2022-09

ENLISTING AI IN COURSE OF ACTION ANALYSIS AS APPLIED TO NAVAL FREEDOM OF NAVIGATION OPERATIONS

Allen, John T., II

Monterey, CA; Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

ENLISTING AI IN COURSE OF ACTION ANALYSIS AS APPLIED TO NAVAL FREEDOM OF NAVIGATION OPERATIONS

by

John T. Allen II

September 2022

Thesis Advisor:	Christian J. Darken
Co-Advisor:	Jonathan K. Alt
Second Reader:	Kenneth Maroon, U.S. Naval Academy

Research for this thesis was performed at the MOVES Institute.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2022	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE ENLISTING AI IN COURSE OF ACTION ANALYSIS AS APPLIED TO NAVAL FREEDOM OF NAVIGATION OPERATIONS	**5. FUNDING NUMBERS**
6. AUTHOR(S) John T. Allen II	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

Navy Planning Process (NPP) Course of Action (COA) analysis requires time and subject matter experts (SMEs) to function properly. Independent steamers (lone destroyers) can soon find themselves lacking time or more than 1–2 SMEs or both. Artificial Intelligence (AI) techniques implemented in real-time strategy (RTS) wargames can be applied to military wargaming to aid military decision-makers' COA analysis. Using a deep-Q network (DQN) and the ATLATL wargaming framework, I was able to train AI agents that could operate as the opposing force (OPFOR) commander at both satisfactory and near-optimal levels of performance, after less than 24 hours of training or 500000–learning steps. I also show that under 6 hours or 150000–learning steps does not result in a satisfactory AI admiral capable of playing the role as the OPFOR commander in a similarly sized freedom of navigation operation (FONOP) scenario. Applying these AI techniques can save both time onboard and time for reachback personnel. Training AI admirals as quality OPFOR commanders can enhance the NPP for the entire Navy without adding additional strain and without creating analysis paralysis. The meaningful insights and localized flashpoints revealed through hundreds of thousands of constructive operations and experienced by the crew in live simulation or simulation replays will lead to real world, combat-ready naval forces capable of deterring aggression and maintaining freedom of the seas.

14. SUBJECT TERMS AI, artificial intelligence, COA, course of action, COA development, USN, Navy, FONOPS, freedom of navigation operations, wargaming, RL, reinforcement learning	**15. NUMBER OF PAGES** 79
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

ENLISTING AI IN COURSE OF ACTION ANALYSIS AS APPLIED TO NAVAL FREEDOM OF NAVIGATION OPERATIONS

John T. Allen II
Lieutenant, United States Navy
BA, University of Texas, Austin, 2006

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MODELING, VIRTUAL ENVIRONMENTS, AND SIMULATION

from the

NAVAL POSTGRADUATE SCHOOL
September 2022

Approved by: Christian J. Darken
Advisor

Jonathan K. Alt
Co-Advisor

Kenneth Maroon
Second Reader

Gurminder Singh
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Navy Planning Process (NPP) Course of Action (COA) analysis requires time and subject matter experts (SMEs) to function properly. Independent steamers (lone destroyers) can soon find themselves lacking time or more than 1–2 SMEs or both. Artificial Intelligence (AI) techniques implemented in real-time strategy (RTS) wargames can be applied to military wargaming to aid military decision-makers' COA analysis. Using a deep-Q network (DQN) and the ATLATL wargaming framework, I was able to train AI agents that could operate as the opposing force (OPFOR) commander at both satisfactory and near-optimal levels of performance, after less than 24 hours of training or 500000–learning steps. I also show that under 6 hours or 150000–learning steps does not result in a satisfactory AI admiral capable of playing the role as the OPFOR commander in a similarly sized freedom of navigation operation (FONOP) scenario. Applying these AI techniques can save both time onboard and time for reachback personnel. Training AI admirals as quality OPFOR commanders can enhance the NPP for the entire Navy without adding additional strain and without creating analysis paralysis. The meaningful insights and localized flashpoints revealed through hundreds of thousands of constructive operations and experienced by the crew in live simulation or simulation replays will lead to real world, combat-ready naval forces capable of deterring aggression and maintaining freedom of the seas.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

AI	artificial intelligence
BLUFOR	blue forces
COA	course of action
CSG	carrier strike group
DDG	guided missile destroyer
DESRON	destroyer squadron
DOD	Department of Defense
DON	Department of the Navy
DOS	Department of State
DQN	deep Q-network
ESG	expeditionary strike group
FON	freedom of navigation
FONOPs	freedom of navigation operations
LOAC	law of armed conflict
ML	machine learning
NPP	Navy Planning Process
OPFOR	opposing force
POTUS	President of the United States
PPO	proximal policy optimization
RL	reinforcement learning
RTS	real-time strategy game
SME	subject matter expert
TPU	tensor processing unit
TTP	tactics, techniques, and procedures
UNCLOS	United Nations Convention on the Law of the Sea
USN	United States Navy

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I am grateful to Dr. Chris Darken for his teachings, time, and advice during my time at NPS. I am so thankful for all the technical and professional assistance he provided week after week to help complete this thesis and forge my limited understanding of AI. Thank you to Professor Jonathan Alt, CDR Ken Maroon, Mr. Will Leonard, Mrs. Christina Rinaudo, and Mrs. Elizabeth (Betsy) Wallace for all your guidance, knowledge, and resources. Additionally, I would like to thank my family. My wife, Kathryn, and my daughters, Margot and Zaida, kept me in good spirits and with a fresh outlook which equipped me to tackle each day. Finally, thank you to my MOVES cohort for standing shoulder to shoulder and enriching my life as an officer, a Sailor, and a man.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. PLANNING FREEDOM OF NAVIGATION OPERATIONS

Since 1979, the President of the United States (POTUS) has annually directed the United States government to challenge excessive maritime claims made by a broad range of littoral states. The U.S. Oceans Policy of 1983 firmly states that the United States "will exercise and assert its rights, freedoms, and uses of the sea on a worldwide basis in a manner that is consistent with the balance of interests" [1].

Annually the Department of Defense (DOD) releases an unclassified freedom of navigation report to Congress. This report delineates the two prongs of the FON program: the Department of State (DOS) diplomatically protests claims inconsistent with international law, and the DOD conducts operational challenges against excessive maritime claims. The service primarily responsible for DOD operations against excessive maritime claims is often the United States Navy (USN), and a platform that is often called to accomplish these operations is a guided missile destroyer (DDG). These DDGs, usually independent steamers, are conducting operations in support of a strike group commander or a geographic combatant commander at the operational and tactical level when they receive a superior's directive via message traffic to plan and execute a Freedom of Navigation Operation (FONOP).

The Navy's publication on Naval operational planning [2] asserts that planning is fundamental to leadership and expected at every echelon of command. While it may be debated whether FONOPs are developed at the DDG, at the destroyer squadron (DESRON), or at the carrier or expeditionary strike group staff (CSG/ESG), the expectation is clear that this capability must be able to be accomplished even at the echelon of an individual DDG. Within Naval operational planning, the Navy planning process (NPP) [2] directs commanders to rigorously develop their options for the mission, more commonly referred to as courses of action (COAs) to include analyzing their COAs against possible enemy COAs (ECOAs) to analyze and determine the COA's feasibility and

acceptability. These planning steps are visually organized in Figure 1 to include this COA analysis which is often referred to as wargaming.

Figure 1. Navy Planning Process (NPP). Source: [2].

Per the NPP [2], wargaming can be either manual or computer assisted. All commanders can use computer-based wargames, thereby affording them every emerging analytical tool at the Navy's disposal, to include artificial intelligence (AI).

B. PROBLEM STATEMENT

FONOPs are executed inside the maritime claims of one nation that are in dispute by at least one other nation and viewed as excessive by the United States. In the maritime domain, the USN conducts FONOPs by sailing in one of three areas: international waters, a claimed exclusive economic zone (EEZ), or claimed territorial waters, see Figure 2. For this thesis, all explored scenarios represent FONOPs the United States conduct within 12 nm of land.

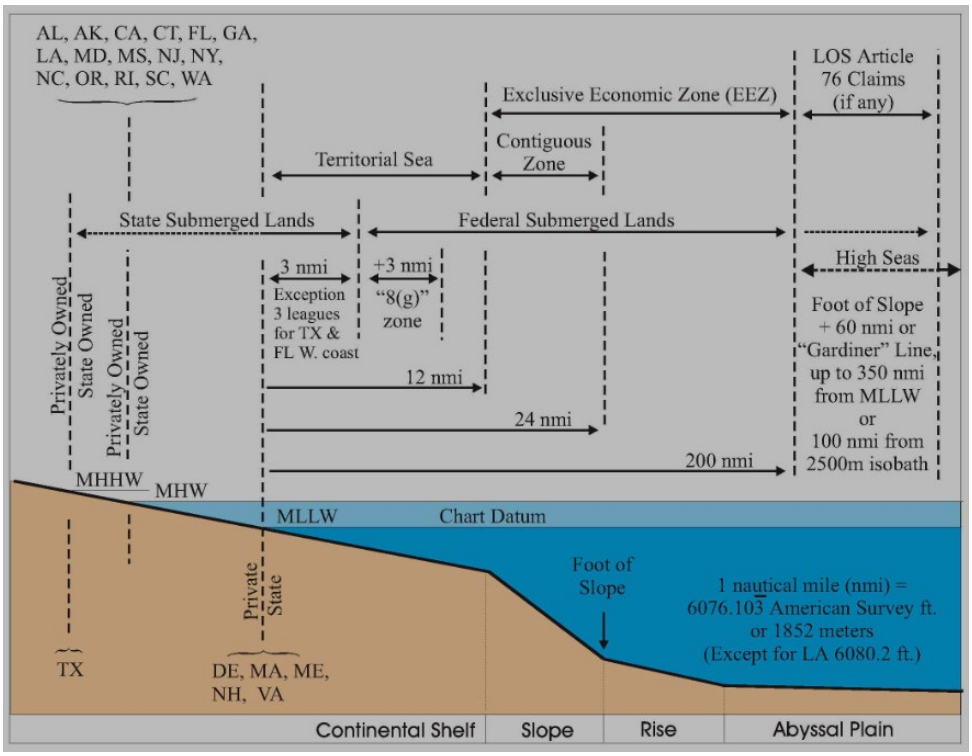


Figure 2. Maritime zones. Source: [3].

I have concerns about the challenges of a superior clearly communicating their intent of executing a strategic-level mission whose effects should be informational or diplomatic to a subordinate commander who is operating at the operational or tactical level and sees the mission's effects as maneuver and communication (both non-verbal and potentially verbal). These concerns aside, FONOPs are a nuanced problem space that cannot be hastily answered as its second and third order effects quickly exit the immediate battlespace and enter the domain of international politics and public opinion.

NPP course of action (COA) analysis [2] requires time and subject matter experts (SMEs) to function properly. Current independent steamers and their planned future replacements are manned with a declining number of senior Sailors with relevant operational experience in the region, and they are deploying more frequently on increasingly contested seas. These DDGs can soon find themselves lacking time for developing sound COAs or lacking even one or two SMEs to lead COA analysis and COA

wargaming or lacking both time and SMEs. AI can aid military decision-makers in COA analysis and wargaming as applied to the FONOPs scenario described above.

Representing an opposing force (OPFOR) and its behaviors in a wargame often takes one of two paths. Either a human player controls the OPFOR, or a predetermined script of choices, attitudes, and objectives that determine the overall behavior controls the OPFOR. Both methods run into problems as old as warfare itself, illuminated by Dr. Joyce Sampson, namely the problems of mirror imaging and scriptwriting [4]. The first is mirror imaging bias where the human player acts only as they would through the lens of their own experience and does not fully take on the objectives and attitudes of the adversary and so performs unrealistic behaviors. The second is sometimes referred to as scriptwriting. When an adversary is written into a static deterministic script, it can be challenging to implement enough stochastic elements to prevent the adversary's behaviors from being predictable and eventually unrealistic or ineffective, because the script cannot assess or reassess its opponent's behaviors.

Using AI, we can address both issues and better resolve each. The problem of the mirror imaging bias can be approached by well informed, researched, and written rules so that adversaries behave in accordance with their governing doctrine, laws, and known tactics, techniques, and procedures (TTPs). Coding this into the rules and structure of the war game and the neural network studying it, can result in intelligent agents that not only follow the laws of physics as presented in the game but also the OPFOR's positions, including their interpretation or application of the United Nations Convention on the Law of the Sea (UNCLOS) and the law of armed conflict (LOAC). Scriptwriting can next be mitigated by the nature of neural networks and their ability to make observations of the battlespace to inform their next steps, to better assess and then reassess what occurred not only from one engagement to the next, but also from one time step to the next within an engagement.

The next challenge lies in time. All military planners know that the true adversary is the clock. Analysis, planning, and wargaming could all be done superbly if only we had enough time. The turnaround on operational planning to include FONOPs planning may be as little as 24 hours. To best represent this limitation, I include wall clock time otherwise

known as the real-world time it takes to train an AI agent as a vital factor with the goal to train a "good enough" AI agent as quickly as possible with a personal laptop computer. In this thesis, I have elected to use a deep Q-network (DQN) because in other research and in comparisons done in laboratory experiments, I participated in here at NPS, DQN consistently achieves a "good enough" solution must faster and much more consistently than other architectures.

To best scope this problem, this thesis explores and answers the following:

1. What are the characteristics of surface maneuver needed to be translated to an AI admiral for FONOPs?
2. How does heuristic decision making for patrolling territorial waters impact red AI admiral's ability to learn interdiction behavior?
3. How quickly can a stand-alone, FONOPs COA analysis neural net learn a realistic, near optimal path and course to interdict as red?

C. SCOPE AND METHODOLOGY

For this thesis, I have elected to build on the ATLATL architecture pioneered by Dr. Chris Darken and used by several researchers at NPS. ATLATL is a turn-based, military, constructive simulation designed for AI research. The hex-based mapping and turn-based structure will help level the capabilities of a human and an AI player by limiting the decision space and therefore the cognitive load for each (see Figure 3).

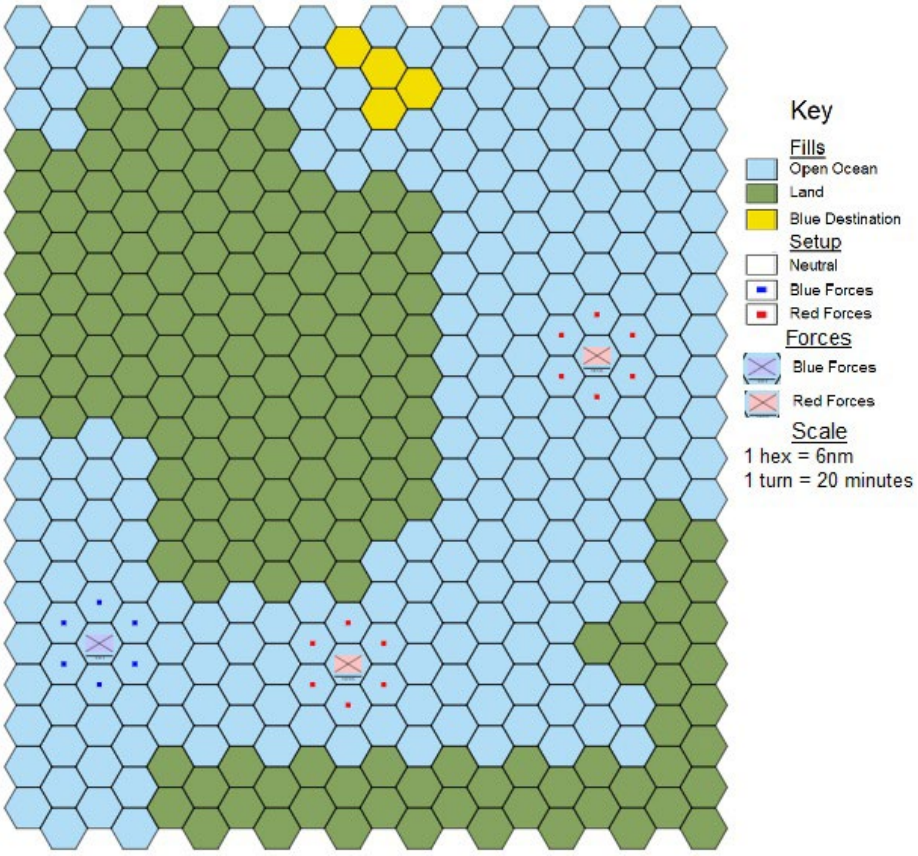


Figure 3. ATLATL FONOPs scenario

In this thesis I train an AI agent referred to as an AI admiral to play as the red or OPFOR commander through reinforcement learning (RL) techniques, using a DQN neural network, without example play or data libraries, and reach near-optimal scores for the AI admiral in under 24 hours of training. For both the player and the AI admiral, the space is divided into hexes which are 6 nm across, turns or phases are sequential between blue acting and red reacting with each pair of turns or phases representing 20 minutes in the real-world scenario. While red and blue are not allowed to fire upon each other as the war game is to analyze FONOPs COAs, red is given the ability to negatively threaten blue which is still shown in the wargame as an attrition bar. This represents the ability for the OPFOR to harass the blue ship and engage in unsafe or unprofessional actions at sea which blue would notice as more hostile. This is worth noting in applications of the simulation, because this represents a flashpoint where the blue ship may seek to de-escalate the

situation or act in its self-defense. Even if the ship fails in the simulation, seeing this flashpoint can assist the actual ship in real life operations.

D. BENEFIT AND STRUCTURE OF THESIS

Several research papers have been published studying the application of reinforcement learning in real-time strategy games (RTS), and a growing number have been written regarding the application of the same techniques to the domain of military constructive simulations. Of the later, few if any, have been written with the intent to place the AI capability in the hands of a forward deployed tactical unit to assist in planning, analysis, strategy, or training. The greatest benefit of this thesis is in its assessment that quality tactical performance can be achieved in a short amount of time without a priori knowledge in the form of data libraries for the AI admiral.

A forward deployed AI solution is feasible, and it also saves both time onboard and time for reach back personnel. Forward deployed AI systems enhances the NPP for the entire Navy without adding additional strain and without creating analysis paralysis. The meaningful insights and localized flash points revealed through hundreds of thousands of constructive operations and experienced by the crew in live simulation will lead to real world, combat ready naval forces capable of deterring aggression and maintaining freedom of the seas.

That said, due to my limited resources, to include data libraries, time, and existing code structure, this thesis will serve to benefit basic research within the field of military constructive simulations. While immediate implementations and applications may not be achieved during this thesis, it is my firm belief that follow on work will be able to deploy lessons learned here to future researchers, this framework can be deployed to the fleet, and this research will result in a future savings of manpower resources and a transfer of human capital to digital knowledge management for the DOD.

The first chapter of this thesis is an introduction to the scope and nature of the topic. The second chapter provides a general survey of the modern research in military constructive simulation, applications of AI in this domain, as well as key concepts such as neural networks, machine learning (ML), and reinforcement learning (RL). Chapter III will

elaborate on the hardware used as well as the software framework of ATLATL and the AI RL tools. In Chapter IV, two scenarios, OPFOR with one ship and OPFOR with two ships, and their individual results will be explored. In the last chapter, a general conclusion will be provided based on the previous chapters.

II. BACKGROUND

This chapter explains key concepts for this thesis such as reinforcement learning (RL) as well as explores research in the domain of military constructive simulations and private and academic domains of AI applications in RTS and similar wargames.

A. WARGAMES

Military wargames have historically found their place among the larger population outside of professional soldiers. These games function both as a medium to test tactics and strategy as well as a recreational pastime. While Go or Wei Qi, Chaturanga, chess, Kriegsspiel, and the like may not be seen as much in the public, turn-based or real-time strategy (RTS) video games are emerging in the sphere of war games, not only as a pastime, but as a profession. Within eSports, the most difficult and long running RTS is StarCraft [5]. At the start of a match, StarCraft players choose one of three races—Terran, Protoss, or Zerg—each with distinct, strategic advantages and disadvantages.

We can draw many parallels between what a player must accomplish in an RTS, and skills and behaviors a commander requires for operations. No matter which race is chosen, in order to play StarCraft, every player must order their units and structures, maneuver the map, find and track their opponent through the fog of war, decide what to prioritize in training with their defense industrial base, research technologies through their race's tech tree, and ultimately conduct skirmishes and battles until they annihilate the OPFOR or force their opponent to resign [6]. These skills and behaviors align to five of the six functions and duties of all commanders stated in JP 5-01 [2]: personnel, intelligence, operations, logistics, and plans and policy.

Wargame limitations that players operate under include whether information is open, game conditions, and rate of taking actions. In wargames, information could be open to all players (e.g., chess), partially open (e.g., Stratego or Lu Zhan Qi), or closed (e.g., StarCraft II). Knowledge of the forces that you and your opponent have and their location in space are often the information that can be closed to one another and create a fog of war. In chess, players can always see the entire board (the space) and all the pieces (forces). In

Stratego or Lu Zhan Qi, each player can see all the forces located in the space, can see the types of their own forces, but the types of their opponent's force are closed to them. In StarCraft II at the match start, each player can only see their starting location and force. The rest of the space and forces are completely closed to each player, and they must explore the space to discover this information. Game conditions and rate of actions taken are either limited by real time and the pressure of opposing players acting simultaneously, by timed matches or timed turns (e.g., chess clocks), or are simply limited by turn-based actions with no source of pressure on players. For an AI agent to participate in lieu of a human player in wargames, it must be constrained by the same limitations as a human player.

1. Real-time strategy

RTS wargames are fertile testing grounds for AI research to develop robust behaviors for RTS and other strategic and tactical applications. A large corporate contributor to the growing field of AI is Alphabet Inc. and their subsidiary DeepMind Technologies [7]. After applying their technical prowess and AI methodologies to Atari video games and even the classic strategy game Go, in 2016 DeepMind announced a collaboration with Blizzard, the developer of StarCraft, to apply its AI techniques to develop an artificial agent capable of competing with and besting the top professional StarCraft players in the world [5], [7].

DeepMind's Vinyals et al. [5] used StarCraft II as a "proving ground" for AI systems and techniques because "many real-world applications require artificial agents to compete and coordinate with other agents in complex environments" with imperfect information. DeepMind chose the domain of StarCraft after other researchers had limited success and viewed it as a modern-day challenge for AI research, "owing to its iconic and enduring status among the most difficult professional eSports and its relevance to the real world in terms of its raw complexity and multi-agent challenges" [5]. Vinyals et al. used online StarCraft II matches against human players to test their agent, AlphaStar, who performed incredibly, eventually earning the rating of Grandmaster for all three races and ranking in the top 0.2% of players. "The success of AlphaStar in StarCraft II suggests that

general-purpose ML algorithms may have a substantial effect on complex real-world problems and domains" [5].

AlphaStar was developed with the wargame limitations all human players experience in StarCraft II to fully challenge the AI agent and ensure fair play with human opponents. DeepMind consulted with professional StarCraft II players and Blizzard employees to develop AlphaStar's limitations inside the RTS [5]. These limitations included match conditions, interface for the AI system, camera view, action rate limits, and delays [5]. These limitations ensured that the way a human player views the map in StarCraft II and the way AlphaStar views it are analogous. The rate at which a human could click a mouse to issue a command and the rate AlphaStar can act are similar. With these limitations, AlphaStar's evaluation as Grandmaster with all three races is attributed to actual in-game performance and not the exploitation of a computer program able to perform outside of the realm of human capabilities.

While DeepMind's success is undeniable, it is important to recognize its resources and the limited domain of that success. First, the forty-two computer and data scientists who worked on AlphaStar over three years committed DeepMind to the problem and gave their mission continuity and resilience [7]. Second, the 971,000 replays of StarCraft II games from Blizzard that DeepMind used as a data set to train AlphaStar during supervised learning advanced AlphaStar's entry point to StarCraft II, allowing DeepMind's efforts to focus more on high-level strategy than low-level tactics [7]. Third, the hardware applied to the problem, Google's proprietary tensor processing units (TPUs), increased efficiency, allowing AlphaStar to train for years of game time in one human week [6]. According to DeepMind:

> For every training agent in the league, we run 16,000 concurrent StarCraft II matches and sixteen actor tasks (each using a TPU v3 device with eight TPU cores) to perform inference. The game instances progress asynchronously on preemptible CPUs (equivalent to 150 processors with twenty-eight physical cores each), but requests for agent steps are batched together dynamically to make efficient use of the TPU. Using TPUs for batched inference provides large efficiency gains over previous work. [5]

This hardware setup enables a single DeepMind learner to process 50,000 agent steps per second. Finally, the extensive number of days reported that DeepMind trained each agent contributed to the league of agents that birthed AlphaStar. Per my analysis of DeepMind's reports [5], [7], they trained over 80 "main agents," which were supported by hundreds of separately identifiable agents for over 40 training days, as seen in Figure 4.

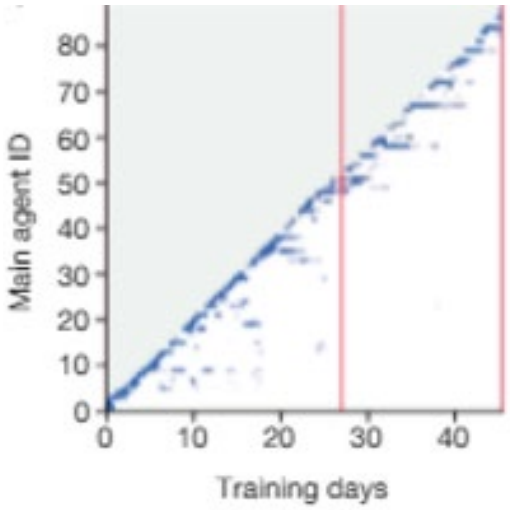


Figure 4. DeepMind's Nash distribution of AlphaStar agents. Source: [5].

I cannot foresee this large of a capability existing at the disposal of a DDG without reach-back or shore-based support. I can, however, see the implementation of the methodology and techniques employed by DeepMind applied to the problem sets at scale in hardware, software, workforce, and time to come to a "good enough" solution that will allow a DDG to develop and analyze sound COAs and to make more informed decisions.

2. Military constructive simulations

While DOD does not currently employ RTS, its war fighters and their commanders are trained via simulations, and these same simulations can be used for planning and to enhance and discover TTPs. Military simulations are often categorized as live, virtual, or constructive. In its body of knowledge (BOK), the DOD Modeling and Simulation Coordination Office (MSCO) describes the differences between the three as follows: "live simulation involves real people operating real systems, virtual simulations involve real

people operating simulated systems, and constructive simulations involve simulated people operating simulated systems" [8]. In other words, a military constructive simulation is often a computer program where a military planner or wargamer will provide guidance, direction, orders, or a plan and then the simulation will construct how that might work out based on the rules of the simulation or wargame itself.

The audience of the simulation determines whether to represent a DDG in a wargame as the individual Sailors or as a single ship. Military constructive simulations and wargames are usually divided based on the echelon of the audience and their interaction with their forces. According to Page and Smith [9], if the base unit in a wargame is a collection of doctrinally identifiable military assets, then the simulation is referred to as an aggregate level simulation. If the base unit is an individual soldier, tank, or singular object, it is an entity level simulation. The unique problem for the USN is that a DDG or other ship is both an aggregate representing hundreds of people and an entity, a single identifiable military asset, and the audience can view themselves as a Sailor or as the ship.

Military constructive simulations can yield meaningful training and planning outcomes as long as their capabilities and limitations are balanced in consideration with the desired outcome for the service member. Constructive simulations share three key attributes: lower fidelity or discretized environments, mathematical models or algorithms, and limited times for inputs by a player or wargamer. First the environments can be lower fidelity because the base unit is a collection. Like a board game, the space or environment the wargame is taking place in can be divided into discrete sectors, squares, hexes, or any other subdivision of the whole. This discretization reduces the processing power needed in the computer simulation and shifts the cognitive load for the player. The player does not need to determine where everyone should move to in a continuous space, but instead orders the aggregate unit to a space in the environment. In their research, Cannon and Goericke [10] found that in "constructive simulations, mathematical models and algorithms represent units, weapons, equipment, attrition rates, and the combat behaviors portrayed by each force operating in the designated environment". Finally constructive simulations may be open-form and allow wargamers to provide inputs and orders throughout the

scenario, or they may be closed-form and limit when inputs and orders can be provided to each player's forces [10].

An unavoidable challenge for all wargames is finding a quality opponent, especially when the end goal is to train against an adversary whose TTPs and overall strategy may be foreign to the wargamer's. Training AI agents to act as the OPFOR in military constructive simulations can provide a library of artificial admirals and generals informed and shaped by individual historical figures or real-world data and trends from training exercises and real-world operations.

B. ARTIFICIAL INTELLIGENCE CONCEPTS

Arriving at a quality OPFOR commander requires considerations in how the AI admiral will be structured, educated, and encouraged, as well as the wargame and computer hardware environment in which it will live. AI agents are developed through hardware and software in conjunctions with different processes or methodologies. Key AI concepts include agent architecture, ML, and neural networks.

1. Agent architecture

Before we train an AI admiral, it is crucial to consider how it exists in the shared environment of the wargame. Computer agents are a behavior or set of behaviors written in computer code that take an observation of the environment as an input and select an action as an output that changes its environment. As shown in Figure 5, this cycle exists with four key components: the environment, the agent, the agent's raw input stream, and the agent's output stream.

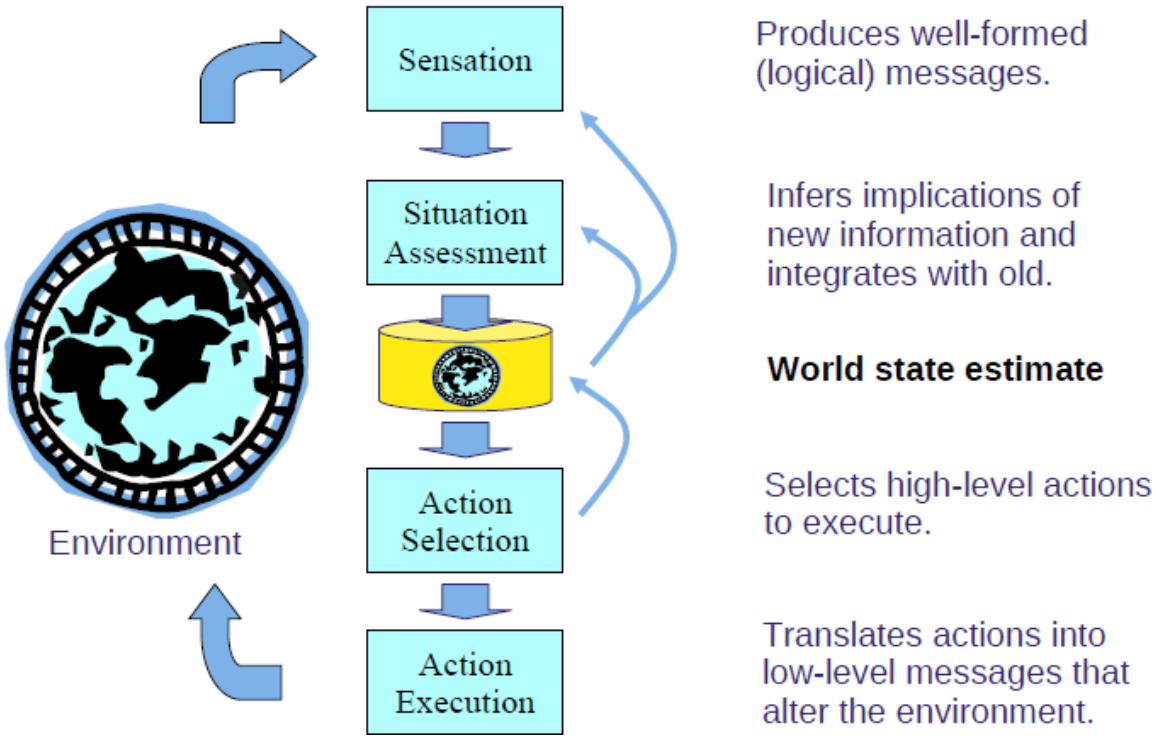


Figure 5. Dr. Chris Darken's generic agent architecture. Source: [11].

To train a quality AI admiral, key considerations must be given to what inputs the agent has at its disposal, and what inputs a similar human wargamer would enjoy. Further, any differences must be attributable to their natures. Agents can sense their environment in limited ways [11]. This could be the agent inside a computer program taking a survey of what it is allowed to see limited by the application itself. In a wargame this would include the agent's forces, the location and environment of those forces, the status of those forces, any hostile forces, the location and environment of hostile forces, the status of those forces, and any information on time. Time information an agent might observe includes the original start time, the current time, any limits such as a time for termination of the wargame or an objective in the wargame, and the time to accomplish an action. All the above are sensations expected of commanders or wargamers in charge of their forces as well.

Designing a quality AI admiral requires us to foster in it an open-minded nature that considers traditional as well as novel actions. Agent assessments can include the consideration of all possible actions, the agent's history in the specific decision, the agent's

history in the wargame, any scoring, and any weighting of sensations and inputs to form its situational awareness and then select its action [11]. Darken [11] refers to the simplest method for the agent to make an assessment as direct-action selection, which is equivalent to Sampson's [4] scriptwriting. In direct action selection, the agent does not consider each possible action, but instead selects an action based on its current circumstances. These are often written into code as "else and if" decision trees with some weighting of inputs to model a desired behavior. In a FONOPs scenario, a blue ship could be coded for direct action selection to move to within 12nm of shore if OPFOR ships are greater than a preselected distance, or if the BLUFOR outnumber the OPFOR in a preselected radius from the blue ship.

A more robust form of agent assessment is found in scoring algorithms and ML algorithms where an AI admiral can be constructed to consider all possible actions. Instead of beginning with the agent sensations and observations, these assessments sometimes begin with all possible actions and then begin to score and rank them based on both the observations and any and all weights and scoring functions that have been given to the agent [11]. Scoring and ML algorithms may result in more nuanced behaviors, such as a blue ship in a FONOPs scenario observing the environment and all other forces and determines the highest score results from taking no action to command a more favorable position on a future turn.

To develop an AI admiral exhibiting nuanced and even human-like behavior, an even more advanced assessment can be produced through ML. Here actions are rated based on past results or scores in addition to everything that is currently observed. An agent estimates what might happen in the current state and potentially removes choices from the list of possible actions based on its past observations or any hard coded criteria. The action is selected and executed, which changes the environment, and a new sensation or observation of the environment begins.

2. Machine learning

If our desire is to educate our AI admiral and not simply give it a list of orders to execute, we must employ ML. According to Dr. Thomas Mitchell in *Machine Learning*

[12], “ML is a subdivision of AI that uses computer algorithms to train an agent and uses its past experiences to improve its performance” [10]. ML is typically subdivided into three categories: supervised learning, unsupervised learning, and reinforcement learning (RL) [13]. These ML algorithms act as educational techniques that will teach our AI admiral how to think and not simply what to think.

For a clearly definable operation with a well-developed data set, our AI admiral benefits most from supervised learning. Per Russell and Norvig in *Artificial Intelligence* [13], supervised learning is when an agent receives “a data set where inputs are always associated with an output that is known to be correct.” Like a human being provided with the questions and a list of solutions for each question, the agent can use the information to discover patterns and associations and develop a rule set or function to make future determinations. An example of this may be an agent given a set of different geometric shapes and their names. The agent could then begin to create a rule set for what is a circle, square or triangle and then be able to recognize a shape outside of its originally provided data set and apply its rule set. Supervised learning applied to an AI admiral could help it develop a common operational language with the warfighters it is designed to support.

In contrast, if the operation is not well defined, but a large data set exists and the Navy needs assistance in determining patterns and relationships, our AI admiral benefits from unsupervised learning. Unsupervised learning provides the agent with a data set without an answer key [13]. Although no known correct answers are provided, the agent still processes the data inputs, attempting to associate them with any solution, pattern, or function. An example of this could be an agent given a set of different geometric shapes with no names or answers. The agent could then begin to create a rule set or pattern for grouping the shapes it received as inputs. Although the agent would have no knowledge that the round shapes were circles and the three-sided shapes were triangles, it could still associate them and even extend these rules and associations to shapes it did not receive in its initial data set. Unsupervised learning can teach an AI admiral to develop a coup d’oeil in an environment and its terrain, assisting the Navy with indications and warnings or determining a near-optimal location for an operating area.

If our AI admiral will oversee analogous operations and needs to learn from its past performances, especially if there is no existing data set and it is going in blind, then the agent will require reinforcement learning (RL). OpenAI [14] describes RL as "training an agent to learn specific behaviors through trial and error". These agent interactions in their environment (or state) are either returned with rewards, punishments [13], or nothing [10]. Like Pavlov's dog, the agent trained by RL learns through exploration until it receives a reward. After receiving this reward, it tries to recreate the action or sequence of actions that led to it. If the agent receives rewards of differing amounts, it will seek to obtain the maximum amount of reward. Timing of the reward and number of rewards are key for training an agent the desired behavior, even more so when a solution is unknown. It is my belief that a nuanced operation such as FONOPs requires RL for an AI admiral to learn how to perform as the OPFOR and assist with developing our most likely red COAs and our most dangerous. For these reasons, this thesis employs RL to train the AI admiral.

Educating our AI admiral via RL or ML will require a neural network to develop and store our agent's developing policy for the operation or problem it is tasked with. In his thesis, Johnathan Boron [15] explained the underlying RL concepts of a policy function clearly and succinctly:

> As it pertains to reinforcement learning and other methods of machine learning, there exists a model of behavior, often termed a policy and labeled as $\pi\theta$. The overarching goal is to find a good parameter vector, that is θ, which causes the policy to produce the desired actions given a particular state. This parameter vector θ may be manifested in different ways. However, in systems where a deep neural network represents the policy, θ specifically consists of the weights in that neural network. In order to optimize θ, reward functions are implemented such that the sum of rewards are maximized over all time steps. [15]

As seen in Figure 6 as well as our Pavlov's dog example, the agent is defining its model of behavior, or policy, from the state of the world (s) that it observes before it takes an action (a). This sensing or observing may be complete or only partial. An environment that is only partially observable is due to limitations from the wargame itself (closed or partially open information wargames). When the agent takes an action, as in Figure 6, this in turn changes the environment, which in RL leads to a function returning a reward, punishment, or

nothing to the agent. As an agent increases its history of experiences in an environment, its policy will become more refined, and our AI admiral will be more resolved in how best to act as OPFOR in our FONOPs scenario.

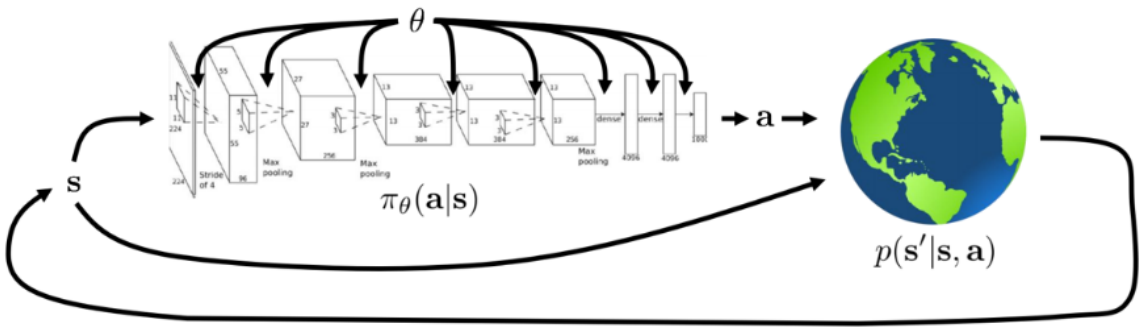


Figure 6. Illustration of reinforcement learning. Source: [16].

3. Neural networks

The mind of our AI admiral that will enable it to make decisions as the OPFOR commander in a FONOPs scenario wargame is the neural network. Neural networks are defined by the policy or algorithm that they implement and the number and manner of observations they make on their environment. A neural network is a parameterized function, which can be written as f(x, Θ), where x is a vector of inputs and Θ a vector of parameters [17]. This Θ is the same as the Θ illustrated in Figure 6. The complex, parameterized function that forms the neural network is made up of simpler functions that follow a sequence where Θ uniquely resides between the simpler functions and may vary locally. Every neural network has an objective function, and the parameters of the neural net are set to yield that function's maximum value [17]. Two of the leading policies that define and determine the objective function are proximal policy optimization (PPO) and deep Q-network (DQN). Understanding the capabilities and limitations of each will help in determining which will benefit the AI admiral and its task of learning to command the OPFOR in our FONOPs scenario.

a. Proximal policy optimization (PPO)

If our AI admiral has more than 24 hours to learn about its environment and its intended operation, PPO is likely the policy of choice to implement in the RL neural network. In its current state, PPO can arrive at the optimal solution and benefits from longer training times with greater amounts of learning steps, but like its policy gradient ancestors, it can overstep in its learning and oscillate around the optimal solution without ever finding it [21]. Introduced in 2017 by Schulman et al. [22], PPO algorithms are a family of policy gradient methods that are simpler to implement and outperform older policy gradient algorithms such as trust region policy optimization (TRPO). Such policy gradient algorithms repeatedly take actions until they arrive at a good estimate of the RL reward, which forms an initial policy, and then they gradually take steps to improve this policy [17]. Essentially our AI admiral learns over time that X may be the average reward for certain actions and so takes slight variations on those actions in those states to see if they might earn larger rewards.

b. Deep Q-network (DQN)

For action spaces that can be finitely summarized into a table, or if our AI admiral is pressed for time, DQN is likely the winning policy for its RL neural network. Sun et al. state "The Q-learning method can provide basic ideas for the construction of intelligent wargames; the traditional table form stores the state and corresponding Q-value to find the best executive action" [20]. Although Sun et al. assert that in cases where the state is very complex that the DQN algorithm will struggle with limited computer memory, it is my opinion that the FONOPs scenario as designed is within the acceptable range of state complexity for DQN.

In this application of DQN, the AI admiral will learn much like a human player with little guidance. The agent's learning materials are limited to the video input of the wargame map, the reward it receives, signals that start and stop wargame matches, and the set of legal actions it is allowed to take. DeepMind [23] used DQN in a similar way to train AI agents to play Atari games with these same limited inputs and found DQN outperformed all previous approaches on six games and surpassed a human expert on three. As DeepMind

reported mastering "difficult control policies for Atari" games using this limited input for their agent, and Sun et al.'s success with both DQN and a modified DQN in training an AI agent to master a hexagonal wargame, it appears to be the policy of choice for this thesis.

C. STATE OF HARDWARE

Just as those that study humankind must evaluate both the mind and the brain, here we review the current state of hardware to determine the conditions of our AI admiral's storage and processing as well as any capabilities or limitations that may arise with a non-networked solution or at least one that cannot rely on a cloud solution at sea.

1. Memory

Neural networks depend on memory for agent storage, copies of the agent, replay storage, data storage, as well as short term buffers to be able to view the world state and consider its own lessons learned or policy to assess the situation and select an action, as illustrated in Figure 5. For this thesis we used 20GB out of 500GB of flash storage, as well as had access to 16GB of RAM. Throughout the research for this thesis, I discovered that several researchers mention the need for memory or data storage and challenges that limited amounts of memory impose, but few list the memory requirements or availability they enjoyed.

2. Processing units

Neural networks and RL speeds or rates impact time, and in turn are impacted by the hardware they use for processing. Traditional processing resides at the CPU, with improvements in CPU performance increasing performance of AI systems, but some researchers, including DeepMind, have begun to utilize other sources to include GPUs and TPUs.

a. Application of graphical processing units (GPUs)

As emphasized by Greg Allen, Chief of Strategy and Communications at the DOD Joint AI Center (JAIC) [24], ML and RL algorithms require a lot of computing power, and around 2010 it became possible to effectively run these algorithms on GPUs. In 2017, Larry

Hardesty [18] at MIT explained the impact GPUs with their "thousands of relatively simple processing cores" had on network size and how larger networks led to greater performance:

> Modern GPUs enabled the one-layer networks of the 1960s and the two- to three-layer networks of the 1980s to blossom into the 10-, 15-, even 50-layer networks of today. That's what the "deep" in "deep learning" refers to—the depth of the network's layers. And currently, deep learning is responsible for the best-performing systems in almost every area of artificial-intelligence research. [18]

While GPUs are more than capable of running or assisting in running the algorithms, I chose to rely on a CPU for this thesis. This decision was influenced by the performance of the Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz, the CPU requirements for the AI system fluctuated between 15 and 73% of the CPU, and the computer occasionally trained agents while performing other tasks.

b. Emerging tensor processing units (TPUs)

The most capable processing units we could employ to train our AI admiral, Google's proprietary TPUs, are out of reach for our DDG, as these units are only available in the cloud. Used by Google internally by 2018 and brought to market in 2020, TPUs, shown in Figure 7, are ML accelerators that enable training of deep neural networks to deliver optimal results at incredible speeds [25].

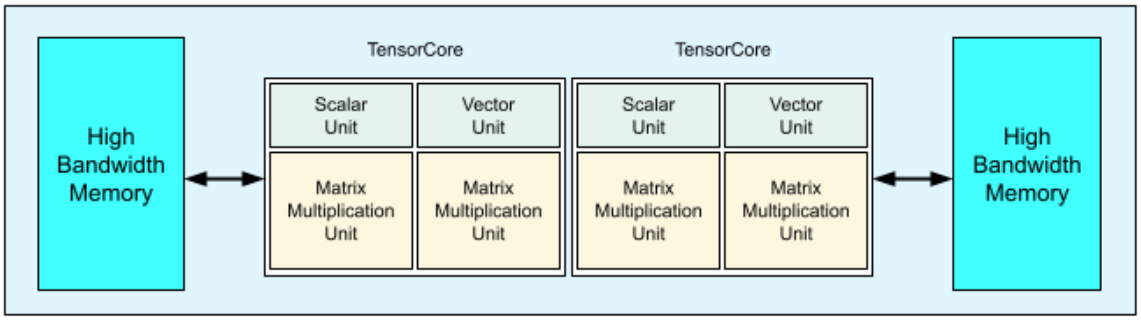
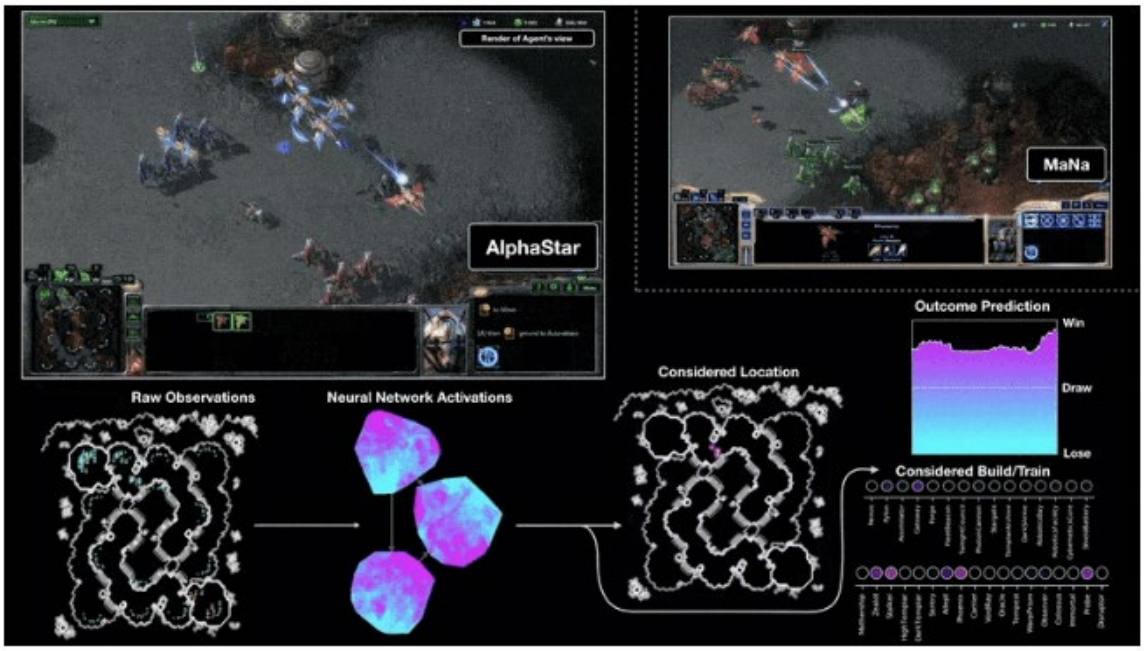


Figure 7. Google TPU v3 used for AlphaStar. Source: [25].

TPUs were instrumental in DeepMind's AlphaStar achieving Grand Master status in StarCraft II and essentially playing 200 years of StarCraft II in 3 actual years [5]-[7].

Google's TPU v3, used to develop AlphaStar benefitted from the TensorCores as well as the high bandwidth memory, as seen in Figure 8. There currently is one other TPU manufacturer, Sophon, a subsidiary of Bitmain Technologies Ltd., headquartered in Beijing, China, but little information could be found regarding their products.



Top left AlphaStar's game GUI, top right MaNa's game GUI, not accessible to AlphaStar, and bottom AlphaStar's sensations, world state estimations and action considerations.

Figure 8. AlphaStar playing StarCraft II. Source: [7].

THIS PAGE INTENTIONALLY LEFT BLANK

III. FRAMEWORK

In this chapter, I apply AI concepts discussed in Chapter II to the military constructive simulation, ATLATL, with the addition of ships in a FONOPs scenario. ATLATL is an AI training environment developed within the MOVES Institute at the Naval Postgraduate School (NPS) pioneered by Dr. Chris Darken [10]. Both human and AI players can compete and train within the ATLATL wargame and in this chapter, I discuss how our AI admiral observes, trains, and learns its policy or strategy.

A. ATLATL WARGAME

ATLATL, named after the Paleolithic tool that allowed early man to throw a projectile farther, is a military constructive simulation designed to allow both human and AI players the ability to command operational forces in a turn-based wargame. Turns in ATLATL are called phases with BLUFOR or blue activating during phase 1, then OPFOR or red activating in phase 2, and continuing with this alternating pattern until the terminal phase is reached. ATLATL as a wargame can be played locally or online as the game code is written in JSON and JavaScript, which is accessible in a web browser. This online interface masks the underlying Python code base, hosted by one of the players, and made up of imports and server files the game requires, (see Figure 9). Python is incredibly popular with RL researchers, furthermore several open-source code bases, tools, and projects are offered in Python. Python as a core language in the code base allows these AI research resources to be utilized.

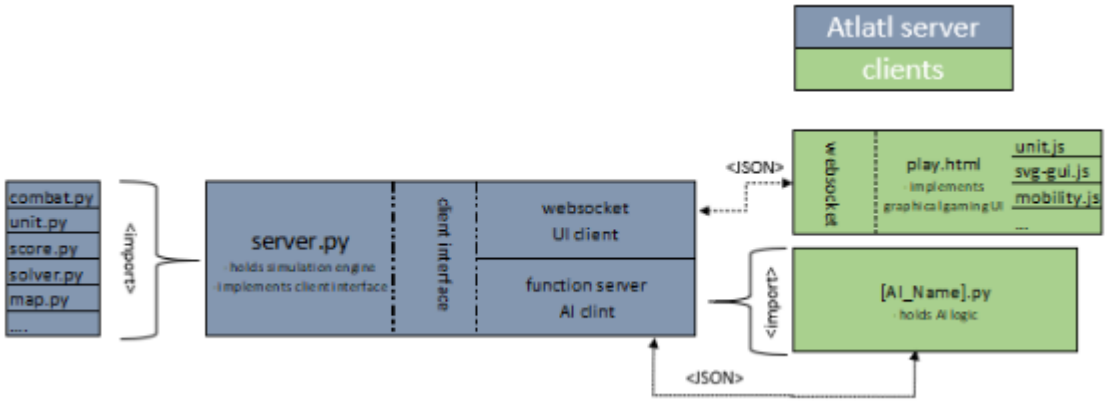


Figure 9. Human versus AI agent ATLATL architecture. Source: [10].

Following the success of other researchers, [10],[11],[15],[17], and [19], using Stable Baselines and its successor Stable Baselines3, I continued the implementation of Stable Baselines3's library which "currently implements twelve different RL algorithms, including" DQN and PPO [10]. These RL algorithms from Stable Baselines3 can only be used in what is known as a gym environment. Gym environments can be sampled or modelled from OpenAI's Gym library. I employ the same Gym environment in ATLATL that has been used by other AI researchers at NPS; the architecture and training setup for my AI admiral is in Figure 10.

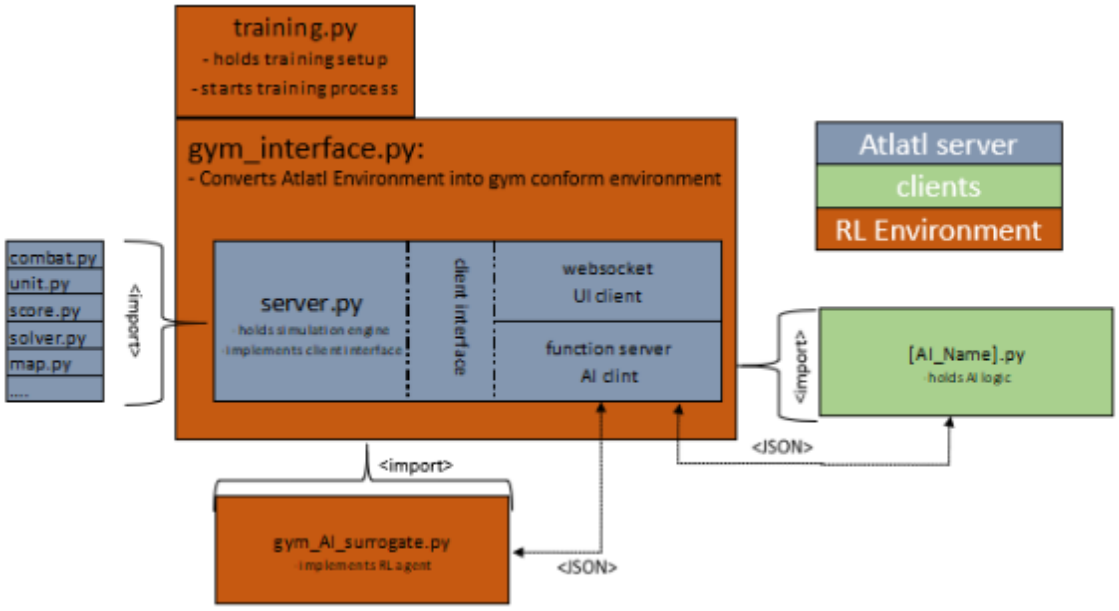


Figure 10. ATLATL gym environment for training. Source: [10].

The AI admiral is trained with the above architecture in the same manner as discussed in the explanation of ML and RL: it observes its environment, chooses an action, and is returned with an observation of its newly changed environment, as well as any reward for that action.

1. State representation

One challenge that was previously addressed by [10] and solved for ATLATL's AI gym environment is the representation of the state space, primarily the coordinate system to best represent the hexagonal-based simulation to the AI agent. Rectangular-based simulations use a certain coordinate system that when adapted to hexagonal simulations creates problems or disadvantages in processing and state space representation. To remedy this, a double coordinate system was used for ATLATL where all hexes adjacent to a central hex are two away as seen in Figure 11.

Figure 11. Hexagonal boards left without double coordinates and right with double coordinates. Source: [10].

In the hexagonal spaces that form the observation space in the gym interface for our AI admiral, I provide a vector of inputs, x, as discussed in Chapter II.B.3 neural network's f(x, Θ). These inputs include the locations, unit strength, terrain, and goals. The locations include that of the active friendly ship, the location of any other friendly ship, the location of any adversary ship(s), and the location of the goal hexes for the end of the FONOPs path. The AI admiral is also provided with the strength of each unit. Each hex also has a type of terrain associated with it that is provided to the AI admiral. I implement Cannon and Goericke's [10] representation of the inputs in a two-dimensional array, with x- and y-axes flipped, and locations of units indicated with a 1.0 (for a unit at 100% strength), 0.7 (for a unit at 70% strength) or a 0.0 (representing an empty space). This representation as worked out by [10] is shown in Figure 12.

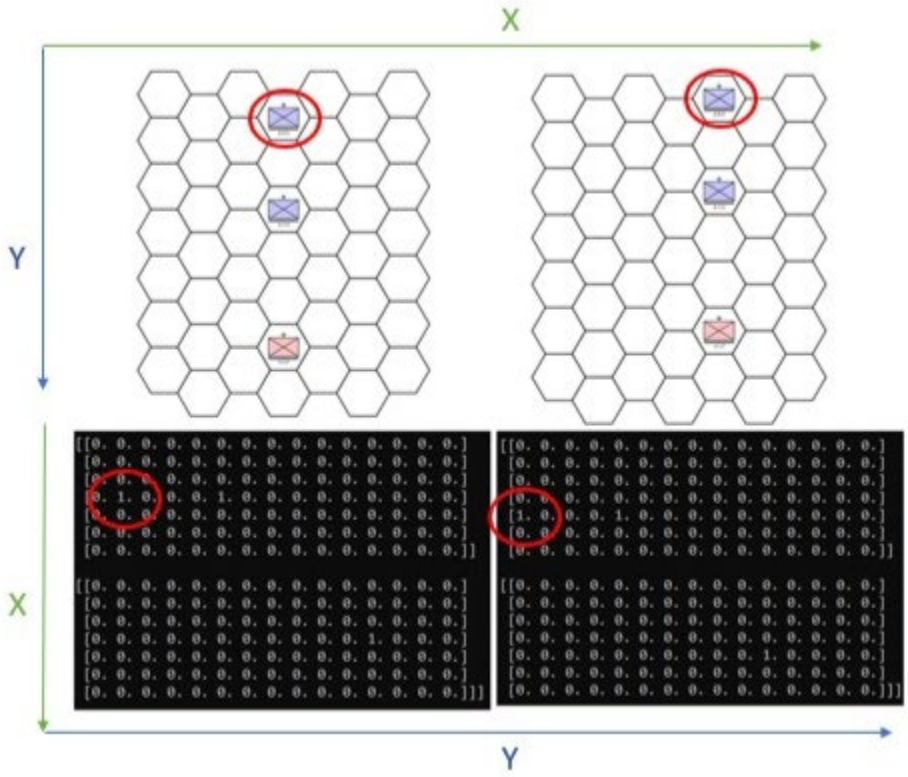


Figure 12. Inputs represented in space and in 2D vectors with x- and y-axes inverted. Source: [10].

Finally, the action space for a ship is represented where it can move one hex in any direction, except for the hex it moved out from, during the previous turn or phase. This restriction represents the following assumptions in the design of this wargame: a ship is moving at an average speed of 20 knots, time-steps represent 20 minutes, each hex of space represents 6 nm, and ships cannot ignore the effects of inertia at sea and so are not allowed to reverse. In other words, we are not allowing the ships to execute a 180-degree turn (see Figure 13).

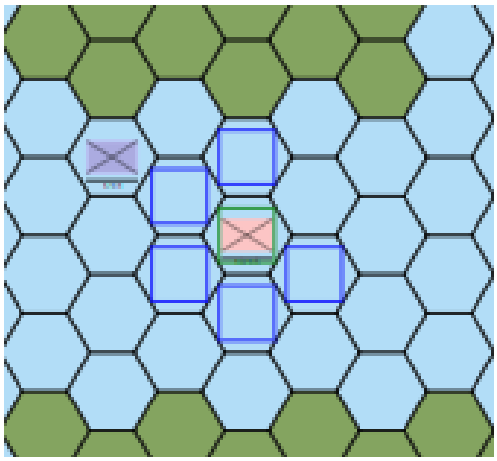


Figure 13. Action space for red (blue boxes mark the five hexes red can move into)

2. Rewards

The AI admiral earns rewards each turn or phase as it trains, and these rewards guide its behavior and hone its policy and future play or strategy. At the end of each turn or phase, the AI admiral receives a reward of 100 points if an OPFOR, red ship is adjacent to the BLUFOR ship. This immediate reward reflects OPFOR ships interdicting a BLUFOR ship navigating in their claimed territorial waters. At the end of a game, the shortest path the BLUFOR ship could have taken is calculated, this amount is compared to the actual distance the BLUFOR had to take to reach its destination. If the BLUFOR ship does not arrive at its destination or was forced to take any path other than the shortest, the AI admiral receives a reward of 200 points. This wargame reward reflects the OPFOR ships' ability to harass and derail the desired course of the BLUFOR ship. Finally, OPFOR ships are allowed to cause an accident with the BLUFOR ship and thereby reduce its overall strength (representative of a collision, a near miss, or other unsafe maneuvers at sea). To dissuade the AI admiral from choosing to attrit the BLUFOR ship completely (possible in ATLATL which includes kinetic combat scenarios), a reward of 50 points was given to the AI admiral only if the BLUFOR ship was still alive or seaworthy at the end of a game.

3. Score

Scoring in the game is the same as the reward function for ease of analysis. Each turn or phase the cumulative score is calculated based on whether an OPFOR ship is adjacent to the BLUFOR ship. In the current game, the earliest contact can be made between OPFOR and BLUFOR is phase 7 of 60. Scoring occurs at the end of each player's turn or phase and so in one round OPFOR can score 200 points for being next to BLUFOR at the end of BLUFOR's turn and OPFOR's turn. At the end of a game, if BLUFOR did not reach its destination or was forced to take any path other than the shortest, 200 points will be added to the overall score. If BLUFOR is still on the board another 50 points will be added to the overall score. The maximum final score an OPFOR commander can earn is 5550 points at the end of Phase 60, which is 30 rounds of 2 turns, or the equivalent of a 10 hour FONOP. Using the standard setup in a 1-versus-1 scenario, I earned a perfect score of 5550 (see Figure 14) by preventing the BLUFOR ship from entering the yellow destination area and maintaining maximum contact with it throughout the scenario.

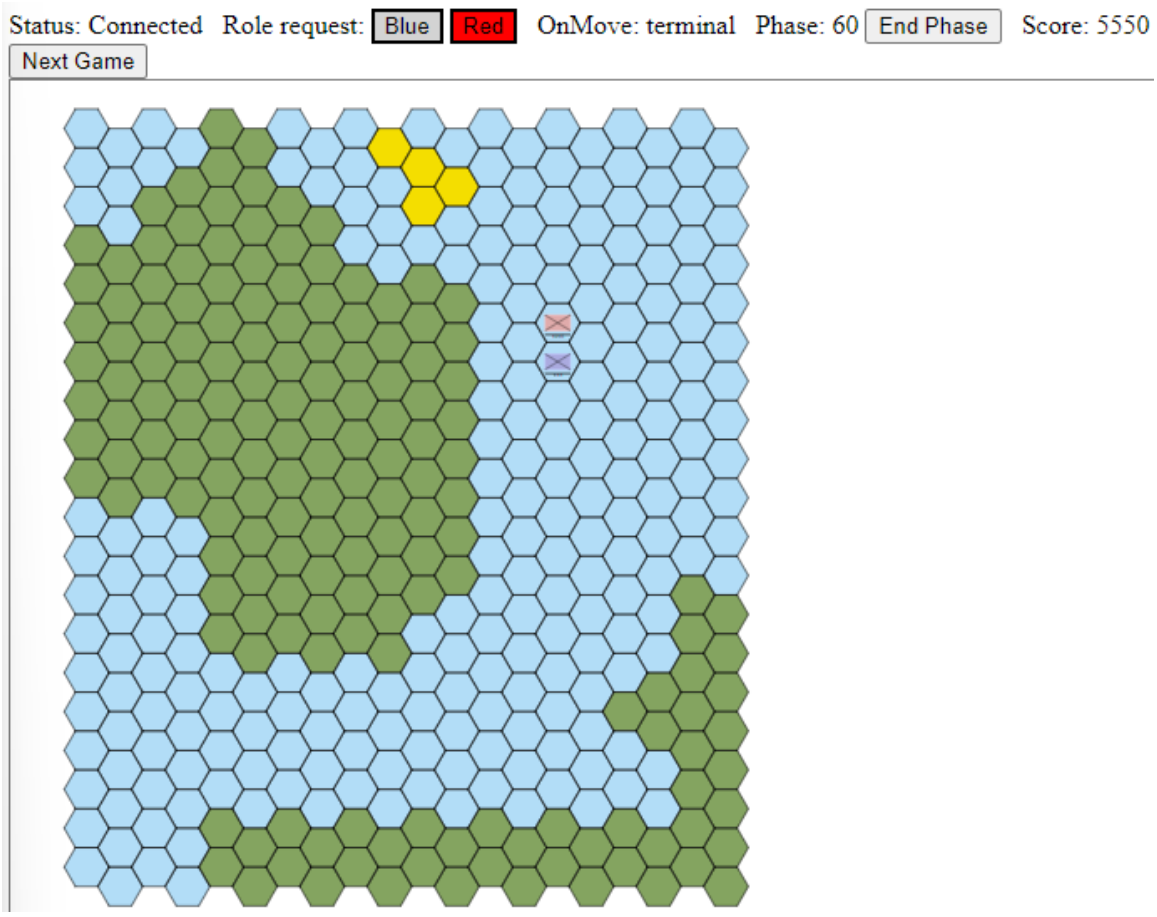


Figure 14. Optimal score in 1v1 scenario with human OPFOR

B. RL TRAINING ENVIRONMENT

To focus the OPFOR AI admiral on a single FONOPs scenario, the training environment was limited to the actual environment it would be tested in with only the number of ships under its command changing during training. The BLUFOR ship ran a complex but scripted behavior where it would take the shortest path from its starting point to the closest yellow destination hex and make every effort to navigate through either the hex adjacent to the coastline or a hex adjacent to such a hex. I chose this behavior, named buffer, to represent the BLUFOR ship sailing within 12 nm of the coastline as it executes FONOPs.

While the only units employed in the training scenario were surface ships, ATLATL does support a growing number of unit types to include infantry, mechanized

infantry, armor, and artillery at the regimental level and are depicted by NATO Joint Military Symbology [10]. The only terrain types utilized in the training environment were blue hexes for the open ocean, green hexes for the land, and yellow hexes to indicate BLUFOR's destination. Other terrain types can be employed for land forces such as clear terrain, marshes, rough terrain, and urban or city terrain. As my scenario was limited to identical unit types for BLUFOR and OPFOR and movement restricted to the blue water hexes and the yellow water destination hexes, there were no mobility adjustments made for the scenario. The BLUFOR and OPFOR ships are restricted from moving on to the green land hexes, however if other units were introduced these considerations could easily be implemented and exist within the ATLATL framework.

Tuning the AI Admiral's training includes selection of several parameters such as its policy and how long the AI admiral should consider its recent experiences. In the ATLATL framework these training parameters can be accessed in the training.py file (see Figures 10 and 15).

```
27  # train params
28  train_policy=["dqn"]#, "mlp"] #, "dqn"]
29  train_iterations_dqn=[100000] # [100000,200000]
30  train_iterations_ppo=[9000] #,[100000,200000]
31  train_gamma_dqn=[0.6] #0.01, 0.3, 0.6, 0.9, 0.99]
32  train_gamma_ppo=[0.01]#, 0.1, 0.3]
33  train_team = "red"
34  train_vs_behavior = ["buffer"]
```

Figure 15. ATLATL training parameters

These parameters include but are not limited to the policy and the factor that impacts retention, also known as gamma. Sun et. al [20] described choosing gamma as:

> Gamma is a decay factor that represents how much future returns affect the current situation, gamma is the decay value used to control the effect of future return on Q. The closer gamma is to 1, the more forward-looking it will be as it focuses on the value of subsequent states. When gamma approaches 0, it will become more focused on the impact of current interests. [20]

In other words, a higher gamma means my AI admiral will care about long-term rewards and its experiences that are building towards those rewards more. A lower gamma means my AI admiral is more short-sighted, preferring to maximize the immediate reward even if it may yield a worse future reward.

C. ALGORITHM SELECTION

During my classroom lab experiments using the ATLATL framework, I found that DQN outperformed PPO for short-duration learning and well-defined problems. In one such lab with a four-person team, I first divided my team into two groups so I could test the two policies: PPO (clip version) and DQN. Initial agents were developed using these network policies, and no changes were made to the default training parameters or only slight changes to a single variable within the policies themselves. Default training parameters for the DQN policy agents and those for the PPO agent were taken from OpenAI Spinning Up [21].

My team had several overarching constraints that directed our approach throughout the experiment. The first set of agents were constrained to a training step budget of 400000 and a set of varied land-based combat scenarios. The second set of agents was limited to a training step budget of 800000 and a different set of varied land-based scenarios.

During the initial 400000–training step runs of both DQN and PPO policies, a high performing scripted behavior named Burt_Reynolds was used to train the AI agents. When using reinforcement learning restricted to 400000–training steps, results showed DQN outperformed PPO in average score against a red force run by a different, scripted agent named pass-agg with a specified scenario Seed 4025 and scenario Cycle 1. A two-sample t-test determined a difference in the average mean for PPO and DQN. The analysis showed DQN a mean 125 higher than PPO. The t statistic was 4.08 and the p-value was 0.0007. This analysis is organized in Table 1:

Table 1. Classroom experiment between DQN and PPO trained AI agents limited to 400000–training steps in a land-based combat scenario

t-Test: Two-Sample Assuming Unequal Variances		
DQN vs PPO Means: 400,000		
	DQN	*PPO*
Mean	-152.825	-277.639
Variance	3695.451	2844.388
Observations	7	7
df	12	
t Stat	4.083466	
P(T<=t) one-tail	0.000758	
t Critical one-tail	1.782288	

After I viewed the PPO playback, optimized the parameters in PPO to the best of my knowledge, and retrained PPO agent with pass-agg, the testing adversary, as the exclusive adversary, the mean reward still did not match that of a DQN policy AI agent, seen in Figure 16.

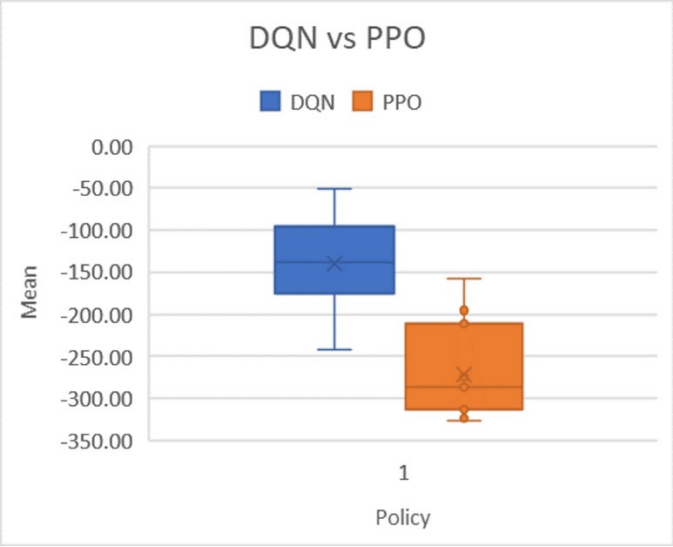


Figure 16. Mean score by policy in a land-based ATLATL scenario

Because of the nature of my thesis, the scope of its problem, and my preliminary findings, I exclusively use DQN as the algorithm to develop the AI admiral's policy.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. RESULTS

Thirty AI admirals were trained on two scenarios: a 1-versus-1 and a 2-versus-1. The scenarios share the same map and environment, the same goals, and the same BLUFOR agent**.** For both scenarios, the OPFOR or red commander was the only AI admiral trained by the neural network. The performance of the AI admiral is scrutinized with its ability to learn in under 24 hours to examine the third guiding question of this thesis: how quickly can a stand-alone, FONOPs COA analysis neural net learn the best path and course to interdict as red? In the structure of this thesis, the variable that most directly impacts the time an AI admiral requires to develop are referred to as the training iterations, training steps, or learning steps. This relationship is shown in Figure 17.

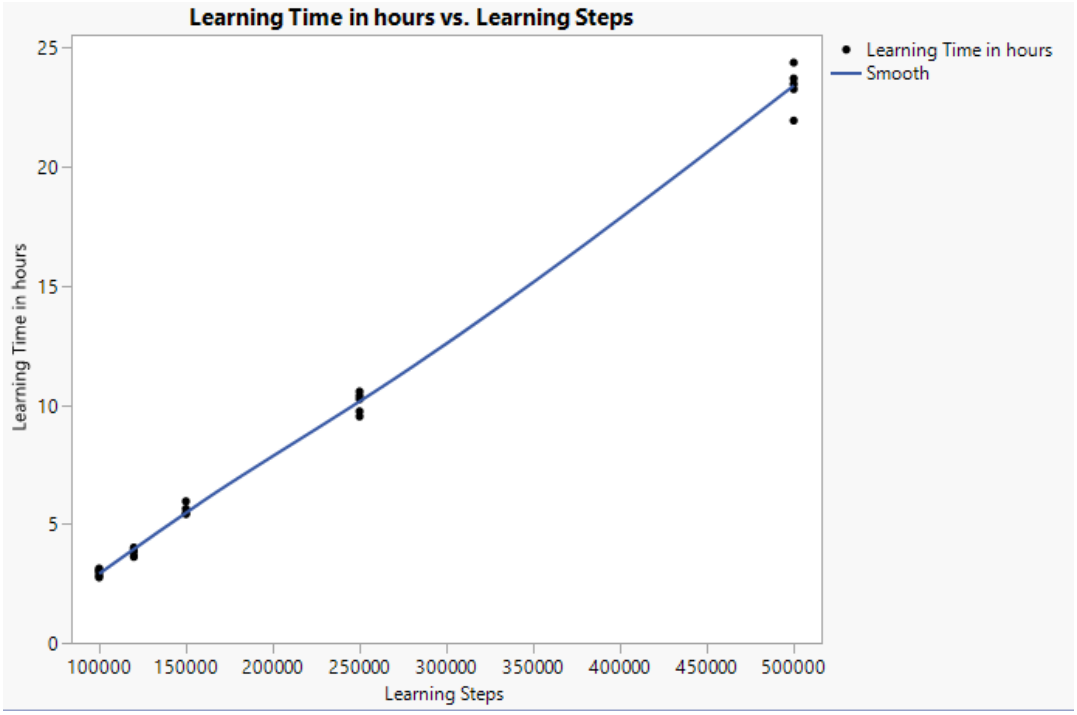


Figure 17. AI admiral learning time in hours as a function of learning steps

For this thesis a 100000–learning step AI admiral required an average of 2.94 hours, a 120000–learning step AI admiral 3.84 hours, a 150000–learning step AI admiral 5.59 hours, a 250000–learning step AI admiral 10.11 hours, and a 500000–learning step AI

admiral 23.42 hours. As seen in Figure 18, the worst range of scores an AI admiral can receive in one FONOPs wargame is below 2000, which represents the OPFOR ships minimally interdicting the BLUFOR ship, the BLUFOR ship successfully conducting FONOPs and arriving at its destination along the shortest path. The satisfactory range of scores for the AI admiral are from 2000 to 5000, representing OPFOR ships successfully interdicting the BLUFOR ship, derailing the BLUFOR ship from the shortest path, and either eliminating the BLUFOR ship (undesired behavior) or allowing the BLUFOR ship to reach its destination (tolerable behavior). Finally, near-optimal scores are between 5000 and 5550, representing maximum OPFOR harassment of BLUFOR ships, prevention of BLUFOR ship moving within the two hexes of the shoreline and diverting the BLUFOR ship from its shortest path as well as preventing the BLUFOR ship from reaching its destination by the end of the wargame, all without eliminating the BLUFOR ship.

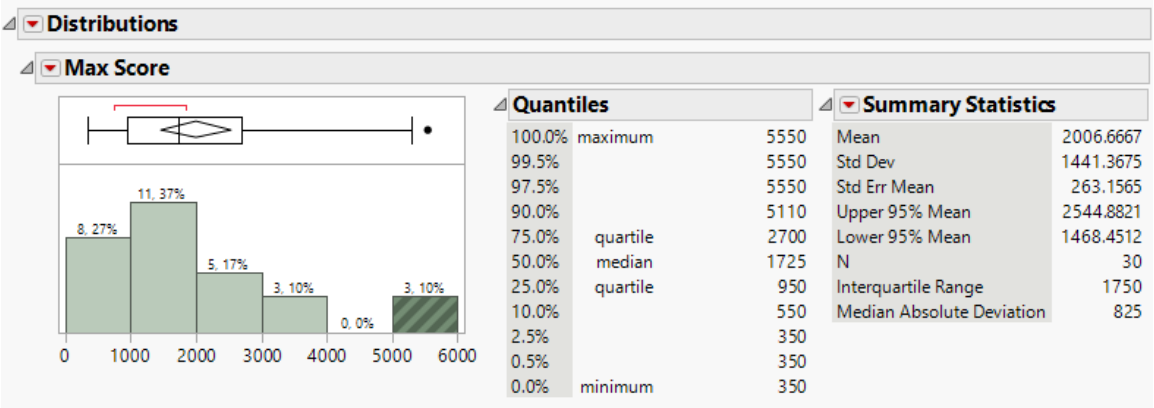


Figure 18. Distribution of the max scores earned by the 30 different AI admirals

The top AI admirals for 1-versus-1 and 2-versus-1, respectively, were both 500000–learning steps with a 90% discount factor. The runner up AI admirals for the two scenarios were both 250000–learning steps with discount factors of 90% for 1-versus-1 and 60% for 2-versus-1.

A. UNDERPERFORMANCE DUE TO TOO FEW LEARNING STEPS

The best score earned by an AI admiral trained for less than 6 hours was still in the worst performance range for both the 1-versus-1 and 2-versus-1 scenarios. To successfully constrain learning time for all AI admirals to 24 hours or less, the learning steps were all 500000 or less. The maximum score of all AI admirals who trained for under 6 hours, or 150000–learning steps or less, was 2050, with other AI admirals' maximum scores below 2000, the worst performance range. Those AI admirals who trained for 11 or 24 hours performed in the satisfactory and near-optimal ranges for all scenarios and gamma values, seen in Figure 19.

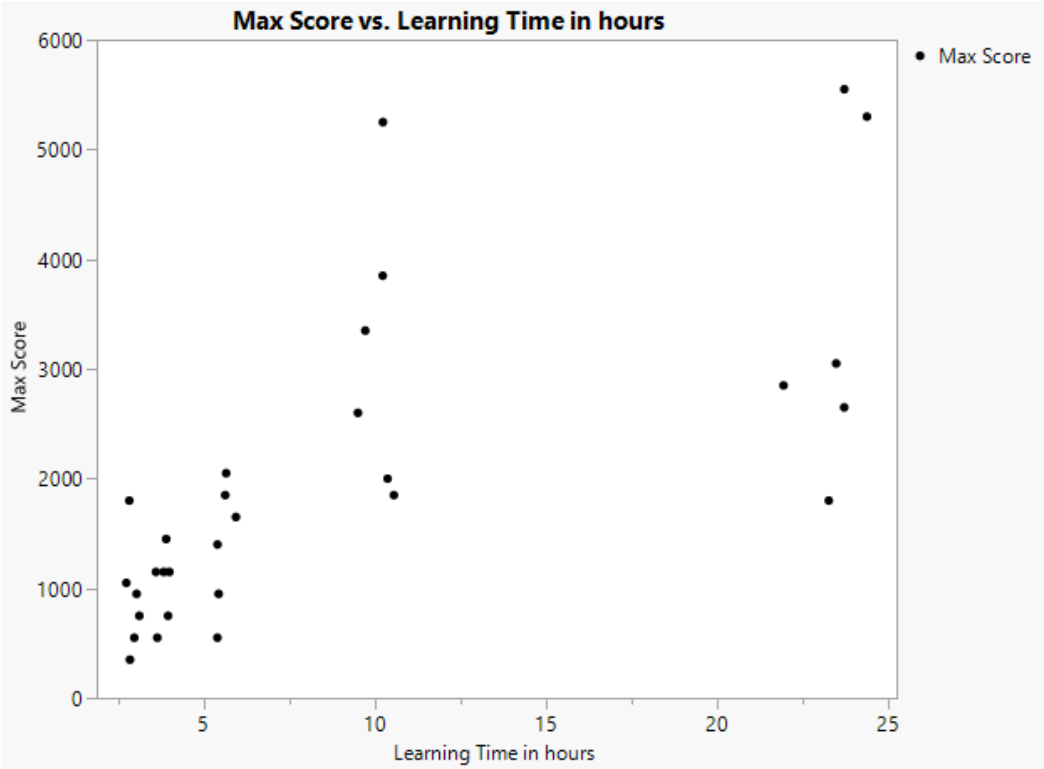


Figure 19. AI admiral performance measured by max score as a function of learning time in hours

AI admirals that also trained with DQN and the same gamma's were able to achieve satisfactory and even near-optimal scores, therefore, the critical factor for time was training AI admirals for more than 6 hours. As none of the AI admirals who trained less than 6

hours possessed maximum scores distributed within the satisfactory range, I evaluated them as underperforming due to too few learning steps.

B. COMPARATIVE PERFORMANCE AT 250K AND 500K LEARNING STEPS

For the top performing AI admirals in each scenario, there was excellent performance by the 250000–learning step AI admirals, although the 500000–learning step AI admirals obtained the better scores. For the 1-versus-1 scenario, the top score for 250000–learning steps was 5250, and the top score for the 500000 was 5550. An increase of 13.47 hours of training, 131.4% increase, to yield a 300–point score increase, 5.7% increase, in the excellent range is not a significant increase in score and performance for the time. For the 2-versus-1 scenario, the top score for 250000–learning steps was 3850, and the top score for the 500000 was 5300. In this scenario, an increase of 14.14 hours of training, 138.1% increase, to yield a 1450–point score increase, 37.7% increase, and moves the AI admiral's score and performance from the acceptable range to the excellent range. Across both scenarios and all other variables, the 250000–learning step AI admiral's score is in the same range as the 500000 (see Figure 20).

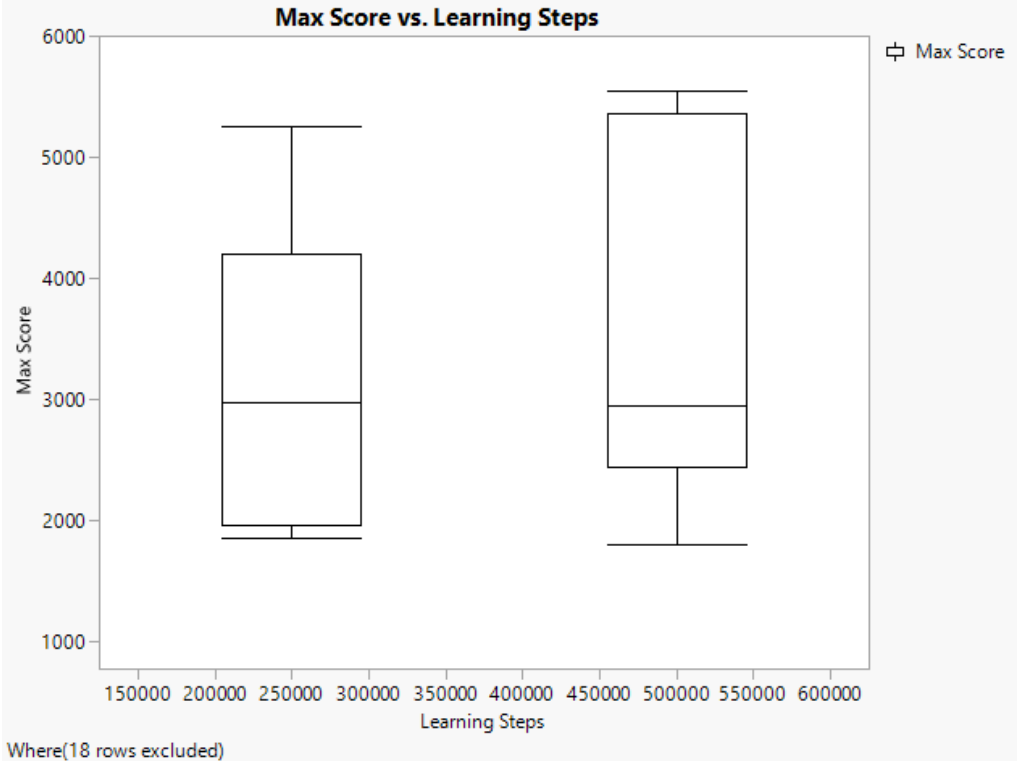


Figure 20. Comparison of AI admirals' performance distribution measured by max score as a function of learning steps

1. 1-versus-1 scenario

In the 1-versus-1 scenario (see Figure 21), AI admirals trained with 250000– and 500000–learning steps performed better with a gamma, or discount factor, of 0.9 or 90% (see Figure 22).

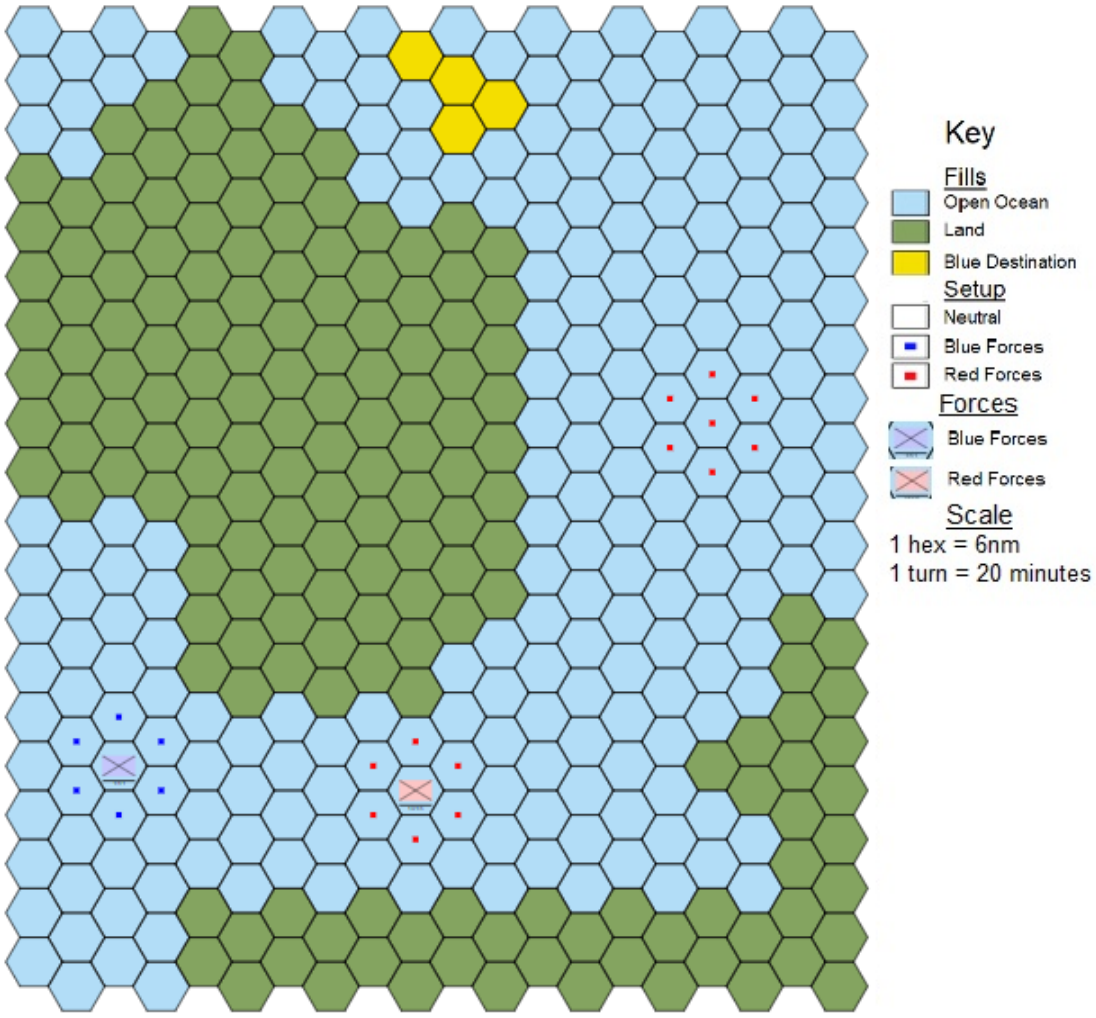


Figure 21. 1v1 scenario

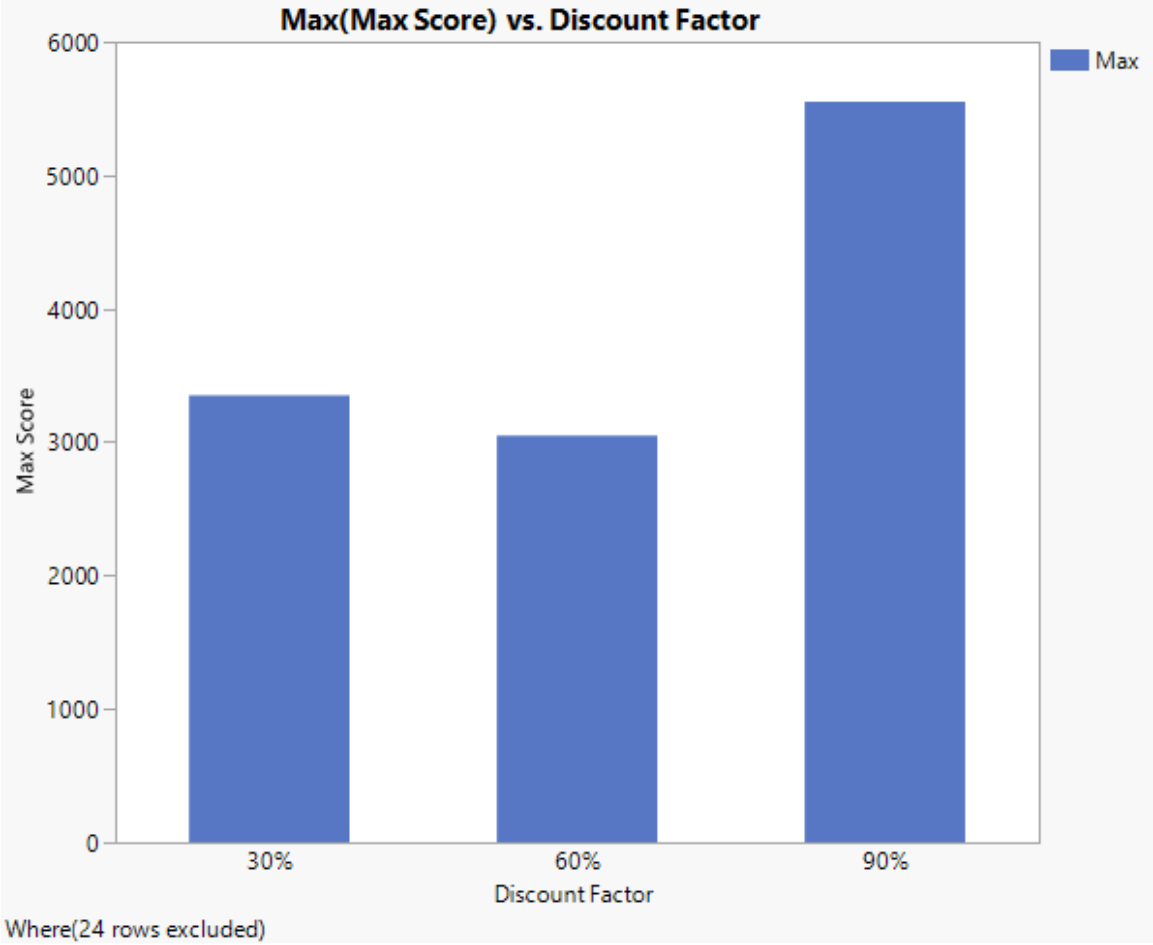


Figure 22. 250000– and 500000–learning step AI admirals' max scores as a function of their discount factor on a 1v1 scenario

2. 2-versus-1 scenario

In the 2-versus-1 scenario (see Figure 23), the results varied with the 500000–learning step AI admiral performing best with a gamma, or discount factor, of 0.9 or 90%, while the runner up for best performance measured by max score was an AI admiral trained with 250000–learning steps and a gamma, or discount factor, of 0.6 or 60% (see Figure 24).

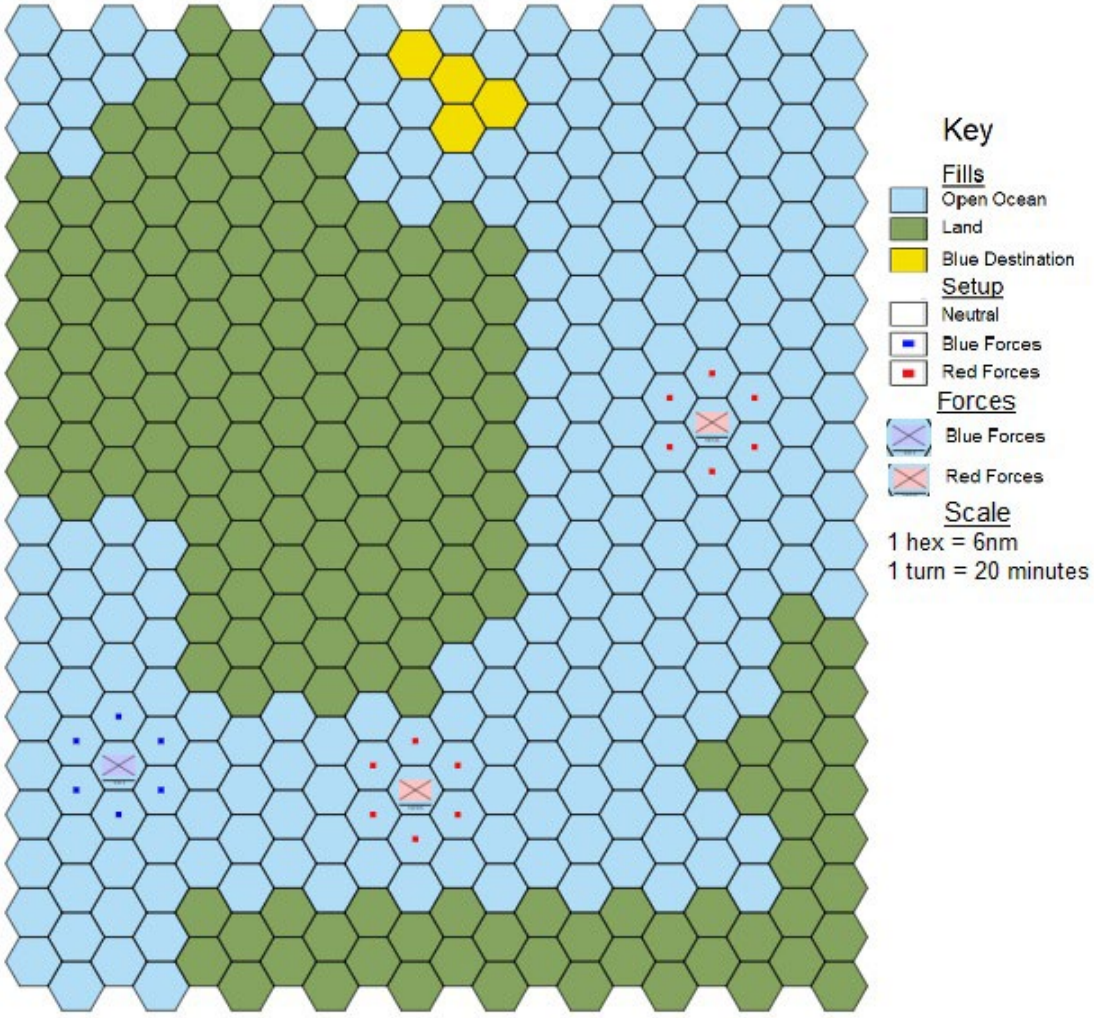


Figure 23. 2v1 scenario

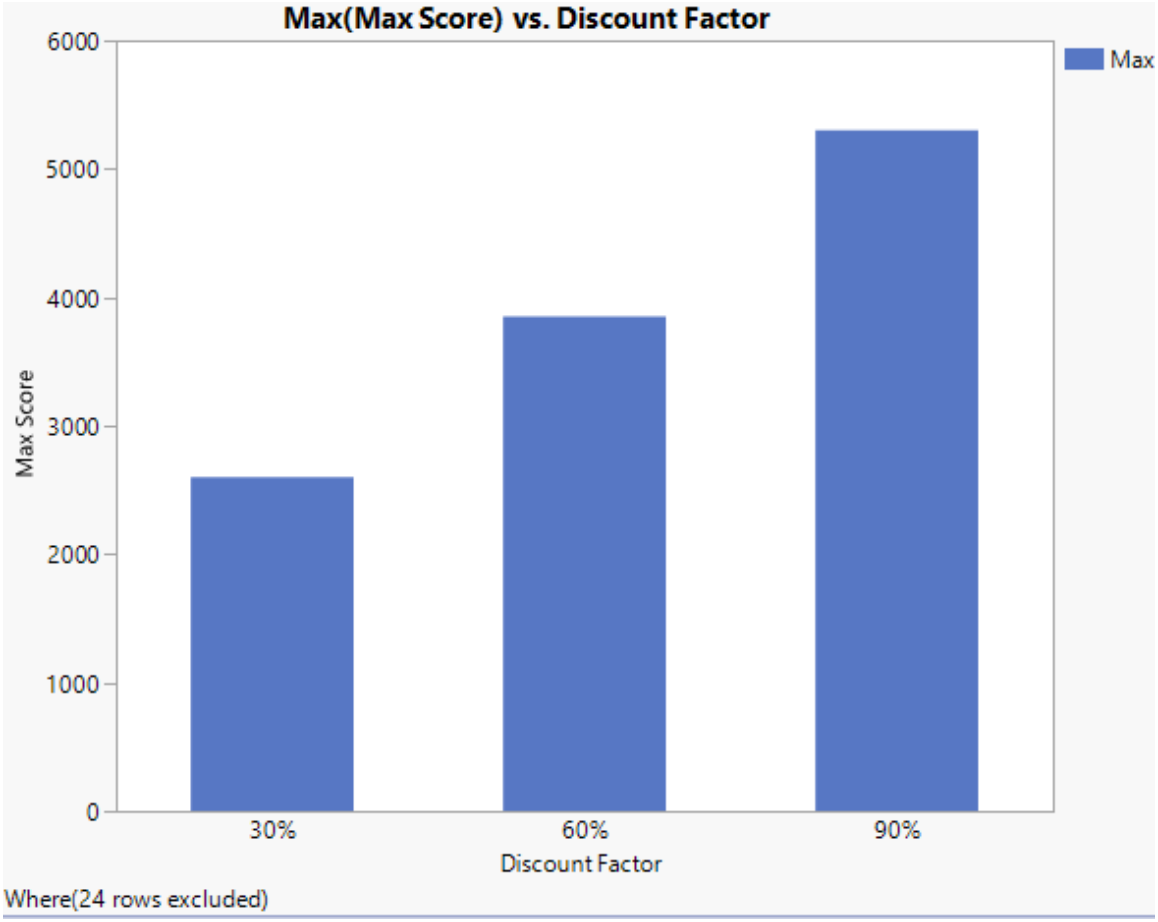


Figure 24. 250000– and 500000–learning steps AI admirals' max scores as a function of their discount factor on a 2v1 scenario

As the performance cannot easily be attributed to a discount factor, I have provided playback captures or battle reports.

a. Battle report 1: AI admiral with 500000–learning steps, 0.9 gamma, earns final score 5300

This AI admiral started its best match against the buffer BLUFOR FONOPs ship by directly interdicting it with its southern ship and attempting to impede BLUFOR beginning on turn 3 or roughly 1 hour after the operation started (see Figure 25). Its second ship remained in position roughly 24 nm east of the coast for the first 12 turns, roughly 4 hours (see Figures 25 and 26).

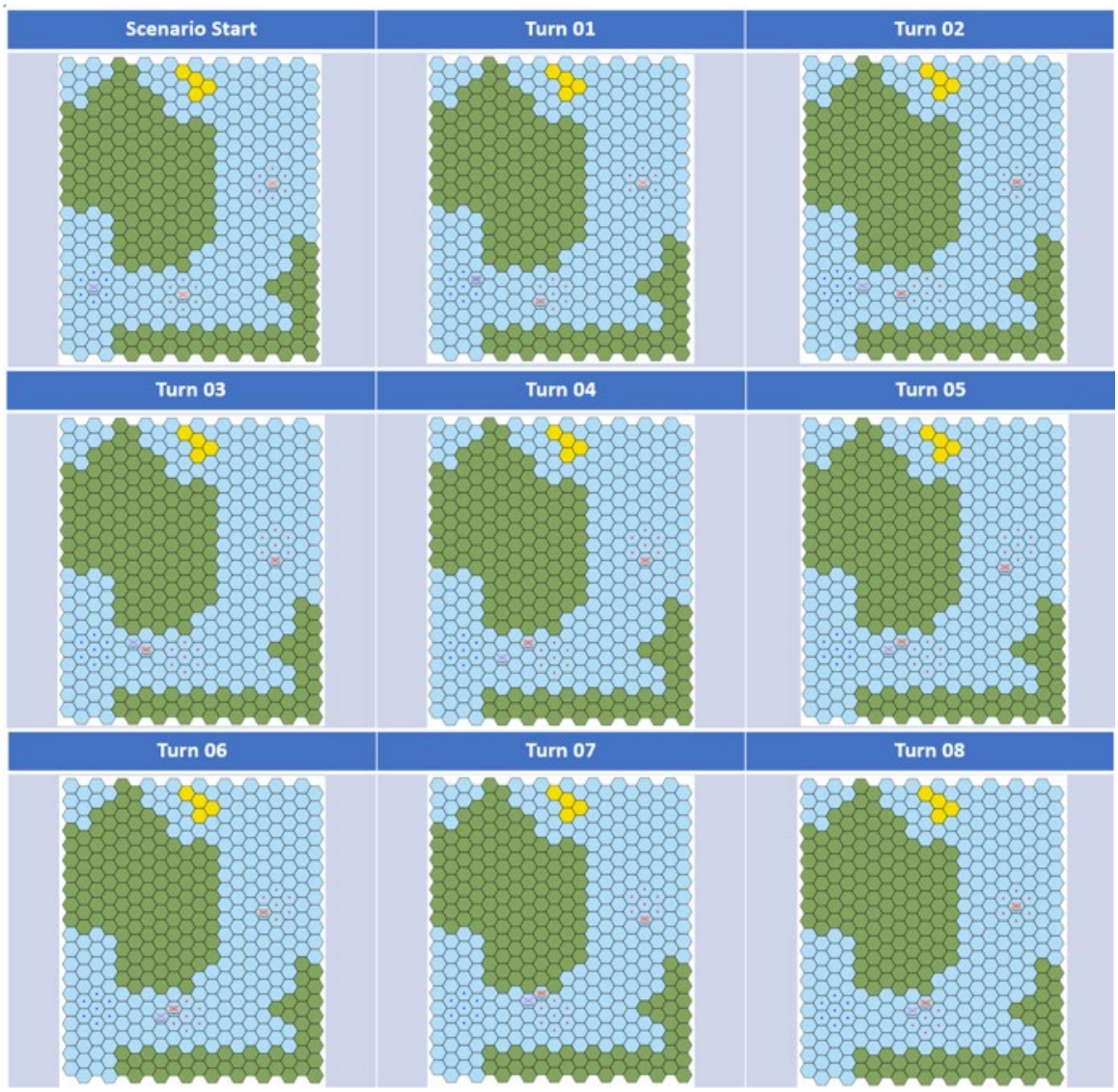


Figure 25. 500000–learning step AI admiral 2v1 battle report through turn 8

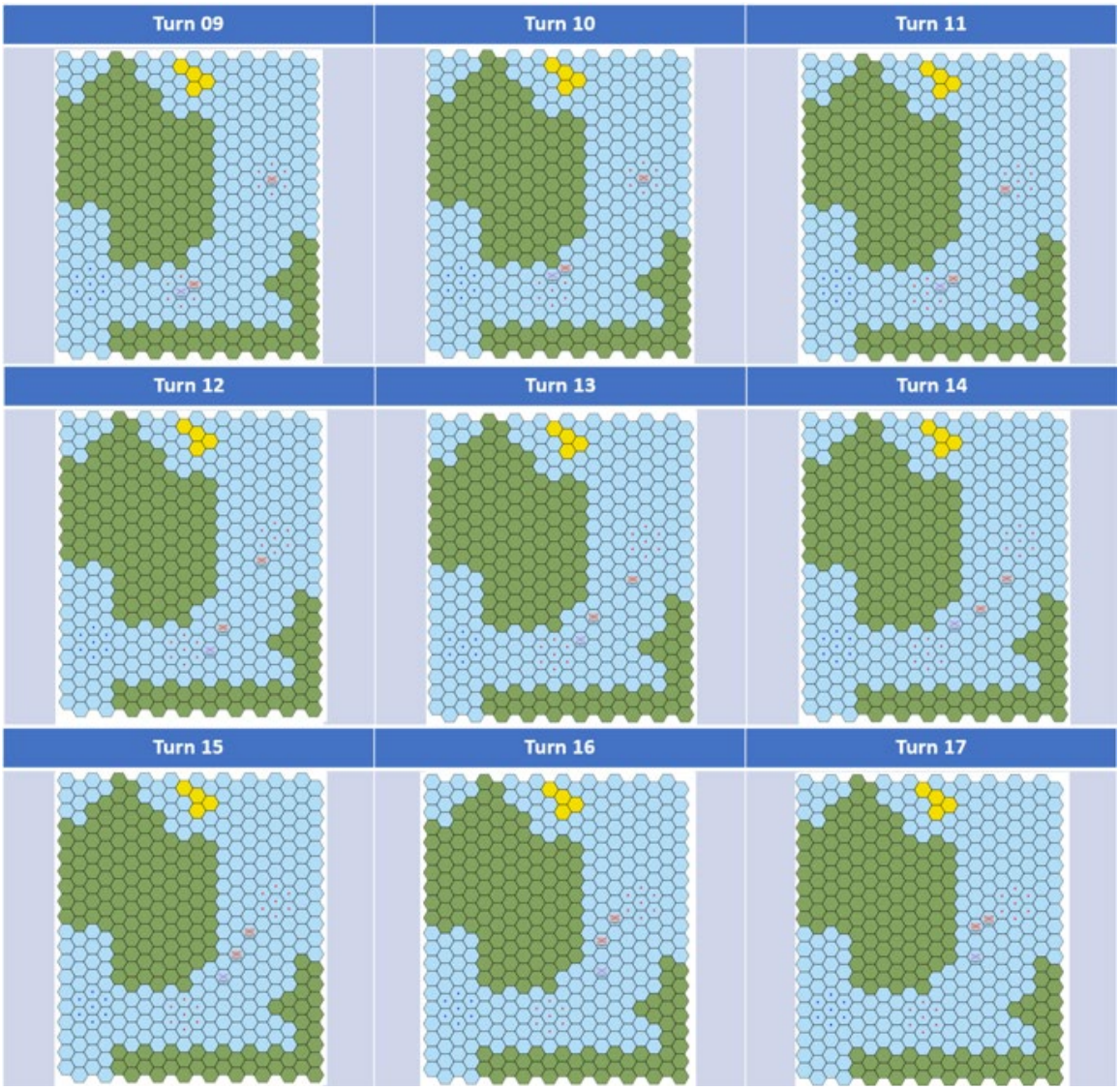


Figure 26. 500000–learning step AI admiral 2v1 battle report through turn 17

Turn 15 marked the beginning of the AI admiral using the 2 OPFOR ships in tandem for interdicting the BLUFOR FONOP. The OPFOR 2 ship screen begins on turn 17 and continues through turn 27, roughly from the operational time marks of 5 hours 40 minutes to 9 hours (see Figures 26-28).

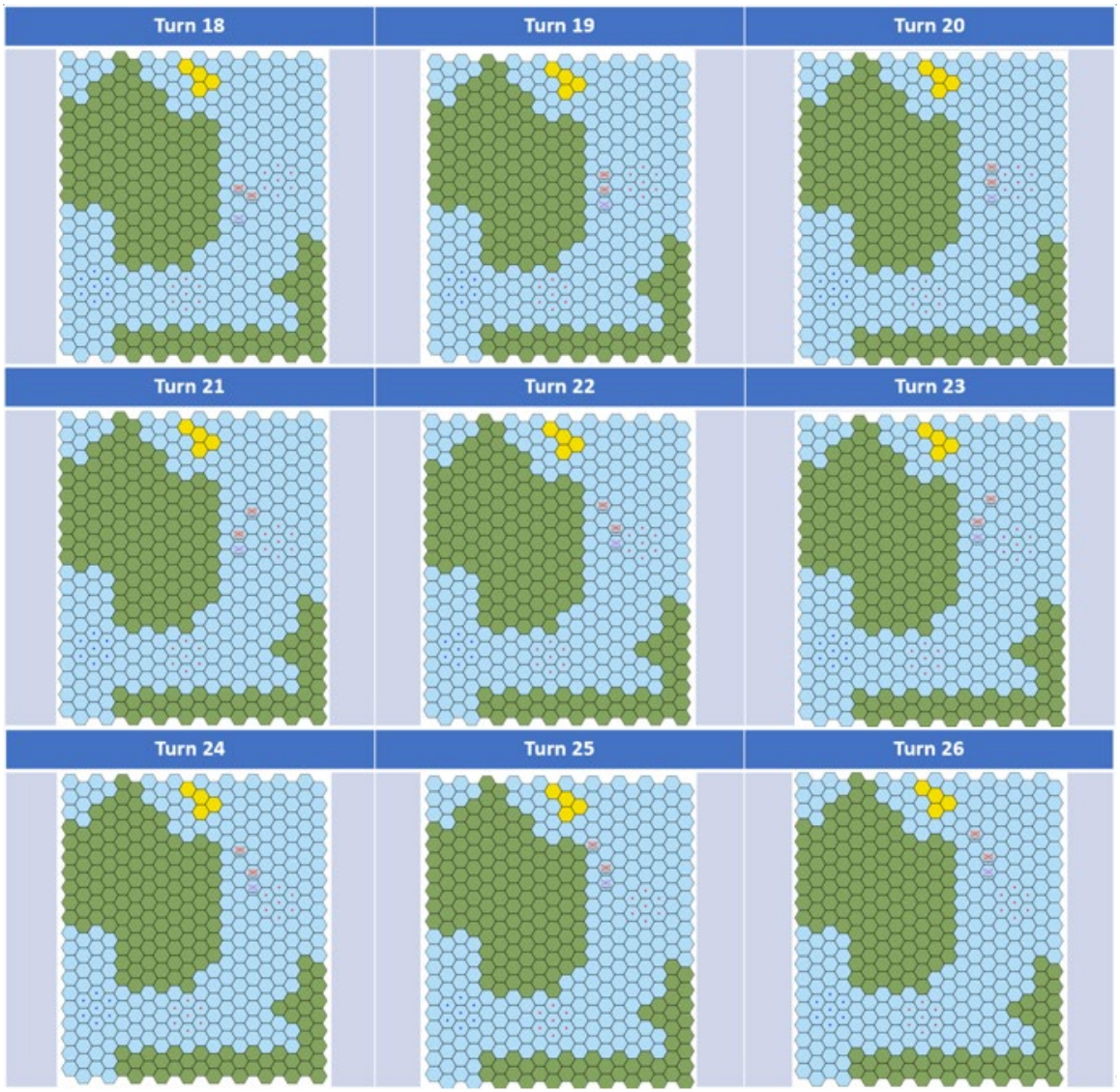


Figure 27. 500000–learning step AI admiral 2v1 battle report through turn 26

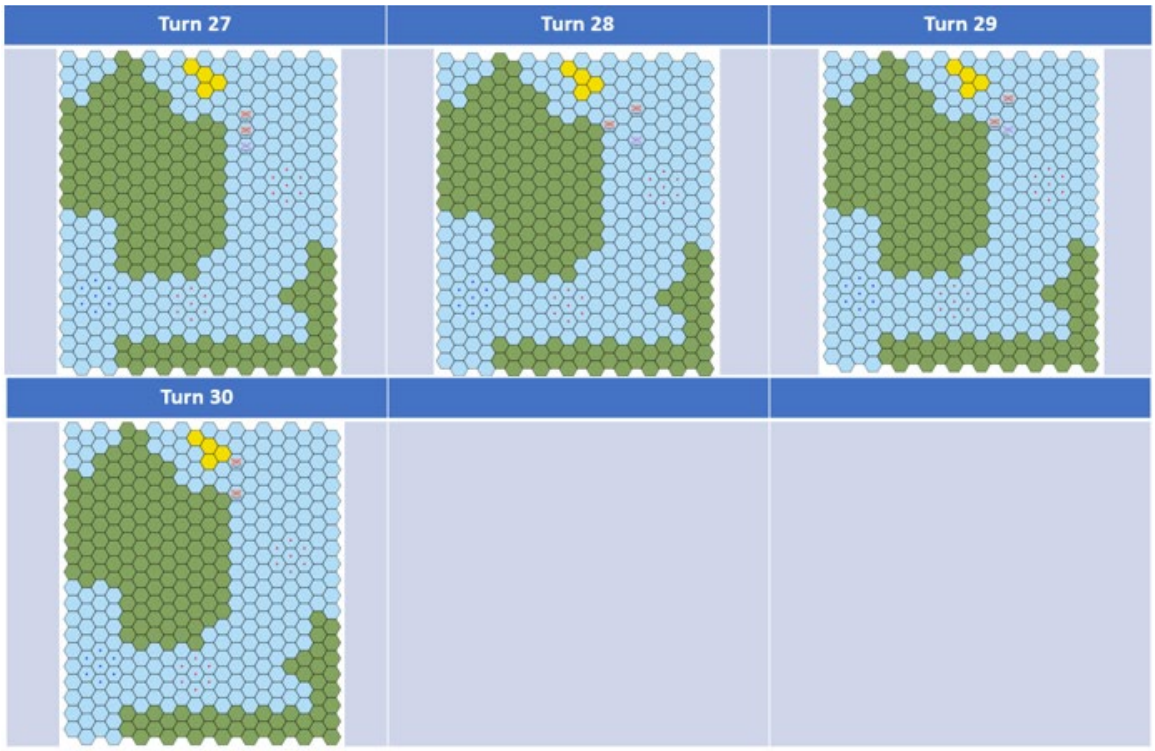


Figure 28. 500000–learning step AI admiral 2v1 battle report through turn 30

During the final 3 turns, the AI admiral arranges the 2 OPFOR ships in a flanking screen, allowing BLUFOR the opportunity to sail around the outside ship or sail between the 2. BLUFOR sails between the 2 in order to maintain its 2 hex or 12nm range of the coast and is aggressively engaged by both ships. This overwhelming use of force actually passes the threshold for attrition in ATLATL and therefore the ship is removed, representing the ship is no longer mission capable.

b. Battle report 2: AI admiral with 250000–learning steps, 0.6 gamma, earns final score 3850

This AI admiral's score is in the satisfactory range, but reviewing its play reveals several places it could have benefited from more learning steps. This AI admiral started its best match against the buffer BLUFOR FONOPs ship similarly to battle report 1, by directly interdicting it with its southern ship and attempting to impede BLUFOR beginning on turn 3 or roughly 1 hour after the operation started (see Figure 29). Its second ship sails south but then halts all movement until turn 18 (see Figure 31).

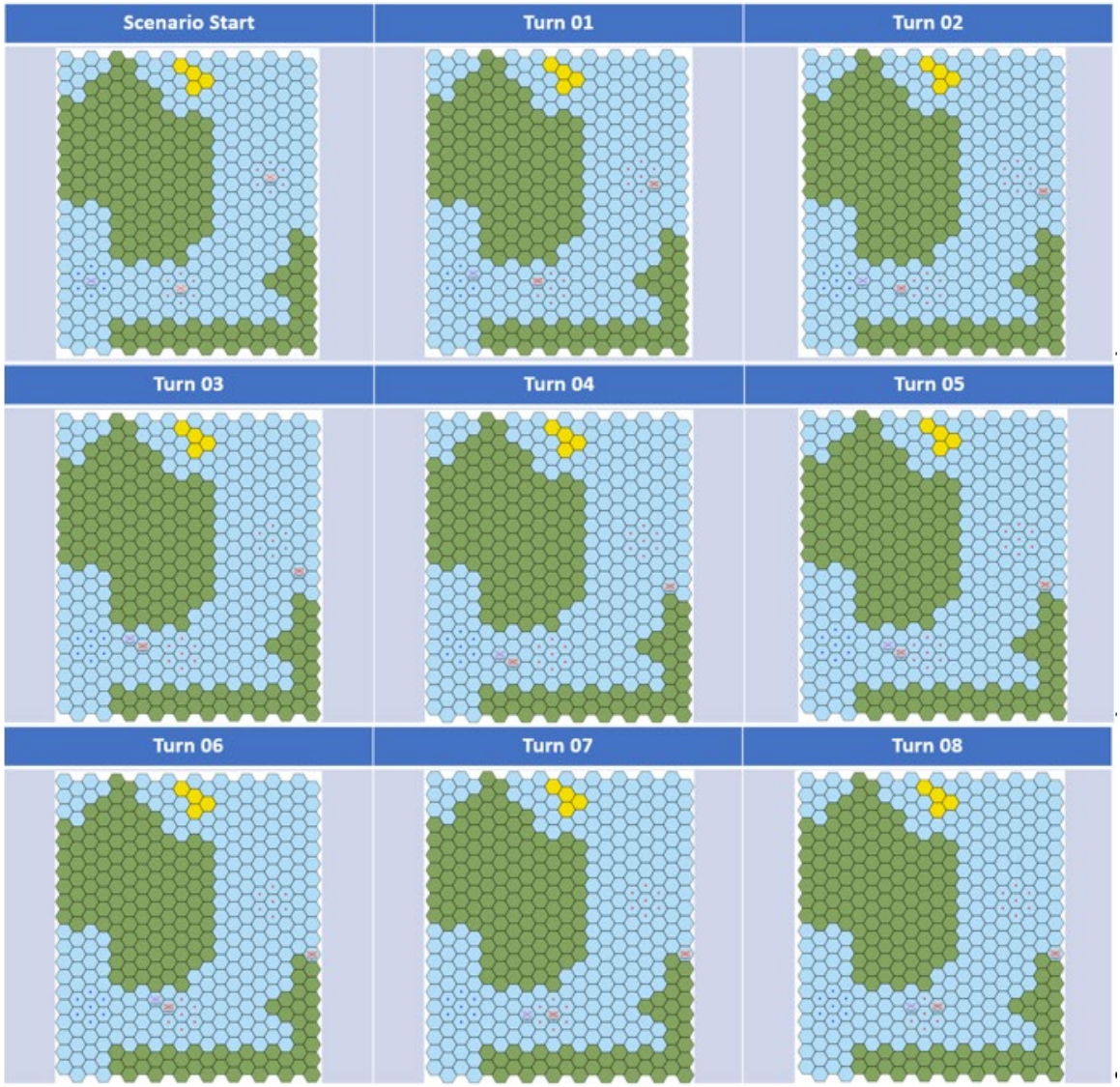


Figure 29. 250000–learning step AI admiral 2v1 battle report through turn 8

The AI admiral directs its southern OPFOR ship to screen the BLUFOR ship's anticipated FONOP route to the yellow destination hexes, however, between turn 15 and turn 16, it misplays by moving into a hex 3 away from the shore (see Figure 30).

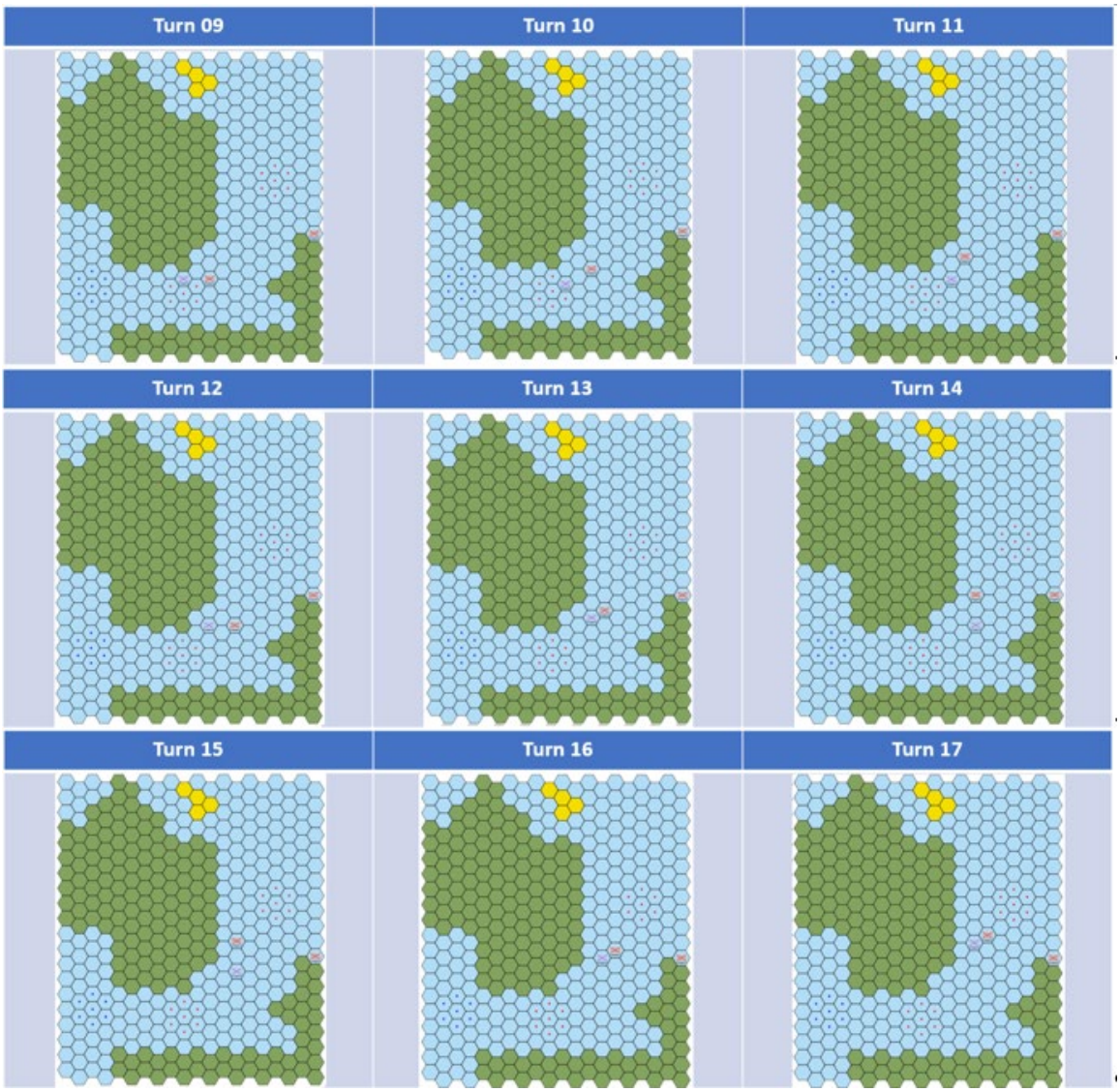


Figure 30. 250000–learning step AI admiral 2v1 battle report through turn 17

These 2 misplays by the AI admiral result in a mismanagement of its forces that allow BLUFOR to complete its FONOP with minimal harassment or impediment during the northern leg of its maneuver. This AI admiral has not learned how to best control 2 ships, apparent from the lack of engagement and tactical use of its northern ship. Furthermore, this AI admiral has learned how to screen and funnel the BLUFOR ship to where it desires, however, employing only 1 ship requires it to also remain within 2 hexes of the coast at all times if it is to succeed. The AI admiral fails to do this on turn 16 (see Figure 30).

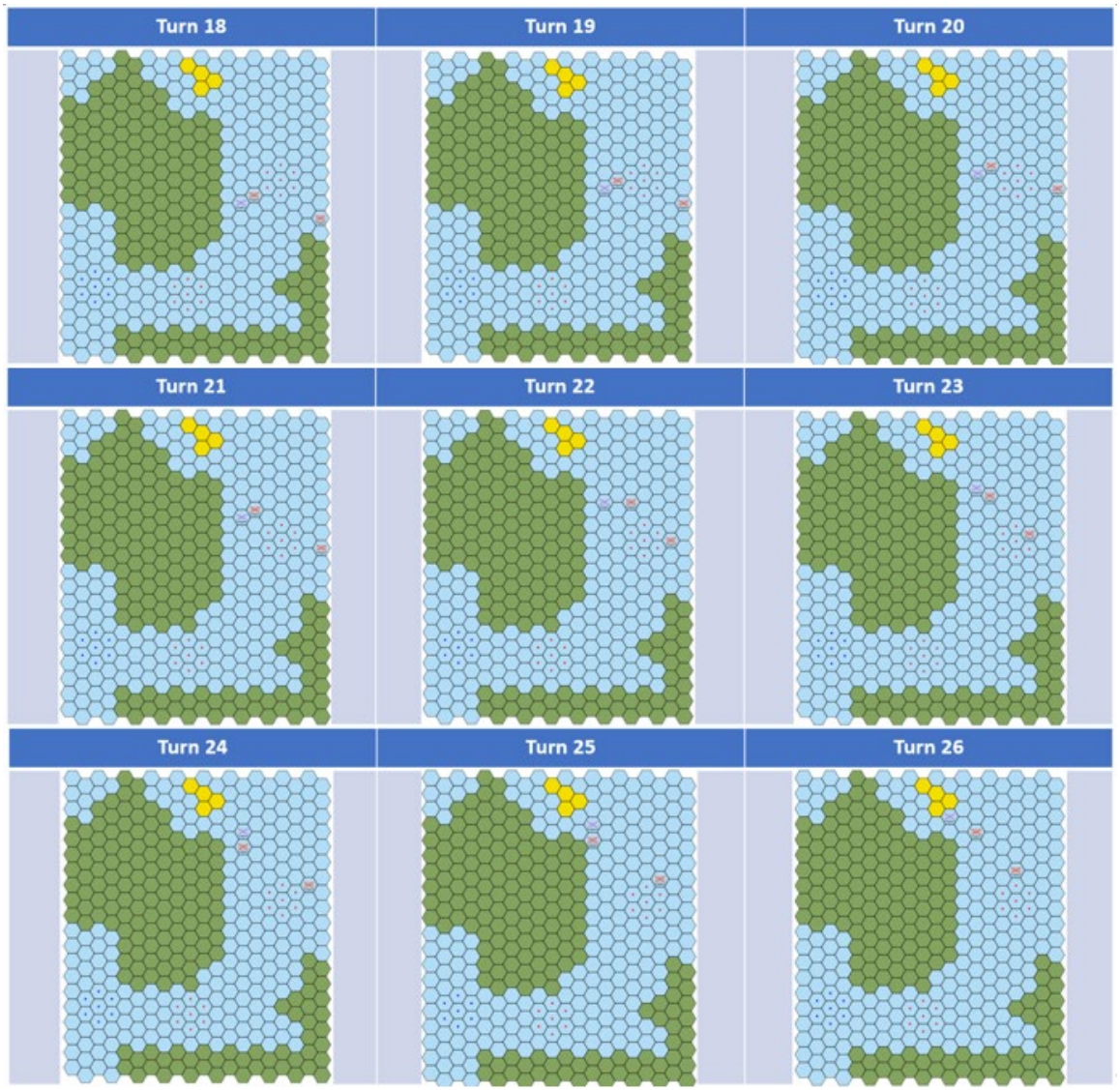


Figure 31. 250000–learning step AI admiral 2v1 battle report through turn 26

The differences in max score earned by these 2 AI admirals are reflected in their play, and while the 500000–learning step AI admiral sub-optimally eliminates its BLUFOR ship (see Figure 28), the 250000–learning step AI admiral fails to successfully slow or interdict the BLUFOR ship for the northern leg of its FONOP (see Figures 30–32).

Figure 32. 250000–learning step AI admiral 2v1 battle report through turn 28

These differences in achievable max score between the 2-versus-1 and 1-versus-1 scenarios lead me to believe that a larger action space, created by the difference in number of units, increases the complexity of the environment for the AI admiral and in turn requires more time for it to successfully learn "good enough" behaviors through RL to create a successful policy. The current environment results in 250000–learning steps or 10.25 hours of training sufficient for an AI admiral to learn to maneuver one red ship to interdict BLUFOR FONOPs, however, that amount of training is not sufficient for an AI admiral to learn to maneuver a squadron of 2 OPFOR ships.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSIONS AND FUTURE WORK

For similar-sized, time-sensitive planning problems, DQN will perform as well or better than PPO. For similar-sized, time-sensitive scenarios as the 1-versus-1, 250000–learning steps will perform as well as 500000–learning steps, and the time savings will be worth it to choose 250000. For similar sized problems as the 2-versus-1, more time steps will be needed, however 500000 should suffice.

A. CONCLUSIONS REGARDING ATLATL AND FONOPS APPLICATIONS

In his presentation "Toward Tactically Savvy AI," Darken [26] presented the notion that tactically savvy AI agents are not often attempted because most AI agents perform well only for a single scenario. While the goal of a future AI admiral capable of reliable performance over a broad range of scenarios, OOB and objectives is the desired end goal, I believe we can accept the current limitations and focus on training AI admirals on the scenarios we care about, free to define the scenario as a forward deployed commander and their crew observe it. This approach will allow forward deployed planners to brief and prepare an AI admiral on the current situation, just like planners would their own ship. Military planners can define the problem and use the tools and techniques presented here to arrive at a quality solution in under 24 hours.

When hard coding limitations into a simulation or wargame such as ATLATL, it is crucial to restrict those to limitations primarily in physics (see Figure 13) and not in tactics. While the aggressive maneuver executed by the AI admiral in battle report 1 (see Figure 28) may appear unlikely between two forces not at war, it is important to remember that AI admiral was trained to be a most dangerous ECOA, and unsafe and unprofessional maneuvers that might result in a collision were permitted. The margin for these errors in real world navigation are slim, and ignoring the threat they pose can be deadly, as seen between USN ships and merchant ships (e.g. USS Fitzgerald and USS John S. McCain). The possibility of these actions should be permitted in code, and the commanding officer should then be equipped to decide to train and teach to it, or to mitigate it. Whether to

remove its consideration from the COA analysis and wargaming or to instruct the real ship's crew to learn from the folly of their digital counterparts.

B. RECOMMENDATIONS AND FUTURE WORK

One possible way to expedite short turnaround training of AI agents in large maritime scenarios might be in a tiered AI system approach: tactical use of a unit might be trained at an AI captain level, a group may be trained at an AI DESRON level, and the strategic implementation of them might be conducted at the AI admiral level. Dividing and conquering a larger task with a team of various AI agents that focus on limited pieces of the environment may reduce overall training time, increase processing speeds, and yield better final results. This approach of creating AI objects and organizing projects as such may also yield agents that can be transported from scenario to scenario with various AI admirals calling on the DESRON or captains to be recycled opposed to trained with each new problem or scenario.

LIST OF REFERENCES

[1] U.S. President. Proclamation. "U.S. oceans policy," Department of State Bulletin 83, no 2075, pp. 70-71, Jun. 1983 [Online]. Available: https://archive.org/details/sim_department-of-state-bulletin_1983-06_83_2075/page/70/mode/2up

[2] Navy Planning, NWP5-01, Department of the Navy Office of the Chief of Naval Operations, Norfolk, VA, USA, 2013 [Online]. Available: https://archive.org/details/501Dec2013NWPPromulgated/mode/2up

[3] National Oceanic and Atmospheric Administration Office of General Counsel, "Maritime zones and boundaries," Accessed on Jul. 14, 2022 [Online]. Available: https://www.gc.noaa.gov/gcil_maritime.html

[4] J. Sampson, "Themes and theorists: Clausewitz, Sun Tzu, on strategy and war," presented at Naval Postgraduate School, Monterey, CA, USA, 2021 [Online].

[5] O. Vinyals, I. Babuschkin, W. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. Choi, et al, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, Nov. 2019 [Online]. Available doi: 10.1038/s41586-019-1724-z

[6] DeepMind, "DeepMind StarCraft II demonstration," YouTube, Jan. 24, 2019 [Online]. Available: https://www.youtube.com/watch?v=cUTMhmVh1qs&t=2004s

[7] O. Vinyals, I. Babuschkin, W. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. Choi, et al, "AlphaStar: Mastering the real-time strategy game StarCraft II," blog, Jan. 24, 2019 [Online]. Available: https://www.deepmind.com/blog/alphastar-mastering-the-real-time-strategy-game-starcraft-ii

[8] *Modeling and Simulation Body of Knowledge*, Modeling and Simulation Coordination Office (MSCO), Alexandria, VA, USA, May 17, 2011 [Online]. Available: https://www.msco.mil/DocumentLibrary/MSReferences/MSEducation/_25_MSBOK20101022DistA.pdf

[9] E. H. Page and R. Smith, "Introduction to military training simulation: a guide for discrete event simulationists," 1998 Winter Simulation Conference. Proceedings (Cat. No.98CH36274), pp. 53-60 vol.1, Dec. 13, 1998 [Online]. Available doi: 10.1109/WSC.1998.744899.

[10] C. Cannon and S. Goericke, "Using convolution neural networks to develop robust combat behaviors through reinforcement learning," M.S. thesis, Dept. of Comput. Sci., Naval Postgraduate School, Monterey, CA, 2021.

[11] C. Darken, "Introduction: Architecture," class notes for MV4025: Cognitive and Behavioral Modeling for Simulations, Dept. of Comp. Sci., Naval Postgraduate School, Monterey, CA, USA, summer 2021 [Online]. Available: https://cle.nps.edu/access/content/group/2e8bd4cb-0859-4eeb-8700-7336e9895f68/Lectures/2-agent-architecture.pdf

[12] Thomas Mitchell, *Machine Learning*, 1st ed. McGraw-Hill, Inc., 1997.

[13] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ: Prentice Hall Press, 2009.

[14] OpenAI, "Introduction," OpenAI Spinning Up, 2018 [Online]. Available: https://spinningup.openai.com/en/latest/user/introduction.html

[15] J. A. Boron, "Developing combat behavior through reinforcement learning," M.S. thesis, Dept. of Comput. Sci., Naval Postgraduate School, Monterey, CA, 2020 [Online]. Available: https://apps.dtic.mil/sti/pdfs/AD1114652.pdf

[16] S. Levine, "CS285 deep reinforcement learning," Robotic AI & Learning Lab, Berkeley, CA, USA, 2019 [Online]. Available: http://rail.eecs.berkeley.edu/deeprlcourse/

[17] C. Darken, "Machine learning tools," class notes for MV4025: Cognitive and Behavioral Modeling for Simulations, Dept. of Comp. Sci., Naval Postgraduate School, Monterey, CA, USA, summer 2021 [Online]. Available: https://cle.nps.edu/access/content/group/2e8bd4cb-0859-4eeb-8700-7336e9895f68/Lectures/8-machine-learning-tools.pdf

[18] L. Hardesty, "Explained: Neural networks," *MIT News*, Apr. 2017 [Online]. Available: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414

[19] K. J. Maroon, "Predicting opponent position and modeling uncertainty," Ph.D. dissertation, Dept. of Comput. Sci., Naval Postgraduate School, Monterey, CA, 2020 [Online]. Available: https://apps.dtic.mil/sti/pdfs/AD1126513.pdf

[20] Y. Sun, B. Yuan, T. Zhang, B. Tang, W. Zheng, and X. Zhou, "Research and implementation of intelligent decision based on a priori knowledge and DQN algorithms in wargame environment," *Electronics*, vol. 9, no. 10, p. 1668, Oct. 2020 [Online]. Available doi: 10.3390/electronics9101668.

[21] OpenAI, "Proximal policy optimization," OpenAI Spinning Up, 2018 [Online]. Available: https://spinningup.openai.com/en/latest/algorithms/ppo.html

[22] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv, Aug. 28, 2017 [Online]. Available: http://arxiv.org/abs/1707.06347

[23] V. Minh, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing Atari with deep reinforcement learning,” DeepMind, London, UK, Dec. 19, 2013 [Online]. Available: https://www.cs.toronto.edu/~vmnih/docs/dqn.pdf

[24] G. Allen, “Understanding AI technology,” Joint Artificial Intelligence Center (JAIC), The Pentagon, USA, Apr. 1, 2020 [Online]. Available: https://apps.dtic.mil/sti/pdfs/AD1099286.pdf

[25] Google, “Cloud TPU System Architecture.” Accessed Aug. 12, 2022 [Online]. Available: https://cloud.google.com/tpu/docs/system-architecture-tpu-vm

[26] C. Darken, “Towards tactically savvy AI,” presented at MOVES open house, Dept. of Comp. Sci., Naval Postgraduate School, Monterey, CA, USA, May 24, 2022.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California