Human-AI Collaboration in Real-World Complex Environment With Reinforcement Learning

by

Md Saiful Islam

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science
University of Alberta

Abstract

Recent advances in reinforcement learning (RL) and Human-in-the-Loop (HitL) learning have made human-AI collaboration easier for humans to team with AI agents. Leveraging human expertise and experience with AI agents in intelligent systems can be efficient and beneficial. Still, it is unclear to what extent human-AI collaboration will be successful and how such teaming performs compared to humans or AI agents only. In this work, we show that learning from humans is effective and that human-AI collaboration outperforms human-controlled and fully autonomous AI agents in a complex simulation environment. In addition, we have developed a new simulator for critical infrastructure protection, focusing on a scenario where AI-powered drones and human teams collaborate to defend an airport against enemy drone attacks. We develop a user interface to allow humans to assist AI agents effectively. We demonstrate that agents learn faster while learning from policy correction compared to learning from humans or agents. Furthermore, human-AI collaboration requires lower mental and temporal demands, reduces human effort, and yields higher performance than if humans directly controlled all agents. In conclusion, we show that humans can provide helpful advice to the AI agents, allowing them to improve learning in a multi-agent setting.

Preface

Part of this work has taken place in the Intelligent Robot Learning (IRL) Lab at the University of Alberta, which is supported in part by research grants from the Alberta Machine Intelligence Institute (Amii); a Canada CIFAR AI Chair, Amii; Compute Canada; Huawei; Mitacs; and NSERC. In addition, a part of this thesis is done in collaboration with AI Redefined, Thales Group, JACOBB, Srijita Das, Matthew Guzdial, and Matthew E. Taylor. A portion of this thesis has been presented at the Adaptive & Learning Agents (ALA) Workshop @AAMAS 2023 and submitted to the Neural Computing and Applications journal for publication. The research project of which this thesis is a part received research ethics approval from the University of Alberta Research Ethics Board (REB number: Pro00107555).

Acknowledgements

First and foremost, I would like to thank my advisor, Dr. Matthew E. Taylor, for his constant support throughout this thesis. I am truly grateful to have an advisor that acts as my champion. Secondly, I would like to acknowledge the consistent efforts of my co-supervisor, Dr. Srijita Das. I would also like to thank Dr. Matthew Guzdial, Dr. Carrie Demmans Epp, and my writing coach, Dr. Antonie Bodley, for all their feedback on this thesis.

Most importantly, I thank my family for their ongoing love and support throughout my graduate program. I am incredibly grateful for my spouse, Moriom. On my worst days, you always find a way to show me the light.

Finally, I would like to acknowledge the help provided by the participants of our user study, who had benevolently volunteered to contribute to one of the main findings of this project.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Protecting critical infrastructure, such as an airport, against security threats is a complex, sensitive, and expensive task, leading to a history of exploring automated and autonomous solutions [1]. However, fully automated and autonomous solutions in critical applications are not advisable due to the current limitations in technology maturity and trained operators. These might lead to poor performance, significant infrastructure damage, and increased risks of other collateral damages. Additionally, training humans to use such solutions effectively remains a considerable challenge. On the other hand, continuous surveillance of such systems, quick assessment, and handling of potential threats would benefit from AI capabilities. In many cases, AI agents need assistance achieving full autonomy within a reasonable timeframe due to the system's complexity or the scarcity of data [2]. Another significant challenge is the AI agent's ability to capture contextual understanding. For instance, consider an airport security scenario where an AI system affiliated with the airport authorities detects rapid movement on a surveillance camera or drone during nighttime. This system might classify the movement as an intruder, lacking the contextual nuance to recognize it as a routine patrol by the local police forces at the airport's perimeter.

Humans generally possess domain expertise, experience, and contextual understanding in solving complex problems that are difficult for AI agents to learn or replicate. For example, considering the above example, a human operator might recognize the drone as a routine patrol based on the circumstances surrounding the drone's presence and behavior. At the same time, the AI agent lacks the knowledge to respond appropriately. Human decision-making becomes essential in safety-critical applications, where scenarios may be partially anticipated. Considering the value of human expertise, it is necessary to effectively leverage human knowledge and situational awareness in collaborative environments, especially for critical applications like defense or security. These applications are likely to benefit from systems that combine the strengths of both human operators and autonomous systems. This integration aims to decrease system costs and enhance task performance while maintaining meaningful human control in dangerous or critical operations. Such a hybrid approach is crucial to mitigate potential risks in these high-stakes environments [3].

Recently, reinforcement learning (RL) has successfully solved many complex decision-making problems, such as mastering the game of Go [4], deploying super pressure balloons in the stratosphere [5], and generating synthetic drugs [6, 7]. Although established domains like Atari and Mujoco serve as benchmarks for cutting-edge RL research [8, 9], the introduction of simulators for complex domains facilitating human-AI collaboration has been less explored [10, 11]. In addition, a notable challenge in deep RL is its sample inefficiency [12], requiring millions of interactions with the environment, making it difficult to adapt to real-world problems. To mitigate this, advice giving techniques such as demonstrations [13–15], action-advice [16–18], preference [19–21] and reward shaping [22–25] have been used to guide RL agents to relevant parts of the state-space. However, most of this work has been restricted to game domains and advice by trained agents. A significant and relatively unexplored aspect concerns the potential improvement of human-agent collaboration through human demonstrations in complex, real-world environments. Furthermore, the current literature on human-agent collaboration reveals a noticeable scarcity of intelligent user interface design and integration for humans to provide effective advice. This scarcity frequently leads to misunderstandings between humans and AI agents, hindering the use of the human operator's expertise.

To address the challenges of complex real-world domains, we develop a novel simulator and user interface for the specific problem of the airport's restricted zone protection system. The use case consists of a fleet of ally drones trying to protect restricted airspace against multiple intruding drones. Following recommendations from air defense domain experts, the simulator is designed to mimic the dynamics of a real-world scenario. These dynamics include the drones' velocity, flight dynamics, the specifications of the ground radar sensor, the sensing payloads (radar and electro-optical), and the neutralization payloads embedded in the blue drones. Such real-world dynamics make the environment complex. The complexity of the environment means that a naive RL agent would require many environment interactions to learn an optimal policy. Given the cost and risk associated with these interactions in the specified domain, the trained agent needs to be sample-efficient. We demonstrate that learning from human or agent demonstrations can minimize the number of required environment interactions for the mentioned complex environment.

Prior research [26–28] indicates that when one person oversees multiple agents in complex systems, the increased monitoring demand can negatively affect their workload and cognitive load, ultimately hindering performance. Instead, we demonstrate that better decision-making capabilities of the trained agents can reduce the human operator's workload and increase the performance of the human-agent team. The main goal of creating human-agent collaborations is to capitalize on the strengths of agents and humans while mitigating their weaknesses. For instance, intelligent agents excel in tasks such as analyzing vast data sets and making rapid decisions based on specific patterns, outperforming humans [29]. In contrast, humans exhibit superior decision-making abilities rooted in their moral values and contextual understanding, compared to agents [30]. A characteristic feature of the specific defense domain use case is that operations are versatile, often highly unpredictable, and decision-making capabilities with contextual understanding play a crucial role. To maintain the exercise of authority and direction by humans, we also use human policy correction to correct the trained agent's policy. We show that online policy correction is the most effective form of advice to improve agent learning and achieve the best performance. In addition, we demonstrate that the cognitive workload of humans is lower while doing policy correction than that of humans controlling an untrained agent (drones in this domain). We use non-expert human and agent demonstrations to showcase the robustness of our approach to address the limited availability of human experts.

1.1 Thesis Contributions

The key contributions of this thesis include the following:

1. Introduces a novel multi-agent simulator for defense-specific airport protection use case modeling real-world dynamics with multiple ally and enemy drone agents.

2. Uses state-of-the-art deep RL algorithms to train multiple agents inside the novel simulator.

3. Develops a user interface inside the simulator, which enables human operators to dynamically take control of single or multiple agents to produce in-context demonstrations, thus enabling human-agent collaboration.

4. Demonstrates empirically that trained agent demonstrations or a mixture of human and agent demonstrations help the agent to learn faster.

5. Compares and evaluates multiple advice-giving techniques, i.e., learning from demonstration and policy correction.

6. Compares the human cognitive workload for various advice-giving techniques using a user study demonstrating that policy correction requires less effort than humans having full control over the agents.

1.2 Thesis Outline

We start with background in Chapter 2, laying the foundation by of reinforcement learning. Then, we transition into the deep reinforcement learning algorithms in Chapter 2.2.2. Chapter 2 continues with a review of learning from humans, highlighting different ways of getting demonstrations from humans and agents. We then review human-AI collaboration approaches, highlighting how they differ from our work. Chapter 3 focuses on the problem formulation, encompassing the design of the relevant environment, the intricacies of the Markov decision process (MDP) formulation, and the platforms that enable Human-in-the-Loop interactions. Chapter 4 presents an overview of the experimental setup, describing the environment configurations, and explaining the performance metrics. In Chapter 5, we report the experimental results of our proposed method. In addition, we describe our user study design and report the results. Lastly, in Chapter 6, we reflect upon the limitations encountered throughout the research journey and suggest potential future work and advancements in complex real-world airport security scenarios.

Chapter 2

Background

In this chapter, we introduce the necessary background details on which our work is built. First, we present the concepts of RL and its extension into deep RL. We then transition into our deep RL algorithm and it's mathematical formulations for agent training. Following this, we explore how agents can learn from human demonstrations. In addition, we review existing literature on human-AI collaboration, highlighting its effectiveness in complex sequential decision-making tasks.

2.1 Reinforcement Learning

Reinforcement learning is a paradigm in which agents learn by interacting with an environment to maximize the expected sum of rewards. The RL framework often models problems as a Markov decision process (MDP), defined by a tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$. Here, $\mathcal{S}$ represents the state space, and $\mathcal{A}$ denotes the action space. A state space $s \in$ state space S is considered Markov if it encapsulates all past agent-environment interactions necessary for future decisions. Formally, a state s is Markov if:

$$P(S_{t+1}|S_t, A_t) = P(S_{t+1}|S_1, A_1, S_2, A_2, ..., S_t, A_t)$$

At each time-step t, an agent in state $s \in \mathcal{S}$ selects an action a from action-set $\mathcal{A}$. The environment's transition probability, T, then determines the probability $p(s', r|s, a)$ of transitioning to state s' and receiving reward r given the current state s and action a. The reward function $R : \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ maps states and actions to scalar-valued rewards. The agent receives a reward r_t after interacting with the environment. The agent's objective is to maximize the expected sum of discounted rewards, G_t, at any time-step, t:

$$G_t \doteq \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

2.2 Deep Reinforcement Learning

Deep RL enhances the capabilities of traditional RL by integrating the representational power of deep learning. This combination empowers agents to tackle complex sequential decision-making tasks in single-agent and multi-agent environments [31, 32]. Recent studies [33–36] have highlighted its potential in critical real-world applications and use a variety of algorithms to train DRL agents. The subsequent sub-sections contain details related to Deep Q Networks and Duelling Double Deep Q Networks.

2.2.1 Deep Q Networks (DQN)

Deep Q Networks [9] (DQN), a value-based RL method, have been instrumental in achieving state-of-the-art results in various RL tasks, showcasing the potential of combining deep learning with RL. DQN employs a deep neural network to approximate the Q-value function, thereby mapping state-action pairs to their anticipated rewards. Given a state s and an action a, the Q-value undergoes an update based on the Bellman equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right) \tag{2.1}$$

Here, α denotes the learning rate, r signifies the reward, γ represents the discount factor, and s' is the subsequent state. The introduction of neural networks in DQN, as opposed to traditional Q-tables, is pivotal for handling environments with large or continuous state spaces. Neural networks, with their function approximation capabilities, enable DQN to generalize across such vast spaces, making it feasible to learn effective policies even when storing or updating a Q-value for every possible state-action pair is impractical. This aspect of DQN is particularly beneficial in complex environments with high-dimensional inputs, such as raw pixel data from video games, where it has shown remarkable proficiency, notably in mastering Atari 2600 games using only pixel inputs.

2.2.2 Duelling Double Deep Q-Network (D3QN)

In this work, we employ Duelling Double Deep Q-Network [37] (D3QN) for agent training, a significant advanced variation of DQN. D3QN builds upon the foundational principles of double DQN [38] and duelling DQN [39], integrating the strengths of both architectures to achieve enhanced performance in complex environments, solving the high dimension catastrophe.

DQN uses deep neural networks to estimate Q values, updated via the Bellman equation. DQN adopts two separate neural networks: the prediction and target networks. The *prediction Q-network* is employed to predict the Q value of each action corresponding to the current state and update every iteration. In contrast, the target Q-network is used to predict the Q value of each action in the subsequent state and updated after N iterations to prevent the non-stationary target from dimensional calculation explosion. However, in performing the maximization operation, the agent tends to select an overestimated value higher than the actual value. In DQN, overestimation occurs because it uses the same Q-values to select and evaluate actions, leading to optimistic bias and inflated reward predictions. Double DQN solves this overestimation problem by decoupling the action selection and evaluation of the target network in DQN. Specifically, Double DQN uses two Q-networks, each with different weights θ and θ^{-}. The network with weights θ is used for action selection, while θ^{-} estimates the greedy policy's value. The Q-value update in Double Q-Learning is given by:

$$Q(s, a; \theta) = R_t + \gamma Q(s', \text{argmax}_{a'} Q(s', a'; \theta); \theta^-) \tag{2.2}$$

Where θ and θ^- represents the parameters of the prediction network and target network, respectively.

On the other hand, duelling DQN further refines the DQN approach, adding a duel neural network architecture known as the dual network and optimizes the Q network. While DQN has shown promise, certain scenarios render the Q-value dependent solely on the state. Duelling DQN addresses this by decomposing the Q-value representation into two distinct functions: the state value function $V(s)$ and the action advantage function $A(s, a)$. This decomposition allows for a more nuanced representation, enhancing the agent's ability to learn and make decisions. This architecture is expressed as:

$$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \alpha) + A(s, a; \theta, \beta) \tag{2.3}$$

Here, $V(s)$ represents the value of state s, and $A(s, a)$ represents the advantage of taking action a in state s. The parameters θ are shared across both functions, while α and β are unique to the state value and advantage functions, respectively.

However, the equation in 2.3 struggles to distinguish between the roles of value and advantage in the final output. To address this, the actual combination used in duelling DQN is:

$$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \alpha) + \left(A(s, a; \theta, \beta) - \frac{1}{|A|} \sum_{a'} A(s, a'; \theta, \beta) \right) \tag{2.4}$$

Modifications in 2.4 ensure that for each state s, the advantages of all actions average to zero.

D3QN effectively combines the dueling architecture with double DQN to solve overestimation, instability and the difficulty of convergence problems of DQN. Based on the above improvements, we select D3QN as our baseline algorithm for agent learning in this complex real-world airport security scenario. One primary reason for choosing D3QN as a baseline was our discrete action space, as the DQN series of algorithms are suitable for tasks with a discrete action space.

2.3 Learning from Humans

In this section, we discuss using demonstrations to guide RL agents and then review prior work on their applications in DRL. In addition, we give an overview of the technique we adopted to learn from human demonstrations and how we modify it for our airport security scenario.

2.3.1 Leveraging Demonstration to Guide Deep RL

Learning from demonstrations [40] has been a common approach to making Deep RL sample-efficient. Hester et al. [13] first leveraged demonstrations inside a deep Q-network (DQN) using a supervised loss function to account for deviations from expert demonstrations. Later, demonstrations were used to speed up training with DDPG in complex robotics tasks [41, 42] using specific techniques like behavior cloning loss and prioritized replay buffers. Goecks et al. [43] provided a unified loss function by integrating loss function components from prior works. Existing literature [44–48] focuses on feedback or demonstration efficient algorithms where the authors primarily investigate several sampling and exploration strategies [49–52], improve feedback efficiency using imitation learning [19], unsupervised pre-training [21], and reward relabelling [53].

Some existing work has used dueling double deep Q network [37] (D3QN) to solve real-world complex tasks like security patrolling [54], path planning for unmanned aerial vehicles (UAV) in a dynamic environment [55], unmanned ground vehicle (UGV) control [56], manufacturing [57], vehicle-to-vehicle communication [58], UAV autonomous aerial combat [59, 60]. However, existing literature primarily focuses on improving autonomous agent performance, while our goal is for an agent to learn from humans and improve human-AI collaboration performance in multi-agent settings. Our approach differs from those mentioned above, as we do not focus on sampling techniques or improving feedback efficiency through imitation learning. Our objective is to leverage human demonstrations in complex real-world defense scenarios to enhance the performance of human-AI collaboration.

2.3.2 Deep Q-Learning from Demonstration (DQfD)

We used Deep Q-learning from demonstration [13] for the discrete action scenario, combining human demonstrations with agent experiences for stability and efficiency. DQfD, integrated into the D3QN framework, accelerates policy optimization by leveraging expert experience. It employs an additional supervised loss function alongside Q-learning loss, ensuring that the agent prioritizes actions from expert demonstrations. In DQfD, the agent is pre-trained with demonstrations using a margin classification loss to mimic expert behavior closely. The margin classification loss is defined as below:

$$J_E(Q) = \max_{a \in A} \left[Q(s, a) + l\left(a_E, a\right) \right] - Q\left(s, a_E\right)$$

where a_E signifies the action executed by the demonstrator in state s. The margin function $l(a_E, a)$ is zero when $a = a_E$ and positive otherwise. The n-step return is used to propagate the values of the demonstrator's trajectory to preceding states. The n-step return is defined as:

$$r_t + \gamma r_{t+1} + \ldots + \gamma^{n-1} r_{t+n-1} + \max_a \gamma^n Q\left(s_{t+n}, a\right)$$

The subsequent n-step loss, accounting for the n-step return, is denoted as $J_n(Q)$. An L2 regularization loss was also introduced to prevent over-fitting to the limited demonstration dataset. The comprehensive loss used for network updates is:

$$J(Q) = J_{DQ}(Q) + \lambda_1 J_n(Q) + \lambda_2 J_E(Q) + \lambda_3 J_{L2}(Q) \tag{2.5}$$

where λ_1, λ_2, and λ_3 parameters control the weighting between the losses. Hester et al. [13] employed DQfD in a single-agent RL context. In this work, we extend its application to a multi-agent RL setting, training all five ally drones based on demonstrations and the collective experiences of all agents. Further details are in Chapters 3 and 4.

2.4 Human-AI Collaboration

In this section, we summarize existing works investigating human-AI collaboration and its performance on various tasks.

Human-AI collaboration is emerging as a critical field, integrating human and AI capabilities for diverse applications [61–63]. Research has concentrated on enabling natural and efficient collaboration in human-AI systems, focusing on the communicative impact of shared actions [64] and balancing performance gains with compatibility with human mental models [65]. Effective human-AI collaboration in complex physical scenarios relies on seamless agent performance in simulations [66]. Notably, human-AI collaboration can expedite sequential manipulation tasks [67] and enhance resource distribution [68, 69]. Autonomous AI agents, distinct from expert systems, can learn tasks without preprogramming [70]. Choudhury et al. [71] compare deep learning-based human models with structured "theory of mind" models. Sandrini et al. [72] address the human-AI teaming planning and allocation problem using a minimum-time formulation. Jahanmahin et al. [73] explore human-centered robot interaction in industrial settings, while Tambe et al. [74] did human-AI teaming in stackelberg game settings. Our research examines the utility of human involvement in such settings, highlighting the need for shared autonomy and clear agent communication in real-time human-AI collaboration.

For tasks requiring human-agent collaboration, autonomous agent pilots are often paired with human counterparts to work together effectively [75, 76]. Similarly, cognitive assistants are used to support astronauts during pivotal space missions, enhancing their decision-making capabilities [77]. Another burgeoning area of research encompasses deploying human-AI collaborative teams in rescue operations [78] and the evolution of semi-autonomous vehicles, where human operators and autonomous agents collaboratively navigate to destinations. Numerous existing studies [79–81] employ DQN or its variations to empower UAVs to autonomously formulate control commands and execute air combat tasks, responsive to the information derived from their environmental contexts. However, human involvement was missing or limited in all these prior works. Recently, Zhang et al. [82] used D3QN with an expert experience storage mechanism for decision-making in a simulated UAV Air Combat. They use expert experience with D3QN similar to ours and demonstrate effective use of training data and faster algorithm convergence. However, we use humans for providing demonstrations and policy correction, while existing works only use trained agent experiences. In addition, they also differ from our problem settings and layered architecture for the defined use case.

Policy correction in RL is essential in safety-critical applications aiming to adjust strategies for optimal results [83, 84]. Off-policy correction, addressing policy divergence due to function approximation, bootstrapping, and off-policy errors, are often managed through importance sampling [85–87]. Techniques such as inverse RL, behavior cloning, policy shaping, and constrained RL employ expert demonstrations, feedback, or constraints for policy guidance [42, 88–92]. Bai et al. propose a dynamic constraint set method for policy refinement using probability metrics [93]. In recent work, Zawalski et al. [94] used importance sampling for off-policy correction in multi-agent RL. In contrast to prior work, our focus shifts toward the impact of human correcting a trained agent's policy. We aim to assess the effects of human interventions on human-AI collaboration performance compared to human-only and agent teams. Agents can learn desired behaviors by observing human experts, making demonstrations a valuable tool in policy correction [95]. In this work, we use humans to correct the policy of a trained agent in our airport security scenarios.

Chapter 3

Problem Formulation

In this chapter, we describe the airport defense use case and formulate the problem as an MDP. We also describe the user interface and relevant details about the Human-in-the-loop interactions. Details related to the system architecture of the environment are provided in Appendix B.

3.1 Environment Design

In this work, we introduce an airport defense simulator and explore the impact of human demonstration on this domain by formulating it as a multi-agent RL problem. The environment used for our problem formulation is shown in Figure 3.1. In this stochastic environment, a team of ally drones collaborates to achieve the shared objective of securing the airport's restricted zone from enemy intrusions. At the sametime, the enemy drones plan attacks with knowledge about the defender's strategy. The ally (blue) drones and humans aim to ensure airport security by working as a team to counter any threats posed by enemy (red) drone attacks. The ally (blue) drones and humans aim to ensure airport security by working as a team to counter any threats posed by enemy (red) drone attacks.

The blue team comprises five blue drones, a ground radar sensor, and a ground control station (GCS). The blue drones can be autonomous, remotely piloted, or controlled through occasional human interventions. Each drone is equipped with a

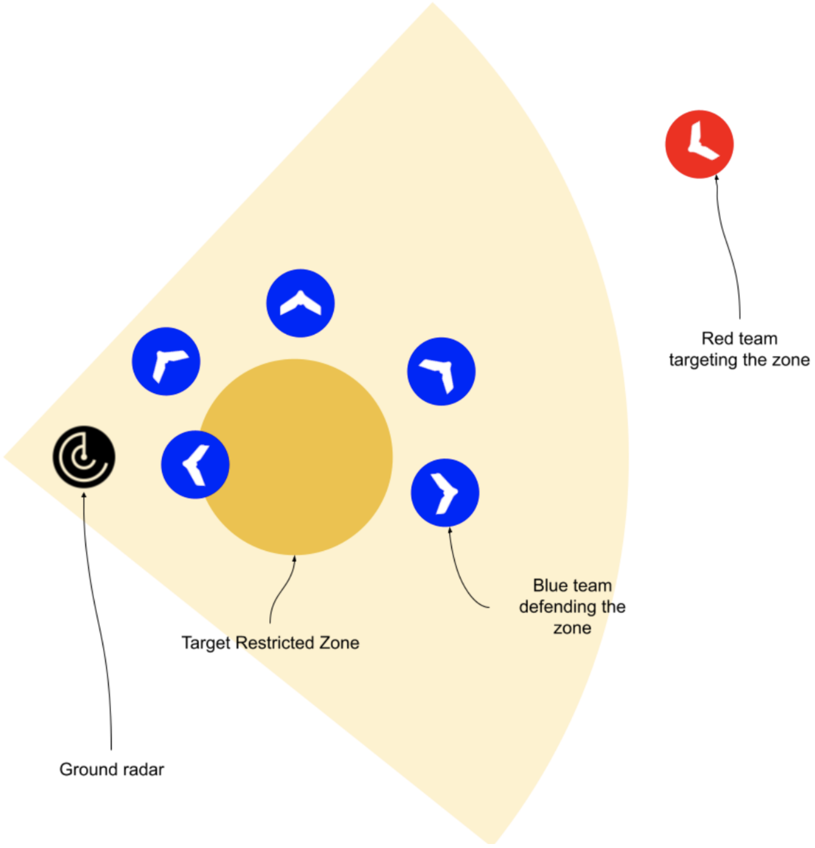


Figure 3.1: Environment: airport defence scenario

gimbal-mounted electro-optic sensor, allowing it to capture data that can be leveraged for surveillance or threat-level assessment. In addition, each blue drone has several neutralization payloads (i.e., devices capable of neutralizing enemy drones when they are within a specific range). The red team comprises a single drone equipped with its radar sensor and a potentially hazardous payload. The goal of the blue team is to detect, localize, intercept, and neutralize the enemy drones before they reach the restricted zone of the airport. Details of the ally drones team and enemy drone are given in Appendix A.

The experimental platform is built around a simplified airspace simulator operating in 2D. Although simplified, several aspects have been modeled following real-world specifications based on feedback from domain experts, such as the detection capabilities of the drone sensors and the radar, as well as the dynamics of the fixed-wing drones. In this environment, the blue and red drones have a partially observable view of the environment. The detection and localization of the red drone provided by the radar and EO sensors embed noise and uncertainty. As per the defense expert's suggestions, we introduce errors into the system to simulate these real-world factors. Specifically, the radar detection probability is 95%, and there is a 5% probability of the radar failing to detect the red drone in 1 second. Moreover, the radar fails to detect the enemy drone if it is outside the radar range. The detection frequency is set to 1 Hz, and the maximum speed of the drones is 10 meters per second. The range of the neutralization payloads embedded on the blue drones is set to 10 meters. All these dynamics make the scenario complex and require the blue team to anticipate the trajectory of the red drone to neutralize it.

3.2 MDP Formulation

We model our airport security scenario problem as a Markov decision process as defined below:

1. **State space:** The state space consists of the relative positions of 1) the red drone, 2) the blue drones, and 3) the restricted airspace over three time steps. In our multi-agent setting comprising five drones, each drone has a partial observation of the environment, where it lacks the capability to perceive the presence and actions of its peer drones directly. Each blue drone has an observation that includes 1) the relative distance to the red drone if detected along the x and y coordinates (in meters) and 2) the relative distance to the center of the restricted zone along the x and y coordinates (in meters). To capture the context beyond the current drone position, we aggregate drone positions over 3 consecutive time steps, resulting in a state with $(2 + 2) \times 3 = 12$ features by stacking three consecutive time steps together.

2. **Action space:** The action space consists of a single continuous action, rotation,

which lies in the range $[-1, 1]$. We discretize this into two discrete actions: positive and negative rotation. The agent must choose between a positive or negative rotation at each time step.

3. **Reward function:** The blue drones receive a positive reward if they successfully neutralize the red drone and a negative reward if the red drone enters the designated target area.

   The team's reward function is defined as follows:

   $$R(s) = \begin{cases} +1 & \text{if any blue drone neutralizes the red drone} \\ -1 & \text{if the red drone enters the restricted zone} \end{cases}$$

   Additionally, at every time step, the blue drones receive a shaping reward, $R_I(s)$, proportional to their relative distance from the red drone in consecutive time steps

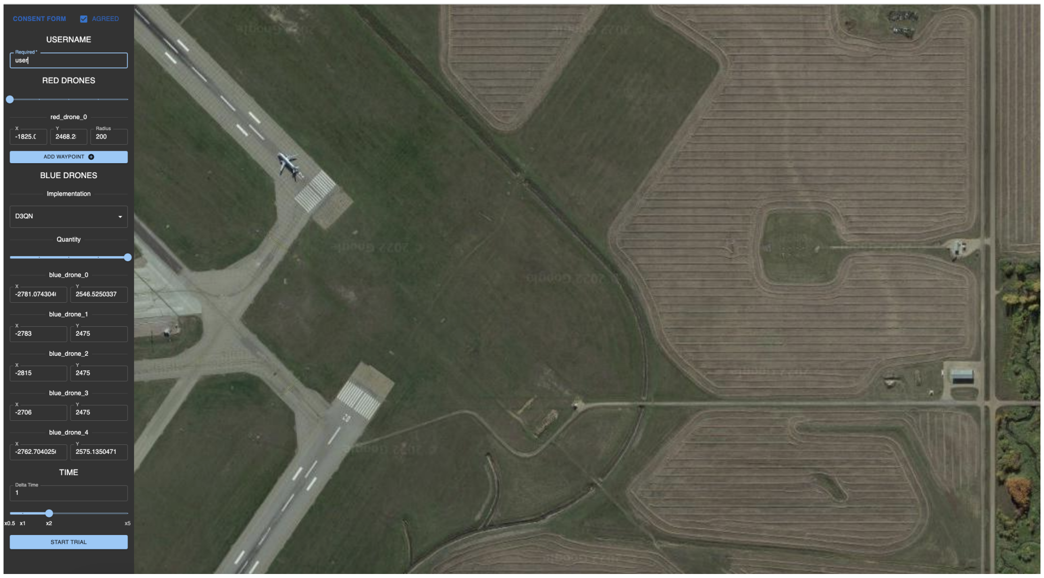
   $$R_I(s) \propto (d_{t-1}(b, r) - d_t(b, r))$$

   where $d_t(b, r)$ and $d_{t-1}(b, r)$ refer to the relative Euclidean distances between the blue drone and the red drone at time steps t and $(t-1)$, respectively. We use potential-based reward shaping as the optimal policy is guaranteed to be invariant [22–25].

We train the agents in multi-agent centralized training and decentralized execution (CTDE) settings where each agent has its own observation space as defined in the MDP formulation. Each agent has a similar reward function based on their current location and action. These agents are trained in parallel using Cogment [96]. Cogment is an open-source platform enabling training and operating various multi-agent RL and human-in-the-loop learning algorithms in a distributed way due to its microservice architecture.

3.3 User-interface for Advice

To foster effective human-agent collaboration within a simulated airport environment, we developed a user interface that comprehensively visualizes the airport and its nearby areas. This interface, developed using JavaScript and primarily leveraging the React front-end framework, serves as a nexus where autonomous ally drones and human operators collaboratively safeguard restricted zones, which are inherently vulnerable to potential adversarial drone incursions.



USERNAME
Required*: user

red_drone_0
X: -1825.0 | Y: 2468.2 | Radius: 200

Implementation: D3QN

blue_drone_0
X: -2781.074304 | Y: 2546.5250337

blue_drone_1
X: -2783 | Y: 2475

blue_drone_2
X: -2815 | Y: 2475

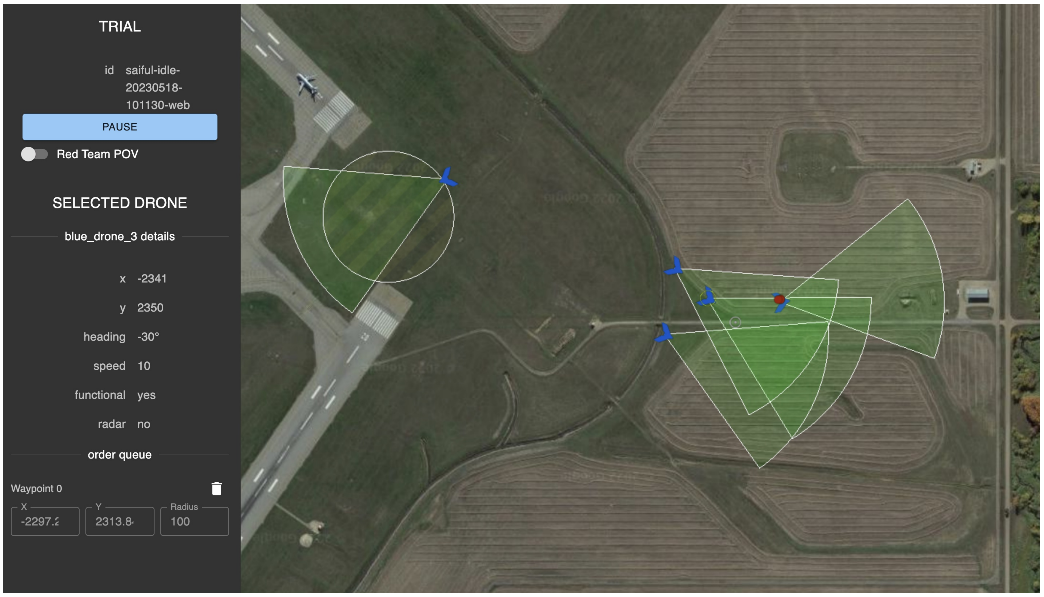
blue_drone_3
X: -2706 | Y: 2475

blue_drone_4
X: -2762.704025 | Y: 2575.1350471

Delta Time: 1

Figure 3.2: A trial configuration view of user interface.

Our user interface has two distinct views: a trial configuration form and an interactive trial run time view. Figure 3.2 shows the starting view of the trial configuration form that empowers users with the flexibility to tailor various parameters. Specifically, users can modify the composition and positioning of the blue team drones, select their underlying AI algorithm, and decide on the potential involvement of a human operator. These options allow the user to operate and control the agent, where the agent is trained with the D3QN algorithm. Additionally, configurations extend to determining the number of adversarial red drones and charting their trajectory toward the restricted zone, with an added provision to stipulate the simulation's update frequency.



id saiful-idle-20230518-101130-web

blue_drone_3 details

x -2341
y 2350
heading -30°
speed 10
functional yes
radar no

Waypoint 0
X -2297.2
Y 2313.8
Radius 100

Figure 3.3: An interactive trial run time view user interface for human operators to control the agents.

Transitioning to the trial run time interface, as depicted in Figure 3.3, presents an aerial perspective of the environment. This view delineates the drones, their respective detection range ground radar, and any identified red drones. Enhanced interactivity is facilitated by enabling users to select individual drones, assign or eliminate waypoints by interacting directly with the map, and manage the simulation's progression by pausing or resuming it. To control the ally drones or modify the drone trajectories, humans can define the waypoints for each drone separately. To *add* a waypoint, a human can select the drone they want to control using a right mouse click and choose the point where the agent needs to go as a next step using a left mouse click. Users can add as many waypoints as they want to control the drone. Similarly, humans can delete waypoints from the left side of the panel using the delete button shown on the bottom left side. To *delete* a specific drone's waypoint, operators need to select the drone using the mouse first and delete the waypoint shown in the bottom left panel. Operators can also delete any number of waypoints of the selected drone. Furthermore, users can seamlessly pan and zoom within the map for a more detailed inspection. This interface aims to provide a platform for managing mission-critical data and facilitating user interaction and control over multiple agents. Users can act as *operators*, providing demonstrations to guide agents or intervene to correct the policy of a trained agent.

Chapter 4

Experiments

In this chapter, we report the environment configuration, experimental settings, and performance metrics used in our experiments.

4.1 Environment Configuration

We used two distinct environment configurations for our experiments based on recommendations from defense experts:

Three Waypoint Scenario (Simple Scenario): In this setup, the starting positions of the five blue drones are determined within a circular region of a 200-meter radius near the restricted zone, as shown in Figure 4.1. The red drone starts from a similar circular region, positioned 1,000 meters to the right of the restricted area. The starting positions of the drones are selected randomly at the beginning of every episode. We added three fixed waypoints between the starting positions of the blue and red drones. The Euclidean distance between two waypoints is 200 meters. These waypoints guide the red drone towards the restricted space within the time limits.

Continuous Waypoint Scenario (Complex Scenario): For this scenario, the blue drones' starting positions mirror those in the three waypoint scenario. However, the setup introduces only one random waypoint selected from a 200-meter radius circle between the initial positions of the blue and red drones, as shown in Figure 4.2. All the starting points for blue drones, red drone, and waypoints are selected randomly

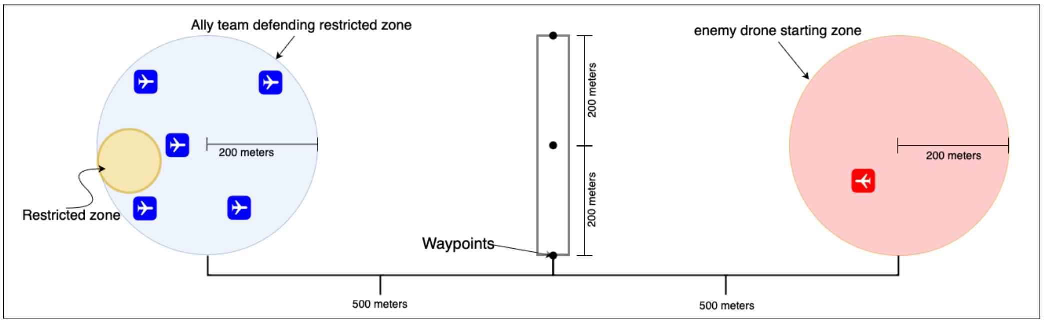


Figure 4.1: Three waypoint scenario (Simple scenario).

at the beginning of every episode. The random waypoint increases the uncertainty with respect to the location and makes the problem more complex. This waypoint is chosen strategically to prevent collisions between the blue and red drones.

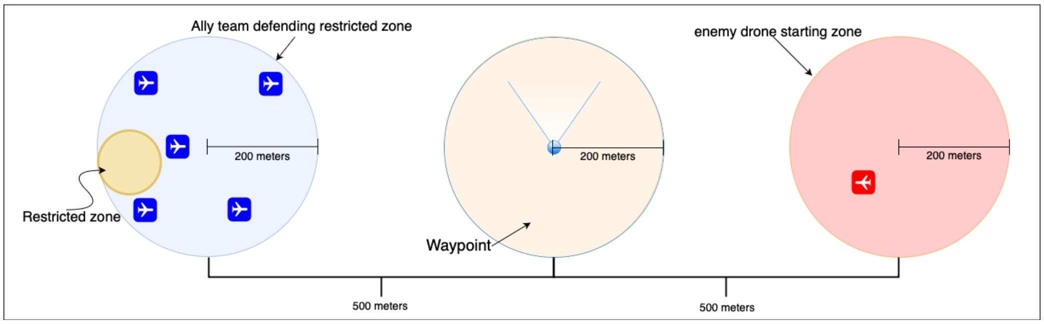


Figure 4.2: Continuous waypoint scenario (Complex scenario).

4.2 Performance Metric

We evaluate our results across two dimensions: (1) task success and (2) human effort while giving advice. We use team performance across the first dimension and cognitive workload measures based on NASA Task Load Index [97, 98] questionnaires along the second dimension.

4.2.1 Team Performance

We evaluated the performance of the trained agent using the *success rate* as the performance metric. The success rate is the percentage of times the blue team wins over all evaluation trials. This metric was chosen because it provides a clear and intuitive measure of the agents' ability to defeat the red drone and is directly proportional to the average reward. We execute 30 evaluation episodes per 100 training episodes to compute the success rate. During the evaluation episodes, agents execute the learned policy without exploration. Our learning curves show the performance metric reported across the evaluation episodes.

4.2.2 Cognitive Workload

The cognitive workload measures are assessed using the NASA Task Load Index [97, 98]. This index evaluates six types of workload: mental demand, physical demand, temporal demand, performance, effort, and frustration. Each participant selects a score ranging from 1 to 21 using a 21-point slider for all six workload types.

4.3 Demonstration Collection Methodology

To collect demonstrations, a human teacher either controls the agent prior to training or a partially trained agent trained in the same environment with randomized starting position of the drones. We collected demonstrations in three ways: (1) the trained agent records the demonstration using agent's sensors in the agent buffer. No mapping is required because the associated state-action pairs are captured directly from the trained agent's sensors. For our experiments, we use a fully trained D3QN agent to generate $2,500$ agent demonstrations. These demonstrations are referred to as **agent demonstrations** in our experiments. (2) A human teacher demonstrates the task using our developed user interface, and the demonstration is collected in a database using a Cogment trial datastore and converted with embodiment mapping later for use. These demonstrations are referred to as **human demonstrations** in our experiments. (3) A human teacher controls a trained agent through the user interface and corrects its policy, thereby providing demonstrations, also referred to as **policy-corrected demonstrations**. We engaged 11 individuals for collecting demonstrations, collectively gathering 500 human demonstrations, with each participant completing at least 30 episodes. Similarly, we collected 500 human demonstrations while correcting the policy of a trained D3QN agent. All the human demonstration data collection and user study were done under the approval of the University of Alberta Research Ethics Board (REB number: Pro00107555).

4.4 Experimental Setup

To establish a foundational benchmark and validate the functionality of the simulated environment, we implemented a heuristic-based decision-making algorithm tailored for drone operations. Our heuristic agents prioritize minimizing the distance to the enemy drone and either intercept their trajectory or follow their tracks for strategic positioning. In the subsequent sections, we use "HA" to denote a heuristic-based agent. The "HM" refers to the average winning percentage of human demonstrators who demonstrated through the designed interface. The winning percentage for average human demonstration is 62%, with a standard deviation of 17% in the simple scenario, and 64%, with a standard deviation of 7% for the continuous complex scenario. We only considered demonstrations where either the agent or human won the game to account for good quality demonstrations.

Algorithm abbreviation	Human Demo	Agent Demo
D3QN-0-0	0	0
D3QN-0-2500	0	2,500
D3QN-500-2000	500	2,000
D3QN$_{HM}$-500-0	500	0
D3QN$_{PC}$-500-0	500	0
D3QN-0-500	0	500

Table 4.1: Algorithm abbreviation with the number of human and agent demonstrations for each algorithm. Here, the subscripts HM represent human demonstrations, and PC represents policy-corrected demonstrations.

This study incorporated an autonomous agent programmed to learn and defend the airport from enemy attacks using DRL. Leveraging the capabilities of Q-value-centric DRL, we aim to achieve rapid policy convergence [54, 85, 99], thereby enhancing the feasibility of its application in more expansive systems. Our baseline was the D3QN algorithm, training from scratch without additional guidance, denoted as D3QN-0-0. We adopt deep Q-learning from demonstration [13] with human experience replay buffer and agent experience reply buffer to leverage demonstrations inside D3QN from either agent, human or from a mix of human and agent demonstrations[1]. We use D3QN-0-2500 and D3QN-0-500 to represent D3QN agents trained with additional agent demonstrations of 2,500 and 500, respectively. D3QN-500-2000 denotes D3QN agents trained with a mix of 500 human demonstrations and 2,000 agent demonstrations. Table 4.1 displays the proportions of human demonstrations and agent demonstrations used by each algorithm. In addition, we use D3QN$_{HM}$-500-0 and D3QN$_{PC}$-500-0 to represent D3QN agents trained with human and policy correction demonstrations, respectively. For experiments with D3QN-0-2500, D3QN-0-500, D3QN-500-2000, D3QN$_{HM}$-500-0, and D3QN$_{PC}$-500-0, we sampled 30% demonstration samples from expert experience replay memory and 70% from agent replay memory consisting of past agent experiences.

After testing our models with various proportions of demonstration data, we determined that the learning algorithm's performance was not significantly affected by the agent experience and demonstrations ratio, detailed as in Section 5.4.1. Hence, we set these proportions to 70% agent experience and 30% experience from expert experience replay memory for the experiments. To update the network, the training algorithm sampled mini-batches from the demonstration data and applied the double Q-learning loss and the n-step double Q-learning loss described in Section 2.3.2. For a fair comparison, we did not use any pre-training for the reported experiments. The Q-function of D3QN is updated by equation 2.3; D3QN-0-2500, D3QN-0-500, D3QN-

[1] Agent demonstrations refer to demonstrations generated by a D3QN agent that was trained to an average success rate of 85% +/- 10% in our airport security scenario.

500-2000, D3QN$_{HM}$-500-0, and D3QN$_{PC}$-500-0 are updated using equation 2.5. The expert margin in equation 2.3.2 was set to $M = 0.8$ in alignment with prior work [13].

All the reported experimental results are averaged over five runs with different seed values, and standard deviations are reported. The results in Figure 5.1 and Figure 5.2 report the mean and standard deviation over five runs, with the y-axis indicating the winning percentage and the x-axis denoting the number of training episodes. To determine significant differences between various baselines and our method, we employed the Mann-Whitney U test (also known as the Wilcoxon rank-sum test) [100]. This non-parametric statistical test was chosen for its robustness against deviations from normality assumptions, unlike other tests such as the t-test and ranked t-test. We observed that the normality assumption was violated in certain results, making the Mann-Whitney U test the preferred choice over the other tests. The use of the Mann-Whitney U test was twofold: firstly, it was used to assess the final performance of evaluation episodes, and secondly, to analyze the area under the learning curve (AUC), reflecting learning progress during training episodes. To calculate AUC, we use the mean reward on 30 evaluation episodes per 100 training episodes by executing the current policy without exploration. We computed the average AUC from five independent runs and use the composite trapezoidal rule included in numpy and scipy libraries for implementation. Hyper-parameter tuning was done using grid search to identify the best parameters for the algorithms, which were then consistently applied across all experiments in both scenarios, as detailed in Appendix C.

Chapter 5

Results

In this chapter, we discuss the experiments and report the computational results of our proposed method. Experiments were conducted in two scenarios (described in Section 4.1) to study the impact of different kinds of teaming. In addition, we present an overview of our user study design and report the results.

We aim to investigate the following research questions:

RQ1: How well does a trained RL agent perform in this environment?

RQ2: Does agent and/or human demonstration help the RL agent learn more efficiently?

RQ3: Does human policy correction help agents learn more efficiently, and how does this compare to learning with demonstration advice?

RQ4: How do humans experience this process?

5.1 Computational Results

We start by discussing the outcomes of the simple scenario, followed by an analysis of the results of the complex scenario. To evaluate the effectiveness of the agent and human demonstrations in the simple scenario, we compare D3QN-0-2500, D3QN-0-500, and D3QN-500-2000 with HA, HM, and D3QN-0-0 as baselines.

To answer RQ1, we trained D3QN-0-0 on the simple scenario and reported its success rate in Figure 5.1 based on 30 evaluation episodes per 100 training episodes. The D3QN-0-0 agents reach a success rate of approximately 90% in 3,500 episodes. The trained agent outperforms the baseline HA and HM, which have a success rate of 60% and 63%, respectively. D3QN-0-0 agents outperform both HA and HM.

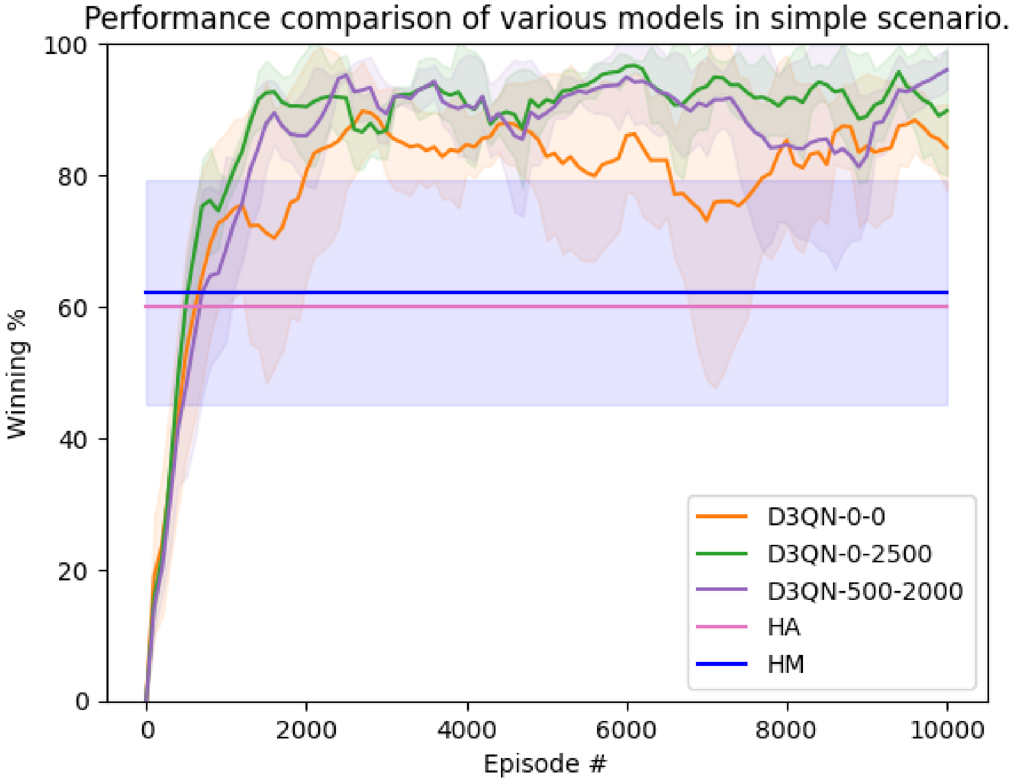


Figure 5.1: The success rate comparison for D3QN-0-0, D3QN-0-2500, D3QN-500-2000, HA, and HM performance in a simple scenario. Here, the number in the suffix represents the number of demonstrations from humans and the number of demonstrations from an agent. HA and HM represents heuristic based agent and the average winning percentage of human demonstrators, respectively.

To answer RQ2, we see that D3QN-0-2500 reaches a success rate of more than 90% in 1,600 episodes, outperforming D3QN-0-0 on average final performance as shown in Figure 5.1; the average AUC computed over five independent runs, as shown in Table 5.1 also supports the above claim. In examining the levels of winning percentage between D3QN-0-0 and D3QN-0-2500, the Mann-Whitney U test results in Table 5.2

show a significant difference between them. At the end of learning, both algorithms converge to the same final performance (around 90%). This supports our claim that agent demonstrations make the RL agent more sample efficient in our environment, consistent with existing results in the literature [13, 41].

Algorithm abbreviation	Mean final performance(± SD)	Mean AUC(± SD)
D3QN-0-0(s)	74.18(± 9.80)	0.72(± 0.08)
D3QN-0-2500	90.97(± 7.58)	0.88(± 0.02)
D3QN-500-2000	96.0(± 2.95)	0.87(± 0.04)
D3QN-0-0(c)	74.18(± 9.80)	0.69(± 0.03)
D3QN$_{HM}$-500-0	83.77(± 2.17)	0.74(± 0.05)
D3QN-0-500	86.22(± 9.01)	0.75(± 0.02)
D3QN$_{PC}$-500-0	95.33(± 3.54)	0.85(± 0.02)

Table 5.1: Comparison of different algorithm mean final performance of 5 runs and mean area under the learning curve (AUC). Here, (s) and (c) represent the simple and complex scenarios respectively. The subscripts HM represent human demonstrations, and PC represents policy-corrected demonstrations.

We also trained the learning agent with a mix of agent and actual human demonstrations, denoted D3QN-500-2000, as shown in Figure 5.1. We sampled an equal proportion of human and trained agent demonstrations in every mini-batch used for training. We used a mix of both types of demonstrations due to the lack of human demonstrations collected in our user study. The Mann-Whitney U test shows no significant learning improvement when compared to D3QN-0-2500 in a simple scenario; however, the performance is still statistically significant than the baseline D3QN-0-0, shown in Table 5.2.

Similarly, for our complex scenario, we trained D3QN-0-0 and plotted its success rate in Figure 5.2. The agent reaches a success rate of 80% in $3,500$ episodes, which is 10% less than the D3QN-0-0 performance in a simple scenario. The trained D3QN-0-0 agent outperforms the baseline HA and HM, with a success rate of 53% and 64%,

Algorithm		P-value	U-statistic	Effect size	Significance
D3QN-0-2500 vs. D3QN-0-0	Performance	0.036	2.0	0.84	Yes
	AUC	0.011	0.0	1.0	Yes
D3QN-500-2000 vs. D3QN-0-0	Performance	0.011	0.0	1.0	Yes
	AUC	0.023	1.5	0.88	Yes
D3QN-500-2000 vs. D3QN-0-2500	Performance	0.400	8.0	0.36	No
	AUC	0.916	11.5	0.08	No
D3QN-0-500 vs. D3QN-0-0	Performance	0.035	2.0	0.84	Yes
	AUC	0.036	2.0	0.84	Yes
D3QN$_{HM}$-500-0 vs. D3QN-0-0	Performance	0.673	10.0	0.19	No
	AUC	0.036	2.0	0.84	Yes
D3QN$_{HM}$-500-0 vs. D3QN-0-500	Performance	0.140	20.0	0.60	No
	AUC	0.828	11.0	0.12	No
D3QN$_{PC}$-500-0 vs. D3QN-0-0	Performance	0.011	0.0	1.0	Yes
	AUC	0.011	0.0	1.0	Yes
D3QN$_{PC}$-500-0 vs. D3QN-0-500	Performance	0.036	2.0	0.84	Yes
	AUC	0.011	0.0	1.0	Yes
D3QN$_{PC}$-500-0 vs. D3QN$_{HM}$-500-0	Performance	0.011	0.0	1.0	Yes
	AUC	0.011	0.0	1.0	Yes

Table 5.2: Comparison of different algorithm significance using the Mann-Whitney U test. Here, subscripts PC and HM represent policy corrected demonstration and human demonstrations, respectively.

respectively. With the complex scenario, the performance of HM remains the same or improves marginally, whereas the performance of HA and D3QN-0-0 drops significantly. This performance changes suggests that humans improve their performance playing more times and outperform HA in more complex environments that require greater generalization capabilities. In conclusion, despite the inherent randomness and complexity of the scenario that affects performance, our trained D3QN-0-0 agent shows superior performance compared to both HA and HM, supporting our earlier finding regarding RQ1.

Figure 5.2 presents the learning curves for D3QN-0-0, D3QN-0-500, D3QN$_{HM}$-500-0, and D3QN$_{PC}$-500-0 in our complex scenario, allowing a comparison of their

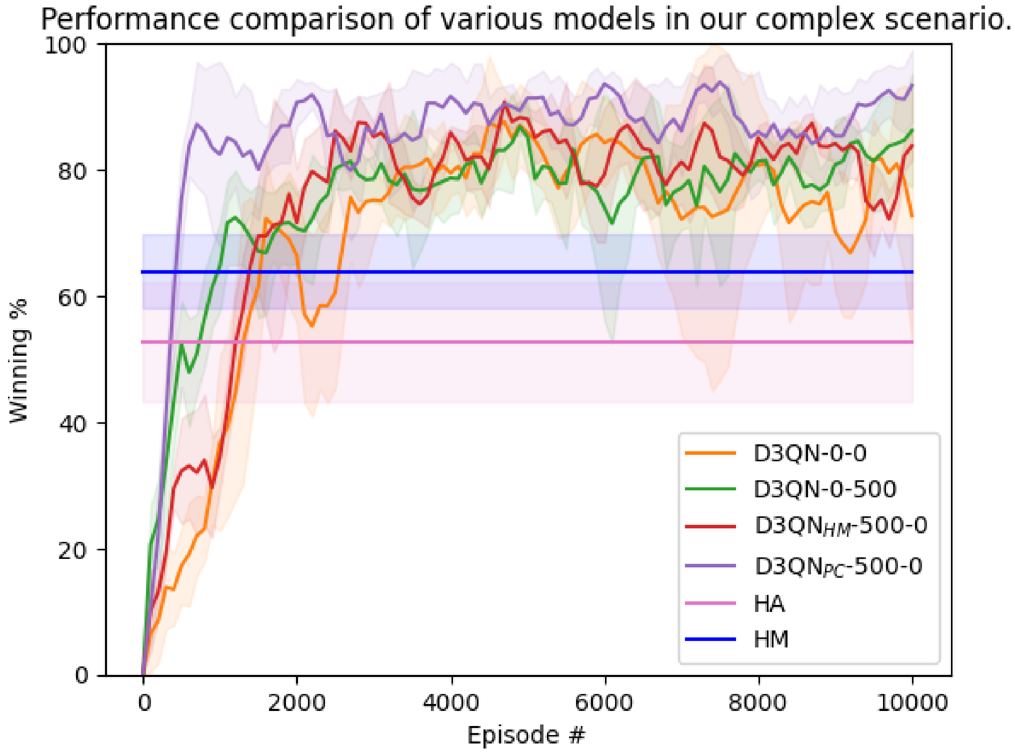


Figure 5.2: The success rate comparison for D3QN-0-0, D3QN-0-500, D3QN$_{HM}$-500-0, and D3QN$_{PC}$-500-0, HA, and HM performance in a complex scenario. Here, the number in the suffix represents the number of demonstrations from humans and the number of demonstrations from an agent.

performances. In particular, D3QN-0-500 shows a higher success rate compared to D3QN-0-0, as evidenced by the results of a Mann-Whitney U test (U=2.0, p=0.035, effect size=0.84), which confirms significant performance differences between these two models. From the Table 5.2, we see that D3QN$_{HM}$-500-0 shows no statistically significant difference on final performance, but shows statistically significant difference in AUC (U=2.0, p=0.036, effect size=0.84). However, the performance gap between the D3QN$_{HM}$-500-0 approach and D3QN-0-500 is not statistically significant in our complex scenario. These findings are consistent with our conclusions from the simple scenario and answer our RQ2.

To address RQ3, we use policy-corrected demonstrations to train our agent, denoted as D3QN$_{PC}$-500-0. We find a significant performance increase in D3QN$_{PC}$-500-

0 compared to all our baseline methods and learning from agent and human demonstrations. D3QN$_{PC}$-500-0 achieves a success rate of more than 80% in $1,000$ episodes, while D3QN-0-0 takes $5,000$ or more episodes. The results of the Mann-Whitney U test show significant differences in performance between D3QN$_{PC}$-500-0 and D3QN-0-0 and also between D3QN$_{PC}$-500-0 and D3QN-0-500 (see Table 5.2). Similarly, the Mann-Whitney U test between D3QN$_{HM}$-500-0 and D3QN$_{PC}$-500-0 (p=0.011, U=0.0, effect size=1.0) indicates significant differences between them. These performance supports our claim that human demonstrations make the RL agent more sample efficient in our complex environment, as seen from the success of D3QN$_{PC}$-500-0 and D3QN$_{HM}$-500-0, which is consistent with previous results from the simple scenario. In addition, the policy correction approach shows a 10% performance improvement over D3QN-0-0 and demonstrates the effectiveness of human-AI collaboration in our proposed scenarios, which answers RQ3 affirmatively.

5.2 User Study Design

To investigate whether human demonstrators can make RL agents more sample efficient, we have designed a human user study where users can provide advice in two ways: (1) by providing full demonstrations, referred to as human demonstrations, and (2) by providing partial demonstrations and intervening when necessary, referred to as policy-corrected demonstrations. Furthermore, to study the trade-off between human costs such as metal demand, physical demand, etc., and agent performance, this study contained survey questionnaires based on the NASA-TLX load index [97, 98] and a set of demographic questions. We hosted our developed system on Amazon AWS to conduct our user study and advertised the study on mailing lists of graduate and undergraduate computing science students at the University of Alberta and industry partner organizations. The participants undertook a control task that involved controlling agents to neutralize the enemy in our simulated environment with randomly set environment configurations. Our user study was conducted using a structured approach with the following sequential steps:

1. Read the task details and digitally sign a consent form.
2. Watch videos on how to use the web client and the user interface. In addition, watch videos of trained agents, controlling agent steps (setting up new waypoints or deleting the existing ones), providing a full demonstration, or providing a policy-corrected demonstration to teach the agents. This step is meant to teach the task to the participants.
3. Round one: provide demonstrations for 30–40 episodes, controlling all the ally drones.
4. Complete the NASA-TLX questionnaires based on their experience during the round one task.
5. Round two: provide policy corrections for trained agents for 30–40 episodes.
6. Complete the NASA-TLX questionnaires based on their experience during the round two task.
7. Complete the demographics questionnaire.

In the initial phase of our study, we collected human demonstrations from Round 1, focusing on our simple scenario. Later, we expanded the study to include a complete user study comprising Rounds 1 and 2 in our complex scenario. We used the same set of participants for both rounds, while these rounds were conducted separately. Participants spent approximately 30–40 minutes on each round to complete the experiments. To evaluate team-wise costs, such as mental, physical, and temporal demand, effort, etc., associated with human involvement, we calculate cognitive workload based on NASA-TLX questionnaires. In addition, demographic information collected included gender, age, current job/position, level of defense experience, level of drone experience, level of simulated drone control, and level of gaming experience.

5.3 Measurement of Cognitive Load

To measure cognitive load, we conducted a user study described in Section 5.2 and used the NASA TLX survey questionnaires detailed in Appendix E. We abbreviate standard deviation as SD and interquartile range as IQR to describe the demographic information. In our user study, 30 people consented to participate, and 11 both provided demonstration and completed the survey. The average age was 26.02 (SD of 6.42), ranging from 18 to 50. Analysis of the responses to the questionnaire indicated that 90% of the participants reported being familiar with AI and games. In contrast, 80% of the participants reported no prior familiarity with drone control or defense strategies.

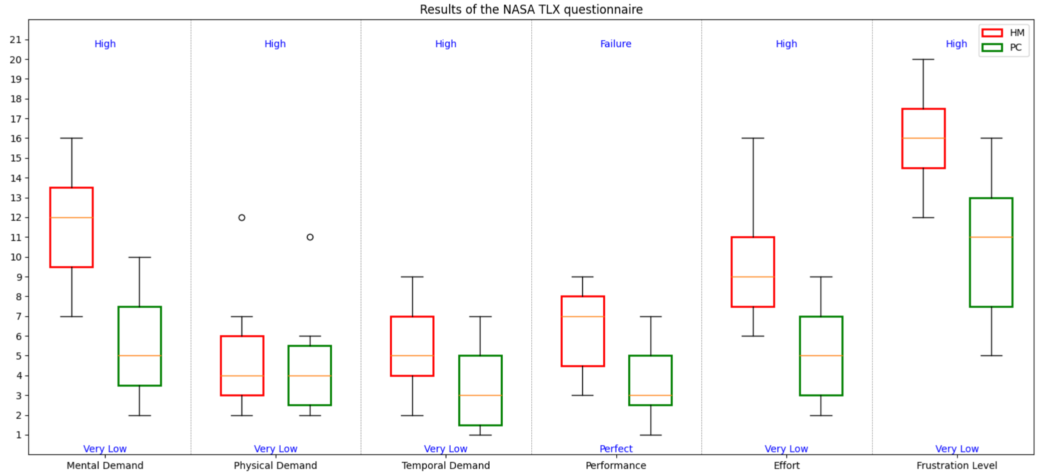


Figure 5.3: Results of the NASA TLX questionnaire. Here, PC represent policy-corrected demonstrations and HM represents human demonstrations.

As presented in Figure 5.3, participants reported much lower mental demand, temporal demand, and effort during policy correction, denoted as "PC", compared to human demonstrations, denoted as "HM". Similarly, participants achieved much higher performance when they corrected the policies than when they fully controlled the agents (i.e., provided demonstrations). To evaluate the statistical significance of these observations, a Mann-Whitney U test was performed, comparing the cognitive load between PC and HM. As detailed in Table 5.3, the results show significant differences between PC and HM across all dimensions of the NASA Task Load Index (NASA-TLX), except for physical demand. These findings suggest that human operators experience more mental demands and frustration when fully controlling drones, especially in our complex simulator. In contrast, PC, a human-AI collaboration approach, requires less effort and yields superior performance outcomes. This addresses our fourth research question, RQ4, regarding the human experience in this process.

Measure	**P-Value**	**U-Statistic**	**Correlation**	**Significance**
Mental Demand	0.0006	113.0	0.8677	Yes
Physical Demand	0.5711	69.5	0.1487	No
Temporal Demand	0.0430	91.5	0.5123	Yes
Performance	0.0098	100.0	0.6528	Yes
Effort	0.0026	106.5	0.7603	Yes
Frustration	0.0014	109.5	0.8099	Yes

Table 5.3: Mann-Whitney U Test for NASA-TLX between PC and HM, here correlation represents Rank-Biserial Correlation.

5.4 Ablation and Analysis

We also conducted several ablation studies to examine the effect of proportions, quantity, and diversity of demonstration data.

5.4.1 Effect of Demonstration Proportions

We evaluated the training models by selecting varying proportions of demonstrations from experience reply memory and the agent's own training experience from the environment in our simple scenario. Specifically, we employ the D3QN-0-2500 algorithm, using 2,500 agent demonstrations in expert experience reply memory. We sampled mini-batches in ratios of 30:70, 50:50, and 70:30 percent from the expert experience reply memory and agent replay memory consisting of past agent experiences, respectively.

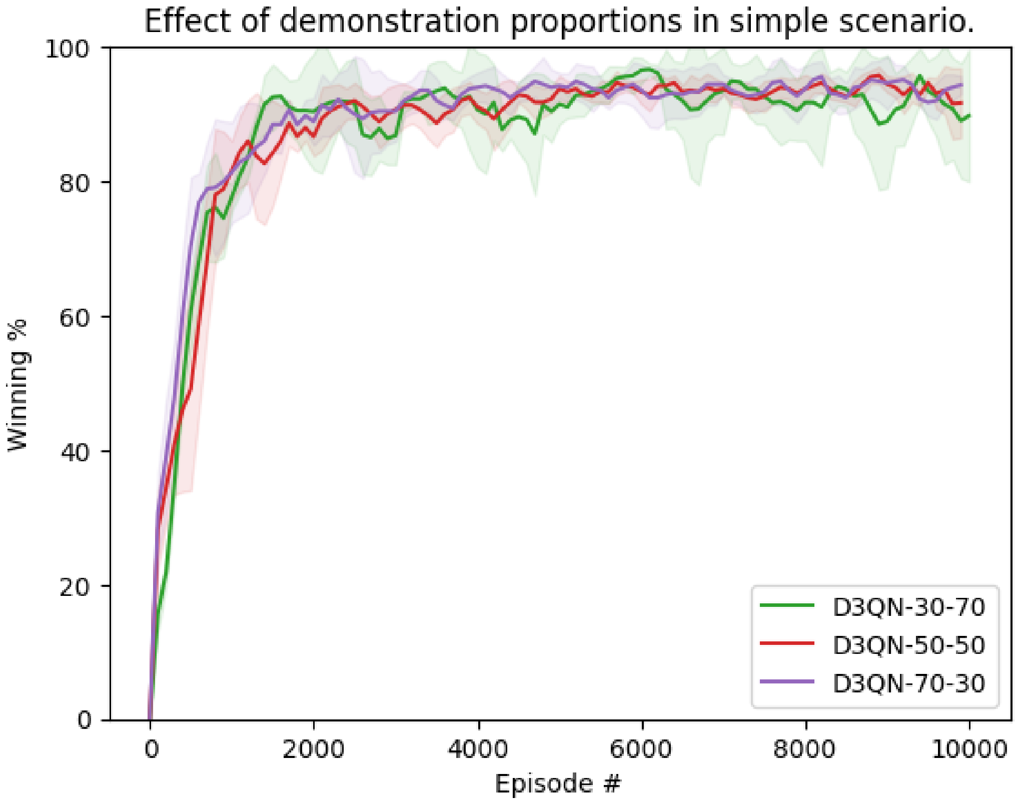


Figure 5.4: The success rate comparison for D3QN trained with 2,500 agent demonstrations (D3QN-0-2500) with various proportions in simple scenario. Here, the suffix represents the proportion of demonstrations from expert experience reply memory and agent's own experience, respectively.

From our experiment, we concluded that trained agent demonstration and agent training experience from the environment proportions did not significantly impact the learning performance of D3QN-0-2500, as seen in Figure 5.4. In addition, we conducted the Mann-Whitney U test on final performance, which yields a U-Statistic of 12.0 with a p-value of 0.99 and effect size of 0.04, indicating that there is no significant difference between D3QN-30-70 and D3QN-50-50 performance. Similarly, D3QN-50-50 and D3QN-70-30 show no significant difference with a U-Statistic of 10.0, a p-value of 0.653, and an effect size of 0.19. We also run the Mann-Whitney U test between D3QN-30-70 and D3QN-50-50 AUC which shows no significant difference with a p-value of 0.67, effect size of 0.19 and U-Statistic of 15.0; similarly, D3QN-50-50 and D3QN-70-30 shows no statistical significant difference in AUC (U statistic:

9.0, P-value: 0.52 and effect size: 0.28)

5.4.2 Effect of Demonstration Quantity

We also evaluated our approach in the simple scenario, as shown in Figure 4.1, by increasing the number of demonstrations in the expert experience reply buffer. In all experiments, we use 30% demonstration proportion from the expert experience reply memory and 70% from the agent experience reply memory. We observed a significant improvement in agent's performance with the increased number of demonstrations, as shown in Figure 5.5.

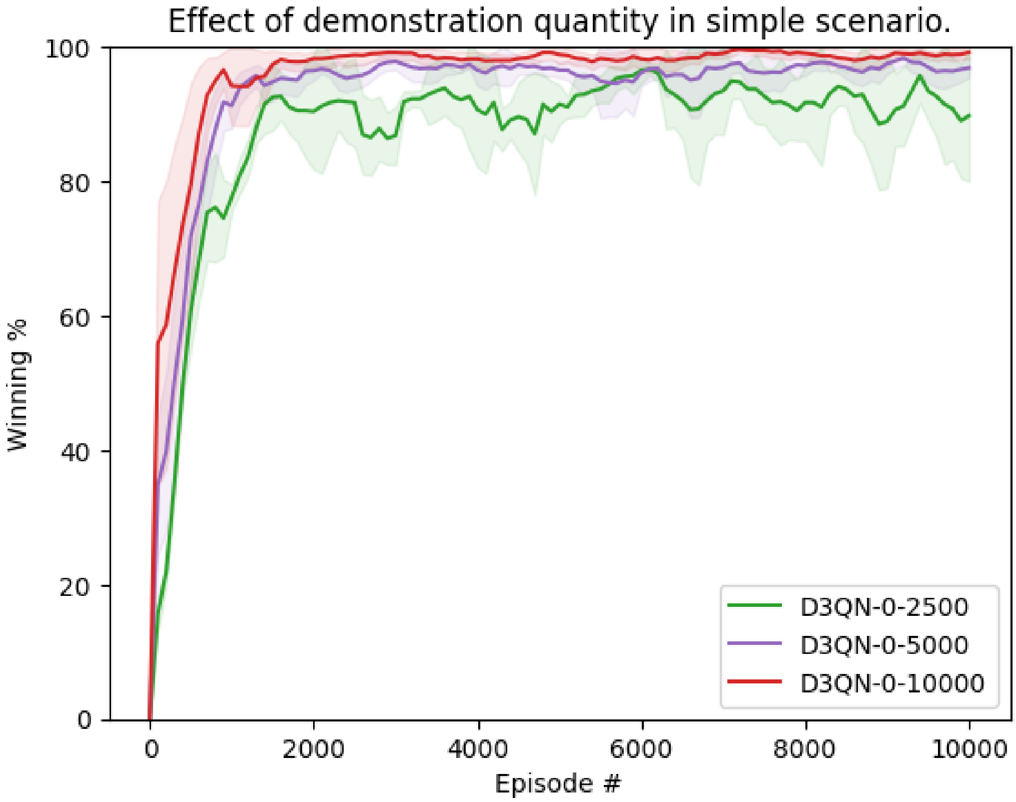


Figure 5.5: The success rate comparison for D3QN with agent demonstrations with various sizes of demonstration data in our simple scenario. Here, the suffix 2,500, 5,000, and 10,000 represents the total amount of agent demonstration in the expert experience reply memory.

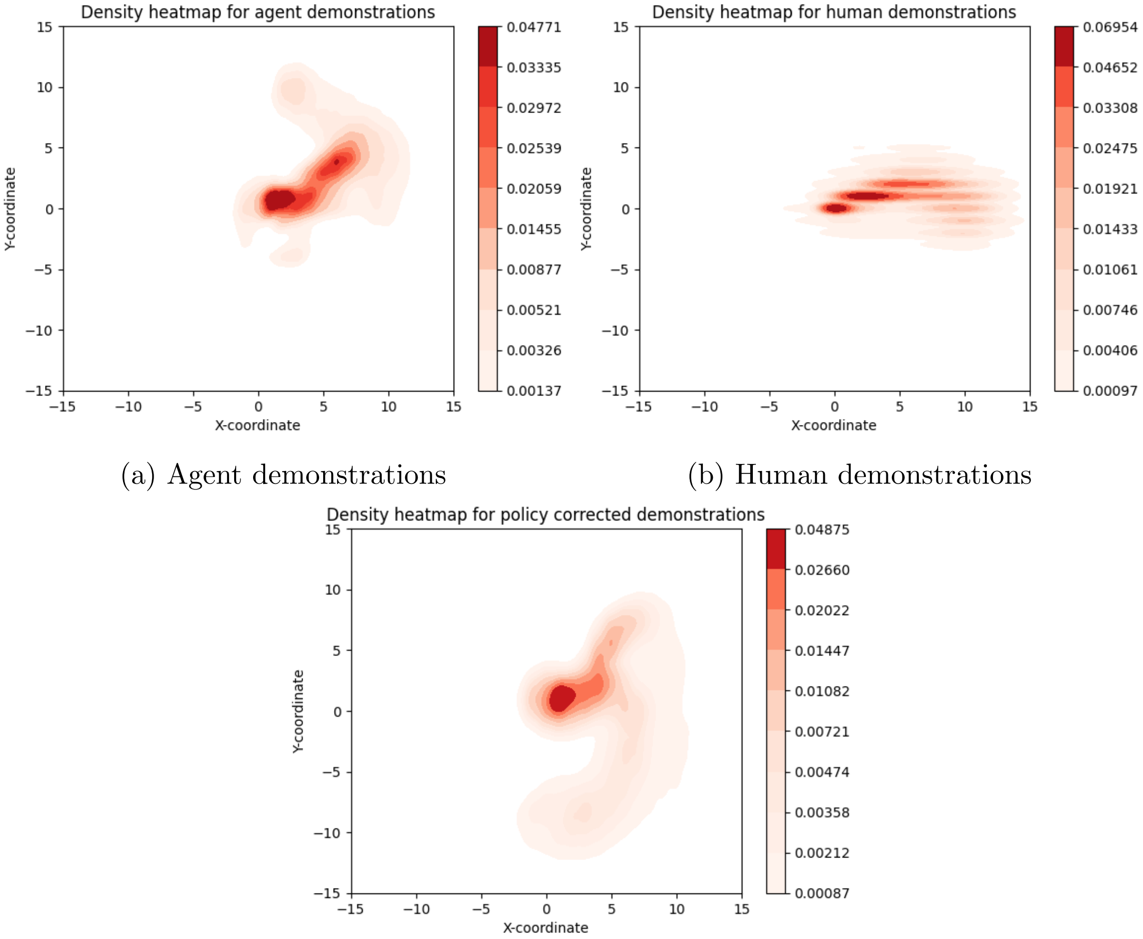
The green curve represents the D3QN with demonstration, where data in the buffer was 2,500. The purple and red curves represent the D3QN guided by the demonstration, where data in the expert experience reply buffer was 5,000 and 10,000, respectively. A Mann-Whitney U test was done on the final performance which yields a U statistic of 2.0 with a p-value of 0.035 and an effect size of 0.84 suggesting significant differences between the 5,000 demonstration data used instead of the 2,500 data in the demonstration buffer. The Mann-Whitney U-Statistic of 2.5 and a p-value of 0.041 with an effect size of 0.8 suggest significant differences between 10,000 and 5,000 demonstration data in the buffer. Similarly, a Mann-Whitney U test was done on the area under the learning curve for D3QN-2500 and D3QN-5000 demonstrating shows significant difference(U statistic: 0.0, P-value: 0.011, effect size: 1.0). Likewise, the Mann-Whitney U test shows a significant difference between D3QN-5000 and D3QN-10000 on the area under the learning curve. From the above experiment, we conclude that increasing the number of demonstrations leads to better performance.

5.4.3 Analysis of Diversity in Demonstration

From the ablation studies in Sub-sections 5.4.1 and 5.4.2, we conclude that the quantity of demonstrations impacts the performance outcomes than the proportion of demonstrations. This leads us to hypothesize that the diversity in demonstrations is the primary driver of enhanced performance.

We investigate the effect of demonstration diversity on the performance of trained agents, guided by agent demonstrations, human demonstrations, and policy-corrected demonstrations. There is often an abundance of trained agent demonstrations in a simulated real-world complex task, while human demonstrations are scarce due to the high human time and cost. We used 500 winning episodes of human demonstrations from 11 persons. Similarly, we use 500 winning episodes of agents and policy-corrected demonstrations.

We employ a qualitative approach to assess the diversity of demonstrations by plotting the trajectories of all three demonstration types and examining the spatial coverage within these visualizations. This method allows us to infer the range of



(a) Agent demonstrations

(b) Human demonstrations

(c) Human policy correction

Figure 5.6: Density heat map of winning five hundred episodes from trained agents, human users, and human policy corrections. Here, the x-axis and the y-axis represent the drones' scaled relative position.

strategies and behaviors exhibited across different demonstrations. In our analysis, we generate density heat maps for each demonstration type, which visually represent the frequency of occurrences at various points in the task space. Specifically, Figure 5.6(a), (b), and (c) correspond to the heat maps for the trained agents, actual human users, and human policy correction demonstrations, respectively. Although our simulated environment is a square space of size 6000 × 6000 meter^2, we plotted them in reduced space for better visualizations and understanding. The heat maps distinctly reveal that policy-corrected demonstrations show a broader coverage area than those

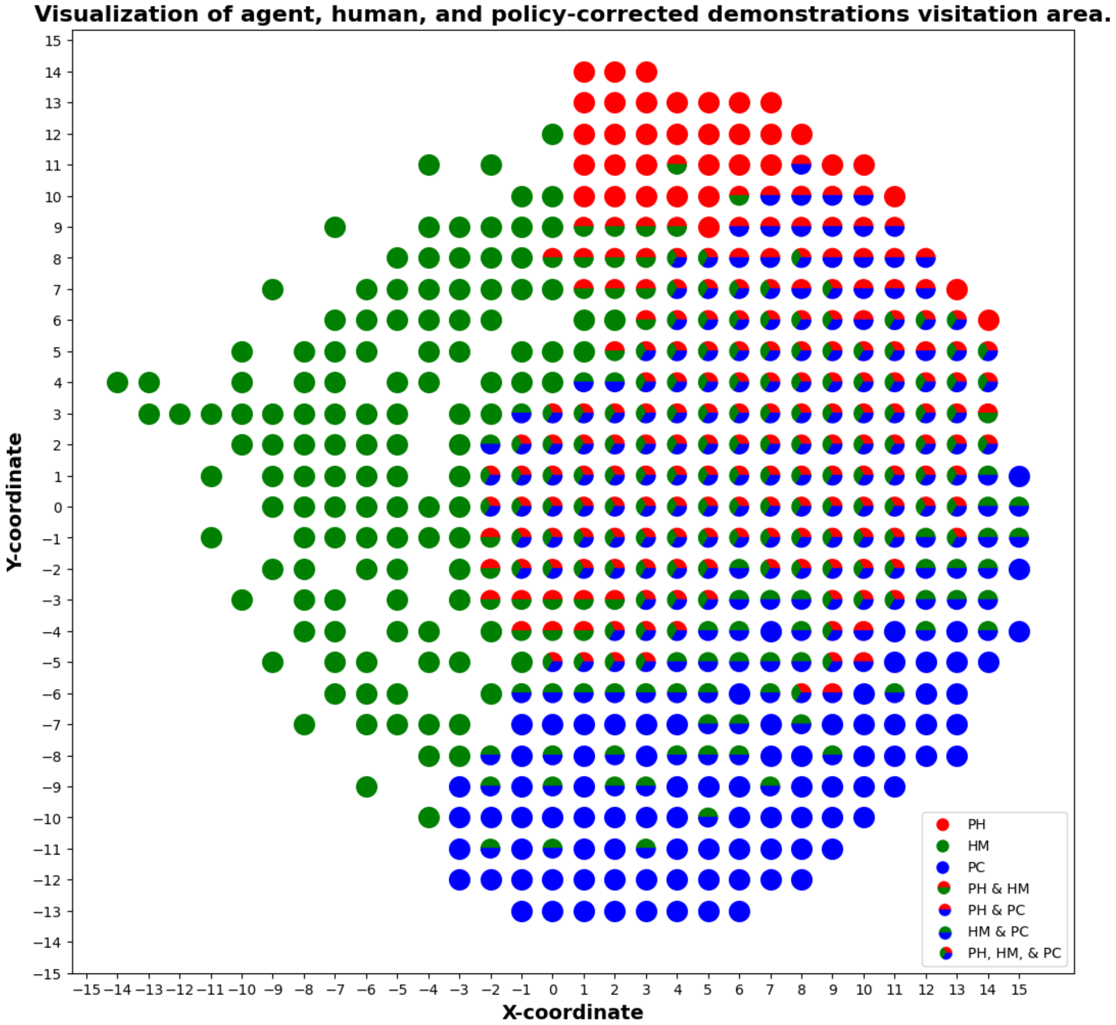
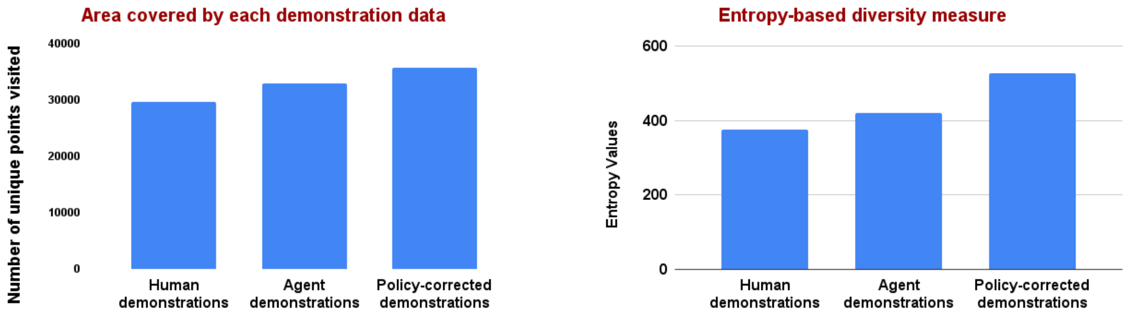


Figure 5.7: The area coverage comparison for human demonstrations (HM), agent demonstrations (PH) and policy-corrected demonstrations (PC) in complex scenario.

from humans and trained agents. This suggests a greater coverage in the demonstrations collected using policy correction, contributing to the enhanced performance observed earlier in experimental results for this advice-providing technique.

To better visualize the visitation area (by agents, humans, and human policy correction), we plotted the x and y coordinates of winning demonstrations in a projected space and marked them with different color codes, as shown in Figure 5.7. The plot clearly shows that the trajectories of humans and trained agent demonstrations overlap less. A diverse range of human actions, from individual to individual, underscores the complexity and variability inherent in human gameplay. In contrast, policy cor-



(a) Area covered by human demonstrations, agent demonstrations and policy-corrected demonstrations.

(b) Entropy-based diversity measure for human demonstrations, agent demonstrations and policy-corrected demonstrations.

Figure 5.8: Quantitative measure of diversity in human demonstrations, agent demonstrations and policy-corrected demonstrations. Here, higher entropy values represent more diversity.

rection effectively uses the strengths of both humans and agents, resulting in more state-area coverage encompassing all critical zones identified by both. Subsequently, we conducted a quantitative analysis of the area covered by each type of demonstration, the results of which are presented in Figure 5.8 (a). From Figure 5.8 (a), we can observe that policy-corrected demonstrations visited 4,116 unique points higher than agent demonstrations and 2,761 points higher than human demonstrations. We also employed an entropy-based approach to quantify the diversity of demonstrations, as suggested by Neumann et al. [101]. As shown in Figure 5.8 (b), the entropy values for policy-corrected demonstrations were over 20% higher than those for human and trained agent demonstrations. This empirical evidence confirms our previous observations, indicating that human policy correction introduces significantly diverse states, improving the learning process with various experiences.

Chapter 6

Conclusions and Future Work

In this work, we demonstrated that human-AI collaboration can be better than humans or agents alone in a complex multi-agent task. By developing a novel simulator and user interface, we have established a platform where humans and AI agents can collaborate and incorporate advice effectively with real-world dynamics. Our experimental results show that a trained RL agent performs better than a heuristic agent and humans in complex airspace-simulated environments. Our empirical findings underscore the value of incorporating human and policy-corrected demonstrations into the training of AI agents, revealing a marked improvement in the agents' learning efficiency and operational performance. In addition, we demonstrated that policy-corrected demonstrations, a human-AI collaboration approach, require less mental demand, temporal demand, and effort, yielding superior performance compared to humans alone. While significant, this study's findings are limited by its focus on a single enemy with full and free communication, not fully reflecting the complexity of real-world scenarios. In addition, only a small number of human participants were involved in the study, which may not represent the diverse strategies employed by various individuals in real-life situations. Future research directions include addressing these gaps by exploring more complex scenarios and diverse human expertise to enhance human-AI collaboration in real-world settings.

Boarder Ethics Statement: In transitioning our research from theory to practice, particularly in areas where AI is integral to critical defense operations, it's essential to acknowledge the potential risks and ethical concerns that come with this integration. To mitigate such potential risks and ethical considerations, our simulator environment is deliberately designed to ensure drones are not armed with life-threatening weaponry, focusing on defensive tasks to reduce the likelihood of misuse by malicious or non-malicious actors. Moreover, as much as we tried emulating real-life defense scenarios, the actual environment dynamics are still noticeably different from real defense applications. In addition, it would be much harder to get physical drones working and communicating together than actually getting the drones to learn via RL in a simulation. Our approach, while innovative, does bring with it certain risks, including decreased human attention, issues with data privacy, and ethical challenges in decision-making. Central to these risks is the potential for over-reliance on AI systems, which could lead to poor decision-making. This overreliance on AI can create a false sense of security, leading to diminished alertness and oversight by human operators, which could exacerbate the risks. To solve these concerns, we encourage research in the direction of developing explainable decision-making frameworks for such safety-critical applications to understand the impact of biased or erroneous human inputs on the model's decision-making capability.

Contribution: Saiful conducted a comprehensive literature review, formulated the research questions, and defined both the model and computational framework. He took the lead in data analysis, collected agent and human demonstration data, and was responsible for the implementation and execution of experiments. Saiful planned and conducted a user study, completed the data analysis from the user study, and interpreted the findings. In addition, he assisted with simulated platform design and tested both the prototype and final simulator.

Bibliography

[1] R. H. Kewley and M. J. Embrechts, “Computational military tactical planning system,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 32, no. 2, pp. 161–171, 2002.

[2] S. K. Gottipati, L.-H. Nguyen, C. Mars, and M. E. Taylor, “Hiking up that hill with cogment-verse: Train & operate multi-agent systems learning from humans,” in *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, 2023, pp. 3065–3067.

[3] F. Santoni de Sio and J. Van den Hoven, “Meaningful human control over autonomous systems: A philosophical account,” *Frontiers in Robotics and AI*, vol. 5, p. 15, 2018.

[4] D. Silver *et al.*, “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[5] M. G. Bellemare *et al.*, “Autonomous navigation of stratospheric balloons using reinforcement learning,” *Nature*, vol. 588, no. 7836, pp. 77–82, 2020.

[6] S. K. Gottipati *et al.*, “Learning to navigate the synthetically accessible chemical space using reinforcement learning,” in *International conference on machine learning*, PMLR, 2020, pp. 3668–3679.

[7] S. K. Gottipati *et al.*, “Maximum reward formulation in reinforcement learning,” *CoRR*, vol. abs/2010.03744, 2020. arXiv: 2010.03744. [Online]. Available: https://arxiv.org/abs/2010.03744.

[8] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” in *2012 IEEE/RSJ international conference on intelligent robots and systems*, IEEE, 2012, pp. 5026–5033.

[9] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[10] B. Schelble, L.-B. Canonico, N. McNeese, J. Carroll, and C. Hird, “Designing human-autonomy teaming experiments through reinforcement learning,” in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, SAGE Publications Sage CA: Los Angeles, CA, vol. 64, 2020, pp. 1426–1430.

[11] T. O’Neill, N. McNeese, A. Barron, and B. Schelble, “Human–autonomy teaming: A review and analysis of the empirical literature,” *Human factors*, vol. 64, no. 5, pp. 904–938, 2022.

[12] J. Ibarz, J. Tan, C. Finn, M. Kalakrishnan, P. Pastor, and S. Levine, "How to train your robot with deep reinforcement learning: Lessons we have learned," *The International Journal of Robotics Research*, 2021.

[13] T. Hester *et al.*, "Deep Q-learning from demonstrations," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, 2018, pp. 3223–3230.

[14] C. Kharyal, T. Sinha, S. K. Gottipati, F. Abdollahi, S. Das, and M. E. Taylor, "Do as you teach: A multi-teacher approach to self-play in deep reinforcement learning," in *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '23, London, United Kingdom: International Foundation for Autonomous Agents and Multiagent Systems, 2023, 2457–2459, ISBN: 9781450394321.

[15] A. Mandlekar *et al.*, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning*, PMLR, 2022, pp. 1678–1690.

[16] L. Torrey and M. Taylor, "Teaching on a budget: Agents advising agents in reinforcement learning," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, 2013, pp. 1053–1060.

[17] S. Frazier and M. Riedl, "Improving deep reinforcement learning in minecraft with action advice," in *Proceedings of the AAAI conference on artificial intelligence and interactive digital entertainment*, vol. 15, 2019, pp. 146–152.

[18] E. Ilhan, J. Gow, and D. Perez Liebana, "Action advising with advice imitation in deep reinforcement learning," in *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 2021, pp. 629–637.

[19] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei, "Reward learning from human preferences and demonstrations in atari," *Advances in neural information processing systems*, vol. 31, 2018.

[20] M. Palan, G. Shevchuk, N. Charles Landolfi, and D. Sadigh, "Learning reward functions by integrating human demonstrations and preferences," in *Robotics: Science and Systems*, 2019.

[21] K. Lee, L. M. Smith, and P. Abbeel, "PEBBLE: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training," in *International Conference on Machine Learning*, PMLR, 2021, pp. 6152–6163.

[22] A. Y. Ng, D. Harada, and S. J. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proceedings of the Sixteenth International Conference on Machine Learning*, 1999, pp. 278–287.

[23] S. M. Devlin and D. Kudenko, "Dynamic potential-based reward shaping," in *Proceedings of the 11th international conference on autonomous agents and multiagent systems*, IFAAMAS, 2012, pp. 433–440.

[24] T. Brys, A. Harutyunyan, H. B. Suay, S. Chernova, M. E. Taylor, and A. Nowé, "Reinforcement learning from demonstration through shaping," in *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[25] Y. Hu *et al.*, "Learning to utilize shaping rewards: A new approach of reward shaping," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 931–15 941, 2020.

[26] J. Humann and K. A. Pollard, "Human factors in the scalability of multirobot operation: A review and simulation," in *2019 IEEE international conference on Systems, Man and Cybernetics (SMC)*, IEEE, 2019, pp. 700–707.

[27] M. J. Barnes, J. Y. Chen, and F. Jentsch, "Designing for mixed-initiative interactions between human and autonomous systems in complex environments," in *2015 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE, 2015, pp. 1386–1390.

[28] T. Porat, T. Oron-Gilad, M. Rottem-Hovev, and J. Silbiger, "Supervising and controlling unmanned systems: A multi-phase study with subject matter experts," *Frontiers in psychology*, vol. 7, p. 568, 2016.

[29] J.-J. C. Meyer and R. J. Wieringa, *Deontic logic in computer science: Normative system specification*, 1994.

[30] J. van der Waa *et al.*, "Moral decision making in human-agent teams: Human control and the role of explanations," *Frontiers in Robotics and AI*, vol. 8, p. 640 647, 2021.

[31] A. P. Badia *et al.*, "Agent57: Outperforming the Atari human benchmark," in *International conference on machine learning*, PMLR, 2020, pp. 507–517.

[32] O. Vinyals *et al.*, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[33] C. Wu *et al.*, "UAV autonomous target search based on deep reinforcement learning in complex disaster scene," *IEEE Access*, DOI: 10.1109/ACCESS.2019.2933002. [Online]. Available: https://dx.doi.org/10.1109/ACCESS.2019.2933002.

[34] D. Martinez, L. Riazuelo, and L. Montano, "Deep reinforcement learning oriented for real world dynamic scenarios," *arXiv preprint arXiv:2210.11392*, 2022.

[35] A. Oroojlooy and D. Hajinezhad, "A review of cooperative multi-agent deep reinforcement learning," *Applied Intelligence*, vol. 53, no. 11, pp. 13 677–13 722, 2023.

[36] J. Orr and A. Dutta, "Multi-agent deep reinforcement learning for multi-robot applications: A survey," *Sensors*, vol. 23, no. 7, p. 3625, 2023.

[37] M. Hessel *et al.*, "Rainbow: Combining improvements in deep reinforcement learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[38] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, 2016.

[39] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*, PMLR, 2016, pp. 1995–2003.

[40] M. E. Taylor, H. B. Suay, and S. Chernova, "Integrating Reinforcement Learning with Human Demonstrations of Varying Ability," in *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, May 2011.

[41] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 6292–6299.

[42] M. Vecerik *et al.*, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv preprint arXiv:1707.08817*, 2017.

[43] V. G. Goecks, G. M. Gremillion, V. J. Lawhern, J. Valasek, and N. R. Waytowich, "Integrating behavior cloning and reinforcement learning for improved performance in dense and sparse reward environments," in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 2020, pp. 465–473.

[44] W. B. Knox and P. Stone, "Reinforcement learning from simultaneous human and mdp reward," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, 2012, pp. 475–482.

[45] G. Warnell, N. Waytowich, V. Lawhern, and P. Stone, "Deep TAMER: Interactive agent shaping in high-dimensional state spaces," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[46] R. Arakawa, S. Kobayashi, Y. Unno, Y. Tsuboi, and S.-i. Maeda, "DQN-TAMER: Human-in-the-loop reinforcement learning with intractable feedback," *arXiv preprint arXiv:1810.11748*, 2018.

[47] J. MacGlashan *et al.*, "Interactive learning from policy-dependent human feedback," in *Proceedings of the International Conference on Machine Learning (ICML)*, Aug. 2017.

[48] D. Arumugam, J. K. Lee, S. Saskin, and M. L. Littman, "Deep reinforcement learning from policy-dependent human feedback," *arXiv preprint arXiv:1902.04257*, 2019.

[49] B. Kim, A.-m. Farahmand, J. Pineau, and D. Precup, "Learning from limited demonstrations," in *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*, 2013, pp. 2859–2867.

[50] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[51] M. Andrychowicz *et al.*, "Hindsight experience replay," *Advances in neural information processing systems*, vol. 30, 2017.

[52] X. Liang, K. Shu, K. Lee, and P. Abbeel, "Reward uncertainty for exploration in preference-based reinforcement learning," in *International Conference on Learning Representations*, 2021.

[53] J. B. Martín, R. Chekroun, and F. Moutarde, "Learning from demonstrations with sacr2: Soft actor-critic with reward relabeling," in *Deep RL Workshop NeurIPS 2021*, 2021.

[54] A. Venugopal, E. Bondi, H. Kamarthi, K. Dholakia, B. Ravindran, and M. Tambe, "Reinforcement learning for unified allocation and patrolling in signaling games with uncertainty," in *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 2021, pp. 1353–1361.

[55] C. Yan, X. Xiang, and C. Wang, "Towards real-time path planning through deep reinforcement learning for a UAV in dynamic environments," *Journal of Intelligent & Robotic Systems*, vol. 98, pp. 297–309, 2020.

[56] H. Yuan, J. Ni, and J. Hu, "A centralised training algorithm with D3QN for scalable regular unmanned ground vehicle formation maintenance," *IET Intelligent Transport Systems*, vol. 15, no. 4, pp. 562–572, 2021.

[57] H. Oliff, Y. Liu, M. Kumar, M. Williams, and M. Ryan, "Reinforcement learning for facilitating human-robot-interaction in manufacturing," *Journal of Manufacturing Systems*, vol. 56, pp. 326–340, 2020.

[58] Y. Ji *et al.*, "Multi-agent reinforcement learning resources allocation method using Dueling Double Deep Q-Network in vehicular networks," *IEEE Transactions on Vehicular Technology*, 2023.

[59] W. Kong, D. Zhou, Z. Yang, Y. Zhao, and K. Zhang, "UAV autonomous aerial combat maneuver strategy generation with observation error based on state-adversarial deep deterministic policy gradient and inverse reinforcement learning," *Electronics*, vol. 9, no. 7, p. 1121, 2020.

[60] Y. Jiang, J. Yu, and Q. Li, "A novel decision-making algorithm for beyond visual range air combat based on deep reinforcement learning," in *2022 37th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, IEEE, 2022, pp. 516–521.

[61] E. Kamar, S. Hacker, and E. Horvitz, "Combining human and machine intelligence in large-scale crowdsourcing," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, 2012, pp. 467–474.

[62] W. Lasecki, J. Bigham, J. Allen, and G. Ferguson, “Real-time collaborative planning with the crowd,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 26, 2012, pp. 2435–2436.

[63] E. Kamar, “Directions in hybrid intelligence: Complementing AI systems with human intelligence.,” in *IJCAI*, 2016, pp. 4070–4073.

[64] C. Liang, J. Proft, E. Andersen, and R. A. Knepper, “Implicit communication of actionable information in human-ai teams,” in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–13.

[65] G. Bansal, B. Nushi, E. Kamar, D. S. Weld, W. S. Lasecki, and E. Horvitz, “Updates in human-AI teams: Understanding and addressing the performance/-compatibility tradeoff,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 2429–2437.

[66] G. Hoffman and C. Breazeal, “Collaboration in human-robot teams,” in *AIAA 1st intelligent systems technical conference*, 2004, p. 6434.

[67] B. Hayes and B. Scassellati, “Effective robot teammate behaviors for supporting sequential manipulation tasks,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2015, pp. 6374–6380.

[68] M. F. Jung, D. Difranzo, S. Shen, B. Stoll, H. Claure, and A. Lawrence, “Robot-assisted tower construction—a method to study the impact of a robot’s allocation behavior on interpersonal dynamics and collaboration in groups,” *J. Hum.-Robot Interact.*, vol. 10, no. 1, 2020. DOI: 10.1145/3394287. [Online]. Available: https://doi.org/10.1145/3394287.

[69] S. Herse, “Optimising outcomes of human-agent collaboration using trust calibration,” Ph.D. dissertation, UNSW Sydney, 2022.

[70] K. Phulera, H. Singh, and A. Bhatt, “Analytical study on artificial intelligence techniques to achieve expert systems,” *International Journal on Emerging Technologies (Special Issue NCETST-2017)*, vol. 8, no. 1, pp. 137–140, 2017.

[71] R. Choudhury, G. Swamy, D. Hadfield-Menell, and A. D. Dragan, “On the utility of model learning in HRI,” in *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, IEEE, 2019, pp. 317–325.

[72] S. Sandrini, M. Faroni, and N. Pedrocchi, “Learning action duration and synergy in task planning for human-robot collaboration,” in *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2022, pp. 1–6.

[73] R. Jahanmahin, S. Masoud, J. Rickli, and A. Djuric, “Human-robot interactions in manufacturing: A survey of human behavior modeling,” *Robotics and Computer-Integrated Manufacturing*, vol. 78, p. 102 404, 2022.

[74] M. Tambe, *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge university press, 2011.

[75] M. Tambe, “Implementing agent teams in dynamic multiagent environments,” *Applied Artificial Intelligence*, vol. 12, no. 2-3, pp. 189–210, 1998.

[76] M. Tambe *et al.*, “Intelligent agents for interactive simulation environments,” *AI magazine*, vol. 16, no. 1, pp. 15–15, 1995.

[77] J. Van Diggelen, J. M. Bradshaw, T. Grant, M. Johnson, and M. Neerincx, “Policy-based design of human-machine collaboration in manned space missions,” in *2009 Third IEEE International Conference on Space Mission Challenges for Information Technology*, IEEE, 2009, pp. 376–383.

[78] A Hong, O Igharoro, Y. Liu, F. Niroui, G. Nejat, and B. Benhabib, “Investigating human-robot teams for learning-based semi-autonomous control in urban search and rescue environments,” *Journal of Intelligent & Robotic Systems*, vol. 94, pp. 669–686, 2019.

[79] J. Hu, L. Wang, T. Hu, C. Guo, and Y. Wang, “Autonomous maneuver decision making of dual-UAV cooperative air combat based on deep reinforcement learning,” *Electronics*, vol. 11, no. 3, p. 467, 2022.

[80] B. Xin and C. He, “DRL-based improvement for autonomous UAV motion path planning in unknown environments,” in *2022 7th International Conference on Control and Robotics Engineering (ICCRE)*, IEEE, 2022, pp. 102–105.

[81] Y. Cao, Y.-X. Kou, Z.-W. Li, A. Xu, *et al.*, “Autonomous maneuver decision of UCAV air combat based on Double Deep Q Network algorithm and stochastic game theory,” *International Journal of Aerospace Engineering*, vol. 2023, 2023.

[82] J. Zhang, Z. Meng, J. He, Z. Wang, and L. Liu, “UAV air game maneuver decision-making using Dueling Double Deep Q Network with expert experience storage mechanism,” *Drones*, vol. 7, no. 6, p. 385, 2023.

[83] Y. Liu, A. Halev, and X. Liu, “Policy learning with constraints in model-free reinforcement learning: A survey,” in *The 30th International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[84] L. Zhang, Q. Zhang, L. Shen, B. Yuan, and X. Wang, “Saferl-kit: Evaluating efficient reinforcement learning methods for safe autonomous driving,” *arXiv preprint arXiv:2206.08528*, 2022.

[85] R. Sutton and A. Barto, “Reinforcement learning: An introduction,” *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 1054–1054, 1998.

[86] R. Munos, “Q (λ) with off-policy corrections,” in *Algorithmic Learning Theory: 27th International Conference, ALT 2016, Bari, Italy, October 19-21, 2016, Proceedings*, Springer, vol. 9925, 2016, p. 305.

[87] A. B. Owen, *Monte Carlo theory, methods and examples*, 2013.

[88] B. Kang, Z. Jie, and J. Feng, “Policy optimization with demonstrations,” in *International conference on machine learning*, PMLR, 2018, pp. 2469–2478.

[89] T. Cederborg, I. Grover, C. L. Isbell Jr, and A. L. Thomaz, “Policy shaping with human teachers.,” in *IJCAI*, 2015, pp. 3366–3372.

[90] W. B. Knox and P. Stone, “Interactively shaping agents via human reinforcement: The tamer framework,” in *KCAP*, 2009.

[91] H. Satija, P. Amortila, and J. Pineau, “Constrained Markov decision processes via backward value functions,” in *International Conference on Machine Learning*, PMLR, 2020, pp. 8502–8511.

[92] S. Marwan, Y. Shi, I. Menezes, M. Chi, T. Barnes, and T. W. Price, “Just a few expert constraints can help: Humanizing data-driven subgoal detection for novice programming.,” *International Educational Data Mining Society*, 2021.

[93] F. Bai, H. Zhang, T. Tao, Z. Wu, Y. Wang, and B. Xu, “PiCor: Multi-task deep reinforcement learning with policy correction,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023, pp. 6728–6736.

[94] M. Zawalski, B. Osiński, H. Michalewski, and P. Miłoń, “Off-policy correction for multi-agent reinforcement learning,” in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022, pp. 1774–1776.

[95] C. Celemin and J. Ruiz-del Solar, “An interactive framework for learning continuous actions policies based on corrective feedback,” *Journal of Intelligent & Robotic Systems*, vol. 95, pp. 77–97, 2019.

[96] A. I. Redefined, S. K. Gottipati, S. Kurandwad, C. Mars, G. Szriftgiser, and F. Chabot, “Cogment: Open source framework for distributed multi-actor training, deployment & operations,” *CoRR*, vol. abs/2106.11345, 2021. arXiv: 2106.11345. [Online]. Available: https://arxiv.org/abs/2106.11345.

[97] S. G. Hart, “NASA-task load index (NASA-TLX); 20 years later,” in *Proceedings of the human factors and ergonomics society annual meeting*, Sage publications Sage CA: Los Angeles, CA, vol. 50, 2006, pp. 904–908.

[98] D. Richards, “Measure for measure: How do we assess human autonomy teaming?” In *HCI International 2020–Late Breaking Papers: Cognition, Learning and Games: 22nd HCI International Conference, HCII 2020, Copenhagen, Denmark, July 19–24, 2020, Proceedings 22*, Springer, 2020, pp. 227–239.

[99] I. Kim and J. R. Morrison, “Learning based framework for joint task allocation and system design in stochastic multi-UAV systems,” in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2018, pp. 324–334.

[100] P. E. McKnight and J. Najab, “Mann-Whitney U Test,” *The Corsini encyclopedia of psychology*, pp. 1–1, 2010.

[101] A. Neumann, J. Bossek, and F. Neumann, “Diversifying greedy sampling and evolutionary diversity optimisation for constrained monotone submodular functions,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO ’21, Lille, France: Association for Computing Machinery, 2021, 261–269, ISBN: 9781450383509. DOI: 10.1145/3449639.3459385. [Online]. Available: https://doi.org/10.1145/3449639.3459385.

Appendix A: Environment Design: Team Description

In this chapter, we discuss the team details of our UVA-based airport security systems shown in Figure 3.1. In the subsequent discussion, we present a mathematical representation of the aforementioned scenario. Let $p_{RA} \in \mathbb{R}^2$ and $r_{RA} \in [0, 1)$ be the center and radius of a circle representing the restricted area, respectively. Two teams are present in this scenario: the ally team (blue team) and the enemy team (red team).

A.1 Ally Team (Blue Team)

The blue team comprises five aerial drones, a ground radar (GR) sensor, and a ground control station (GCS). Each ally drone also has several neutralization payloads (i.e., devices capable of neutralizing enemy drones when they are within a certain range). The goal of the blue team is to protect the restricted zone of the airport from the red team by detecting, localizing, and neutralizing the enemy drones.

Ally Ground Control Station (GCS): There is one GCS located at position $p_{GC_t} \in \mathbb{R}^2$ at time step $t \in \mathbb{N}$. In addition, let $r_{GC} \in (0, 1)$ denote the circle's radius representing the GCS operating range.

Ally Ground Radar (GR): Consider $n_{GR} \in \mathbb{N}$ ground radars whose jobs are to gather information about the ally and enemy drones. Denote by $p_t^{i,GR} \in \mathbb{R}^2$ the position and by $\phi_t^{i,GR} \in [-\pi, \pi]$ the heading angle (orientation) of the i-th GR at time $t \in \mathbb{N}$, $i \in \mathbb{N}_{nGR}$. Let $u_t^{i,GR} \in [-1, 1]$ denote the action of the i-th GR, where $u_t^{i,GR}$ is the ratio of angular speed with respect to its maximum value. In particular, $u_t^{i,GR}$ controls the rotation of the i-th GR as follows:

$$\phi_{t+1}^{i,GR} = \phi_t^{i,GR} + v_{GR}^{max} \times u_t^{i,GR} \tag{A.1}$$

where $v_{GR}^{max} \in [0, 1)$ is the maximum angular speed. Denote by $\rho_{GR} \in [0, 2\pi]$ the field of view of the GR and by $r_{GR} \in [0, 1)$ the radius of a circle representing its sensing range.

Ally Drone (Blue Drone): The ally system has $n_{AD} \in \mathbb{N}$ drones. The state of the i-th ally drone includes:

1. Position: $p_t^i \in \mathbb{R}^2$.
2. Heading angle: $\phi_t^i \in [-\pi, \pi]$.
3. Relative orientation of the electro-optic (EO) sensor: $\phi_t^{i,EO} \in [-\phi_{EO}^{\max}, \phi_{EO}^{\max}]$.
4. Functionality status: $f_t^i \in \{0, 1\}$, where 0 indicates non-functional.
5. Control by GCS: $g_t^i \in \{0, 1\}$, where 0 indicates no control by GCS.
6. Position of controlling GCS: $p_t^{i,GC} \in \mathbb{R}^2$.
7. Radar status: $\text{radar-enabled}_t^i \in \{0, 1\}$, where 0 means off.
8. EMP usage: $\text{emp-used}_t^i \in \{0, 1\}$, indicating if EMP has been used.

In particular, there is an EMP-auto-destruction probability $pr_{EMPD} \in [0, 1]$ according to which the drone may destroy itself upon using EMP, i.e.,

$$f_{t+1}^i = \begin{cases} 0 & \text{if } f_t^i = 0; \\ 1 & \text{if } f_t^i = 1 \text{ and } \textit{emp-used}_t^i = 0; \\ B & \text{if } f_t^i = 1 \text{ and } \textit{emp-used}_t^i = 1; \end{cases} \tag{A.2}$$

where $B \in \{0,1\}$ is a Bernoulli random variable with success probability of $1 - pr_{EMPD}$.

Denote by $\rho_{EO} \in [0, 2\pi]$ the field of view of the EO sensor and by $r_{EO} \in [0,1]$ the radius of a circle representing its sensing range. In addition, let $\rho_{AD} \in [0, 2\pi]$ be the field of view of the radar and $r_{AD} \in [0,1)$ be the radius of a circle representing its sensing range. The action set of drone i at time t is described below:

1. Control signal for the movement angle $u_t^{i,MA} \in [-\pi, \pi]$,

2. Seed-ratio of the drone, i.e., $u_t^{i,SR} \in [0,1]$ such that

$$p_{t+1}^i = p_t^i + v_{SR}^{max} \times u_t^{i,SR} \times \cos(\sin(u_t^{i,MA})), \tag{A.3}$$

   where $v_{SR}^{max} \in (0,1)$ is the maximum speed of drone,

3. Angular speed-ratio $u_{i,t}^{heading} \in [-1,1]$ such that

$$\phi_{t+1}^i = \phi_t^i + v_{AS}^{max} \times u_t^{i,heading}, \tag{A.4}$$

   where $v_{AS}^{max} \in (0,1)$ is the maximum angular speed,

4. Angular speed-ratio of EO sensor $u_t^{i,EO} \in [-1,1]$ s.t.

$$\phi_{t+1}^{i,EO} = \begin{cases} -\phi_{EO}^{max} & \text{if } \phi_t^{i,EO} \leq -\phi_{EO}^{max}; \\ +\phi_{EO}^{max} & \text{if } \phi_t^{i,EO} \geq \phi_{EO}^{max}; \\ \phi_t^{i,EO} + v_{ASEO}^{max} \times u_t^{i,EO} & \text{otherwise;} \end{cases} \tag{A.5}$$

   where $v_{ASEO}^{max} \in (0,1)$ is the maximum angular speed of EO sensor,

5. Turn off/on the drone's radar $u_t^{i,ER} \in \{0,1\}$, where 0 refers to turning the radar off, i.e.,

$$\text{radar-enabled}_{t+1}^i = u_t^{i,ER}. \tag{A.6}$$

6. Turn off/on the EMP $u_t^{i,EMP} \in \{0,1\}$, where 0 refers to turning the EMP off, i.e.,

$$\text{emp-used}_{t+1}^i = u_t^{i,EMP} + \text{emp-used}_t^i(1 - u_t^{i,EMP}). \tag{A.7}$$

7. Turn off/on jamming, i.e., $u_t^{i,EJ} \in \{0, 1\}$, where 0 refers to turning the jamming off.

8. Turn off/on GPS spoofing, i.e., $u_t^{i,GPSS} \in \{0, 1\}$, where 0 refers to turning the spoofing off.

9. Turn off/on hacking, i.e., $u_t^{i,EH} \in \{0, 1\}$, where 0 refers to turning the hacking off.

A.2 Enemy Team (Red Team)

The red team comprises a single drone equipped with its own radar sensor and a potentially hazardous payload. The enemy team consists of $n_{EGCS} \in \mathbb{N}$ GCSs and $n_{ED} \in \mathbb{N}$ drones.

Enemy Ground Control Station (GCS): The position of the i-th enemy GCS at time t is $p_i^{EGCS}(t) \in \mathbb{R}^2$. The radius of its operating range is denoted by $r_{EGCS} \in [0, 1)$.

Enemy Drone (Red Drone): The state of the i-th enemy drone at time t includes:

1. Position: $p_t^{i,ED} \in \mathbb{R}^2$.

2. Payload: $l_t^{i,ED} \in \{0, 1, 2, 3\}$, where values represent ”safe”, ”unknown”, ”moderate”, and ”dangerous”, respectively.

3. Control by GCS: $g_t^{i,ED} \in \{0, 1\}$.

4. Position of controlling GCS: $p_t^{i,EGCS} \in \mathbb{R}^2$ where 0 means the drone is not controlled by the GCS

5. Functionality status: $f_t^{i,ED} \in \{0,1\}$ where 0 means it is not functional; In particular,

$$f_{t+1}^{i,ED} = \begin{cases} 0 & \exists j \in N_{nAD} \text{ s.t. } f_t^j \times u_t^j = 1 \text{ and } \|p_t^{i,ED} - p_j\|_2 \leq r_N, \\ 0 & \|p_t^{i,ED} - p_t^{i,EGCS}\|_2 > r_{EGCS} \text{ and } gt_t^{i,ED} = 1, \\ ft_t^{i,ED} & \text{otherwise.} \end{cases} \tag{A.8}$$

where $r_N \in [0,1)$ is the radius of a circle representing the neutralization range of ally drones and $u_j^t \in \{0,1\}$ is defined as whether or not the ally drone $j \in N_{nAD}$ is enabled to neutralize enemy drone i, i.e.,

$$u_t^j := u_t^{j,EMP} \vee u_t^{j,GPSS} \vee (gt_t^{i,ED} \wedge u_t^{j,EJ}) \vee (gt_t^{i,ED} \wedge u_t^{j,EH}). \tag{A.9}$$

Appendix B: Experimental Platform and Architecture for HitL Interactions

The experimental platform is a distributed application built using the Cogment platform. We use Cogment as it simplifies the conduction of large-scale experiments in multi-agent systems and HitL. Cogment dispatches observations of the environment from the simulation to the agents, as well as instructions from higher-level agents such as humans or decision-makers. It then dispatches agents' actions to the environment, which updates the simulation and agent's instructions. Furthermore, a priority-ordered list comprising multiple agents can be allocated simultaneously to a single drone entity. When an agent with higher priority issues a command for velocity or rotation change, it overrides those from lower-priority agents. This mechanism facilitates dynamic takeover by the human operator, optimizing communication channels, data storage, and processing pipelines in the process.

In order for the human operator to control the ally drones and online policy correction, we have developed a user interface as a part of the experimental platform shown in Figure B.1. The experimental platform is built around a simplified airspace simulator operating in 2D, simulating two types of entities: drones from both blue and red teams and the ground radar on the blue side. While simplified, several aspects have been modeled following real-world specifications provided by defense experts, such as the detection capabilities of the drone sensors and the radar, as well as the dynamics of the fixed-wing drones. The communication between the different agents

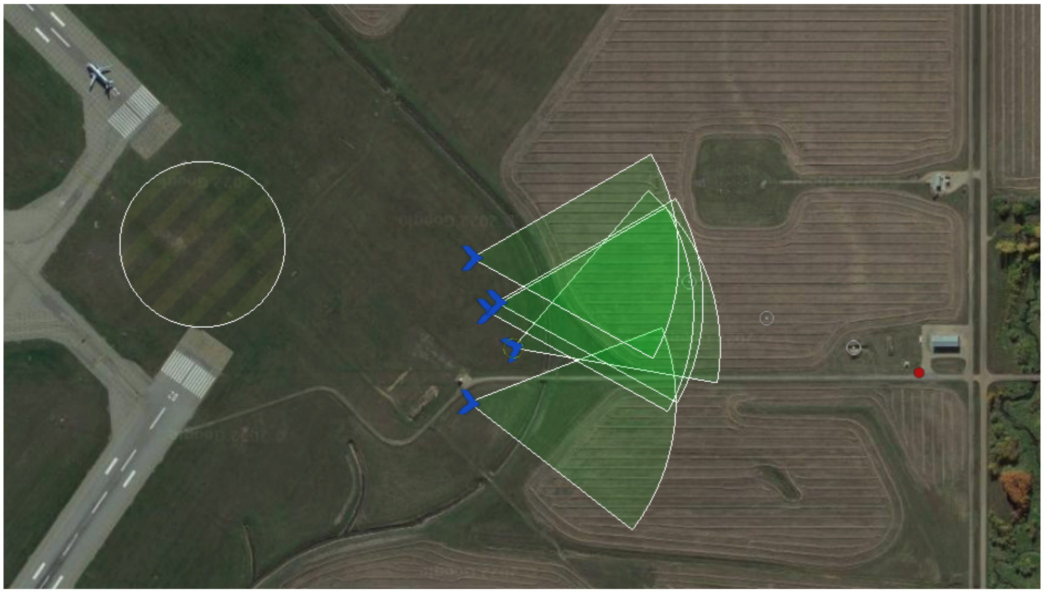


Figure B.1: User interface for human operators to control the agents

is limited due to their partial observation of the environment but is perfect and instant. The experimental platform models the scenarios as a multi-agent system with a three-layer architecture. The primary agent type controls the drones through velocity and rotation changes. The bottom layer is named the drone agent layer, and it receives partial observation of the environment "through the lens" of its sensors. In this layer, the blue drones can be either fully autonomous, fully human-operated, or hybrid (i.e., control is shared by the human-agent team). This is implemented by two Cogment actor implementations (drone and human actors), shown in Figure-B.2. For each ally drone agent in the simulation, Cogment instantiates two actors using those implementations; it can then dynamically assign the control of the drone entity to one of them.

The system supports two types of high-level decision-makers: the Control and Command agents. The Control Agent simplifies drone-level tasks by assigning targets, a capability that can be leveraged by either a human or an agent. Meanwhile, an additional layer is incorporated to allow the Command Agent to designate areas of interest to the Control Agents, facilitating more nuanced and strategic operations.

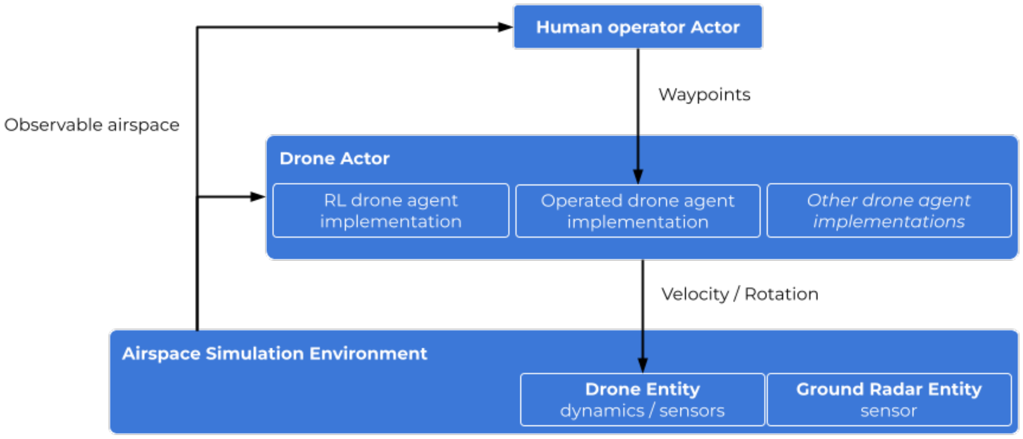

Human operator Actor
Waypoints
Observable airspace
Drone Actor
RL drone agent implementation
Operated drone agent implementation
Other drone agent implementations
Velocity / Rotation
Airspace Simulation Environment
Drone Entity dynamics / sensors
Ground Radar Entity sensor

Figure B.2: Airspace simulation and hierarchical multi-agent modeling

Human operators were introduced to facilitate the control of individual drones and to gather demonstrations of various drone behaviors. These operators can select specific drones and set waypoints for them by predicting the anticipated trajectories of enemy drones. In Figure B.1, the waypoints are denoted by grey circles on the map. Once a specific waypoint has been defined, Cogment dynamically gives control of the associated ally drone to the operated agent, causing it to move towards the defined waypoint via the shortest path under the standard physical dynamics constraints. Similarly, the human operator can delete existing waypoints through the interface to rectify any errors in predicting the enemy drone trajectory. Conversely, when no waypoints are defined, the control is given back to the autonomous agent. The interface also allows the human to operate at three simulation speeds 1x, 2x, 5x, according to their preferences.

Figure B.2 represents the architecture of the developed experimental platform. The Cogment platform [96] handles the orchestration of the execution and communication between the different components:

1. The drone agents can each use one of multiple implementations; they are encapsulated in dedicated micro services as Cogment actors.

2. The simulation, encapsulated in a dedicated micro service as a Cogment envi-

ronment (that uses MDP formalism).

3. The human operator, interacting through the UI, encapsulated as a client Cogment actor.

The experimental platform, along with a user interface is developed for this project enabled the team to easily implement a heterogeneous, hierarchical multi-agent system. This allowed the integration of multiple types of agents, each with their own specialized capabilities and roles, within a single system while the hierarchical properties enabled task decomposition. Furthermore, a priority ordered list of multiple agents can be assigned to a single drone entity at once. One key feature of the platform is the dynamic agent or human "takeover" capability, which supports human-AI teaming during operations and provides advice during training. This allows for the seamless integration of human operators and AI agents within the system, allowing the operator to take over when needed. This allowed us to evaluate the human-AI team. A detailed description of the user interface can be found in Section 3.3.

Appendix C: Hyperparameters

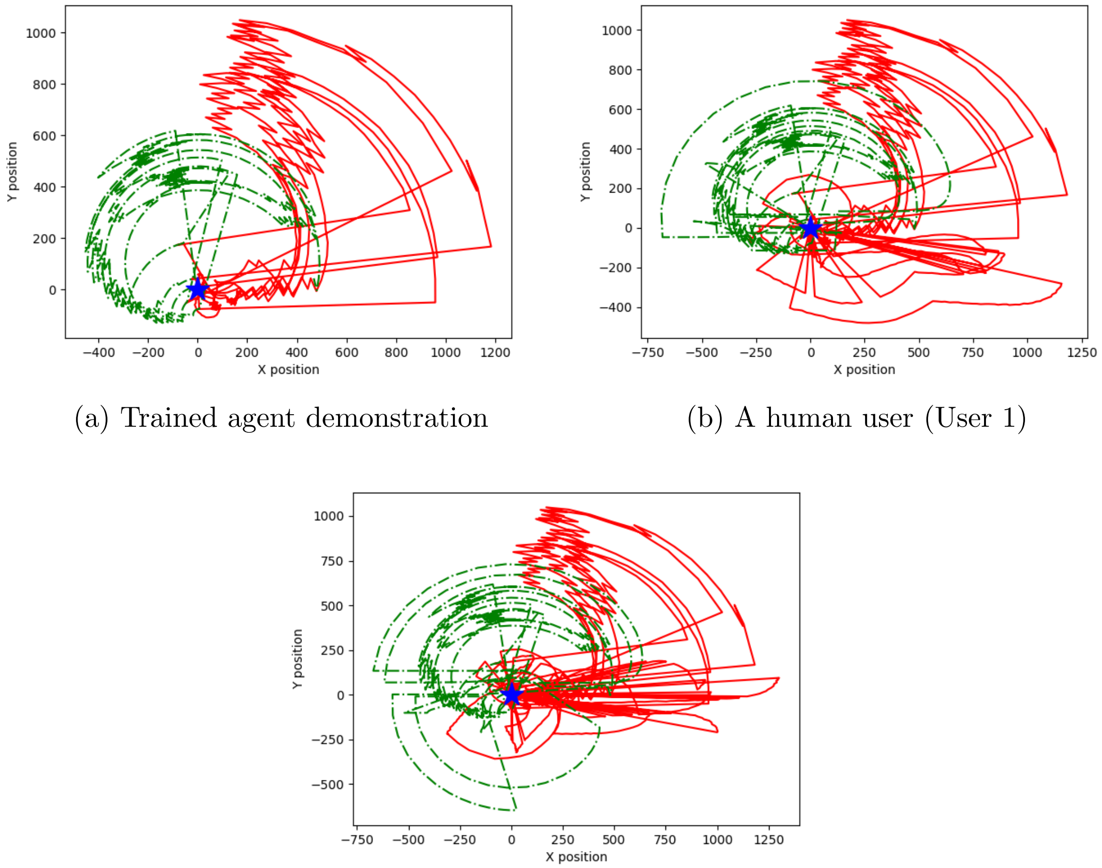
We perform hyper-parameter tuning using grid search to find best parameters for the algorithms and use the same settings for all the models in scenarios. We considered learning rate values of[0.4, 0.04, 0.004, 0.0004, 0.00004], epsilon decay values of [0.99, 0.995, 0.9995, 0.99995, 0.999995], and discount factor values of [0.9, 0.99, 0.999]. The supervised loss coefficient weight (λ_2 in equation 2.5) varied between 10^5 to 1, and we set λ_3 to 0. The same network structure was used across D3QN-0-2500, D3QN-0-500, D3QN-500-2000, D3QN$_{HM}$-500-0 and D3QN$_{PC}$-500-0, consisting of two hidden layers with 64 fully connected neurons. A final fully connected layer was added to represent each action's Q-values. The non-linearity function used in all layers was rectified linear units (ReLU). During training, we use the Adam optimizer and applied an epsilon-greedy policy, gradually reducing epsilon from 1 to 0.05. The batch size was 64, and the replay memory size was $100,000$.

Parameter	Value
Training Episodes	10,000
Replay Memory size	100,000
Batch size	64
Learning rate	0.0004
Discount factor	0.99
Target network update frequency	10
Initial ϵ	1.0
Final ϵ	0.05
ϵ decay per episode	0.999995

Table C.1: Model hyper-parameters.

Appendix D: Demonstration Visualization

We visualized five trajectories of all blue and red drones from trained agent demonstrations and two actual human demonstrations from different users who played more than 30 games, as shown in Figure D.1. In this figure, the blue star denotes one of the ally drones that neutralized the enemy drone and is the frame of reference (located at $(0, 0)$). The red and green lines represent the relative position of the enemy drone and the restricted airspace (with respect to the blue drone's position). Figure D.1(a) shows five trajectories of ally drones generated from the trained D3QN agent. We note that across all the figures, the red drone starts moving towards the restricted zone while being chased by the blue drones until it is neutralized. The low density of red lines around the blue star indicates that the blue drones quickly neutralize the red drone without following it for a long time. From Figure D.1(b) and D.1(c), which depicts trials by two different human participants, we notice that there is more movement (high-density) around the blue star, suggesting that the human tries setting waypoints in different areas (using the whole team of five blue drones) of the map to neutralize the red drone. These trajectories are sub-optimal (longer trajectory length) as compared to the trajectories from trained agent demonstrations. However, these might be helpful to neutralize the red drones in challenging environment configurations where the trained RL agents fail to catch the enemy drone (trained agents have a failure rate of around 10% in this task).



(a) Trained agent demonstration

(b) A human user (User 1)

(c) A human user (User 2)

Figure D.1: Visual representation of five episodes from trained agent and two different real human users

Appendix E: NASA-TLX Load Index Questionnaires:

Participants will complete a 6-question NASA-TLX workload assessment, with each question featuring a 21-point slider ranging from “very low” to “very high” . These questions are modified to facilitate comparison with the previous round of the task:

1. Mental Demand: How mentally demanding was the task compared to the previous round?

2. Physical Demand: How physically demanding was the task compared to the previous round?

3. Temporal Demand: How hurried or rushed was the pace of the task compared to the previous round?

4. Performance: How successful were you in accomplishing what you were asked to do compared to the previous round?

5. Effort: How hard did you have to work to accomplish your level of performance compared to the previous round?

Demographic Questionnaires:

- Gender? [Multiple Choice: Woman, Man, Transgender, Prefer to describe myself, Prefer not to respond]

- Your current position/post

- What is your age? [slider 18-65]
- Are you experienced in defence strategies/drone control, development? [Yes/No]
- Years of experience in defence strategies/drone control, development? [slider or box]
- Did you experience drone controls in real life or simulated tasks in the past ? [yes/no]
- Do you have experience with video games? [Yes/No] If yes, years of experience with video games.