Opposed Artificial Intelligence: Developing Robustness to Adversarial Attacks in Attacker-Defender Games via AI-based Strategic Game-Playing

Phillippa Spencer1, Prithviraj Dasgupta2, Michael McCarrick2, Michael Novitzky3, David Hubczenko4, Signe Redfield2, John James3, Adam Jeffery1, Ranjeev Mittu2

1 Defence Science and Technology Laboratory
UNITED KINGDOM

2 Distributed Intelligent Systems Section (Code 5583)
Information Technology Division
U. S. Naval Research Laboratory
UNITED STATES OF AMERICA

3 U S Military Academy
UNITED STATES AMERICA

4 Defence Science and Technology Group
AUSTRALIA

pspencer@dstl.gov.uk; raj.dasgupta@nrl.navy.mil; michael.mccarrick@nrl.navy.mil; michael.novitzky@westpoint.edu; David.Hubczenko@dst.defence.gov.au; signe.redfield@nrl.navy.mil; john.james@westpoint.edu; apjeffery@dstl.gov.uk; ranjeev.mittu@nrl.navy.mil

ABSTRACT

Real-time strategy games have emerged as an attractive environment for developing and analysing artificial intelligence (AI) and deep machine learning based algorithms for competitive, attacker versus defender scenarios. Similarities between the features of computer-based real-time strategy games and war games used for military training exercises also provide a means of transitioning results and lessons from AI-based real-time strategy games to aid and inform decision making capabilities of the warfighter. Our paper investigates this intersection of AI-based real-time strategy games and strategic planning in military decision making through an area called opposed AI. We describe the issues and challenges for developing effective opposed AI in real-time strategy games in the context of an opposed AI competition we had organized recently, using a simulated version of the Capture The Flag game within a marine environment. We discuss the competition entries, results, and lessons learned from feedback given by the competitors, and, prescribe future directions and open challenges for AI-based, complex, and opposed, real-time strategy games.

1.0 INTRODUCTION

In recent years, artificial intelligence (AI) has emerged as the major enabling technology behind automated systems used in military and civilian applications. Automated systems have to continuously interact with other entities in their environment including humans, smart devices, computers and other AI. Traditionally, AI-based systems have been designed while assuming that other entities interacting with them are benign. In other words, the interacting entities do not intentionally behave antagonistically to defeat or subvert the AI. In the real world,

however, as AI-based systems become more pervasive, adversarial actors consistently come up with novel methods to confound AI-based systems, to make them fail and operate in incorrect, unsafe or even dangerous ways. Our paper describes the efforts that are underway to address these challenges as part of Five Eyes (FVEY) The Technical Cooperation Program (TTCP) AI Strategic Challenge (AISC), within a technical area called opposed artificial intelligence (OAI).

The objective of OAI is to develop a better understanding of issues that arise when AI-based systems from different stakeholders with non-aligned, possibly opposing mindsets and objectives interact with each other in environments characterized by noisy and poor quality data. One of the major directions in the OAI pillar is to model the OAI problem as a defender versus attacker game and to develop and analyze different game-playing strategies using reinforcement learning techniques. Towards this objective, we are using a multi-player game called Aquaticus Capture-the-Flag (CTF) game. The game programming interface is written in Python and OpenAI Gym to allow easy and flexible integration with reinforcement learning algorithms that can intelligently learn to play and win the game by analysing the space of possible attack and defence strategies. In this paper, we describe the issues and challenges related to developing effective AI-based techniques that can enable a player to gain a decisive advantage in an OAI scenario along with our experiences from organizing the first OAI Aquaticus CTF competition. We conclude by discussing some of the lessons learned from the competition and by identifying future directions that would generalize OAI research and make it more amenable for transitioning for effective decision making in opposed scenarios on the battlefield.

2.0 AI-BASED GAME PLAYING FOR REAL-TIME STRATEGY GAMES

Over the past few years, deep reinforcement learning (RL) based AI has yielded very promising results towards achieving human-level machine intelligence within game-like scenarios. The first breakthrough was with deep RL-based agents that successfully defeated champion-level human players in 2-player, competitive board games like Chess, checkers, and Go [Silver, 2018]. More recently, RL-based agents have been shown to be successful in defeating human champion players in multi-player real-time strategy (RTS) games like StarCraft-II [Vinyals, 2019], Dota 2 [Berner, 2019] and Capture the Flag [Rojas Herrera, 2019].

RTS games are significantly more complex than 2-player board games. To illustrate the increased complexity with an example, in the game of chess, the board size is 8 x 8, there are 32 game pieces to start with and the game's branching factor is 35, which means that there are 10^{35} possible player moves. In contrast, in an RTS game like StarCraft-II, the environment size is usually 64 x 64, each player can have up to 200 active game pieces, and the worst-case branching factor could reach 10^{50} [Ontanón, 2017]. Besides the size of the game board and moves, there are a few challenges that make RTS games more difficult to play than 2-player board games, as described below:

- *Imperfect information:* In most RTS games, a player might not be able to observe the moves that its opponent makes and, consequently, might not be able to infer the state of the game-play. For example, in Capture the Flag, if an automated player uses a line-of- sight sensor like LASER to detect the location of its opponent, then it cannot detect opponents that are outside the LASER sensor's range. As a result, it is not aware of where the opponent is, which direction it is moving in and whether it has already grabbed the flag[1]. Because the player cannot perceive the opponent's move and the game's current state, it cannot decide with certainty which of its moves would be best. For example, the player does not have sufficient information to decide whether it is more beneficial to attack by moving towards the opponent's flag, whether to defend by moving towards its home flag zone, or, whether to move towards

[1] In many RTS games, imperfect information is simulated via a setting called fog-of-war.

an opponent to intercept and tag it. To solve this uncertainty arising from imperfect information, a player needs to maintain a list of all possible game states that could be reached from the last observed game state (for example, since the player was last able to observe its opponent), reason over these possible game states to estimate the most likely game state and make a move to react effectively to its estimated state. Clearly, these operations introduce significant computational overhead in playing RTS games.

- *Uncertainty with long-term predictions due to sparse rewards:* In RTS games, like Capture the Flag most of the moves made by a player, such as moving towards the opponent's flag or retreating towards the home flag, do not accrue any points. Moves that result in significant points like grabbing and/or capturing the opponent's flag, or tagging the opponent happen relatively infrequently during the game. The sporadic nature of significant points manifests in a challenge called sparse rewards within a reinforcement learning algorithm. Sparse rewards make it difficult for a player to decide which of its currently available low-reward moves would lead to larger reward in the long term.
- *Real-time, strategic and explainable decisions:* Finally, in board games like chess and Go a player has a finite, albeit short time to decide its own move after observing its opponent's move. In contrast, in RTS games like Capture the Flag, players on both sides make their moves simultaneously - a player needs to observe and react effectively to its opponent's moves (e.g., defending its own flag) and continue to pursue its own objectives (e.g., grabbing opponent's flag) at the same time. The rapid, real-time nature of RTS games requires players to make strategic decisions to predict its opponent's long-term objectives from observing the opponent's past and ongoing moves, and at the same time, to chalk out its own objective towards winning the game while maintaining a balanced offense and defence. Once again, strategic decision making amplifies the computations required by a player in an RTS game.

Deep RL algorithms provide useful features that could be used to tackle some of these aforementioned challenges in designing an intelligent RTS game play. First, deep RL algorithms can quickly learn intelligent game plays by analysing moves from past game-plays along with leveraging heuristic search algorithms like Monte Carlo Tree Search [Silver, 2018] to tackle exceptionally large game state and action (or move) spaces. Deep RL-based agents have also been successfully shown to dynamically learn game playing strategies in real-time, when the game scenario evolves. For example, in [Jaderberg, 2019], researchers showed that in a Capture the Flag game, when a ladder was made available to players in the middle of an ongoing game, the players quickly learned to put the ladder to use by using it to climb a fence and use a shorter path to reach the opponent's flag. Finally, in [Vinyals, 2019], researchers showed that a software agent could learn to intelligently play a complex, multi-player RTS game like StarCraft-II with at least human level competence using a variant of the self-play technique from game theory, called league play. Taken together, these advances in deep RL make it a suitable choice for designing intelligent play in RTS games.

RTS games are closely related to war-games [Schwartz, 2020] as both share several common features such as players having to make decisions under time-constraints; players having to make decisions with imperfect information about opponents and about the game's state; and, not knowing precisely the long-term consequences of a player's immediate move. Although deep RL techniques have shown promising results for intelligent play in multi-player RTS games, its application to war-gaming is not yet extensive [Schwartz, 2020; Goodman 2020].

3.0 CAPTURE THE FLAG GAME AND GAME RULES

Capture the Flag (CTF) is a classic war-game that is played between two opposing teams within a bounded playing field. The playing field is divided into two halves, one for each team, and each team has a flag located within its half, inside an area called the flag zone. The objective of each team is to capture its opponent's flag

by grabbing the opponent's flag and returning into its own flag zone while carrying the opponent's flag, without being tagged by the opponent team members. Each flag grab and flag capture accrues points for the team capturing the flag, while getting tagged and losing the flag after grabbing it diminishes points. The game continues for a specific time and the team with the highest points at the end of the game is the winner.[2] CTF has been used extensively for military training over several decades. The attractiveness of CTF as a game-based training exercise draws from the fact that while the game rules and objectives are straightforward and easy to learn, continued interactions between intra- and inter- team members regarding timing and logistics of attacking versus defending can yield complex winning strategies over the course of the game.

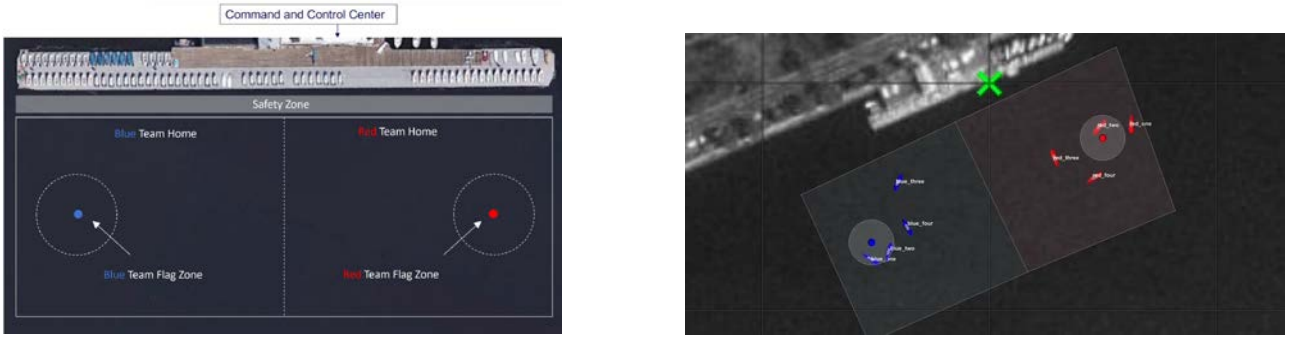

Figure 1. *Screenshot of an Aquaticus Capture The Flag game inside MOOS-IvP simulator (Left) and an Aquaticus CTF game in progress with the MOOS-IvP simulator interface superimposed on a real marine environment; each team has four players (Right).*

3.1 Aquaticus Capture the Flag (CTF) Game

Aquaticus Capture the Flag is a simplified version of the CTF game where human players are replaced by marine vehicles that are controlled either remotely by a human or autonomously by an on-board software agent. As in conventional CTF, there are two teams referred to as red and blue. As shown in Figure 1, the playing field is a 160 X 80 square meters rectangle, divided into two halves measuring 80 X 80 square meters each. Each team's flag is located roughly three quarters of the way from the centerline; the flag zone is a 10 meter circle centered at the flag's initial location.

3.1.1 Aquaticus CTF Agent Capabilities

Aquaticus agents can move at variable speeds; different agent speeds can be set from a configuration file. The speed of an Aquaticus agent is limited by the dynamics of the modeled vehicle. An agent can have a heading in the range of 0 – 360 degrees; agent headings can discretized into intervals. Agent actions are given by the cross product of the agent speeds and agent headings. For example, with two possible agent speeds, high and low, and six different agent headings, the agent would end up having 12 possible actions. The game simulator handles collision avoidance between the agents during the game.

3.2 Aquaticus CTF Game Rules

Three types of rules specify the actions performed on the agents and the points system used in the Aquaticus CTF game, as described below:

[2] More complex versions of CTF involve players being captured and sent to a jail if they are tagged inside the opponent's half, along with rescuing players from jail. While such complexities are planned for later iterations of our project, we do not consider them in our current CTF game as we are interested to explore the feasibility of using RL algorithms to learn strategies that can perform effectively in the simpler game.

Tagging Rules: An agent (tagger) may tag an opponent (taggee) when the following conditions are met:

- Tagger must be in their team's zone.
- Taggee must be in their opponent's zone.
- Tagger must be untagged.
- Tagger must be within 10 meters of the taggee.

If an agent that was carrying a flag gets tagged it will drop the flag; an agent that is tagged is not able to grab a flag. A tagged agent can get untagged only if it re-enters its flag zone. Tagging can be attempted by an agent once every 30 seconds. An agent gets automatically tagged if it leaves the boundaries of the game.

Flag Grab and Flag Capture Rules: An agent may grab the opponent's flag if it enters the opponent's flag zone while still being untagged and the flag is still available (not being carried by another agent). An agent captures a flag when it enters into its own team's flag zone while carrying its opponent's flag.

Table 1. Points scored for different actions in the Aquaticus CTF Game

Action	Points
Grab opponent flag	+1
Capture opponent flag	+2
Tagged, not carrying flag	-1
Tagged, carrying flag	-2

Scoring Rules: A team scores points for grabbing and capturing the opponent's flag while it loses points when it gets tagged by the opponent. The points for the different scoring-related actions are given in Table 1.

After a team has captured the flag, the game is reset - each flag is returned to its initial location inside its home flag zone, team members return to their initial locations within their respective home zones and a new round of the game starts. The game is run for a specified time of T_{game} minutes and the team that gets the higher score at the end of the game is the winner.

4.0 SOFTWARE SIMULATION PLATFORMS: MOOS-IVP, AQUATICUS GYM

The Aquaticus CTF game is implemented on the MOOS-IvP (Mission Oriented Operating Suite-Interval Programming) software. MOOS-IvP is a messaging based, distributed, middleware developed at MIT for simulating communications and decision making on multiple, coordinating autonomous marine vehicles [Benjamin, 2010]. MOOS-IvP consists of two main components – a message passing service called MOOSDB that enables two or more distinct applications (e.g., autonomous marine vehicle's controllers) to communicate with each other, and a decision-making component called IvP that encapsulates the behaviours of autonomous agents and resolves conflicts between decisions made by those behaviours when multiple agents interact with each other. MOOS-IvP provides a scripting based interface to implement behaviours that control the movement of marine vehicles and a graphical interface to observe the progress of the simulation. The Aquaticus CTF software implements the game environment including the playing field boundaries and zones; ranging and location sensors on the vehicles; game rules for grabbing, capturing and tagging; and game states such as locations of players and flags during the game. It interacts with MOOS-IvP via messages and provides the

capability of implementing behaviours of the players using a high-level language like C++ or Python. Aquaticus CTF software was originally developed to facilitate human-robot interactions in a marine CTF game [Novitzky, 2019]. Subsequently, it was adapted to enable deep reinforcement learning-based tools to learn and control the behaviour (direction and heading) of autonomous vehicles playing the Aquaticus CTF game [Gupta, 2018].

For the integration of deep reinforcement learning techniques with MOOS-IvP, we developed an AI-Gym [Brockman, 2016] and Stable-baselines [stable-baselines] based interface to the MOOS-IvP software. AI-Gym is a widely used software library written in Python that provides a standard interface to the main operations within an RL algorithm such as initializing an environment, taking actions inside the environment, and, receiving the observation, reward and completion status of the task being performed by the RL agent as a result of different actions within the environment. It abstracts away the details of the environment dynamics from the developer and helps to modularize the program design. Stable-baselines is a Python implementation of state-of-the-art deep RL algorithms that supports modern deep learning libraries such as Tensorflow and PyTorch. Like AI-Gym, Stable-baselines helps to abstract away implementation details of RL algorithms from the developer by providing a standard interface for training and evaluating deep neural network learning models. To enable compatibility between the MOOS-IvP software that is written in C++ and the Python-based software in AI-Gym and Stable-baselines, we used a tool called PyMOOS that is provided with MOOS-IvP and enables communication with the MOOSDB, via sending and receiving messages. A schematic of the different software components of MOOS-IvP, Aquaticus CTF game logistics and the AI-Gym and Stable-baselines is illustrated in Figure 2.

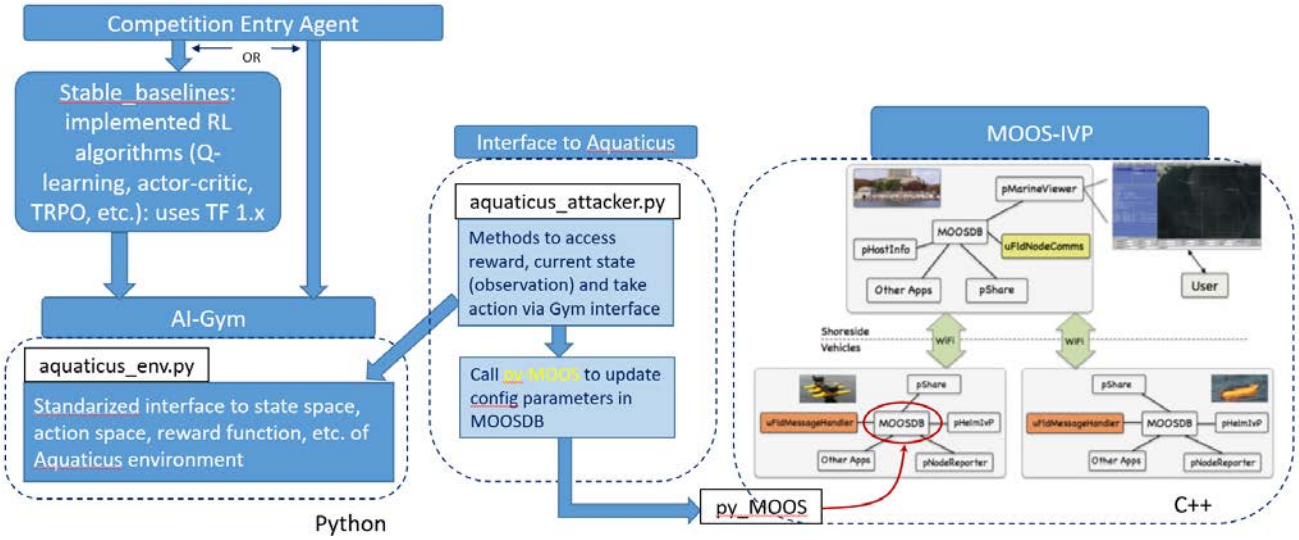

Figure 2. Schematic showing the integration of Stable-baselines and AI Gym with the Aquaticus CTF game engine and MOOS-IvP simulation platform

5.0 FIRST OPPOSED AI AQUATICUS CTF COMPETITION AND RESULTS

The first Opposed AI Aquaticus CTF competition was held on July 14, 2021. The competition was entirely simulation-based using the MOOS-IvP simulation platform for simulating agent movements and inter-agent communication, Aquaticus CTF for game mechanics and Aquaticus-Gym for implementing RL algorithms for agent decision making. Each team had only one player (agent). Moreover, to keep the game simple, the blue

agent was always in a defender role, circling at a fixed radius around its flag on the periphery of its flag zone. The objective of each competitor was to design the control algorithm for the red agent that would determine its mobility strategy to start from its home zone, grab the blue flag and return to its flag zone without getting tagged by the blue agent. Each competitor's entry would have to play the game for game time T_{game} = 5 minutes wall clock time, which corresponded to 1250 ticks on MOOSDB's clock; a MOOS-IvP speed up or time warp of 4x was used for the duration of the game.

The competition was hosted on a computer at the United States Military Academy, West Point, NY. We received four submission entries for the competition. On the day prior to the competition all the competitors met virtually with the organizer to ensure that their competition entry agents operated correctly on the competition computer at USMA. This virtual session was crucial to address any potential issues due to differences between the hardware and software drivers across competitors' computers and the competition computer. The competition was live-streamed from USMA to stakeholders within the Five-Eyes community and was attended virtually by nearly 30 attendees. Figure 3 shows a snapshot from one of the games during the competition. Table 2 summarizes the main aspects of the different entries to the first OAI Aquaticus CTF competition.

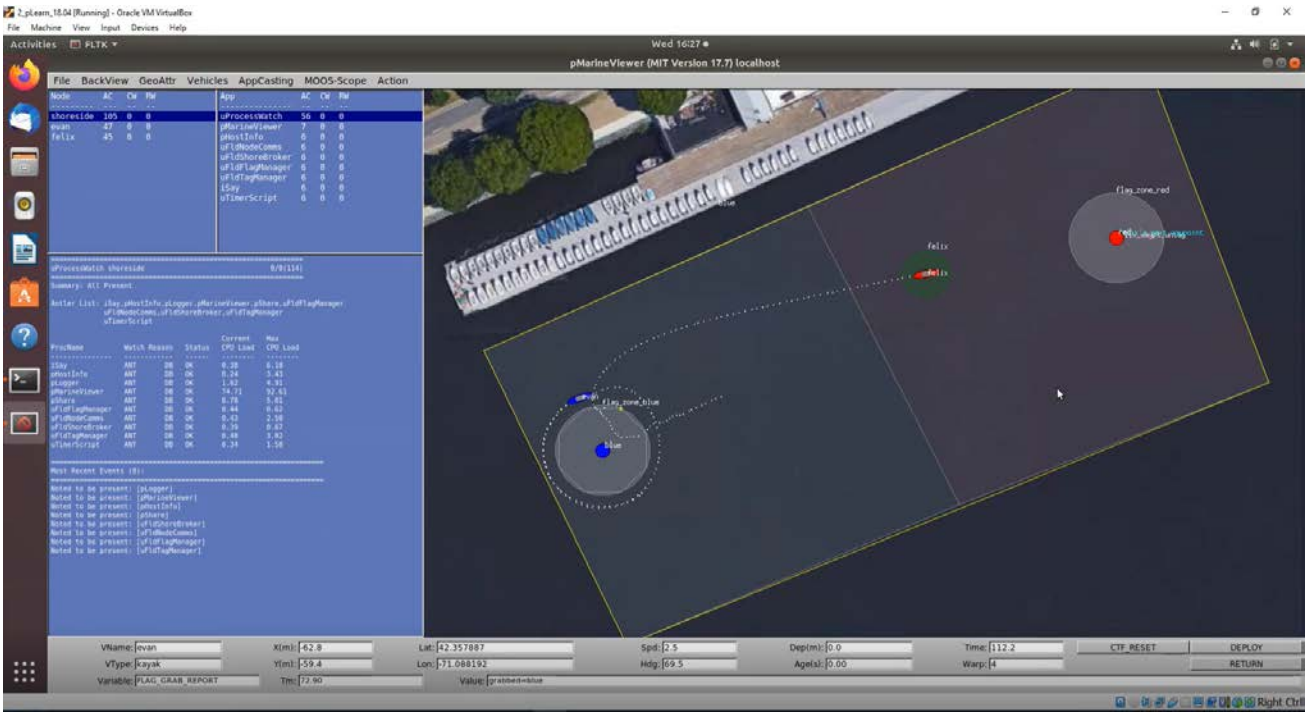

Figure 3. Snapshot of a game-play during the first OAI Aquaticus CTF competition showing the red attacker returning to its home flag zone after a blue flag grab; the blue defender uses a simple defence strategy by circling around its flag at a fixed radius.

6.0 LESSONS LEARNED AND FUTURE DIRECTIONS

The first OAI Aquaticus CTF competition provided some invaluable insights into the feasibility of using intelligent game play in RTS games to inform strategies in opposed, attacker versus defender scenarios. We identify some of these below and discuss a few related challenges and future directions below:

Table 2. Comparison of the main aspects of the different entries at the First OAI Aquaticus CTF Competition

Team Name/Loc	**Team 1**	**Team 2**	**Team 3**	**Team 4**
Framework	Aquaticus-Gym (AI-Gym + Stable-baselines 2.0)	Aquaticus-Gym (AI-Gym + Stable-baselines 3.0)	Aquaticus-Gym (AI-Gym + Stable-baselines 2.0)	Aquaticus
Attacker Control Algorithm (: used in competition)*	Deep RL: Deep Q-network (DQN)* and Actor Critic (A2C)	Deep RL: Proximal Policy Optimization (PPO2)*, Deep Q-network (DQN), Actor Critic (A2C)	All-Aspect Proportional Navigation [Sim 2000] + Potential fields	Manually designed heuristic
Reward Function or Objective Function	Sparse reward between -1 and 1 proportional to distance from flag, enemy and boundary; large reward (penalty) for flag grab or capture (getting tagged)	1) Smooth reward function (gradually increasing positive and negative rewards); 2) Sparse reward function (only rewards for win or lose actions like tagging opponent, flag grab or capture; leaving boundaries)	All-Aspect Proportional navigation [Sim 2000] with repulsive potential field from opponent and field boundaries	Heuristics-based: Head deterministically to fixed location either north or south (chosen randomly) of opponent flag zone, then move head-on towards flag to grab it
Hyper-parameter Tuning	Manual	Manual	Manual	Manual
Training Time (iterations)	1.5 X 10^5 (DQN); 2 X 10^5 (A2C) time steps	0.5 X 10^6 - 2 X 10^6 time steps	N/A (not machine learning based)	N/A (not machine learning based)
Average Score (pre-competition)	~80% wins	Average rewards: ~ -1 (PPO2 and DQN); ~ -4 (A2C)	100% wins	~14% wins
Competition Score	18	6	33	9
Challenges Encountered	-	-	RL was less effective than potential fields for current competition problem	Naïve, manual approach that could be defeated easily by semi-intelligent defensive behaviours

Advanced Capabilities (not used in competition)	Learned to capture flag against a randomization of two defender movement strategies – circle around home flag and 8-shaped pattern around flag	Learned to capture flag against a randomization of three defender movement strategies – circle around home flag; arc north or arc south to opponent flag and back; arc to random location and back	Successful against different attack strategies; could be easily switched between attacker and defender roles; suitable as a baseline for comparison with other RL-based strategies	

- *Countermeasures should match enemy capabilities:* An interesting take away from the competition was that simpler, heuristics-based attack strategies like potential field based navigation (Team 3) could outperform complex strategies learned using more sophisticated techniques like deep RL (Teams 1 and 4), albeit when the defender used a naïve strategy like circling around its flag. Nevertheless, RL-based strategies were found to be more effective against intelligent defender strategies such as combinations of different movement patterns, as reported by Teams 1 and 4 (Table 2, last row). These findings point to the direction that it makes sense for players to train their strategy against a variety of strategies deployed by their opponent. Techniques like league play and curriculum learning [Vinyals, 2019; Goodman, 2020] are essential to build effective defences against such mixed strategy play by the opponent.
- *Parameter and hyper-parameter tuning is paramount:* The importance of parameter and hyper-parameter tuning was found to be crucial by all competition teams. For our competition, all competition entries employed manual parameter tuning. However, as the game is made more complex in future competitions it will becomes infeasible to test the effects of hundreds of different parameter and hyper-parameter values on the player's performance. Automatic deep learning hyper-parameter tuning and optimization tools [Akinremi, 2021] appear to be a useful resource in eliciting improved performance once a player's game playing algorithm, RL-based or otherwise, has been decided by the programmers.
- *Predict and exploit asymmetric situations:* The relative advantage that a player has over its opponent varies dynamically as a game proceeds. In our competition, Team 3, which scored 100% wins during testing and scored the highest during the competition was able to exploit the naïve circle-around-the-flag strategy of the defender to win decisively. As another example, during the course of CTF if a player sees that it has clear line-of-sight to the opponent's unguarded flag while all of its opponents are being intercepted by its teammates, it would make sense for the player to proceed to grab the flag instead of loitering to aid its teammates. These situations called windows of superiority [Atherton, 2020], where the balance between the player and its opponents becomes asymmetric, appear and disappear for very short and fleeting durations during the game. A player that can accurately predict its future windows of superiority, schedule logistics to improve its lethality and degrade its opponents' capabilities during the windows, can score a decisive advantage over its opponents. Techniques like hierarchical RL where a player simultaneously learns two tasks: how to respond to its opponent during its current move as well as how to predict its opponent's behaviour during future moves hold promise to solve towards identifying and exploiting asymmetric situations during the game.
- *Simulation fidelity and extensions:* The fidelity of the simulation environment to physical units' characteristics such as manoeuverability, latency, and network connectivity in the battlefield is essential for validity of results and future transition to realistic scenarios involving marine vehicles and human

machine teaming. However, there is a tradeoff between fidelity and complexity of the simulation software – high fidelity simulation tools usually have a steep learning curve that might deter their widespread adoption for experimentation. To alleviate such hurdles in the adoption and use of powerful gaming and simulation platforms for future competitions, it is useful to develop software that could abstract away specifics related to lower level operations of the simulator. For our competition, the scripting based configuration tools of MOOS-IvP and our integration of AI-Gym and Stable-baselines with MOOS-IvP for the OAI Aquaticus competition helped alleviating issues related to learning a new simulation environment for the environment. A useful and interesting direction for future Aquaticus CTF competitions is to integrate higher fidelity networking simulation capabilities and commensurate abstractions for developers with RTS game simulation tools such as MOOS-IvP. Most existing RTS game platforms that support integration with AI algorithms, such as StarCraft-II Learning environment, and micro-RTS, do not consider networking and communication simulation between game pieces or units. On the other hand, war game simulation environments like Command: Modern Air/Naval Operation or warfighter training environments Next Generation Threat Simulation do not yet offer suitable AI integration interfaces. Integration of RTS game environments with network simulators would afford a greater fidelity to field operations as well as provide methods to simulate attacker versus defender games at the networking level, along with the currently supported decision level.

- *Autonomous players should explain decisions and actions:* A computer algorithm that makes important and consequential decisions must possess the capability to explain to a human why it made certain decisions. Rather than building explainability *post-hoc* into an AI algorithm, it is useful to proactively integrate methods that an intelligent player uses to record and analyse observations, techniques that it uses for reasoning and updating its beliefs about the current situation, alternatives that it faces while making decisions and rules that it uses to make a decision. These explainability-enhanced techniques could then be analysed by human decision makers to correct possible flaws in the AI's decision making algorithm as well as to inform and improve their own decision making skills.
- *Handling non-conformance to game rules:* A key difference between computer-based games and real-life scenarios is that, in the latter, players might not always abide by the rules of engagement. In computer games, an entity such as a game manager or referee enforces game rules and penalizes or disqualifies players for breaking game rules. In contrast, opponents in real-life scenarios like battlefield combat might take decisions that break the conventions of engagement without receiving any penalties. Adversarial AI techniques like adversarial training [Tramer, 2017] have addressed some issues related to developing robustness against malicious opponents that try to fool a machine learning based classifier. Effective defences using approaches like moving target defense [Cai, 2016] and game theory [Ferguson-Walter, 2019; Pawlick, 2021] have been researched extensively in the field of cyber security. It makes sense to investigate commensurate techniques for AI-based RTS game playing and make it robust against malicious adversaries.
- *Verification and Validation in Complex Games*: The reason we use AI to analyse complex settings like RTS games and war-games is that humans cannot predict precisely what circumstances will arise, and, therefore, cannot adequately specify what the desired actions should be. Without adequate specifications, verification and validation of these systems is particularly challenging. The use of a game-based context like Aquaticus CTF provides an arena in which to explore verification and validation techniques in a more controlled environment. In particular, as we explore more complex game rules and scenarios in the Aquaticus CTF game, the system's responses to adversarial tactics will provide unique insight into how performance degrades under adversarial conditions, what kinds of adversarial activities particularly affect AI performance, and whether specific AI development techniques result in systems that are more or less robust to particular adversarial techniques.

In this paper, we discussed the feasibility of using RTS games like CTF as a means to leverage AI-based, machine learning algorithms towards analysing complex interactions between opposed entities like attackers and defenders. Our ongoing work includes enhancing the RL interface to Aquaticus CTF game environment to be able to handle multiple players per team along with addressing some of the open issues and future directions discussed in this section. To encourage the involvement of a larger community of machine learning researchers and developers, and, to promote rapid development of novel AI game-playing algorithms, a series of Aquaticus CTF competitions are also being organized over the forthcoming years. As the AI learns to play the game more effectively, the game will be made harder by introducing complex interactions such as players being able to only partially observe each other's moves due to simulated fog-of-war, and, adversaries surreptitiously using malicious tactics such as deception, forgery and illegal moves to gain unfair advantage in the game. Overall, these efforts are expected to culminate in improved understanding of the effectiveness of different AI strategies in opposed environments, demonstrate how opposed AI operates, and, improve the awareness and engagement of the scientific community in opposed AI. We envisage that with further understanding of these challenges, the results of AI-based game playing in complex, opposed, RTS games can be transitioned successfully to real-life combat scenarios to aid the warfighter.

REFERENCES

[1] Akinremi, B. (2021, July). Best Tools for Model Tuning and Hyperparameter Optimization. URL: https://neptune.ai/blog/best-tools-for-model-tuning-and-hyperparameter-optimization; Last Retrieved: August 30, 2021

[2] Atherton, K. (2020). DARPA Wants Wargame AI To Never Fight Fair. URL: https://breakingdefense.com/2020/08/darpa-wants-wargame-ai-to-never-fight-fair/; Last Retrieved: August 29, 2021.

[3] Benjamin, M. R., Schmidt, H., Newman, P. M., & Leonard, J. J. (2010). Nested autonomy for unmanned marine vehicles with MOOS-IvP. Journal of Field Robotics, 27(6), 834-875.

[4] Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., ... & Zhang, S. (2019). Dota 2 with large scale deep reinforcement learning. arXiv preprint arXiv:1912.06680.

[5] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Openai gym. arXiv preprint arXiv:1606.01540.

[6] Cai, Gl., Wang, Bs., Hu, W. et al. Moving target defense: state of the art and characteristics. Frontiers Inf Technol Electronic Eng 17, 1122–1153 (2016).

[7] Ferguson-Walter, K., Fugate, S., Mauger, J., & Major, M. (2019, April). Game theory for adaptive defensive cyber deception. In Proceedings of the 6th Annual Symposium on Hot Topics in the Science of Security (pp. 1-8).

[8] Goodman, J., Risi, S., & Lucas, S. (2020). AI and Wargaming. arXiv preprint arXiv:2009.08922.

[9] Gupta, A., Novitzky, M., & Benjamin, M. (2018, October). Learning autonomous marine behaviors in MOOS-IvP. In OCEANS 2018 MTS/IEEE Charleston (pp. 1-10). IEEE.

[10] Jaderberg, M., Czarnecki, W. M., Dunning, I., Marris, L., Lever, G., Castaneda, A. G., ... & Graepel, T.

(2019). Human-level performance in 3D multiplayer games with population-based reinforcement learning. Science, 364(6443), 859-865.

[11] Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., & McDaniel, P. (2017). Ensemble adversarial training: Attacks and defenses. arXiv preprint arXiv:1705.07204.

[12] Montvieux, Inc. (2020). Hunting of the Plark. URL: https://github.com/montvieux/plark_ai_public, Last Retrieved: August 30, 2021.

[13] Novitzky, M., Robinette, P., Benjamin, M., Fitzgerald, C., & Schmidt, H. (2019). Aquaticus: Publicly Available Datasets from a Marine Human-Robot Teaming Testbed. 2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 392-400.

[14] Ontanón, S. (2017). Combinatorial multi-armed bandits for real-time strategy games. Journal of Artificial Intelligence Research, 58, 665-702.

[15] Pawlick, J., & Zhu, Q. (2021). Game Theory for Cyber Deception: From Theory to Applications. Springer Nature.

[16] Rojas Herrera, S. (2019). Applying deep reinforcement learning to Berkeley's capture the flag game (Bachelor's thesis, Uniandes).

[17] Schwartz, P. J., O'Neill, D. V., Bentz, M. E., Brown, A., Doyle, B. S., Liepa, O. C., ... & Hull, R. D. (2020, April). AI-enabled wargaming in the military decision making process. In Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II (Vol. 11413, p. 114130H). International Society for Optics and Photonics.

[18] Silver, D., Hubert, T., Schrittwieser, J., *et al*. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science, 362(6419), 1140-1144.

[19] Sim, Y.C., Leng, S.B. & Subramaniam, V. An All-Aspect Near-Optimal Guidance Law. *Dynamics and Control* **10,** 165–177 (2000).

[20] Stable-Baselines; URL: https://stable-baselines.readthedocs.io/en/master/; Last retrieved: August 25, 2021

[21] Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., ... & Silver, D. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature, 575(7782), 350-354.