26th International Congress on Modelling and Simulation, Adelaide, SA, Australia, 30 Nov–4 Dec 2025
mssanz.org.au/modsim2025

ARES: Aerial Response and Engagement Simulator

Ishan Honhaga [a], **Anh Viet Do** [a], **Eyoel Teffra** , **Anthony Perry** [b], **Rob Baker** [b], **Claudia Szabo** [a]

[a] *School of Computer Science, The University of Adelaide, South Australia*
[b] *Defence Science and Technology Group, Defence, Adelaide, South Australia*
Email: ishan.honhaga@adelaide.edu.au

Abstract: We introduce the Aerial Response and Engagement Simulator (ARES), a highly customizable and extensible simulation environment specifically designed for training reinforcement learning (RL) algorithms in diverse Counter Uncrewed Aerial Systems (CUAS) scenarios. ARES prioritizes extensibility, providing users the ability to tailor simulations by creating custom agents, adversaries, countermeasures, and reward structures. This inherent flexibility allows RL algorithms to effectively learn and optimize their strategies against evolving drone threats within a competitive framework.

ARES features a stationary friendly Blue Team equipped with a range of customizable countermeasures, tasked with defending against a mobile adversary Red Team of attacking drones. The primary objective for both teams in an episode is to eliminate the other team. The environment provides extensive options for defining agent behaviors, custom countermeasures, and reward structures, which collectively support the exploration and development of novel CUAS tactics.

Our simulator is structured around several classes and wrappers as shown in Figure 2. The Client class initiates the process by feeding the specifications into the Playground class, which is responsible for instantiating all the essential entities that populate the world in ARES simulation environment.

ARES allows for learning algorithms to be deployed within both Blue and Red teams, facilitating effective testing of the robustness and resilience in varied environments. The provision of a dynamic and intelligent adversary improves the fidelity and utility of simulated training and evaluation against Uncrewed Aerial Systems.

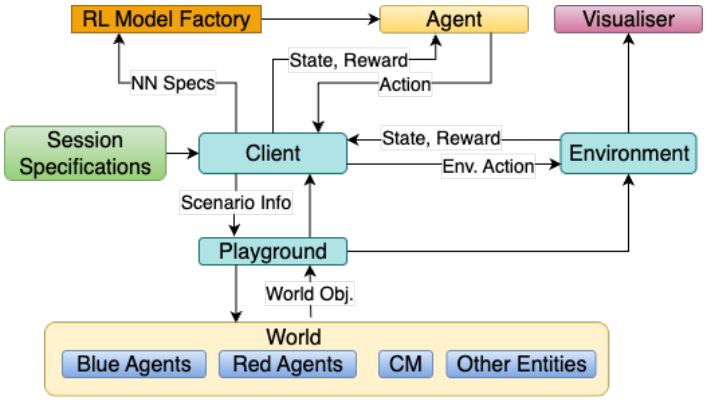


Figure 1. ARES Operational Flow

To accommodate various research and development needs, ARES supports both visual and headless (non-visual) simulation modes. The visual mode aids in qualitative analysis, while the headless mode enables efficient, efficient algorithm training and quantitative analysis, making ARES a versatile tool for CUAS research.

Keywords: *Simulation environment, Reinforcement Learning, Drone simulation*

1 INTRODUCTION

The current landscape of military operations has undergone a profound transformation, moving decisively towards Uncrewed aerial systems (UAS)-centric operations. This shift not only showcases a leap in technological advancement but also a fundamental re-conceptualisation of battlefield dynamics, strategic planning, and the nature of engagement (Molloy (2024)). UAS are increasingly indispensable, with their utility expanding from surveillance and logistical support to direct combat engagement. This trend is not only established but is projected to intensify significantly in the future (Molloy (2024)).

The rapid technological advancements of the past few decades that have expanded the usage of UASs from niche capabilities to central components of global military arsenals. Today, UAS can be used for various roles such as intelligence gathering, surveillance, and reconnaissance (ISR), precision strike operations, logistical support, electronic warfare, and search and rescue missions Molloy (2024), Castrillo et al. (2022) and Pong (2022). Their utility is demonstrated across various conflict zones, from counter-terrorism operations to large-scale conventional warfare (Molloy (2024)). Conversely, as UAS systems can and have been used in various offensives, research into counter-UAS systems is picking up significant interest (Pong (2022)). A critical challenge in determining the effectiveness and fault tolerance of existing solutions is the high cost of their testing and optimizataion of pre-deployment configuration in real-life scenarios (Pong (2022)).

The advancements in UAS technology is rapidly evolving, as highlighted by recent works such as Tian et al. (2025), Castrillo et al. (2022), Nichols et al. (2019) and Adoni et al. (2024). The expansion in UAS utility has increased the research investment into two primary areas, developing UASs with sophisticated capabilities (Adoni et al. (2024), Tian et al. (2025)) and building advanced counter-UAS systems (Samaras et al. (2019), Nichols et al. (2019)). While these works demonstrate the significant potential in both domains, a key gap remains. A unified system enabling both UAS and countering systems to learn and adapt by directly playing against each other, thereby pushing the boundaries of autonomous adversarial development is crucial for advancing the state-of-the-art in autonomous combats and developing robust solutions against evolving aerial threats.

In this paper we describe the ARES environment, which is developed as a training ground for counter-UAS Reinforcement learning (RL) algorithms. Due to the versatility of UAS and the advancement in technology, UAS are getting better and smarter at an accelerated rate as discussed by Molloy (2024),Tian et al. (2025), Castrillo et al. (2022), Nichols et al. (2019) and Adoni et al. (2024). To tackle this rapid advancement, a training ground for RL algorithms needs to be highly customizable, capturing diverse scenarios and observations, with an extended ability to evaluate numerous solutions. ARES is a simulation environment based on the OpenAI GYM toolkit Brockman et al. (2016), designed around the Counter-UAS problem with a range of customisation options available to allow users to tailor the scenario to their needs. To the best of our knowledge, no simulator of this type of scenario, that evaluates and integrates RL algorithms exists.

2 ARES ENVIRONMENT

The basic structure of an ARES scenario faces a Blue Team against a number of Red Teams. The Blue Team consists of ground units (tanks) and the Red consists of aerial units (drones). Each blue agent is stationary and is equipped with (CM) to destroy incoming red agents. CM for each blue agent can be selected from those present in the ARES environment, and new ones can be easily added. Similarly, predefined or new reward structures can be used for scoring. The red agents are mobile and attempt to destroy the blue agents via collision. Drone behaviour, speed and target can be specified, with the capability to extend these abilities and add new ones. The Blue and Red Teams compete against each other with the objective of winning an episode. A team wins when it eliminates the other team.

Reinforcement Learning (RL) algorithms rely on the formal structure of a Markov Decision Process (MDP) which dictates how an agent perceives its environment, takes actions, and receives feedback to learn optimal strategies as discussed by Sutton and Barto (2018). ARES provides the MDP framework, allowing RL agents to explore state-action spaces and optimize decision making. This foundational MDP structure, coupled with ARES's customizable features, such as configurable environment parameters, reward functions, and agent behaviors significantly enhances the ability of RL algorithms to adapt and learn robust strategies. Below sections discuss the MDP framework and various customisation available in ARES.

2.1 Markov Decision Process Framework

To facilitate the learning of effective strategies by RL algorithms, the problem of countering uncrewed systems is modeled as a MDP in ARES and characterized by five key elements State (S), Action (A), Transition Probability (T), Reward (R), and Horizon (H) (Sutton and Barto (2018)).

State: A state (S) represents all relevant observable information an agent perceives within its environment at a given moment. An observation (O) defines what the agent perceives. It can be a subset of the true state, depending on the agent's scope of observability. A Blue agent's observation space includes its position, hitpoints, and specifications of its available CMs. For each CM, it can observe its maximum range, hit probability, current ammunition count, type of CM, and its area of impact. It can also perceive these exact same features for other friendly (Blue) agents. Furthermore, the agent has visibility of every Red agent on the screen, including their position, velocity, type, hitpoints, and relative distance.

A Red agent can observe its own location, velocity, type, hitpoints, and its distance from all the Blue agents. Additionally, it can perceive these same features for other friendly (Red) agents present on the screen.

Action: An Action (A) represents a decision made by the agent at a particular state, which, upon execution, causes the environment to transition to a new state. For ARES, the action space is structured as a joint action space. This joint action space includes the following distinct options:

1. *Selection of target, selection of CM*: At any given state, a combination of targets that are moving towards the Blue team and available CM can be selected by the RL policy.
2. *No operation*: This action is available with the RL policy through the entirety of the episode. This is also the default action if the agent is destroyed.

Transition Probability: The transition function (T), formally denoted as $P(s' \mid s, a)$, quantifies the probability of the environment transitioning to a next state (s') given the current state (s) based on the action (a) by the agent. Modelling this function poses a challenge in environments with vast or continuous state spaces, such as ARES. Therefore, in ARES, a model-free approach is adopted during training. Instead of modeling the transition probabilities, the next state provided by the environment is observed.

Rewards: Executing an action (a) from a state (s), leads to a new state (s') in the environment which generates a Reward (R) signal. This value serves as a feedback mechanism for the Reinforcement Learning agent, indicating the utility or "value" of the chosen action within that specific state. Through the accumulation of these rewards over time, the agent learns to optimize its policy to maximize its long term return. The reward for the Blue Team is computed based on red UAS destroyed, cost of firing a CM, penalty for getting eliminated, destroying the Red agents and a penalty for selecting an invalid CM. An invalid selection is determined if the target is outside the selected CM range or if the CM itself is out of ammunition.

ARES offers an interface that enables customization of reward and penalty signals. This capability is crucial for implementing reward shaping, which can accelerate the learning process of RL agents.

Horizon: The Horizon (H) defines the time span within which a given task is expected to be completed. In reinforcement learning, this is commonly referred to as an episode, representing a complete sequence of interactions from an initial state to a terminal state or a predefined maximum number of steps. Each episode consists of a discrete sequence of increments called timesteps.

2.2 Simulator Structure

The Python-based simulator is structured around several classes and wrappers. The Client class initiates the process by feeding the specifications into the Playground class, which is responsible for instantiating all the essential entities that populate the world in ARES simulation environment. It also passes the RL requirements to the RL Model factory which genrates the RL model(s) required by the Blue Team, Red Team or both. This class also manages the simulation's current state and helps state transitions in response to actions taken, by making the state available to the RL model(s) and sending the actions back to the environment which is selected by the agents. The ARES Environment also calculates and delivers rewards to the agents, guiding them in updating their respective policies. The operational flow of the simulation is illustrated in Figure 1.

A training or inference session starts by selecting the desired algorithm type. The session can be for the Blue or Red Team or both. The input also requires a scenario number, which is used to look up and load the corresponding scenario configuration from the scenarios file. ARES, also allows for loading of the Blue/Red RL

models for training/testing on a given scenario, provided the observation and action space of the specified scenario is the same. Following the initialization of the Reinforcement Learning (RL) algorithm, scenario information is passed to the Playground module, where the simulated "world" is constructed as per the scenario's specifications. Once the world is created the session is kicked off by iterating through episodes.

Both the Blue and Red Teams share a set of common entity state attributes, including position, velocity, and speed. In addition to these, each agent possesses a unique identifier, dimensional attributes such as height and width and flags for visibility and operating dimensionality (i.e., 2D or 3D). These fundamental objects for each agent, regardless of team affiliation, are then inherited by an Agent class. This Agent class further specifies attributes essential to all entities, such as current hitpoints, maximum hitpoints, and an operational flag which indicates if the agent is still active. These generic agents are then specialized into either Blue or Red Team agents. Blue agents are then provided with additional details such as CMs and the specifics of their action space and Red agents with their respective action space details.

During each episode timestep, the simulation environment tracks the CM utilized by each Blue agent, updating their status as they engage Red agents with various CM types. Concurrently, the position of each Red agent is calculated based on its speed function, and these positions are updated within the simulation.

ARES utilizes Pygame for rendering the simulation environment, providing a dynamic visual representation of ongoing episodes. Each agent is equipped with dedicated render objects, allowing for customized visualization based on its specific characteristics. While all agents share fundamental rendering attributes such as height, width, and size, their visual elements are specialized by team. Blue Team agents feature assets like operational and destroyed tank images, recoil constants for visual feedback, and a adjustable turret image. Their countermeasures also incorporate specific draw functions to visualise engagement. Conversely, Red Team agents have distinct images for their operational drone state and a separate image to indicate destruction, ensuring each entity is accurately and dynamically represented within the Pygame visualization.

2.3 Environment Configurations

ARES offers customization capabilities to help with diverse research and evaluation. This includes the ability to configure the simulated map, along with control over the adversarial Red Team's operational doctrine, strategies, and responses. ARES also allows for configurable scenarios, enabling users to isolate and investigate particular challenges or test specific hypotheses within a controlled environment.

Map: ARES offers extensive configurability for various aspects of the simulation environment, which allows for tailored experimental setups. Firstly, the *dimensionality of the map* can be selected between two-dimensional (2D) or three-dimensional (3D) representations along with physical dimensions like length, breadth, and height can be adjusted to specific scenario requirements. Furthermore, the *map scale* can be altered, defining the conversion ratio between pixel units and real-world distances (e.g., kilometers). This allows the environment to be scaled up or down as needed, representing different scales of operational areas.

Secondly, ARES supports the creation of *invisibility zones*, which are specific regions within the map where Red Team drones become invisible to Blue Team agents. The presence of such zones creates a rugged terrain that disrupts the line of sight for the Blue Team, introducing a new strategic challenge. The size, shape, placement, and number of these invisibility zones are configurable parameters. Finally, the frame rate of the visualization can be adjusted. This feature allows for speeding up or slowing down the visual representation of the training process, providing flexibility for observation and analysis.

Scenarios: Scenarios in ARES define positioning and specific behaviour of both the Red and Blue Teams and enable RL policies to learn nuanced behaviors that adapt effectively to dynamic changes within the environment. For example, the initial positioning of the Blue Team can be adjusted from a spread-out formation across two distinct columns to a single column, forcing the RL agent to learn different defensive or offensive strategies. This level of control is beneficial to evaluate and train the Blue Team's performance against various Red Team behaviours. Configuration values for two scenarios are shown in Table 2.

Swarm Behaviour: ARES provides flexibility for red agents to have an option to exhibit an evasive Swarm manoeuvre. During their flight to their respective target, they begin to orbit a seemingly arbitrary point in a bizarre fashion.

Table 1. Available Countermeasure Configurations

Property	Gun	Jammer	Frag	Laser	Net
Ammunition	100	80	10	150	10
Maximum Range	200	60	100	200	100
Hit probability	0.6	1	0.85	0.6	0.85
AOI	N.A.	N.A.	10	N.A.	N.A.

Table 2. Scenario 1 (S1) and Scenario 2 (S2) Specifications. N.A. indicates unavailable attributes.

Attribute	Blue Team		Red Team		Description
	S1	S2	S1	S2	
Maximum agents	1	3	5	5	Size of Blue and Red Team
Agents on Screen	N.A.	N.A.	5	5	Maximum agents onscreen simultaneously
Maximum hitpoints	1	1	1	1	Hits an agent can take
CM/agent	gun	gun, laser	N.A.	N.A.	Types of CM an agents holds
Arrangement	equidistant	equidistant	N.A.	N.A.	Equidistant agents in a single column
Drone Type	N.A.	N.A.	Phantom, Parrot	Phantom, Parrot	Types of available agents
Size	N.A.	N.A.	30	30	Size of the drone
Speed	N.A.	N.A.	60 kmph	60 kmph	Speed of the drone
Speed Function	N.A.	N.A.	Callable func./null	Callable func./null	Red agent speed, flexible or constant is null
Spawn Window	N.A.	N.A.	0.8	0.8	Spawn ground denotes map's breadth percentage
Spawn Location	N.A.	N.A.	2	2	Spawn location within spawn window: 1-Top, 2-Middle, 3-Bottom

This behaviour serves multiple objectives critical to the drones' success. Firstly, it creates confusion among ground-based defensive units as the unpredictable movement complicates targeting, making it harder to achieve a successful hit. Moreover, this is also to achieve the objective of emptying the Blue's available ammunition. This coordinated evasive manoeuvre ensures mission success for the red UAS and allows Blue Team to learn novel strategies to counter UAS swarm behaviour. ARES includes a set of pre-defined CM that are adaptable to specific requirements ARES incorporates a set of pre-defined CMs that can be adapted to specific requirements, as shown in Table 1.

2.4 Episode Rollout

Each episode starts with Red Team UASs spawning from the map's edge opposite to the stationed Blue Team. As the Red agents advance, the Blue Team deploys CM to neutralize the incoming red UAS (threats). The primary objective for both teams is to destroy the opposing force. An image of an active episode is shown in Figure 2.

At each timestep, the simulation progresses through a defined sequence of operations. Initially, the Blue Team's action phase is executed, followed by the Red Team's action phase. Thereafter, rewards are collected, first for the Blue Team and then for the Red Team. This iterative cycle repeats itself at each timestep until the episode terminates. An episode terminates if Blue wins (Red Team is destroyed), Red wins (Blue Team is destroyed), or the episode reaches its time limit.

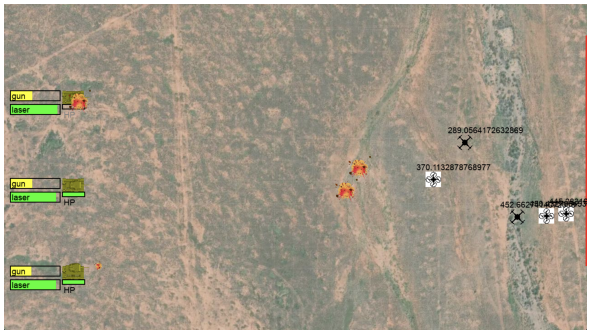


Figure 2. ARES Simulation of an Active Episode. Blue team has three agents (tanks), each equipped with two CM, a gun and a laser, engaging a group of five Red agents comprising two distinct types of UAS.

Blue Team. At the beginning of each episode, agents are initialized within the environment and equipped with CM as defined by the current scenario. If the Blue Team's operating policy selects a no-operation action, no further action is taken. Otherwise, the engagement rules within ARES are applied only if all the following conditions are met. First, the policy selects a target-countermeasure pair. Second, the selected target is within the firing range of the chosen CM. Third, the selected CM has ammunition and fourth, the Red agent is visible. As an episode progresses, the Blue Team continuously selects available actions from its action space.

Red Team. Red Team's policy assigns an operational Blue agent as its target when it spawns. Currently, this assigned target remains fixed for the entire lifetime of the drone. Once a target is selected, the drone proceeds to fly towards it, following the trajectory and speed defined by the scenario. The policy will reassgin a different target only if its current target is destroyed. A Red drone is eliminated if it makes contact with its designated

target, resulting in an explosion or it is intercepted by a countermeasure that can neutralize it.

Training Blue and Red Policy: As ARES represents a Markov Decision Process each episode starts from an initial state, which defines the available action for both the Blue and Red Teams. Upon the execution of selected actions, the environment transitions to a new state, and rewards are generated based on the actions taken by both teams. This interactive cycle persists until the episode terminates, as shown in Figure 3. The environment incorporates a replay buffer to store the experiences (state transitions, actions, rewards) of both teams. These stored experiences are then sampled and fed into the policy's training loop updates. This process is iteratively repeated until the training session concludes.

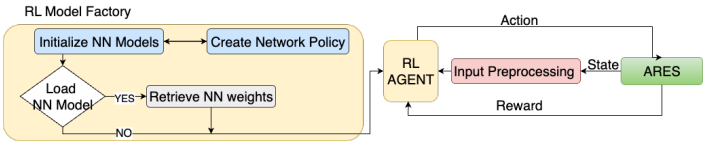


Figure 3. ARES interfacing with the RL Algorithm

A significant feature in ARES is that it allows the Red Teams to be controlled by RL algorithms. This capability allows not only the evaluation of defensive RL agents against heuristic drone behaviors, but also for the rigorous testing of Blue agents against adaptive and learning adversaries. By training the Red Team with diverse RL algorithms, ARES facilitates the exploration of novel attack strategies and nuanced hostile behaviours. This provides a stress-testing environment for blue team algorithms and identifies potential vulnerabilities that might not surface against static or predictable threats. ARES also offers customization options for the Red Team's reward structure, which can easily be extended to incorporate specific adversarial objective, such as selection of targets, depleting resources of Blue Team or evading the fired countermeasures.

3 Experiments

This section highlights the utility of the ARES environment by presenting and discussing results from two distinct scenarios shown in Table 2. These scenarios were specifically designed to validate ARES as a robust simulation environment capable of assisting RL algorithms in developing effective strategies against incoming drone threats, guided by a specified reward structure. The primary objective of the Blue Team's reward structure is to prioritize the elimination of Red Team drones while minimizing Blue Team casualties. The Blue Team operated under different RL policies across the scenarios. In Scenario 1, the Blue Team utilized QMIX (Rashid et al. (2020)) and SAC (Haarnoja et al. (2018)) whereas, in Scenario 2 the Blue Team employs DQN (Mnih et al. (2013)) and SAC algorithms. For comparison, results also include the Blue Team operating on a baseline heuristic policy. In both scenarios, the Red Team consistently operates under a policy that randomly selects an operational Blue agent as a target and moves towards it as specified, until collision or elimination.

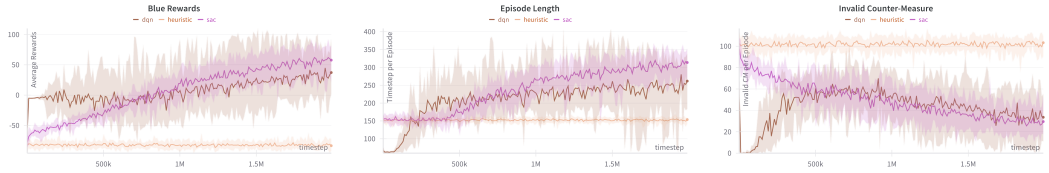
Rewards: The results demonstrate that the Blue Team, operating under various RL policies, surpasses the baseline heuristic policy, indicating successful learning. This establishes ARES's efficacy in framing a complex, real-world problem as a tractable Markov Decision Process (MDP), thereby showcasing its suitability as an environment for RL research and development against UAS. From 4 and 5 it can be seen that SAC achieves higher cumulative rewards compared to QMIX in Scenario 1 and DQN in Scenario 2.

Survivability: A primary objective for the Blue Team is survivability. A longer episode indicates the Blue team survived for a longer duration. As Figure 4 illustrates, RL policies in Scenario 1 consistently demonstrate superior survivability, indicating successful learning. However, the results for Scenario 2, presented in Figure 5, depict a different outcome, the RL policies do not outperform the heuristic. This phenomenon is called a loophole learning (REF), where the Blue Team learns that selecting a no-operation action more frequently than other available actions yields higher cumulative rewards. Consequently, this leads to the premature termination of the episode due to the rapid elimination of the Blue Team.

Selection of CM: A critical aspect of the Blue Team's strategy for survivability is conservation of ammunition. As shown in Figure 4 and 5, both scenarios showcase a reduction in the usage of invalid countermeasures over time, suggesting learning in this regard. However, the decrease in invalid countermeasure usage in Scenario 2 can be misleading. Due to the loophole learning, where the policy favours the no-operation action, there is lower frequency of any countermeasure being chosen. Leading to lower usage of ammunitions while the core issue is the policy's learned inaction rather than strategic ammunition management.

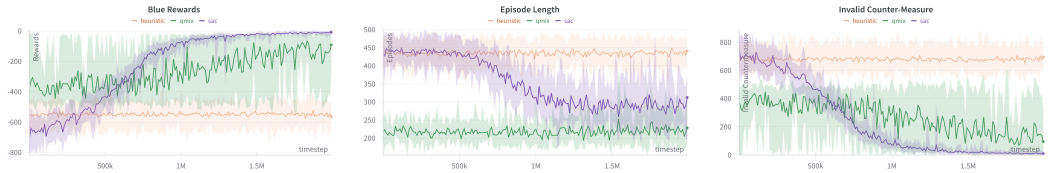
4 Conclusion

ARES is a robust and highly adaptable simulation environment, specifically engineered to facilitate development and evaluation of RL algorithms for countering UAS. Its core strength lies in its extensive customizabil-



(a) Blue rewards (b) Episode length (c) Invalid CM

Figure 4. Scenario 1: Results for the Blue Team operating with the DQN (brown), SAC (pink), and Heuristic (orange) policies, while the Red Team employs a random policy



(a) Blue rewards (b) Episode length (c) Invalid CM

Figure 5. Scenario 2: Results for the Blue Team operating with the QMIX (green), SAC (purple), and Heuristic (orange) policies, while the Red Team employs a random policy

ity, offering researchers and practitioners flexibility in scenario design. The availability of default counter-measures, coupled with configurable environmental parameters, empowers users to construct novel and diverse training scenarios that represent real-world complexities or have the potential to mimic hypothetical challenges. Furthermore, ARES's custom reward structure helps learning strategies with emphasis on specific aspects of an anti-UAS strategy. ARES allows RL agents to learn behaviors that align with desired operational outcomes, such as survivability and resource conservation. This control over the learning objective allows for focused research and experimentation against sophisticated UAS threats.

Future work will leverage ARES to explore more complex swarming UAS behaviors, integrate advanced sensor models, and investigate the efficacy of decentralised control policies for robust, adaptive counter-UAS systems. Blue and Red team features will be enhanced to include ground units and troops with a different set of capabilities and restrictions (Blue) and toggle between a homogeneous and heterogeneous behaviour (Red). ARES not only provides a sophisticated testing ground for current RL advancements but also lays the groundwork for strategising the next generation of autonomous defence strategies against evolving UAS.

REFERENCES

Adoni W.Y. H, Fareedh J. S., Lorenz S., Gloaguen R., Madriz U, Singh A and Kühne T. D. (2024), Intelligent swarm: Concept, design and validation of self-organized uavs based on leader–followers paradigm for autonomous mission planning, *Drones* **8**(10), 575.

Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J and Zaremba W (2016), Openai gym.

Castrillo V.U., Manco A., Pascarella D. and Gigante G. (2022), A review of counter-uas technologies for cooperative defensive teams of drones, *Drones* **6**(3), 65.

Haarnoja T, Zhou A, Abbeel P and Levine S (2018), Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor — arxiv.org, *Proceedings of Machine Learning Research* .

Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D and Riedmiller M (2013), Playing atari with deep reinforcement learning, *arXiv preprint arXiv:1312.5602* .

Molloy O (2024), Drones in modern warfare: Lessons learnt from the war in ukraine, *Australian Army Occasional Paper No. 29* .

Nichols R. K, Mumm H. C, Lonstein W D, Ryan JCH, Carter C and Hood JP (2019), *Unmanned aircraft systems in the cyber domain*, New Prairie Press.

Pong B (2022), The art of drone warfare, *Journal of War & Culture Studies* **15**(4), 377–387.

Rashid T., Samvelyan M., De W. Christian S, Farquhar G, Foerster J and Whiteson S (2020), Monotonic value function factorisation for deep multi-agent reinforcement learning, *Journal of Machine Learning Research* **21**(178), 1–51.

Samaras S., Diamantidou E., Ataloglou D., Sakellariou N. and Vafeiadis A. et al. (2019), Deep learning on multi sensor data for counter uav applications—a systematic review, *Sensors* **19**(22), 4837.

Sutton R. S. and Barto A. G. (2018), *Reinforcement Learning: An Introduction*, second edn, The MIT Press.

Tian Y, Lin F, Li Y, Zhang T, Zhang Q, Fu X, Huang J, Dai X, Wang Y and Tian C (2025), Uavs meet llms: Overviews and perspectives towards agentic low-altitude mobility, *Information Fusion* **122**, 103158.

