Using AI in Wargaming Simulation as a Multi-Domain Decision Support Tool

Greg Chance, Callum Pender, Rory Holland, Chris Halliday

Frazer-Nash Consultancy, UNITED KINGDOM

{g.chance, c.pender, r.holland, c.halliday}@fnc.co.uk

ABSTRACT

Command & Control (C2) planning presents an increasingly complex challenge, such as the growing availability of relevant data and how to process this in useable timeframes. By understanding the likely Red Force responses to a potential Blue Force Course of Action (CoA), planners are empowered to make better strategic decisions using insights from Artificial Intelligence (AI) wargaming. Modelling and Simulation (M&S) tools in combination with AI can rapidly predict Red Force CoA by consuming and processing observational data of the operational picture. We present Red Force Response (RFR), a decision support tool that exploits AI in a wargaming simulator to find potential Red Force CoAs. Using state-of-the-art Deep Neural Network (DNN) algorithms including Proximal Policy Optimisation (PPO) and Curiosity Learning, integrated into a Multi-Agent Reinforcement Learning (MARL) environment, the RFR agent finds both high performing and novel CoAs based on the reward and action selection diversity respectively. A 91% Red Force win probability was achieved in a tactical air scenario when trained for 17,587 episodes against a superior Blue Force. The concept demonstrates an effective use of AI for C2 planning, how cloud computing can be used to effectively train agents and how the concept could scale to larger problems.

1.0 INTRODUCTION

The Dstl Machine Speed Command & Control (MSC2) project was established to transform Command and Control (C2) by enabling faster and more effective C2 processes across all environments, domains, and levels of command. This aligns with the objectives to utilise modelling and simulation technology advantage for the defence-related capability development, threat mitigation and security posture of NATO nations. This work under the MSC2 project aimed to explore the potential for using Artificial Intelligence (AI) technology to control the behaviour of agents within a wargaming simulation to establish the feasibility of introducing complex behaviours, similar to those experienced in real environments. The main contribution of this work is a conceptual 'AI assistant' decision support tool that will help to understand 1) potential adversarial (Red Force) CoA that could be most damaging to a Blue Force, 2) possible novel Red Force CoA and 3) explainability techniques to help understand the effectiveness and novelty of the AI generated Red Force CoA. We present results that demonstrate the application of machine learning to this domain is highly effective.

1.1 Technologies Utilised

The following section details some of the key technologies utilised in this work. Reinforcement Learning (RL) involves an agent learning to make decisions by interacting with its environment to maximise rewards. Multi-Agent Reinforcement Learning (MARL) extends this concept to multiple agents who collaborate and share experiences within a shared environment, leading to more robust and creative solutions. [1]

Environment vectorisation involves running multiple instances of an RL environment in parallel, allowing the agent to interact with several simultaneously. This reduces the time taken to reach training convergence by collecting more diverse data from different environments. This improves sampling efficiency & training stability and reduces the risk of overfitting. Multi-core processing can further increase the performance benefits of vectorisation. [2]

The reward profiles used typically by RL are directly related to actions taken by the agents and the feedback from the environment. This is known as an *extrinsic* reward. Curiosity learning was developed in 2017 as a method to introduce an *intrinsic* reward. The intrinsic reward is designed specifically as a small, regular reward to encourage exploration. Curiosity learning is achieved by calculating the difference between the next state as estimated by a predictor neural network, and the actual in-game next state. The larger the difference between the two, the larger the intrinsic reward, as it seeks to motivate actions in unexplored game state/action pairs. The predictor neural net is updated incrementally during RL training, to reduce curiosity around states already explored during training, to continually encourage novel exploration. Over time, the total curiosity reward trends downwards, as the RL explores more of the available state-action space; the goal is for the battle-winning behaviours to increase simultaneously, so that as the RL explores, it learns useful strategic behaviours, and is rewarded for these.

Therefore, intrinsic rewards and extrinsic rewards are combined and provided for RL, to encourage curious behaviour in the simulator, while learning to reach broad strategic objectives. [3]

1.2 Command: Modern Operations

Command: Modern Operations is an advanced military simulator, used by a range of organisations, including the US Army, Air Force, Navy and Marine Corps, the UK MOD, Dstl, RAAF, Boeing, BAE, Lockheed Martin and parts of NATO. It offers the user the ability to recreate any post-WWII scenario, with detailed control over the operations of each unit. The user can also create and implement any custom scenario to allow for extensive mission planning, simulation and analysis.

A key advantage of CMO is its fidelity – it offers some of the best-in-class wargaming capability. This is reflected by its advanced battle mechanics, which model engagements as probabilistic encounters decided based on unit type, sensors, weapons, and operator experience. Furthermore, CMO offers an expansive range of both modern and historic units, is supported by a large online user base, and is highly customisable for new scenario and unit creation.

The simulator can be run in both Graphical User Interface (GUI) mode and headless mode; headless mode enables CMO to run much faster without loading graphics packages and is called directly via the command line. CMO combat mechanics are stochastic; the game contains a Monte-Carlo simulation feature to exploit this stochasticity to explore a range of possible scenario outcomes. The GUI was used to generate the scenario, described fully in Section 0. A key limitation of CMOs use is the associated license fees to run the software for RL training in headless mode: to undertake this work, the Professional Edition of CMO was necessary[1].

1.3 Rapid Exploratory Modelling Toolset

Rapid Exploratory modelling Toolset (RET) is a Frazer-Nash developed open-source defence agent-based modelling framework, built in Python. RET extends the open-source Mesa library, inheriting core components. RET provides a headless, faster-than-real-time simulation to provide operational analysts a simple and flexible framework to assess a large problem space. RET is a stochastic model, allowing the likelihood of a particular outcome to be measured. It is intended to provide a solution between low fidelity & intensity modelling tools such as spreadsheet modelling and high fidelity & intensity modelling tools such as tools including CMO.

[1] CMO, Available: https://command.matrixgames.com/?page_id=3822

The most significant limitation of using RET as a platform for a multi-domain decision support tool was its fidelity. RET is a mid-fidelity operational analysis, agent-based modelling framework and not a high-fidelity wargaming simulator. RET does not model many of the more complex aspects of wargaming including experience, morale, health, fatigue, maintenance and consumables such as fuel, food and water. Hence a deployed RET decision support tool would be intended to be used alongside a richer tool such as CMO. [2]

2.0 METHOD

2.1 Problem Scenario Formulation

Two scenarios were created, with the aid of a Frazer-Nash military advisor, to model both a Coastal Area Defence (CAD) & Tactical Air Interdiction (TAI) mission. The CAD scenario was implemented in CMO & the TAI scenario was implemented in RET. Both scenarios were abstract and were not representative of any real-world historical engagements.

Key aims for the development of the scenarios were to incorporate multiple domains, with multiple unit types that used a variety of weapons and sensor systems. The inclusion of several unit types was intended to introduce a complex range of strengths and weaknesses that the RL agent would have to understand to demonstrate campaign winning tactics. Another key aim was to create an interdependence of units such that the campaign cannot be won without a level of collaboration between units, which if achieved, would demonstrate a high-level of understanding of the problem space by the RL agent. Success in these scenarios would be a collaborative, targeted RFR, capable of eliminating Blue units by learning and maximising potential strategic advantages.

2.1.1 Command: Modern Operations – Coastal Area Defence

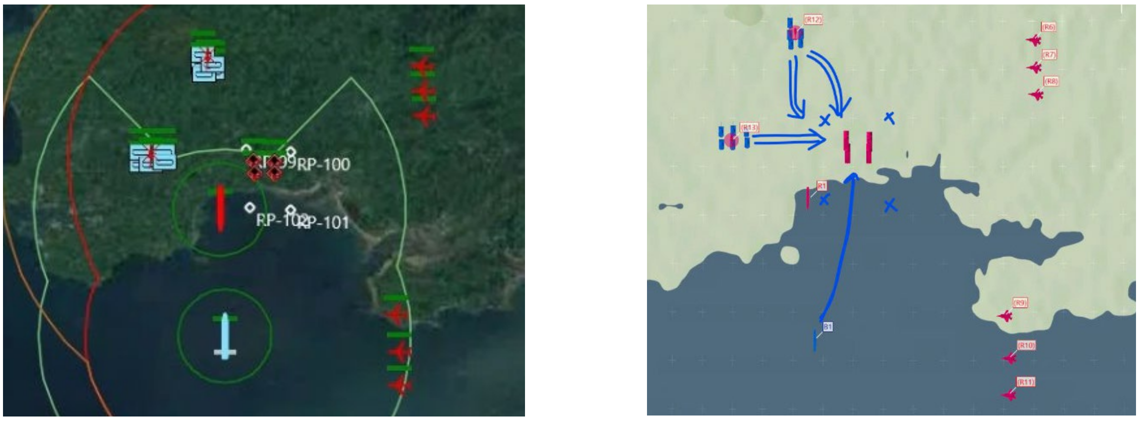
The developed CAD scenario in CMO models the territorial defence of a small coastal area, shown in Figure 1 (a). Units, weapons and sensors in the scenario are closely modelled to real unit capabilities, using the CMO DB3000 database [4]. Teams, units and loadouts are listed in Table 1. The scenario is unbalanced, with each side having relative advantages to exploit.

Table 1: CAD scenario force composition.

Red Force			Blue Force		
Icon	Unit (No.)	Weapons	Icon	Unit (No.)	Weapons
	Type 23 Frigate (1x)	16x 55 Mk8 HE/ER 32x 75 DS30B HE Burst 2x Stingray Mod 1 32x Sea Ceptor		Type 23 Frigate (1x)	16x 55 Mk8 HE/ER 32x 75 DS30B HE Burst 2x Stingray Mod 1 32x Sea Ceptor
	F-35C (6x)	2x AIM-120C-5 ARAAM 2x AIM-132A ASRAAM 6x Paveway IV		AMX-30B2 Tank Group (11x4 Tanks)	30x 105mm Mle F2 APFSDS-T 20x 105mm Mle F2 HE 15x 20mm Single Burst 20x 7.6mm MG Burst
	M777A2 Towed Howitzer Artillery (4x)	200x 155mm/39 HE 60x 155mm/39 Base Bleed 20x 155mm/39 M982 Excalibur Ia-2 Base Bleed			

[2] RET, Available: https://github.com/dstl/RET.

An attacking Blue Force follow a fixed course of action; Blue Force advance two tank groups separately, while simultaneously moving their frigate northward, to assault the x-marked territory in a joint attack, shown in Figure 1 (b). Red Force are defending, and without Red Force action, the land area is captured within an hour in scenario time. Red Force have an air advantage but can only win by developing a strategy for focusing their Paveway missile use on Blue land units. The winning condition for Red is to successfully defend the territory against the Blue Force assault; conversely, the wargame is lost if the territory is captured by Blue Force.



(a) (b)

Figure 1: Showing scenario (a) in CMO and (b) in Tacview for postprocessing of unit CoAs, with the Blue CoA shown using arrows to the contested territory, marked with crosses.

2.1.2 Rapid Exploratory modelling Toolset - Tactical Air Interdiction

The scenario sees the Red Force coordinating a large strike package of several different classes of aircraft with varying loadouts, that aim to deliver a tactical air interdiction mission upon Blue Force heavy armour units. The Blue Forces advance under the protection of air defence upon two key bridges, in aim of capturing them. To win, the Red Force must interdict the Blue Forces before they can capture the key bridges. The composition of forces, including the ammunition capacity of each unt modelled is shown in Table 2.

All platforms, weapons, and sensors modelled in RET were generic and were not intended to represent the performance of any real-life systems.

Figure 2 shows the starting position of the units within the simulation. The circles around units shown, demonstrate the maximum horizontal range of the weapons of each unit. The calculation of range to target in RET includes altitude, so maximum weapon range within the simulation forms a sphere of influence around the unit, rather than a two-dimensional circle. The circles shown also highlight the maximum identification range of the sensors of each unit. Sensors in RET have three confidence levels associated with them. Detection is the lowest level of observation possible, revealing only the position of object. Recognition allows the unit type to be discerned. Identification, which is the highest level of observation, reveals the affiliation and casualty status of the object. The TAI scenario was simulated over an hour window from 12:00 to 13:00 in 10s timesteps in a 200x100km zone of interest.

Table 2: TAI scenario force composition.

Red Force			**Blue Force**		
Icon	**Unit (No.)**	**Target Type (Ammunition)**	**Icon**	**Unit (No.)**	**Target Type (Ammunition)**
A	Strike aircraft (x3)	Land-based vehicles (x6)	F	Air superiority fighter aircraft (x4)	Aircraft (x6)
SEAD	Suppression of enemy air defence aircraft (x4)	Air defence units (x6)	EW	Communications jamming system (x1)	Enemy communications (∞)
J	Electronic warfare aircraft (x1)	Enemy sensing capability (∞)		Surface-to-air missile system (x6)	Aircraft (x5)
F	Air superiority fighter aircraft (x4)	Aircraft (x6)		Armour (x8)	Land-based vehicles (x30)
	Armour (x4)	Land-based vehicles (x30)	C2	Command and control (x1)	None
C2	Command and control (x1)	None	A	Strike aircraft (x1)	Land-based vehicles (x6)
AAW FF	Anti-aircraft warfare frigate (x1)	Aircraft (x50)			
	Reconnaissance satellite (x1)	All units (∞)			
	Ground station (x1)	None			

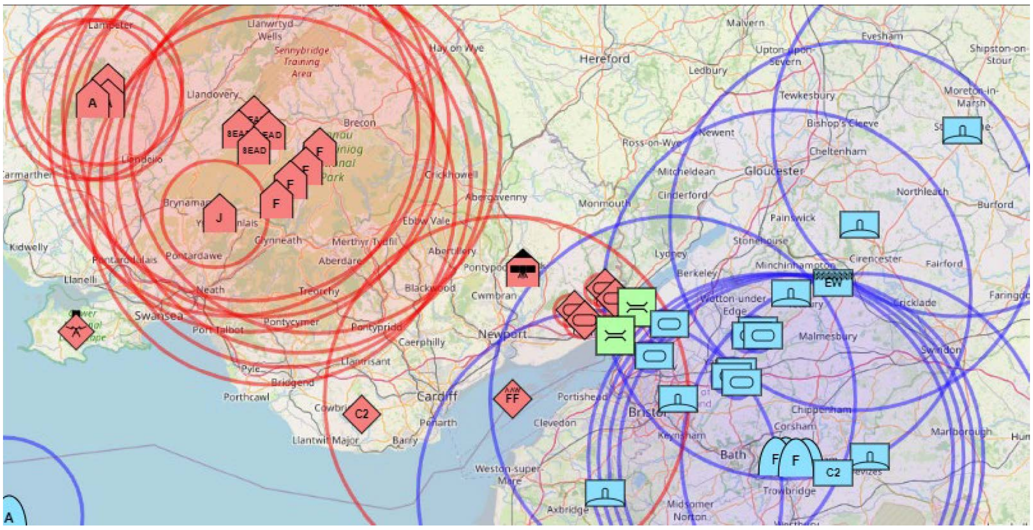


Figure 2: TAI scenario starting laydown, circles indicate weapon ranges.

2.2 Promoting Novel Red Force CoA

Separate actions, observations & rewards were utilised in the different simulation tools, owing to the requirements of the tools and objectives of the scenarios.

2.2.1 Coastal Area Defence

Red units can act each timestep, either to attack Blue units, or to move along the cardinal directions, shown in the action space of Table 3 (a). Three reward profiles were tested: extrinsic, intrinsic, and a combination of extrinsic and intrinsic. Extrinsic rewards are detailed in Table 3 (b). As described in Section 0, intrinsic reward is calculated as the total numerical difference between the predicted next state, as assessed by a neural net using the previous state and actions, and the real next state. Finally, the observation space for each unit, which is an observation of each other unit in the scenario, normalised relative to the scenario x, y, z size, is shown in Table 3 (c).

Table 3: CAD scenario (a) Action space (b) Rewards (c) Observations.

ID	Action
0	Move North One Cell (2km)
1	Move South One Cell (2km)
2	Move East One Cell (2km)
3	Move West One Cell (2km)
4	Attack Enemy Unit

(a)

Unit Action	Reward
Move Unit	+ 0.5
Attack Enemy Unit	+ 5
Winning Game	+ 100 + 10800 - steps
Out-of-Area Penalty	- 10
Out-of-Territory Penalty	= 0

(b)

ID	Observation
0	Linear distance to the observed unit
1	Relative X distance to the observed unit
2	Relative Y distance to the observed unit
3	Relative Z distance to the observed unit
4	Unique ID of the observed unit
5	Affiliation of the observed unit

(c)

If the territory defence was successful, a +100 reward, + max steps – steps taken reward is given: steps taken is the total number of stems taken before the win condition is met. This is designed to motivate efficient winning – the model is rewarded more for winning quickly, than for winning slowly. This only applies to the win condition – no time penalty is applied to a lost game.

The RL was penalised for exploring too far out of bounds, to prevent curiosity exceeding the bounds of a relevant scenario area. For the tested scenario, the bounding area was ±2° latitude, ±2° longitude, from the initial Red unit positions. This is mainly relevant to the F-35Cs in the target scenario.

If a unit tried to move into an inaccessible territory, such as a ship onto land, 0 overall reward was given for the timestep. This was to prevent a unit becoming curious around natural boundaries in the map.

It was hypothesised that the combined extrinsic and intrinsic reward profile would learn fastest and have the best overall result. This hypothesis is supported by [3], since a mix of curiosity-driven learning and overall strategic reward creates a non-sparse, strategy-aligned reward profile.

2.2.2 Tactical Air Interdiction

Observations and rewards were normalised to aid with training performance. The action space, or possible actions for the RL agent to select from the Red Force units allowed the unit to either to move into adjacent grid cells or to use their weapon/countermeasure. The action space implemented is

shown in Table 4 (a).
The extrinsic reward function included a significant yet sparse reward for effectively solving the problem space, by winning the campaign through the destruction of all Blue Force armoured units, awarded to all units that remained alive, at the point of victory. More frequent rewards were included for the destruction of Blue Force units, awarded on a unit-by-unit basis. A small continuous living penalty was also applied to encourage movement and decisive actions. The reward function was intended to reward outcome rather than method to allow the RL agent to generate potentially novel CoAs. The reward function is shown in Table 4 (b).

Red Force units do not have a global view of other units within the game and must rely on their own sensors and communication to detect enemy units. As such, the units must move within sensor range of other units to perceive them. The observation that each unit makes for all other units, which are all expressed numerically, in the simulation is shown in Table 4 (c).

Table 4: Showing (a) action space, (b) reward profile and (c) observation space for the TAI scenario.

ID	Action
0	Move North One Cell (2km)
1	Move East One Cell (2km)
2	Move South One Cell (2km)
3	Move West One Cell (2km)
4	Use Weapon/ Countermeasure

(a)

Unit Action	Reward
Destroy Blue Force unit	+1
Destroy all Blue Force armoured units	+1
Living cost	−0.001

(b)

ID	Observation
0	Linear distance to the observed unit
1	Relative X distance to the observed unit
2	Relative Y distance to the observed unit
3	Relative Z distance to the observed unit
4	Unique ID of the observed unit
5	Affiliation of the observed unit
6	Casualty state of the observed unit
7	Agent type of the observed unit
8	Ammunition remaining of the observed unit

(c)

2.3 Algorithm Selection for a High-Performing Red Force Agent

For this study, the RL agent utilised the Proximal Policy Optimisation (PPO) algorithm as implemented in the Stable-Baselines3 Python library. PPO was primarily selected as the algorithm of choice due to its state-of-the-art nature, training stability, sampling efficiency & versatility.

Unlike other methods, PPO uses a clipped surrogate objective function that prevents excessively large policy changes from a single update. This allows PPO to achieve highly stable and reliable results in complex environments with high-dimensional state spaces. Due to its high sampling efficiency, PPO can learn effectively from fewer interactions with the environment than other methods. PPO has also been demonstrated to be an effective approach to solving a wide variety of problems. [5]

Stable-Baselines3 offers a well-documented, optimised and widely used implementation of PPO. The multi-agent environment wrapper utilised, SuperSuit, also only has native support of environment vectorisation for gymnasium, Stable-Baselines & Stable-Baselines3. [6]

2.4 Architecture & Platform Integration

The architecture of the RFR framework has three main components: the simulation engine, the multi-agent environment and the reinforcement learning agent. We utilised RET & CMO as simulation engines, PettingZoo as a multi-agent environment and Stable-Baselines3's PPO as a reinforcement learning agent.

PettingZoo is a multi-agent equivalent of OpenAI gymnasium utilising an almost identical set of requirements for defining a particular environment, such as a reset and step function. The environment is provided with all of the required methods of each simulation platform to control each of the units in the simulation and advance the model timestep. This offers seamless handling of multiple interacting agents with distinct policies and capabilities.

The user (Blue Force commander) defines the units and the environment, the algorithm and hyperparameters to train with and is returned the data from the evaluation campaigns alongside the trained RL agent.

2.5 Experimental Setup, Configuration & Training

Training was performed on an Azure Windows 10 Virtual Machine (VM). All training was performed on the *Standard NC8as T4 v3* VM, which was allocated 8 virtual CPU cores, 56GB of memory an NVIDIA Tesla T4 GPU. The cloud deployment of training allowed for flexible resource scaling and robust data protection, whilst training utilised the GPU. [7]

Most hyperparameters were left unchanged from the default optimised values provided in Stable-Baselines3. The experimental configurations used, including non-default hyperparameters is shown in Table 5.

Table 5: Experimental configuration of training parameters.

Training Parameter	CAD Scenario	TAI Scenario
Total training steps	10,800,000	40,000,000
Evaluation campaigns run	500	200
Vectorised environments	3	128
Reward horizon (timesteps)	10,800	360
Batch size	10,800	46,080
Entropy coefficient	0.01	0.01

3.0 RESULTS AND DISCUSSION

3.1 Experimental Results

The following section details the results of training from the CAD and TAI scenario. The total time taken for training and evaluation was 127,800s for the CAD scenario and 57,222s for the TAI scenario. Training averaged a mean timesteps per second of 85its/s for the CAD scenario and 735its/s for the TAI scenario.

3.1.1 Coastal Area Defence

To evaluate the effectiveness of RL in the CAD scenario, three reward profiles were used in training and testing. An extrinsic reward profile, intrinsic reward profile, and combined extrinsic and intrinsic reward profile, were used to train three separate agents respectively.

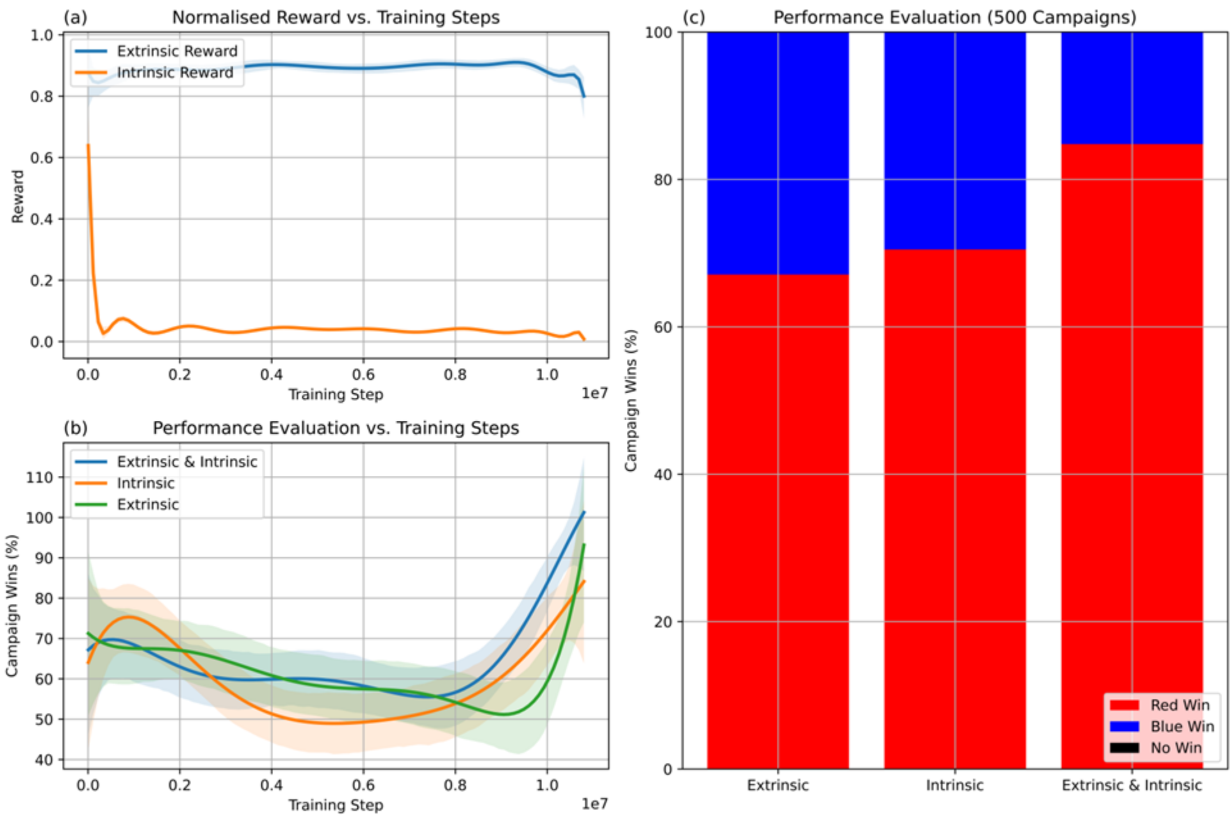


Figure 3: CAD Scenario Results. (a) Normalised reward trends during RL training. (b) Campaign win rate trends during training. (c) Final performance evaluation during 500 evaluations.

The trend in Figure 3 (a) indicates a relatively consistent extrinsic reward during training, with a sharply decreasing intrinsic reward. This indicates that the RL curiosity decreases considerably following an initial exploration of the scenario, while the average reward for achieving strategic objectives remains consistent over training. This supports the intrinsic reward performance evaluation of Figure 3 (b), which shows the intrinsic RL improving quickly initially, but failing to maintain a high campaign win rate. Although the curiosity reward may push the RL towards fast, creative courses of action, it is likely that in lacking an overall long-term strategic reward, the RL fails to adhere to a consistently effective course of action. Conversely, the slower learning rate of the purely extrinsic reward may reflect a lack of creativity and novelty when developing a strategy in the CAD scenario.

It is the combined extrinsic and intrinsic reward profile that outperforms during training, showing a greater overall performance improvement over the same training period. This result indicates that a combined reward profile does encourage faster learning and better outcomes for reinforcement learning, as per [3]. This is reflected by the evaluation results of Figure 3 (c), where the combined extrinsic and intrinsic reward markedly outperforms the separate reward profiles, by 18% and 14% against extrinsic and intrinsic rewards respectively.

3.1.2 Tactical Air Interdiction

The mean total Red Force reward and mean length of one campaign during the training of the RL agent is shown in Figure 4. Mean was a rolling average taken over the last 100 campaigns.

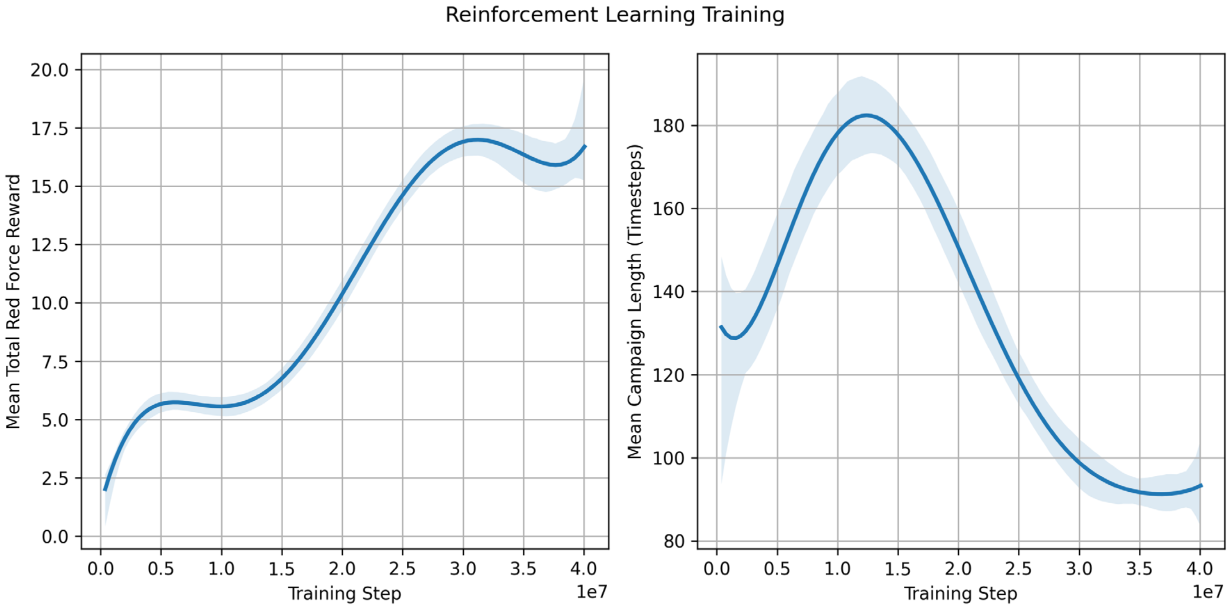


Figure 4: Reinforcement learning training performance monitoring.

Figure 4 shows that training reached a point of relative stabilisation beyond 30 million timesteps, suggesting that the PPO algorithm has converged upon a particular learned behaviour and any further training is unlikely to significantly improve the result. Post training, the RL agent was deployed in an evaluation environment to measure its performance. A comparison of approaches is shown in Figure 5.

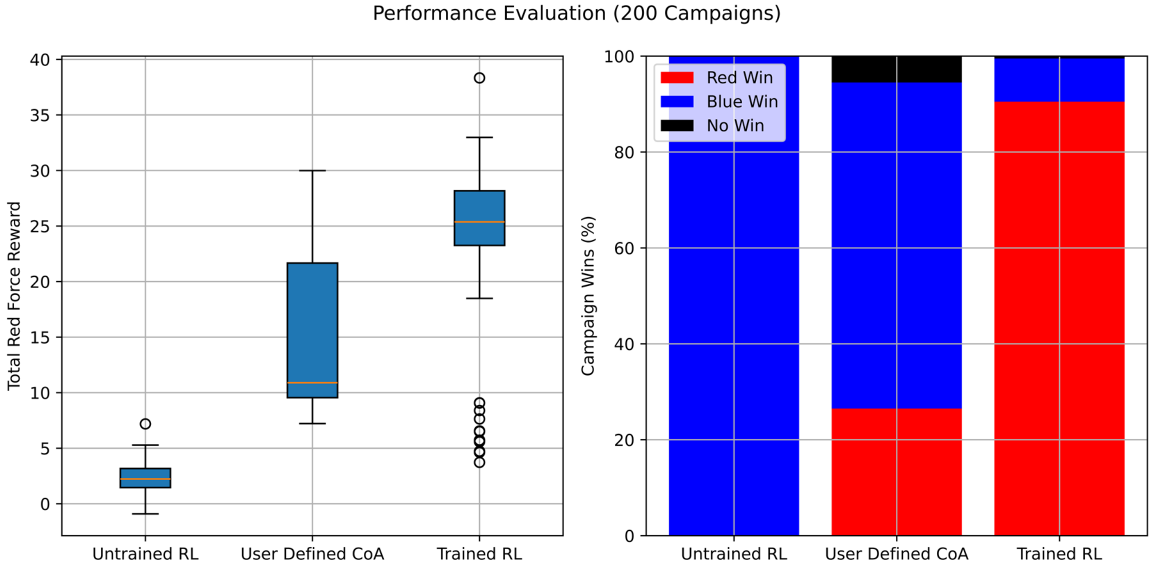
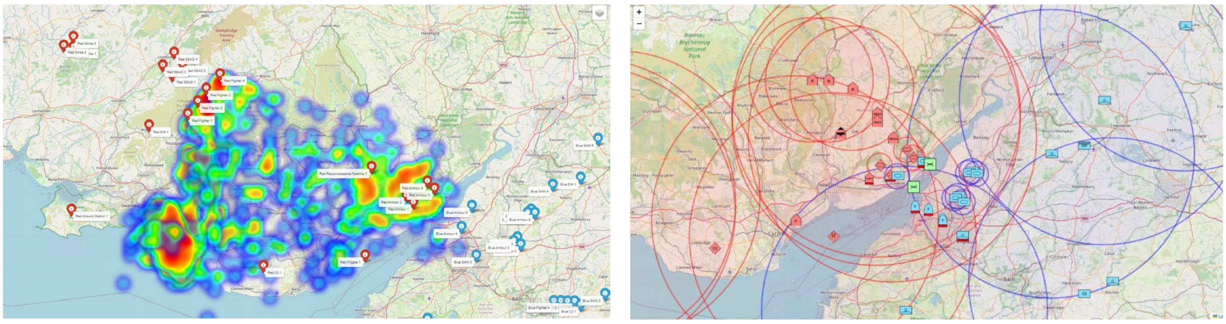


Figure 5: Evaluation of performance of trained RL agent against user and untrained RL agent.

Figure 5 shows that the trained RL agent significantly outperformed the human user-defined and untrained RL agent as expected. The trained RL agent achieved an overall campaign win-rate of 91% compared to 27% for a user-defined CoA. The total reward achieved by the RL agent was significantly higher than the user-defined CoA and the inter-quartile ranges of each distribution did not overlap.

3.2 Develop Explainability Tools & Metrics

A number of techniques were exploited or developed to help interpret the agent generated CoA. The first was an interactive, web-based heatmap which was used to show the position units aggregated over a number of samples. A check box was implemented to switch on or off units which helped to understand the temporal development of individual unit behaviour. A Red Force fighter aircraft example is shown in Figure 6 (a), which indicates that the aircraft have a particular concentration in the southwest area of the map, having learnt to destroy the Blue Force strike aircraft in that region which are tasked with destroying the satellite ground station. These tools allowed us to understand that the RL agent has learnt that campaigns in which it continues to receive periodic global observations from the satellite are more successful than those which it does not and that the ground station must be protected for these observations to be provided for the duration of the campaign.



(a) (b)

Figure 6: RFR explainability tools. (a) Interactive explainability heatmap of Red Force fighter aircraft in the TAI scenario. (b) Model playback tool (timestep 68 of 110).

Other key information on the learnt behaviour can be analysed through model playback visualisation. An instance of a model playback is shown in Figure 6 (b) which shows the RL agent engaging Blue Force air defences prior to successfully engaging Blue Force armour with its strike aircraft.

These tools enabled us to observe the Red Force concentrating forces to limit exposure to different air defence units, protecting the mission critical strike aircraft and waiting at the maximum range of Blue Force air defences to allow all SEAD aircraft units to engage simultaneously. The risks taken by the RL agent may be unacceptable to a real-world Red Force commander. However, the results highlight potential novel high-risk strategies that Red Force may employ to overcome Blue Force, which the Blue Force commander should be prepared to counter. Model playback and heatmap analysis were key to deciphering how Red Force overcame Blue Force and as such can support Blue Force decision making through the visualisation of the learnt CoA.

4.0 CONCLUSION

This work provides initial demonstration of the potential of AI in delivering operational benefits in C2 by using RL to simulate RFR CoAs in custom wargaming scenarios. This work has shown that RL agents can be trained in both a simple (RET) and a more complex (CMO) wargaming simulation environment. In both platforms, novel campaign winning strategies have emerged.

High performance RL agents have been demonstrated across both platforms in multi-domain (Land, Sea & Air) scenarios. The training of these RL agents has included the implementation of curiosity learning and parallelised cloud computing. These technologies have helped lead to the demonstration of an average win rate for the Red Force of 85% in the CAD scenario and 91% in the TAI scenario. Combined intrinsic curiosity and extrinsic strategic rewards have been shown to outperform both individual reward profiles, in learning rate and win percentage, in the CAD scenario. The developed explainability tools have demonstrated deep insight into the learnt behaviours of the RL agents, allowing the user to have a high level of understanding of the Red Force tactics employed and the reason for their successes. Both platforms show potential to be developed into a decision support tool, with increased maturity in particular areas such as improving training efficiency, increasing scenario scale and the inclusion of additional warfighting domains.

5.0 FUTURE WORK

The RFR tool described has shown great success when used with both RET and CMO in a research and development environment. Further development should include work to generalise the tool, increase scale & maturity, and look towards operationalisation. This might involve the inclusion of additional domains; currently the tool has been integrated with the Air, Sea and Land domains, with a limited exposure to electronic warfare. Utilisation of, for example, the cyber domain and intelligence would serve to further increase the usability and effectiveness of the tool.

Operationalisation of the tool could involve at least three use cases:

1. C2 Planner. A desktop application able to suggest possible Red Force CoA to a C2 planner. This idea could be extended to include other interaction methods, such as an API library to use the underlying code in external applications or as a tool to support immersive insights using virtual reality. The tool could also be broadened to support Force Composition Analysis, allowing the C2 planner to understand Red Force strength, command structure and disposition of personnel and equipment.
2. Personnel Training. The tool could also be used as an adversarial training partner used by the wargaming community to facilitate a role as a teaching aid or testing facility. This could be an efficient way to upskill personnel or act as an impartial testing methodology.
3. AI Training. The tool could also be used as a training agent against other computer-based software agents in a machine vs. machine application. This 'tournament' style interaction has proven effective at mastering environments, e.g. AlphaGo [8], and could provide an efficient way to broaden the experience and training data of the model. This is also a route by which a Blue Force agent could be trained adding to the richness and complexity of the overall tool but also moving towards are more generalised system.

6.0 REFERENCES

[1] J. K. Terry, B. Black, N. Grammel, M. Jayakuma, A. Hari, R. Sullivan and L. Santos, "PettingZoo: A Standard API for Multi-Agent," *arxiv,* 2021.

[2] A. Rutherford, B. Ellis, M. Gallici, J. Cook, A. Lupu, G. Ingvarsson and . T. Willi, "JaxMARL: Multi-Agent RL Environments and Algorithms in JAX," *arxiv,* 2023.

[3] Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell and A. A. Efros, "Large-Scale Study of Curiosity-Driven Learning," *arxiv,* 2018.

[4] B. Campaign, "DB3000," 2024. [Online]. Available: https://baloogancampaign.com/command-documentation/db3000/.

[5] J. Schulman, F. Wolski, P. Dhariwal, A. Radford and O. Klimov, "Proximal Policy Optimization Algorithms," *arxiv,* 2017.

[6] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus and N. Dormann, “Stable-Baselines3: Reliable Reinforcement Learning,” *Journal of Machine Learning Research 22,* 2021.

[7] Microsoft, “NCasT4_v3 sizes series,” [Online]. Available: https://learn.microsoft.com/en-us/azure/virtual-machines/sizes/gpu-accelerated/ncast4v3-series?tabs=sizebasic.

[8] D. Silver, A. Huang, C. Maddison, A. Guez and L. Sifre, “Mastering the game of Go with deep neural networks and tree search,” *Nature,* 2016.