifically using drones to locate persons-in-water (PIW). It is intended to investigate various hypotheses regarding the performance of RL, convergence rates, mutli-agent coordination and information exchange for solving this problem.

keywords—*Reinforcement Learning, DQN, PPO, Search and Rescue, Multi-agent*

1. Introduction

The issue of missing PIW is as old as humankind itself, and given the chaotic nature of the ocean, SAR operations have never been optimal, with limitations on the human ability ranging from the creation of proper search paths to visibility and recognition of PIW.

With this in mind, this project aims to research a modern solution to the problem. It is believed that employing artificial intelligence (AI) on drones with the purpose of searching for castaway individuals may significantly improve the efficacy of operations and reduce the time needed to find and save PIW. The application of RL techniques is theorized to produce optimal solutions, because it is hypothesized that the maximization of reward is sufficient to foster generalization abilities [7].

2. Environment

The environment is a 2D grid with a probability matrix representing the area where there is a probability of containing a PIW. The probability matrix is based on a Gaussian distribution and a dispersion increment, which controls the rate of expansion of the Gaussian. The direction of movement is controlled by a vector representing ocean currents. The PIW's movement is influenced by the probabilities in the matrix, selecting a pseudo-random direction weighted by the probabilities of adjacent cells and an independent movement vector.

The environment states are represented as a tuple of two boxes: one for the drone's position (x, y) and another for the probability matrix. This allows the agents to use the probabilities and their own positions to decide the best action, facilitating the orchestration of complex search patterns to find the PIW.

The action space of the environment is discrete, consisting of 9 actions: moving in the cardinal and inter cardinal directions and searching the current cell. With a Probability of Detection (POD) of 1.0, the environment is deterministic, but with lower values, it becomes stochastic.

The environment is inherently a sparse reward problem, as the goal is only reached when the PIW is found. Below is a formal definition of the reward function.

$$R(Ts) = \begin{cases} 1 + 1 \cdot (1 - \frac{Ts}{Ts_{limit}}), & \text{if found} \\ 0, & \text{otherwise} \end{cases}$$

Being, the Ts the current time step, and Ts_{limit} the limit of time steps for the current SAR mission.

All instructions for using the environment, along with detailed documentation and examples, are available in a Python package named Drone Swarm Search Environment (DSSE) [8].

3. Methodology

To evaluate the efficacy of various search algorithms and strategies, the team used three algorithms: a baseline greedy heuristic, Deep Q-Network (DQN) [3], and Proximal Policy Optimization (PPO) [4]. Both DQN and PPO, supporting centralized and decentralized learning configurations, were implemented using the RLlib library [5]. DQN and PPO were chosen to represent the main value-based and policy-based algorithm families, respectively. All scripts for training and testing the agents are available in a GitHub repository [11].

The baseline algorithm, which does not use Reinforcement Learning, calculates agents' actions based on the probability matrix. Each drone is assigned to search in the cell with the highest probability, focusing solely on exploitation. This simple, non-stochastic algorithm was used for comparison to better understand the efficacy of RL approaches.

The neural network architecture chosen was a multi-input convolutional neural network (CNN). It comprised a CNN branch that processed the full matrix and a fully connected (FC) branch that processed the agents' positions. These branches were then concatenated and passed through a joint FC layer.

In PPO, the network produced two outputs: a policy branch determining action probabilities and a value branch for the critic to update policies. In DQN, the output consisted of Q-values for each action in a given state.

In order to comprehensively evaluate the efficacy and learning capabilities of various RL algorithms within the environment, we have formulated and systematically tested several hypotheses. Through rigorous experimentation, we aim to uncover the strengths and weaknesses of RL approaches when compared to baseline algorithms. The findings from these experiments are anticipated to offer valuable insights, showcasing the practical advantages of RL in this context and contributing to the optimization of search strategies using autonomous agents.

3.1. Experiment Setup

All experiments were conducted in the environment described previously using the following base configurations:

- **Grid Size:** 40x40
- **Person in Water (PIW):** 1
- **Dispersion Matrix:** Centralized in the middle of the grid
- **Agents (Drones):** 4, positioned on the four edges of the map
- **Dispersion Increment:** 0.1

3.2. Hypotheses

Hypothesis 1: *Does reinforcement learning (RL) outperform a greedy strategy when the person in water moves away from the region of highest probability?*

To test this hypothesis, we will compare the performance of agents controlled by centralized RL algorithms, such as PPO and DQN against a greedy policy under the specified base configuration. Performance metrics will include search efficiency, time to locate the PIW, and adaptability to the PIW's movement.

Expected Outcome: RL algorithms are anticipated to demonstrate superior adaptability and performance over the greedy approach, particularly as the PIW's location deviates from predicted regions.

Hypothesis 2: *Do agents trained with independent neural networks*

achieve faster convergence compared to those trained with shared networks?

We will analyze the learning curves of agents in the base configuration using both a centralized learning approach and an independent learning approach. Metrics such as the rate of convergence and final policy performance will be evaluated.

Expected Outcome: Preliminary data suggests that the independent approach may converge faster than centralized training. However, literature indicates [9] that centralized learning can converge more quickly and achieve better policy performance. This experiment aims to determine the optimal configuration for our problem and setup.

Hypothesis 3: *What is the impact of trajectory information sharing among agents on search operations' effectiveness?*

Using the centralized version of the PPO algorithm, we will test two approaches in an environment with large dispersion:

1. Modifying observations with the agent's trajectory and directly altering probabilities on the matrix.
2. Utilizing a Long Short-Term Memory (LSTM) network to handle sequential information regarding agents' position trajectories.

Expected Outcome: Sharing historical search data among agents is predicted to enhance search efficiency through a more coordinated and informed approach. The use of LSTM is expected to boost performance in partially observable environments [2].

Hypothesis 4: *Can agents be organized to ensure full coverage is done in minimal time, while prioritizing the cells of highest probability, thereby maximizing coverage efficiency and minimizing search time?*

This hypothesis will be explored in a coverage environment with 2 drones. The focus will be on evaluating how RL can optimize drone movements to avoid redundant searches and prioritize high-probability regions.

Expected Outcome: RL is expected to effectively organize drone movements, reducing search mission time and prioritizing areas with a higher chance of containing PIWs.

Hypothesis 5: *How do agents develop their search patterns when there is more than one PIW?*

We will maintain the base configuration while increasing the number of PIWs to four. The search patterns and task distribution among drones will be analyzed.

Expected Outcome: The drones are expected to organize themselves to divide the search areas efficiently, increasing the coverage area and enhancing the likelihood of detecting PIWs, thereby improving the overall success rate.

3.3. Data Collection and Analysis

The validation data was collected by running the SAR mission simulation 5,000 times using the greedy algorithm, and the trained DQN and PPO algorithms across two neural network modes, within the same environment configurations. The data included rewards received, the number of actions taken, and whether the agents found the PIW in each trial.

The training data was analyzed by calculating the moving average of rewards and actions. The size of the moving average window depended on the training batch, which was collected in parallel using RLlib and stored for later analysis. Learning curves, showing the increase in received rewards and the decrease in the number of actions over episodes, were used to evaluate model performance.

4. Results

The results for the Reinforcement Learning algorithms include the learning curves for each experiment done, as well as an analysis of their success rate and behavior. Each experiment was conducted based on the previously stated hypothesis.

All experiments to test the five final hypotheses were done with both DQN and PPO algorithms. These tests adhered to the previously mentioned base configuration, except for those related to hypothesis 5, which involved four PIWs instead of one.

The hyper parameters used for PPO for the experiments are described in table 1, while the hyper parameters used for DQN are described in table 2.

Table 1. PPO Hyper parameters used for the experiments.

Parameter	Value	Description
B	8,192	Training batch size
Lr	10^{-5}	Learning rate
γ	0.9999999	Discount factor
M	300	Stochastic Gradient Descent (SGD) minibatch
K	10	Number of SGD iterations

Table 2. DQN Hyper parameters used for the experiments.

Parameter	Value	Description
B	512	Training batch size
Lr	10^{-4}	Learning rate
γ	0.9999999	Discount factor
U	500	Update target network every U steps
e_0	1	Initial epsilon for ϵ-*greedy*
e_f	0.1	Final epsilon for ϵ-*greedy*
T	400,000	T time steps for epsilon decay from e_0 to e_f

The results of each experiment are presented alongside their respective hypotheses, accompanied by considerations and analyses.

Hypothesis 1: *Does the RL algorithm outperform the greedy strategy in scenarios where the PIW can move out of the region of highest probability?*

Tests using RL algorithms on a 40x40 grid with dispersion increments of 0.1 and 0.05 show that RL outperforms the greedy approach in more complex environments. Figures 1 and 2 display PPO learning curves for these configurations, and figure 3 compares PPO and DQN learning.

PPO learns a better policy and much quicker than DQN, achieving rewards between 1.4 and 1.75, while DQN reaches only 0.4. PPO training took 3 hours, whereas DQN took 23 hours. PPO's advantage is attributed to its action probability distribution, allowing better exploration on states the agent has less knowledge about. Thus, PPO was chosen for further tests.

Tables 3 and 4 show algorithm performance for 5,000 tests in each configuration.

PPO significantly outperformed the greedy approach, with a 75.44% success rate versus 35.84% for 0.1 dispersion increment and 83% versus 50.18% for 0.05. RL algorithms can learn complex search patterns, suitable for unpredictable SAR missions.

Hypothesis 2: *Does Decentralized Training Improve Convergence Time and Stability?*

To test this hypothesis, we trained a PPO algorithm using a decentralized approach, with each agent having its own policy, as opposed to a shared policy among all agents. This idea stemmed from mixed

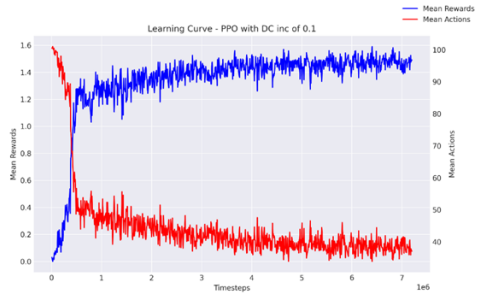

Figure 1. Learning curve for PPO on 0.1 dispersion increment configuration.

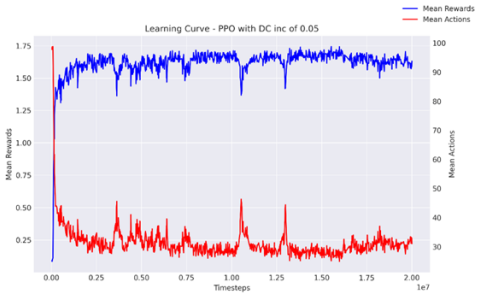

Figure 2. Learning curve for PPO on 0.05 dispersion increment configuration.

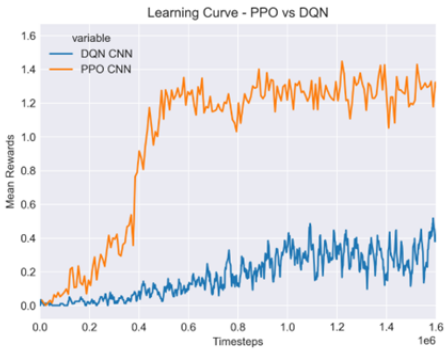

Figure 3. Learning curve comparison of PPO and DQN on 0.1 dispersion increment configuration.

Table 3. Comparison of algorithms for 0.1 dispersion increment.

Metric	PPO CNN	Greedy
Mean reward	1.34	0.59
Action mean	42.47	77.48
Action median	23	100
Success rate (%)	75.44	35.84

Table 4. Comparison of algorithms for 0.05 dispersion increment.

Metric	PPO CNN	Greedy
Mean reward	1.48	0.86
Action mean	35.91	65.07
Action median	22	94
Success rate (%)	83.00	50.18

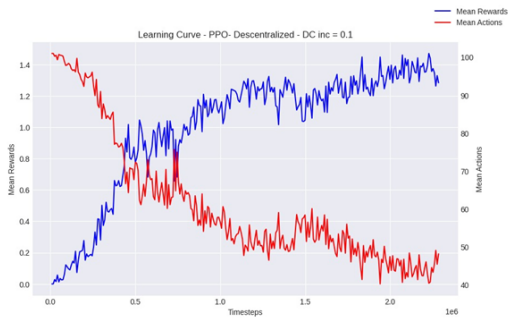

Figure 4. Learning curve for decentralized PPO and dispersion increment of 0.1.

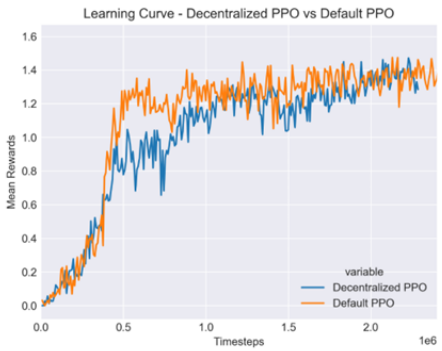

Figure 5. Learning curve comparison Centralized PPO vs Decentralized PPO.

findings in recent literature, with some studies indicating improvements in training time and stability, while others found no benefits or even detriments.

The training setup mirrored that of Hypothesis 1 with a 0.1 dispersion increment, differing only in the use of unique policies per agent. This allowed for direct comparisons with the standard PPO and baseline algorithm. Preliminary results suggested shorter training times and higher stability, but final results showed otherwise.

Decentralized training resulted in slower convergence, reduced stability, and inferior policy performance. Figure 4 shows the learning curve for decentralized PPO, while figure 5 compares centralized and decentralized PPO learning curves.

Both centralized and decentralized approaches reached a mean reward around 1.4, but the decentralized version took around two million steps compared to the centralized version's 500,000 to a million steps. Mean actions were around 40 for centralized and 50 for decentralized, indicating the decentralized approach was less efficient.

Table 5 presents the test results for both approaches, using the same validation tests as Hypothesis 1.

Table 5. Comparison of decentralized vs centralized PPO.

Metric	PPO Centralized	PPO Decentralized
Mean reward	1.34	0.94
Action mean	42.47	60.22
Action median	23	41
Success rate (%)	75.44	53.22

The centralized approach consistently outperformed the decentralized one, with higher mean rewards, lower action counts, and a higher success rate. Consequently, we decided to use centralized training for all subsequent hypotheses.

The longer training times for decentralized training can be at-

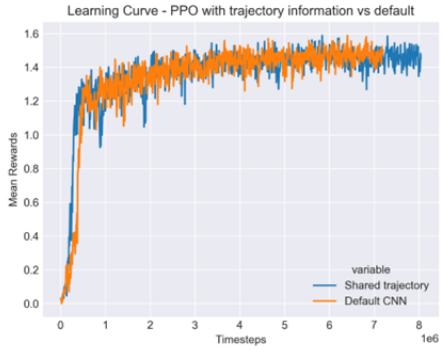

Figure 6. Learning curve comparison - PPO with trajectory information (matrix) vs default.

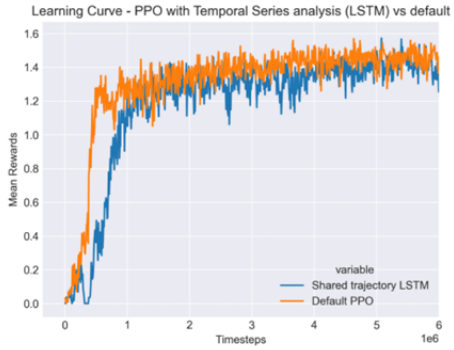

Figure 7. Learning curve - PPO with LSTM vs default network.

tributed to the additional resources required for multiple policies, each necessitating a separate neural network. This impacts hardware and time resources, as more networks need more time and memory. The lower mean reward is likely due to some agents learning sub-optimal policies, which significantly affects overall performance in a complex problem with few agents.

Hypothesis 3: *Does Search Trajectory Information Impact Search Results?*

For this hypothesis, we trained a centralized PPO algorithm using two approaches:

1. Trajectory-Based Probability Adjustment: Modifying the probability matrix based on the drone's trajectory, reducing cell probability after a drone searches it and gradually restoring it over time.
2. LSTM Layer Integration: Adding a Long Short-Term Memory (LSTM) layer to handle the sequential information of agent positions, replacing the second fully connected layer in the positions branch of the neural network.

Results from both approaches, shown in figures 6, 7, and table 6, indicate no significant impact on search performance.

As seen in figures 6 and 7, the learning curves for PPO with trajectory information, both matrix and LSTM, are similar to the default PPO configuration. Table 6 shows the statistics for 5,000 simulations for each approach, confirming no significant impact on search performance.

We hypothesize that the information in the environment's observation is sufficient for the algorithm's Markov Decision Process (MDP) to select optimal actions, rendering additional trajectory information redundant. It is speculated that trajectory information might be more efficient in larger-scale searches with more agents, where search area overlap is greater.

Table 6. Comparison with trajectory sharing methods PPO and default PPO.

Metric	PPO default	PPO trajectory matrix	PPO LSTM
Mean reward	1.34	1.35	1.36
Action mean	42.47	41.99	41.57
Action median	23	23	23
Success rate (%)	75.44	75.98	76.46

Figure 8. Learning curve for PPO optimizing coverage speed and POC.

Hypothesis 4: *Can Agents Organize to Maximize Coverage Area and Speed?*

To test this hypothesis, we used the Coverage Environment available in DSSE to extend state-of-the-art experiments [6] [10]. Agents were tasked with optimizing a NP-Complete problem [1], learning efficient solutions to cover the search area, minimize steps, and prioritize high-probability cells.

The experiment was conducted on a 9x9 grid with 58 cells having a Probability of Containment (POC) greater than 1, generated by a 2-hour drifting simulation from default initial points near Guarujá, São Paulo, Brazil. Two agents used a centralized multi-layer perceptron, which performed better than a default CNN due to the grid's small size.

As depicted in figure 8, trained agents learned to complete the search area efficiently, prioritize high-POC cells, and reduce the number of steps, averaging 34 steps to cover 58 cells. This demonstrated a significant parallelization, achieving 58.62% of the required steps, close to the ideal 50% speed-up.

This experiment aimed to extend previous research by increasing complexity and using a multi-agent approach. We used 58 valid search grid cells, compared to 19 in [10] and 30 in [6], to test if agents could autonomously learn and organize to parallelize tasks effectively. The results indicate that agents learned efficient behaviors and completed tasks faster than a single agent could. This highlights that reinforcement learning algorithms can handle multi-objective problems, learning tasks with multiple optimizations while prioritizing actions that yield the highest sum of discounted rewards.

Hypothesis 5: *How is the Search Pattern for Scenarios with More than One PIW?*

To test this hypothesis, experiments were conducted using the same control parameters as previous hypotheses, employing PPO with a CNN and a sparse reward scheme. The difference was the inclusion of four PIWs instead of one, to determine if agents adapt their behavior for multiple PIWs.

The learning curve in Figure 9 shows agents learning to coordinate their search, as mean actions decrease and mean rewards increase. The theoretical maximum reward is 8, but agents converged around 6 after approximately ten million steps, with mean actions oscillat-

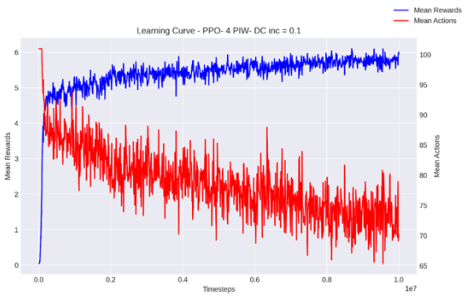

Figure 9. Learning curve for PPO with 4 PIW.

ing between 70 and 80. This instability reflects the complexity of managing four distinct PIWs, each with its own movement pattern.

Table 7. Evaluating results for PPO with 4 PIW.

Metric	4 PIWs
Found all PIW (% of trials)	21.54
Average number of PIW found	2.3
Action mean	86.81

Table 7 shows that agents found all PIWs in only 21.54% of the tests, a lower success rate than in other hypotheses. Agents quickly moved to the center of the probability matrix and followed it, likely due to the neural network architecture, RL algorithm, or training hyper parameters.

Further research and advanced RL features are needed to handle the increased complexity and stochastic nature of realistic environments. Although the agents found an average of 2.3 PIWs per test, achieving a 100% success rate is crucial for real-world scenarios where all PIWs must be rescued. While this result is a step forward in RL research, it highlights the need for ongoing improvement to ensure all PIWs are consistently located.

5. Conclusions

In the first hypothesis, we aimed to test if RL techniques have superior adaptability and performance compared to a pre-determined, informed algorithm, especially when PIWs deviate from regions of highest probability. The results confirm that RL algorithms outperform the greedy approach in all metrics. PPO achieved a 75.44% success rate with a 0.1 dispersion increment and 83% with a 0.05 increment, compared to the greedy algorithm's 35.84% and 50.18%, respectively. DQN failed to learn an effective strategy.

The second hypothesis examined the effect of independent networks on convergence time and training stability. Decentralized networks required more training steps and were less stable than centralized ones. This is due to the additional resources needed for multiple policies, each training its own neural network. The lower performance is likely because not all agents learn an optimal policy, significantly impacting results in complex problems.

For the third hypothesis, we tested the impact of historical search data on search efficiency. The results showed no significant change in behavior or efficiency. The team concluded that the environment's observations provide sufficient information for the algorithm to choose good actions. The additional data did not enhance performance, but it might be more impactful in missions with greater search area overlap.

The fourth hypothesis focused on efficient path planning to minimize redundant searches and search time while prioritizing high-probability areas. Trained agents learned to cover the search area, prioritize cells with higher POC, and accelerate the search process. They completed the task in 58.62% of the required steps, approaching the ideal 50% speedup. This demonstrates that RL algorithms can handle multi-objective problems and optimize multiple tasks.

Finally, the fifth hypothesis investigated agent organization and movement patterns in scenarios with multiple PIWs. Agents consistently located more than half of the PIWs, which is considered a success in research terms. Although the agents didn't find all PIWs in most tests, they effectively coordinated their search, improving their performance over time. The increased complexity of managing multiple PIWs led to sub optimal solutions, but the algorithms showed potential for handling such scenarios.

References

[1] K. Trummel and J. Weisinger, "The complexity of the optimal searcher path problem", *Operations Research*, vol. 34, no. 2, pp. 324–327, 1986.

[2] X. Li, L. Li, J. Gao, *et al.*, *Recurrent reinforcement learning: A hybrid approach*, 2015. DOI: https://doi.org/10.48550/arXiv.1509.03044. arXiv: 1509.03044. [Online]. Available: https://arxiv.org/abs/1509.03044.

[3] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning", *Nature*, vol. 518, pp. 529–533, 2015. DOI: https://doi.org/10.1038/nature14236.

[4] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms", *CoRR*, vol. abs/1707.06347, 2017. arXiv: 1707.06347. [Online]. Available: http://arxiv.org/abs/1707.06347.

[5] E. Liang, R. Liaw, P. Moritz, *et al.*, "Rllib: Abstractions for distributed reinforcement learning", International Conference on Machine Learning, 2018.

[6] B. Ai, M. Jia, H. Xu, *et al.*, "Coverage path planning for maritime search and rescue using reinforcement learning", *Ocean Engineering*, vol. 241, p. 110 098, 2021, ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2021.110098. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801821014220.

[7] D. Silver, S. Singh, D. Precup, and R. S. Sutton, "Reward is enough", *Artificial Intelligence*, vol. 299, p. 103 535, 2021, ISSN: 0004-3702. DOI: https://doi.org/10.1016/j.artint.2021.103535. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370221000862.

[8] M. Castanares, L. F. S. Carrete, E. F. Damiani, L. D. M. de Abreu, J. F. B. Brancalion, and F. J. Barth, *Dsse: A drone swarm search environment*, 2024. DOI: https://doi.org/10.48550/arXiv.2307.06240. arXiv: 2307.06240 [cs.LG].

[9] A. M. Labanca, *Deep reinforcement learning for kamikaze drone decision-making*, São José dos Campos: Instituto Tecnológico de Aeronáutica, 2024.

[10] J. Wu, L. Cheng, S. Chu, and Y. Song, "An autonomous coverage path planning algorithm for maritime search and rescue of persons-in-water based on deep reinforcement learning", *Ocean Engineering*, vol. 291, p. 116 403, 2024, ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2023.116403. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801823027877.

[11] R. R. Rodrigues, J. C. C. de Oliveira, P. H. B. A. Andrade, and R. L. Falcão, "Final reinforcement learning project - droneiros de ibitinga". [Online]. Available: https://github.com/insper-classroom/projeto-final-droneiros-de-ibitinga/.