Search and Rescue Drone using Reinforcement Learning

Abhinav Bhamidipati
University of Maryland
College Park
abhinav7@umd.edu

SriRam Prasad Kothapalli
University of Maryland
College Park
sriram21@umd.edu

Charith Reddy Gannapu Reddy
University of Maryland
College Park
charith@umd.edu

1. Abstract

Search and rescue (SAR) operations in hazardous environments often require efficient and reliable systems to locate and assist victims. Traditional SAR techniques can be slow, dangerous, and limited in their ability to cover large or inaccessible areas. Drones offer a promising solution due to their ability to quickly navigate complex terrains and gather real-time data. However, fully autonomous operation in dynamic, obstacle-rich environments remains a significant challenge. In this work, we address this challenge by applying reinforcement learning (RL) to enable autonomous decision-making in search and rescue drone missions. Specifically, we use Proximal Policy Optimization (PPO), a state-of-the-art RL algorithm, to train a drone to navigate simulated disaster zones, locate victims, and optimize its flight path. The system is implemented within the Robot Operating System (ROS) framework and tested in the Gazebo simulator, which provides a realistic environment for drone training and evaluation. Our approach focuses on enabling the drone to balance exploration, avoid obstacles, and efficiently complete SAR tasks. Experimental results demonstrate that PPO-driven autonomous drones can successfully navigate dynamic environments, improving SAR efficiency and reliability. This work presents a scalable solution for autonomous SAR missions, contributing to the development of more intelligent, adaptable, and effective rescue operations.

2. Introduction

Reinforcement learning (RL) represents a significant paradigm within artificial intelligence, focusing on training agents to make sequential decisions through trial and error. The core of RL lies in the interaction between an agent and its environment, where the agent learns to maximize a cumulative reward by observing state transitions and taking actions accordingly. This method has been successfully applied across diverse domains, including robotics, gaming, and navigation tasks. In RL, the agent's behavior is governed by a policy, which maps states to actions. Policies can be deterministic or stochastic, with stochastic policies defining probability distributions over actions given a state. The objective of reinforcement learning is to optimize this policy such that the expected return, typically modeled as a cumulative reward over time, is maximized. Essential concepts in RL include value functions, which estimate the expected return from a state or state-action pair, and the notion of "advantage," which quantifies how a specific action compares to others under the same policy.

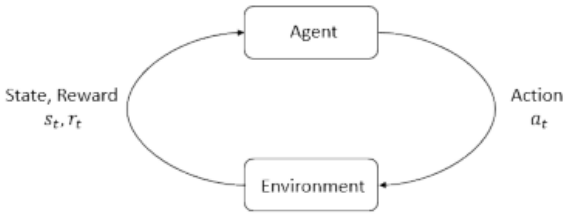


Figure 1: Agent-environment loop

Proximal Policy Optimization (PPO) is a state-of-the-art RL algorithm designed to balance performance and stability. Unlike traditional policy gradient methods, PPO incorporates constraints to ensure that updates to the policy are not excessively large, thereby preventing drastic performance drops. This is achieved through a clipped objective function that limits the policy's deviation from the previous iteration. PPO's approach enables efficient and robust learning, making it suitable for environments with high-dimensional state or action spaces.

In the context of path planning for mobile robots, PPO has demonstrated its effectiveness in generating control laws that enable robots to navigate complex environments autonomously. By training in simulated environments, RL models such as PPO can learn to overcome obstacles, reach targets, and adapt to dynamic scenarios, all while optimizing computational resources and training time. This capability makes PPO a valuable tool for advancing autonomous robotics and intelligent navigation systems.

3. Related Work

Recently, numerous drone control algorithms leveraging deep learning have emerged, finding applications in tasks such as localization and navigation [3]. Among these, supervised learning has shown significant advancements in drone control, while unsupervised learning has primarily focused on auxiliary tasks like feature extraction and action recognition. Deep reinforcement learning (DRL) has achieved notable breakthroughs, establishing itself as a key approach for drone navigation and control.

Supervised learning in UAV Navigation: Supervised learning relies on extensive datasets for training, enabling UAVs to tackle complex tasks. Convolutional neural networks allow UAVs to navigate and locate targets in constrained indoor environments using monocular vision [5], while models trained on collision data enhance obstacle avoidance. Urban navigation systems trained on road datasets help UAVs avoid obstacles in cities, and supervised learning has been integrated into practical applications like IoT systems [1]. However, challenges include the need for manually labeled data, limited generalization, and the frequent need for retraining in new environments.

Unsupervised Learning: Although unsupervised learning cannot independently perform UAV navigation, it plays a supportive role in supervised and reinforcement learning models. It assists in data labeling [2], estimates depth maps from monocular images, and supports rescue missions with human detection capabilities. Techniques such as Variational Autoencoders (VAEs) have been employed to process visual perception data, enhancing DRL training.

Reinforcement Learning: Reinforcement learning (RL) adopts a "trial-and-error" approach inspired by human and animal learning. By interacting with the environment and receiving evaluative feedback, RL enables UAVs to optimize decision-making without requiring predefined teacher signals. RL is particularly suited for addressing complex decision-making problems like UAV navigation.

Traditional RL methods include value-function-based algorithms, such as Q-learning, which have been applied to mobile robot navigation [7]. However, Q-learning's discrete state and action spaces limit its applicability in dynamic UAV navigation. To address these limitations, DRL algorithms were introduced, combining deep learning's capabilities with RL to handle continuous state and action spaces. Early DRL advancements included Deep Q-Networks (DQN) [8], which demonstrated promising results in path planning despite limitations due to their discrete action space.

Policy-based DRL algorithms have emerged to address the challenges posed by discrete action and state spaces in UAV navigation. Methods such as Deep Deterministic Policy Gradient (DDPG) [6] and Distributed Proximal Policy Optimization (DPPO) [4] have advanced UAV control by enabling continuous decision-making and adaptability in dynamic and complex environments. Extending these developments, we propose the application of Proximal Policy Optimization (PPO) to address the demands of search and rescue (SAR) operations in hazardous settings.

Our approach focuses on equipping UAVs with the ability to autonomously navigate disaster zones characterized by dynamic obstacles and intricate terrain. By employing PPO, the UAV learns to optimize its trajectory while balancing exploration and obstacle avoidance, ensuring efficiency and reliability in locating victims. Through rigorous training and testing in realistic simulation environments, our work showcases the potential of reinforcement learning in enhancing UAV navigation for critical SAR tasks.

4. Problem Formulation

Consider the drone navigation problem as a Markov Decision Process (MDP) where the drone must navigate a dynamic and obstacle-rich environment. The problem can be formalized by the tuple (S, A, P, R, γ), where S represents the state space, including the drone's position, velocity, and environmental information (e.g., detected obstacles and victims). The action space, A, corresponds to the possible movements or control commands of the UAV. The transition function, $P(s'|s, a)$, models the likelihood of transitioning from state s to state s' under action a. The reward function, $R(s, a)$, provides feedback to the agent, encouraging it to explore the environment effectively, avoid collisions, and locate victims. The discount factor, $\gamma \in [0, 1]$, governs the importance of future rewards.

The objective is to optimize a policy $\pi(a|s)$ that maximizes the expected cumulative reward over time, formalized as:

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right].$$

Proximal Policy Optimization (PPO) is used to solve this MDP by approximating the optimal policy. PPO operates within the actor-critic framework, where the actor represents the policy $\pi_\theta(a|s)$, and the critic estimates the value function $V^\pi(s)$, which predicts the expected return from a given state. PPO employs a clipped objective function to update the policy while preventing large, destabilizing changes. The objective function for PPO is given by:

$$L^{\text{PPO}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right],$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the probability ratio between the current and previous policies, and $\hat{A}_t$ is the advantage function that measures the relative value of the action a_t in state s_t. The clipping parameter ϵ limits the extent

to which the policy can change, ensuring stability during training.

This approach is particularly effective for UAV navigation in complex environments, as it allows the drone to learn an optimal navigation policy without requiring explicit supervision, enabling it to adapt to dynamic obstacles and evolving mission goals in search and rescue operations.

By leveraging the PPO framework, the drone can iteratively improve its navigation policy through trial-and-error interactions with the environment, gradually refining its ability to balance exploration and exploitation. The use of the clipped objective ensures that the learning process remains stable and avoids catastrophic policy updates, which is crucial for high-stakes scenarios such as search and rescue missions. Additionally, incorporating a well-designed reward function allows the system to prioritize critical tasks, such as reaching victims quickly or avoiding hazardous areas, while maintaining energy efficiency and operational safety.

5. Methodology

5.1. System Architecture

The system architecture integrates ROS2 middleware, Gazebo simulator, OpenAI Gym, and Stable Baselines 3 for PPO training and deployment, as illustrated in Fig. 3. The modularity of this setup ensures efficient communication, simulation, and reinforcement learning application.

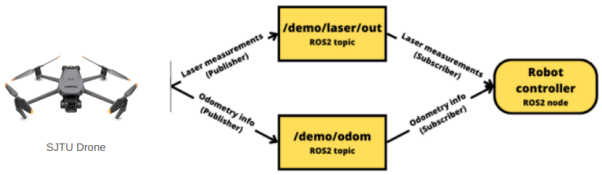


Figure 2: Laser and Odometry topic Communication

The drone interacts with ROS2 nodes for sensor input and control commands. The main ROS2 topics used are: 1. Laser Measurements: Published on /demo/laser/out for obstacle detection and mapping. 2. Odometry Information: Published on /demo/odom for tracking the drone's position. Fig. 2

The robot controller node subscribes to these topics to make navigation decisions and publishes control commands to /demo/cmd_vel, as shown in Fig. 4.

5.2. Simulation Environment

The drone was tested in a simulation environment created in Gazebo. Fig. 5 shows the sample environment setup with multiple rooms, obstacles, and target points.

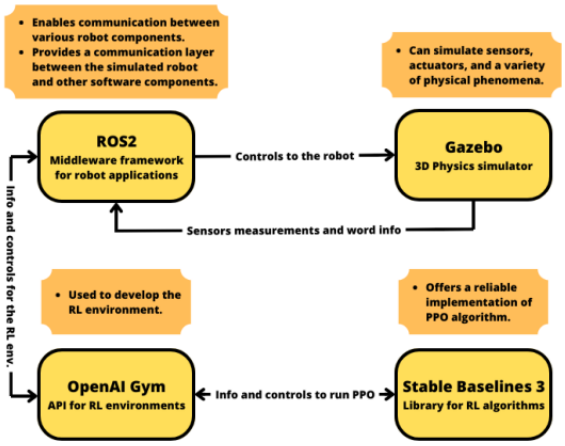


Figure 3: System Architecture for RL-enabled Drone

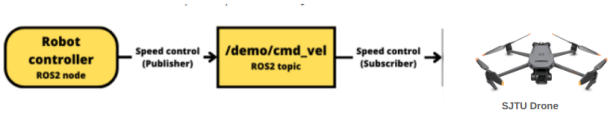


Figure 4: Speed Control flow to SJTU drone

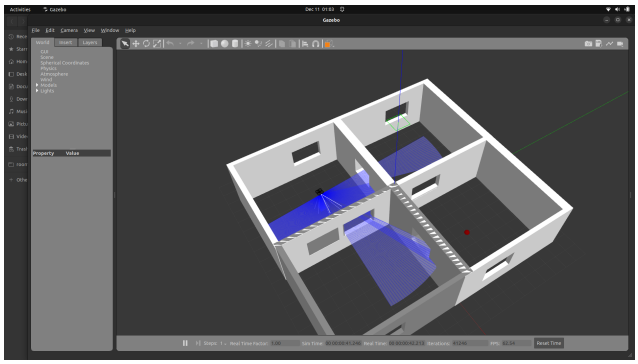


Figure 5: Gazebo Environment

5.3. State Space and Observation

The PPO algorithm operates by leveraging a well-defined state space and observation model to train the drone for effective navigation. The *state space* is designed to capture critical environmental and positional information, including laser scan data to measure distances to obstacles, drone odometry to monitor its position and velocity, and the computed distance to the target. These components enable the agent to perceive its surroundings and plan actions accordingly.

The *observation model* further refines this information by integrating sensor inputs, such as laser measurements and positional data, with the relative location of the tar-

get. This allows the drone to dynamically update its understanding of the environment and respond effectively to obstacles, hazards, and targets. Together, the state space and observation model provide a comprehensive representation of the drone's operational context, ensuring the reinforcement learning algorithm can optimize its decision-making for search and rescue tasks.

5.4. Reward Functions

To train the drone effectively, three reward strategies were designed and evaluated, each tailored to encourage specific navigation behaviors and optimize performance:

1. Simple Reward This strategy focuses on fundamental behaviors for reaching the target and avoiding obstacles. The agent receives:

- A positive reward of $+1$ for successfully reaching the target.
- A negative reward of -1 for colliding with obstacles.
- A neutral reward of 0 for all other cases.

2. Risk-Seeker Reward This approach builds upon the Simple Reward by promoting a slightly riskier behavior to explore challenging areas. The agent receives:

- Similar rewards as in the Simple Reward strategy.
- A smaller negative reward of -0.1 for colliding with obstacles, encouraging risk-taking without significantly penalizing exploration.

3. Heuristic Approach This advanced strategy incorporates time and distance heuristics to balance exploration and exploitation. The agent's reward is calculated as:

- A positive reward of $(1000 - \text{num\_steps})$ for reaching the target, penalizing the time taken to complete the task.
- A large negative reward of -10000 for colliding with obstacles.
- A continuous reward that combines:
  - **Instant Reward:** Proportional to the negative distance from the target, promoting progress.
  - **Attraction Factor:** Activated near the target, providing additional rewards for proximity.
  - **Repulsion Factor:** Activated near obstacles, imposing penalties to discourage unsafe navigation.

Each reward function was carefully designed to address different trade-offs between safety, exploration, and efficiency. These strategies were tested and compared in various simulation environments to identify the most effective approach for search and rescue operations.

6. Experiments

We have conducted 3 different experiments with 3 different reward functions that we have discussed above in the Gazebo world for approximately 7-8 hours and here are our results.

6.1. Simple Reward function

The approximate KL divergence initially increases but eventually stabilizes, showing that the updates adhere to the KL constraints. The final values indicate controlled exploration without overfitting. The loss reduces significantly, reflecting that the value function is learning effectively. By the end of training, it approaches zero, implying good alignment of predicted and actual rewards. The clip fraction starts relatively high, indicating significant policy updates early on. It decreases over time, stabilizing as training progresses, which suggests the model converges to a more stable policy.

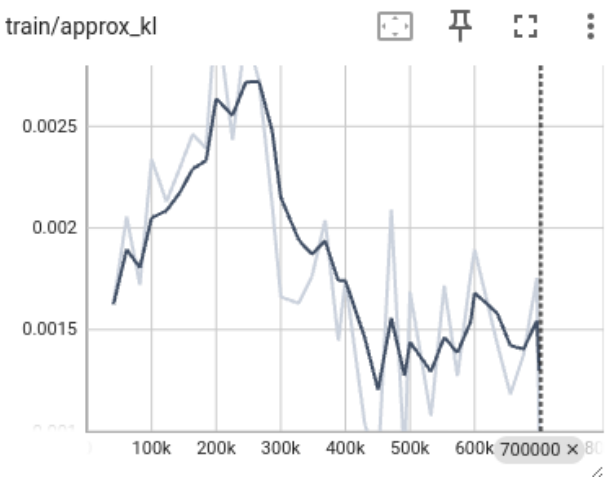
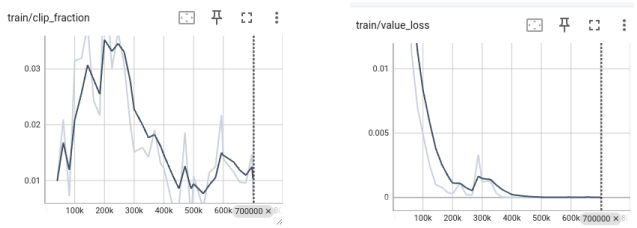


Figure 6: KL-Divergence

6.2. Risk-Seeker Reward

A similar pattern emerges, with higher peaks and fluctuations during mid-training. The decreasing trend at the end suggests better adherence to KL constraints, although the overall variance may impact policy quality.There is an increase in variance mid-training, likely due to riskier exploration strategies. The eventual decline indicates stabilization, but the model's behavior suggests occasional instability in policy updates.Value loss decreases consistently but



(a) Clip fraction (b) Training loss

Figure 7: Clip-fraction and training loss.

with fluctuations due to riskier actions during exploration. This suggests challenges in accurately predicting rewards for high-variance strategies.

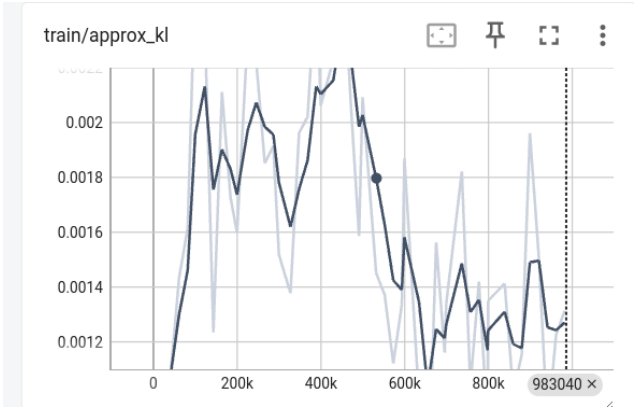
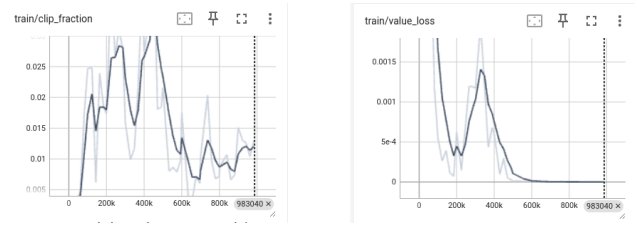


Figure 8: KL Divergence



(a) Clip fraction (b) Training loss

Figure 9: Clip-fraction and training loss.

6.3. Heuristic Approach

The KL divergence also shows irregular patterns, including sharp increases. This suggests the heuristic approach may occasionally violate KL constraints, leading to less consistent learning.The clip fraction shows sharp spikes, particularly during mid-training, which indicates abrupt changes in policy due to heuristic updates. This suggests a less stable optimization trajectory compared to the other approaches.The value loss remains higher for longer and reduces at a slower rate compared to the other approaches. This suggests the heuristic approach struggles to predict rewards effectively due to the non-linear or hard-coded nature of the heuristics.

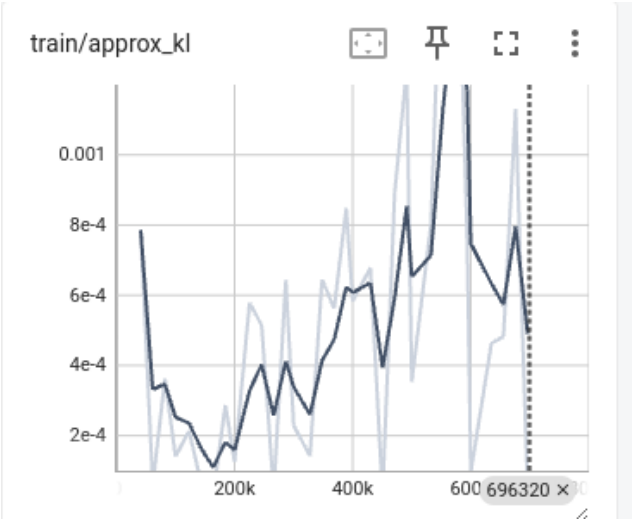
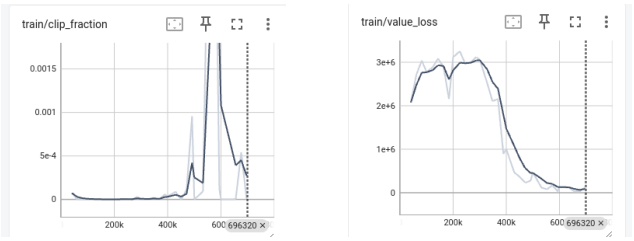


Figure 10: KL Divergence



(a) Clip fraction (b) Training loss

Figure 11: Clip-fraction and training loss.

7. Results

Simple reward-based optimization emerges as the most reliable and consistent approach for training reinforcement learning policies. Risk-taking strategies can be beneficial for exploring larger state-action spaces but require careful tuning to prevent instability. Heuristic-based methods, while potentially useful for domain-specific tasks, may introduce significant challenges in ensuring stable and efficient policy learning.

The drone moves from the start point to the red target position, from room 1 to room 3, which are diagonal to each other. Fig. 12 (a), (b), (c), (d) in sequence.

8. Conclusion

This project demonstrates the potential of reinforcement learning, specifically Proximal Policy Optimization (PPO),

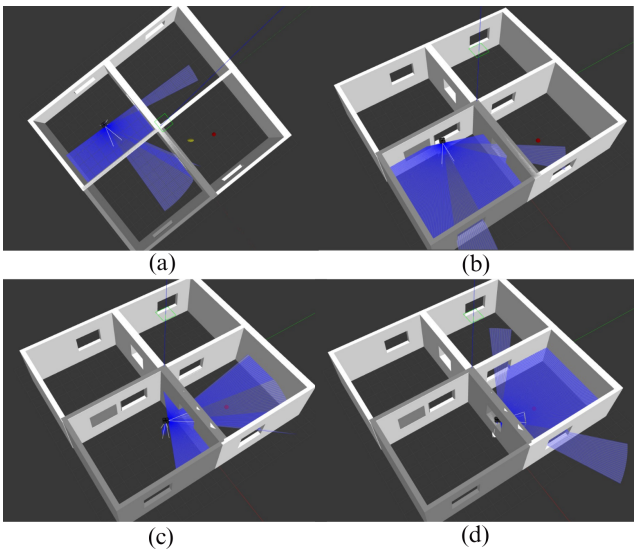


Figure 12: Inference moving from initial to target position in sequence (a), (b), (c) and (d)

in enabling autonomous drone navigation for search and rescue (SAR) operations. By leveraging the robustness of PPO and integrating it within a modular framework combining ROS2, Gazebo, and deep learning techniques, the drone successfully navigates dynamic and obstacle-rich environments to locate targets efficiently. Through the evaluation of various reward strategies, including simple, risk-seeker, and heuristic-based approaches, the system showcases adaptability and scalability for complex SAR tasks.

9. Future work

Future developments for this project will focus on designing advanced reward functions and integrating the reinforcement learning (RL) agent within a hierarchical planning framework. Specifically, enhancing agent performance could involve crafting more tailored reward structures and incorporating a more diverse range of training scenarios. However, the current study's limitations necessitate further exploration to refine the learning environment and training conditions.

Additionally, future research will explore incorporating sensory inputs from devices such as depth cameras and RGB cameras to enhance reward computation. By leveraging data from these sensors, the RL agent can develop a richer understanding of the environment, improving its ability to make context-aware decisions and handle more complex scenarios. Integrating such sensory modalities will pave the way for more robust and adaptable RL-based navigation systems.

10. Individual Contribution

Abhinav: Lead the formulation of the problem, focusing on the foundational aspects of the project. Created the Gazebo simulation environment and developed the initial training setup for the RL model using a simple reward function.

Charith: Focused on supervised learning for the drone by utilizing depth images as input to the model, trained the model using the Risk-Seeker reward strategy, and drafted and structured the project report.

Sriram: Setup of ROS2 environment and SJTU drone, trained the model with a heuristic approach.

References

[1] S. Alsamhi, O. Ma, M. Ansari, and F. Almalki. Survey on collaborative smart drones and internet of things for improving smartness of smart cities. *IEEE Access*, 7:128125–128152, 2019. 2

[2] M. Bah, A. Hafiane, and R. Canals. Deep learning with unsupervised data labeling for weed detection in line crops in uav images. *Remote Sensing*, 10:1690, 2018. 2

[3] A. Carrio, C. Sampedro, A. Rodriguez-Ramos, and P. Campoy. A review of deep learning methods and applications for unmanned aerial vehicles. *Journal of Sensors*, 2017:3296874, 2017. 2

[4] N. Heess, D.T.B, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, and S. E. et al. Emergence of locomotion behaviours in rich environments. *arXiv*, 2017. arXiv:1707.02286. 2

[5] D. Kim and T. Chen. Deep neural network for real-time autonomous indoor navigation. *arXiv*, 2015. arXiv:1511.04668. 2

[6] T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv*, 2015. arXiv:1509.02971. 2

[7] E. Low, P. Ong, and K. Cheah. Solving the optimal path planning of a mobile robot using improved q-learning. *Robotics and Autonomous Systems*, 115:143–161, 2019. 2

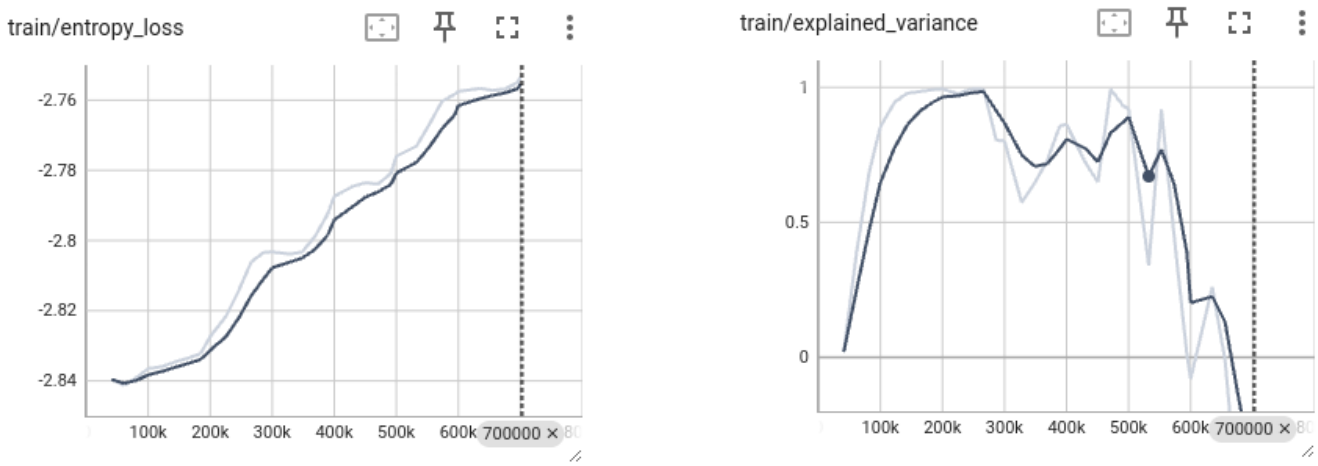
[8] V. Mnih, K. Kavukcuoglu, D. Silver, J. Veness, M. Bellemare, A. Graves, M. Riedmiller, A. Fidjeland, G. Ostrovski, and S. P. et al. Human-level control through deep reinforcement learning. *Nature*, 518:7540, 2015. 2

11. Appendix

For additional details and demonstration, the inference of the drone using a simple reward mechanism is available via the following link:

https://drive.google.com/file/d/1rX_Co7UapMUwAzIdfQrSaqxVZ5ZzpA7F/view?usp=sharing

Few more results Simple Reward

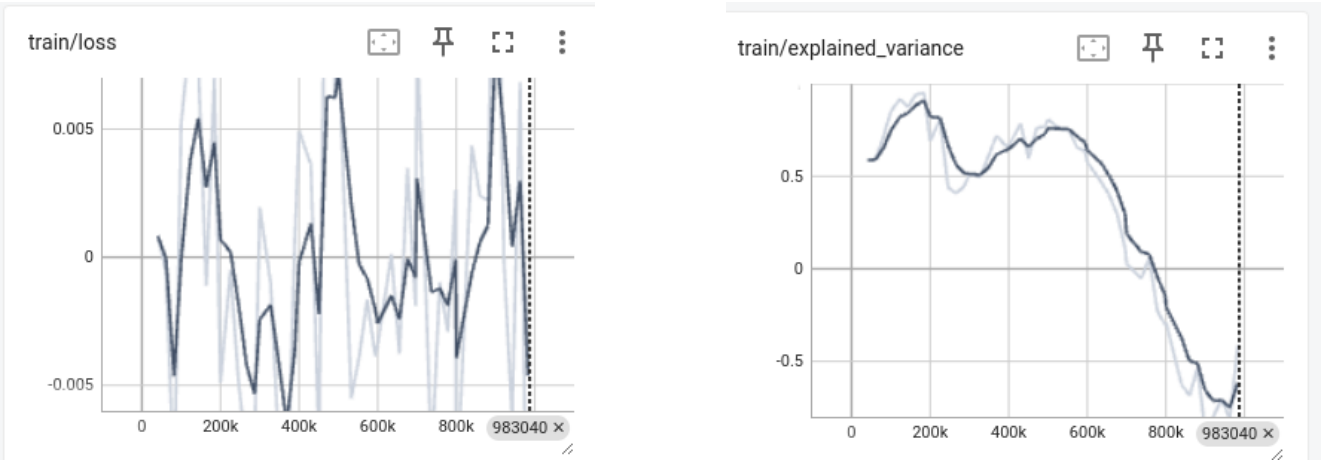


(a) ENtropy Loss

(b) Training loss

Figure 13: Explained variance and entropy loss

Risk-Seeker Reward



(a) Loss

(b) Explained variance and entropy loss

Figure 14: Explained variance and entropy loss

Heuristic Approach

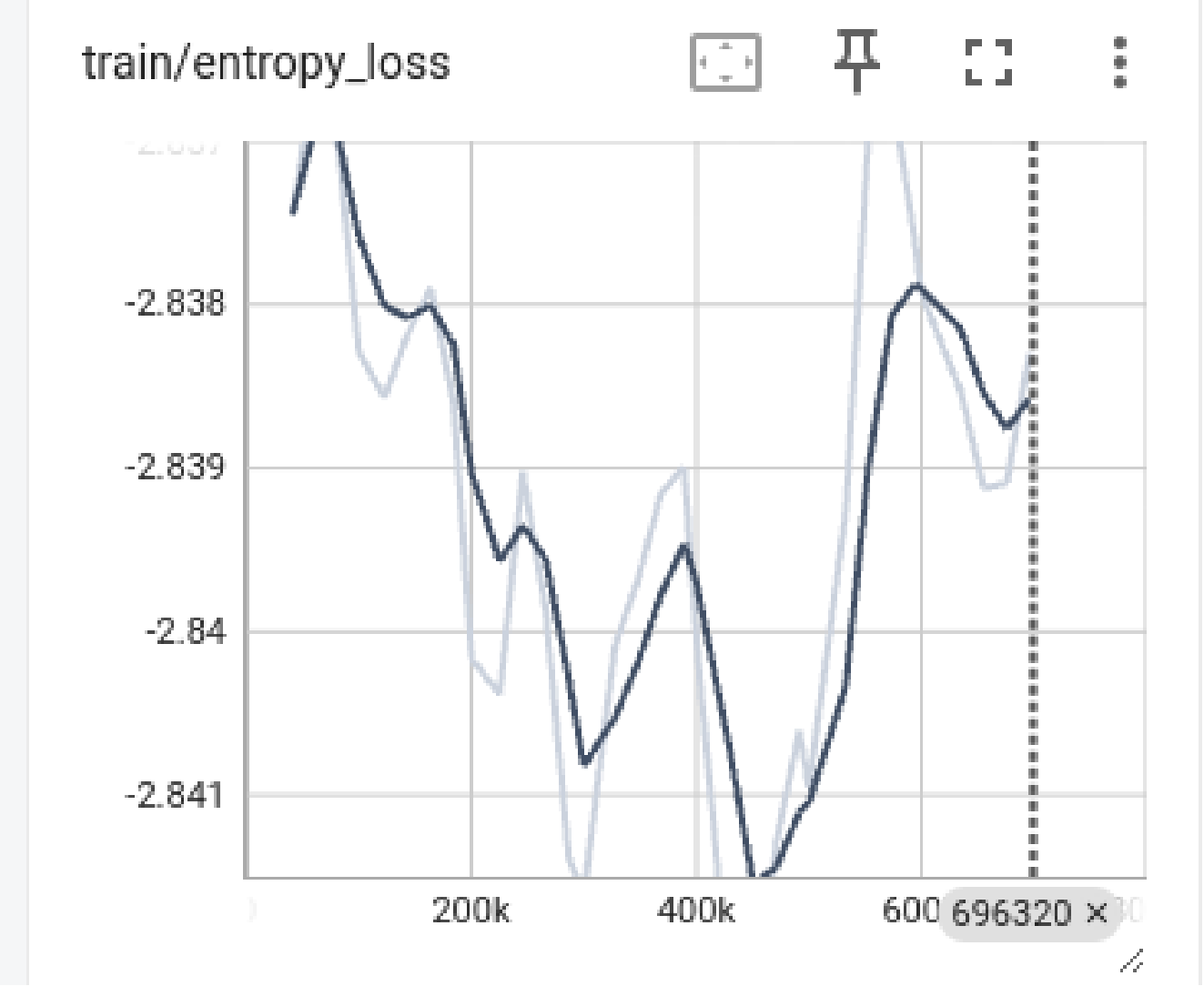


Figure 15: Entropy Loss