2022-09

SCALING REINFORCEMENT LEARNING THROUGH FEUDAL MULTI-AGENT HIERARCHY

Rood, Patrick R.

Monterey, CA; Naval Postgraduate School

http://hdl.handle.net/10945/71091

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

SCALING REINFORCEMENT LEARNING THROUGH FEUDAL MULTI-AGENT HIERARCHY

by

Patrick R. Rood

September 2022

Thesis Advisor: Christian J. Darken
Second Reader: Charles R. Timm

THIS PAGE INTENTIONALLY LEFT BLANK

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

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.

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

4. TITLE AND SUBTITLE SCALING REINFORCEMENT LEARNING THROUGH FEUDAL MULTI-AGENT HIERARCHY	**5. FUNDING NUMBERS**
6. AUTHOR(S) Patrick R. Rood	
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)

Militaries conduct wargames for training, planning, and research purposes. Artificial intelligence (AI) can improve military wargaming by reducing costs, speeding up the decision-making process, and offering new insights. Previous researchers explored using reinforcement learning (RL) for wargaming based on the successful use of RL for other human competitive games. While previous research has demonstrated that an RL agent can generate combat behavior, those experiments have been limited to small-scale wargames. This thesis investigates the feasibility and acceptability of scaling hierarchical reinforcement learning (HRL) to support integrating AI into large military wargames. Additionally, this thesis also investigates potential complications that arise when replacing the opposing force with an intelligent agent by exploring the ways in which an intelligent agent can cause a wargame to fail. The resources required to train a feudal multi-agent hierarchy (FMH) and a standard RL agent and their effectiveness are compared in increasingly complicated wargames. While FMH fails to demonstrate the performance required for large wargames, it offers insight for future HRL research. Finally, the Department of Defense verification, validation, and accreditation process is proposed as a method to ensure that any future AI application applied to wargames are suitable.

14. SUBJECT TERMS deep learning, reinforcement learning, autonomous agents, constructive simulations, wargaming	**15. NUMBER OF PAGES** 109
	16. PRICE CODE

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

THIS PAGE INTENTIONALLY LEFT BLANK

SCALING REINFORCEMENT LEARNING THROUGH FEUDAL MULTI-AGENT HIERARCHY

Patrick R. Rood
Lieutenant Colonel, United States Army
BS, Texas A&M University, College Station, 2004

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2022

Approved by: Christian J. Darken
Advisor

Charles R. Timm
Second Reader

Gurminder Singh
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Militaries conduct wargames for training, planning, and research purposes. Artificial intelligence (AI) can improve military wargaming by reducing costs, speeding up the decision-making process, and offering new insights. Previous researchers explored using reinforcement learning (RL) for wargaming based on the successful use of RL for other human competitive games. While previous research has demonstrated that an RL agent can generate combat behavior, those experiments have been limited to small-scale wargames. This thesis investigates the feasibility and acceptability of scaling hierarchical reinforcement learning (HRL) to support integrating AI into large military wargames. Additionally, this thesis also investigates potential complications that arise when replacing the opposing force with an intelligent agent by exploring the ways in which an intelligent agent can cause a wargame to fail. The resources required to train a feudal multi-agent hierarchy (FMH) and a standard RL agent and their effectiveness are compared in increasingly complicated wargames. While FMH fails to demonstrate the performance required for large wargames, it offers insight for future HRL research. Finally, the Department of Defense verification, validation, and accreditation process is proposed as a method to ensure that any future AI application applied to wargames are suitable.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

AI	artificial intelligence
CNN	convolutional neural network
COA	course of action
CTCE	centralized training, centralized execution
CTDE	centralized training, decentralized execution
DDPG	deep deterministic policy gradient
DOD	Department of Defense
DotA	Defense of the Ancients
DQN	Deep Q-Network
DTDE	distributed training, decentralized execution
EoF	economy of force
FMH	feudal multi-agent hierarchies
HRL	hierarchical reinforcement learning
MARL	multi-agent reinforcement learning
MDP	Markov decision process
ML	machine learning
M&S	modeling and simulation
OODA	observe, orient, decide, act
OPFOR	opposing force
PLA	People’s Liberation Army
PPO	proximal policy optimization
RAI	responsible AI
RL	reinforcement learning
RTS	real time strategy
SB3	Stable-Baselines3
STE	Synthetic Training Environment
TTP	tactics, techniques, and procedures
U.S.	United States
VV&A	validation, verification, and accreditation

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

My thanks to Daniel DeRidder and Wei Ting Goh for helping me understand Atlatl and Stable-Baselines3. Your insights helped me build my agent and test bench. Thank you, MAJ Charlie Timm and Ms. Lauren Callihan, for your hours of help, allowing me to craft an articulate thesis. To my advisor, Dr. Chris Darken, thank you for your thoughts, recommendations, and guidance. This research would not have been possible without you. Finally, to my wife, Kimberly, thank you for your love, patience, and continuous support throughout this endeavor.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Wargames are invaluable training, planning, and research tools for successful militaries. Naturally, the United States (U.S.) Department of Defense (DOD) plans to integrate artificial intelligence (AI) into wargaming. One way to integrate AI into wargaming is to replace the human players with intelligent agents; algorithms capable of demonstrating combat behavior. This thesis examines the feasibility, acceptability, and suitability of replacing human wargame operators with intelligent agents. To do so, this chapter explains why wargames are critical to successful militaries.

A. WHY MILITARIES CONDUCT WARGAMES

Militaries conduct wargames to answer critical questions about warfare that must be understood before actual conflicts arise. Wargames are simulations of actual combat using opposing forces and are shaped by human decisions [1]. While there is a broad spectrum of different types of wargames, they all share a common objective: to "gain valid and useful knowledge" [2]. This delineation is important because the different purposes of the wargames will cause the players and game controllers to behave differently. Figure 1 shows the broad spectrum of wargaming from training to analysis to experimentation.

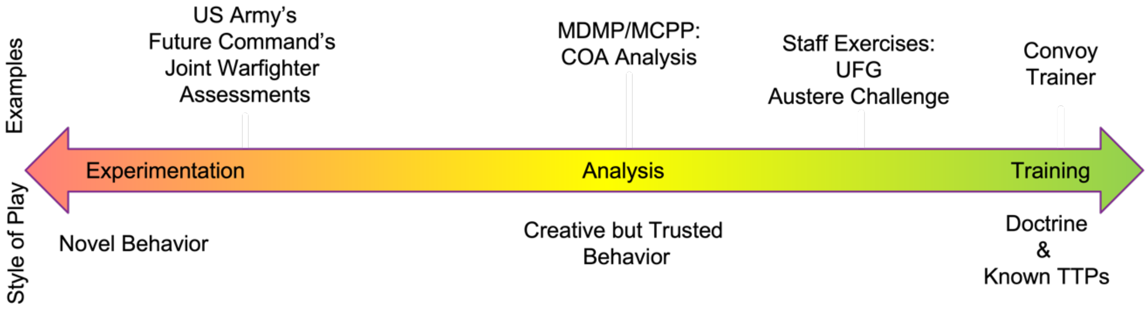


Figure 1. Spectrum of Wargames

1. Wargaming for Training

The most straight forward type of wargames are those for training. Large staffs use constructive simulations (digital wargames) to exercise their staff processes and validate their military readiness. Small crews use virtual simulators to train their battle drills and crew drills. Militaries conduct these wargames to understand warfare and exercise decision-making skills [3]. All players' actions and decisions are generally expected to conform to known doctrine and tactics, techniques, and procedures (TTPs). For large staff exercises, the adversary may push the boundaries of TTPs to challenge the staff (e.g., act more aggressive but still rely on the same TTPs).

2. Wargaming for Analysis

Wargaming may be used for analysis, that is "the determination of what would have happened in a confrontation of forces" [3]. These are the type of wargames most military staffs are familiar with: the wargame conducted as part of course of action (COA) analysis. These types of wargames allow for the assessment of war plans, force structures, or doctrine. In these wargames, both sides are expected to employ known doctrine and TTPs but "within which the creative spirit is free to roam" [4].

3. Wargames for Experimentation

At the other end of the spectrum is experimental wargames. In these wargames, both sides may employ new forces, weapons, and/or tactics to explore potential future wars [5]. Ultimately, organizations conduct experimental wargames to produce "knowledge about the nature of a warfare problem" [2]. The U.S. military integrates these types of wargames in exercises like the U.S. Army's Future Command's Project Convergence and Joint Warfighter Assessment.

4. Benefits of Wargaming

While wargames are neither predictive nor complete replications of reality, they do offer something that cannot be obtained without actual combat: insight into decision-making in war. When wargaming for training, organizations are learning what good decision-making looks like (both the process and final results). When wargaming for analysis, planners are evaluating the decisions they made during planning and what potential decisions need to be made during execution.[1] Finally, wargaming for experimentation gives insight into potential future decisions, specifically those involving future force designs, tactics, and acquisitions.

These benefits were enough that, in 2015, Deputy Secretary of Defense Robert Work issued a memo calling for renewed efforts in wargaming across the DOD [6]. Deputy Secretary Work saw wargaming as beneficial to innovation, risk management, and professional military educations. Eventually, Work believed, wargaming would drive the DOD's planning, programming, budgeting and execution process, the method that informs the DOD's resource allocation. The U.S., and its western allies are not the only militaries to believe in the benefits of wargaming. The Chinese are investing significant resources into wargaming, to include integrating AI into wargaming [7].

B. ARTIFICIAL INTELLIGENCE WITHIN WARGAMING

AI offers a chance to improve military wargaming by reducing costs, speeding up the decision-making process, and offering new insights. Employing human operators for the many roles within wargaming is costly. Organizations must either task their own personnel (taking them away from their normal functions) or pay for external support. This cost can be eliminated by integrating AI into wargames. Wargame analysis can only go as fast as the human operators. Replacing operators with intelligent agents can speed up the wargame and allow for multiple wargames to occur simultaneously, enabling broader analysis. Finally, intelligent agents have been noted for their creativity in game play [8]. Creative agents can enable better analysis of war plans, force formations, or tactics by exploring possibilities that human wargamers might not have considered.

National security organizations within the U.S. recognize the potential for integrating AI into wargaming. The National Security Commission on Artificial Intelligence in their final report advocates for the immediate integration of AI capabilities

[1] A famous example of the importance of wargaming is the failure of the Japanese navy to properly wargame the attack on Midway. They failed to develop a contingency plan for an ambush by the U.S. Pacific fleet even though the possibility was discussed during the wargame [2].

into wargames to ensure the U.S. and its allies remain competitive with its peers [9]. The U.S. Army's future simulation training system, the Synthetic Training Environment (STE) envisions integrating AI to monitor and adjust the difficulty of the training scenarios [10]. The U.S. Army Research Lab has numerous projects investigating the integration of AI into military command and control systems. Specifically, they are exploring the use of a subfield of AI called reinforcement learning (RL) to conduct continuous planning in order to develop "novel plans for Blue Forces" [11]. Continuous planning will require an agent that can evaluate its plans, possibly through wargaming.

Multiple researchers are examining RL agents for wargaming based the success other RL agents have had in human competitive games like StarCraft II [12], the Defense of the Ancients (DotA) [13], and Go [14]. Real time strategy (RTS) games like StarCraft II and DotA best represent wargames. Similar to wargames, RTS games require long term goal planning with short term tactical decisions within a limited information environment. Previous research has demonstrated that RL agents can replicate desired combat behavior within wargames [5], [11]. According to Kania and McCaslin, the Chinese People's Liberation Army (PLA) saw the success of Google's AlphaGo in defeating the world's best Go Master as proof that AI could be applied to wargaming [7]. Since then, the PLA and other Chinese defense organizations have been investing in their AI wargaming capabilities.

C. PROBLEM STATEMENT

While previous research has demonstrated that an RL agent can generate combat behavior, the experiments have been limited to small engagements. The researchers only required the RL agent to control three to five subordinate units. Reinforcement learning agents will need to successfully scale to meet the size requirements for large wargames involving several hundred units.

The problem is that as the number and types of units increases in a wargame, the amount of information and the number of possible moves becomes intractable. Newton et al. proposes scalability as a set of objectives: speed, convergence, and performance while remaining within a set of constraints: cost, compute capacity, and time as the size of the project increases [15]. Hierarchical organization is one approach to scaling. This thesis will investigate the scalability of hierarchical reinforcement learning (HRL). In other words, any feasible and acceptable AI integration into wargaming must still display desirable combat behavior as the number of units in the wargame increases.

Looking beyond the feasibility and acceptability of integrating AI into military wargames, the integration needs to be suitable. It is possible to develop and execute a failed wargame because the knowledge derived from it is invalid or not useful. Weuve et al. [16] explains the different routes that can cause a wargame to fail, which they call wargame pathologies. The design, and implementation, of an intelligent agent, with the purpose of replacing human operators, needs to prevent wargame pathologies, thereby ensuring valid results.

This leads to the following research questions. Does HRL allow intelligent agents to increase the number, and effectiveness, of cooperative units without loss in performance? What framework can ensure that intelligent agents are designed and applied properly to meet the purpose of wargaming?

D. SCOPE OF RESEARCH

This thesis continues past investigation by [17] and [18] of RL within the Atlatl combat environment. Atlatl is a discrete, hexagon based wargame, simulating land combat operations. Initial studies successfully produced combat behavior within RL agents using a simple multi-layer perceptron [17]. Subsequent studies then examined RL agents within complex terrains and dynamic opponents using convolutional neural network (CNN) architecture [18].

While there is a broad range of HRL methods, the focus of this research is on feudal multi-agent hierarchies (FMH). Within FMH, a single RL agent, the Manager, assigns tasks to a collection of subordinate RL agents called Workers [19]. This thesis compared the resources required and effectiveness of employing a rules-based agent, a single RL agent, and a FMH in increasingly larger scenarios within Atlatl.

Wargaming is composed of players and referees [1]. The players for the friendly units are referred to as the Blue Forces, their adversaries are called the Red Forces and any civilians or military units outside of either players are called the Green Forces. While it is possible to fully automate wargaming by using intelligent agents for all the players and the referee, this thesis only evaluated the replacement of a single player.

This thesis also investigates the potential complications that arise when replacing the opposing force (OPFOR), the Red units, with an intelligent agent. The specific wargame pathologies are discussed with the means to mitigate them. The DOD validation, verification, and accreditation (VV&A) framework is applied to the modeling of the OPFOR through RL.

E. FINDINGS

This thesis finds that a FMH agent fails to perform better than a single RL agent when the FMH agent is trained in a distributed manner. The FMH agent's learning improves when both the Manager and Worker train in the same environment. However, the Worker's inconsistent actions prevents the Manager from developing an optimal policy. Additionally, the training requirement for FMH exceeds those for a single RL agent which inhibits FMH's ability to scale to large military wargames. Finally, this thesis finds that methods for integrating AI into military wargames are suitable when processes, like the DOD's VV&A framework are applied. Otherwise, model-based wargame pathologies can invalidate the wargame's objectives with negative consequences for the U.S. military.

F. THESIS CONTRIBUTION TO RESEARCH

This thesis has direct benefit to the U.S. government by furthering research into employing fully autonomous agents within constructive simulations. Fully autonomous wargaming agents that are able to operate at multiple echelons are required to support the full spectrum of wargaming. This easily extends to a decision support tool during military planning by assisting planners in evaluating different COAs rapidly. Additionally, exploring the suitability of using intelligent agents in wargames will facilitate the adoption AI within the wargaming community.

II. BACKGROUND AND RELATED WORKS

This chapter discusses previous research involving the integration of AI into wargaming, specifically RL, while also introducing important terms critical to understanding the problem and potential solutions for integrating AI. First, the chapter breaks down the wargame environment within the context of AI design. After establishing what the wargame environment is, this chapter then looks at advancements AI has made in solving similar challenges including RL. The chapter discusses critical aspects of RL in the wargame environment. This chapter concludes by identifying the gap in RL research regarding scaling methods to meet the needs of military wargaming.

A. WARGAMING ENVIRONMENT

To understand the difficulty of scaling RL to support wargaming, the wargame environment needs to be discussed first. Stuart Russell and Peter Norvig argue that designing the appropriate intelligent agent requires a thorough understanding of the environment [20]. Russel and Norvig [20] claim that partially observable, multi-agent, stochastic, dynamic, and continuous environments are the hardest for an algorithm to achieve human level performance. If [20]'s claim is correct, then wargames are among the hardest environments.

1. Partially Observable Environments

The standard way of thinking about partially observable environments [20] is that in war games, information that is critical for decision-making is limited. In combat, this limitation is commonly referred to as the "fog of war." In wargames, players lack complete information about the current state of the environment, specifically about their opponent. Beyond incomplete information about opponents, friendly units/players may lack information about each other. This lack of information requires players to make assumptions about the environment.

2. Multi-Agent Environments

Large military wargames are multi-player games, consisting of hundreds of cooperative and competitive players with complex interactions which is itself a unique problem. A player must develop a winning strategy while their opponent does the same, which requires coordinating the actions of several players and assessing which of those moves were beneficial or not. Multi-agent RL methods, which seeks to address these unique problems, are discussed in a subsequent section.

3. Stochastic Environments

Effective wargame strategies must account for chance because wargames are often stochastic environments. A stochastic environment is one in which actions do not produce fixed results [20]. Many wargames utilize stochastic environments [1] to reflect the role chance or luck plays in combat [4]. In a classic board game, stochasticity is replicated using the roll of a dice. In modern computer simulations, there are complex probabilistic models to determine if an opposing unit is detected, hit, or the amount of damage it receives [1]. Players need to factor probability into their decision-making or otherwise it can delay learning.

4. Dynamic Environments

It is hard for a player to make an optimal decision because within modern wargames the state of the game is constantly changing. Decisions by other players and outside forces effect the game, disrupting the decision-making process. The state of the game is changing, regardless of any actions taken by the player because, similar to RTS games, its opponent is free to continue taking actions. So the decision a player makes for a given situation may no longer be the current state of play. A player with a faster decision cycle (sometimes referred to as an OODA loop[2]) will be more successful than other players [22].

[2] OODA stands for observe, orient, decide, act. Developed by U.S. Air Force Colonel John Boyd, the OODA loop explained the decision-making cycle of air-to-air combat and has since been extended to decision-making in combat in general [21], [22].

5. Sequential Environments

Player decisions early in the game can have crucial impacts for how the game ends, which AI scholars commonly refer to as sequential decisions [20]. Wargames create sequential environments by limiting available resources, specifically combat strength, fuel, and ammunition. Sequential decision-making differentiates RL from other machine learning (ML) techniques, like a classification task in supervise learning, because each decision an RL agents makes impacts future decisions. RL researchers commonly use the Markov assumption to bypass the problem of sequential environments by assuming the previous state contains all the information needed to predict the next state [23]. How RL researchers apply the Markov assumption is discussed in the section on RL basics.

6. Continuous Environments

Military wargames require players to process a large amount of information while selecting from a large number of possible decisions. Generally, when wargames are played in a constructive simulation the resulting environments are high-dimensional continuous observation and action spaces [5]. A discrete space is one with a limited number of options (e.g., chess), while continuous is the opposite (i.e., an infinite number) [20]. In a continuous environment, it is impossible for a player to conduct an exhaustive search for the best decision, that is, the player cannot try every possible move or experience every possible state no matter how much time and computational power the player has available [20].

B. ADVANCEMENTS IN AI COMPETITIVE GAMING

Today’s RL methods for military wargames are built on the approaches developed for simpler competitive games. Figure 2 shows the achievement of AI programs over select games over time. This section briefly covers the history of AI within competitive gaming to explain why researchers are currently relying on RL.

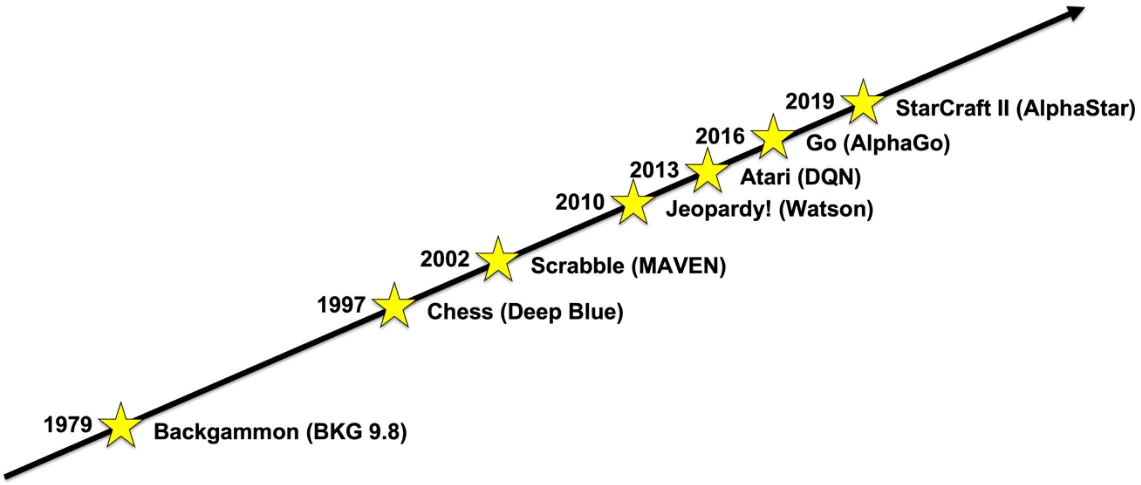


Figure 2. Select Wins of AI Algorithms Over Human Experts. Adapted from [24], [25].

1. Early Efforts Competitive AI Approaches

Early attempts to create human-level competitive AI for simple games involved searching while expert systems were built for increasingly complex ones. Many current AI methods [26] still use heuristic search methods, which involves an effective and efficient search of all the possible moves to determine which is the best one to make. Richard Sutton argues in his essay “A Bitter Lesson” [27], that heuristic search methods are preferred options over complex algorithms, like expert systems, due to the continual increases in computational speed and capacity which makes searching quicker than complex algorithms in the long term.

While expert systems offered a way to simplify or bypass search methods for complex problems, they ultimately failed to meet the challenge. Expert systems utilize rules based on knowledge of the domain to elicit the desired behavior. Nils Nilsson [25] argues that these systems failed because they were “brittle,” that is, these programs struggled or failed when they lacked the requisite knowledge needed for a change to the environment. After the failures of expert systems, interest in the AI community eventually moved toward RL.

2. RL in Gaming

Researchers have used RL methods to make effective game playing agents by leveraging neural networks and the same computational capabilities that make "Big Data" possible. The success of neural networks, combined with the availability of large amounts of data, has led to the current "AI Summer"—the emphasis on AI in the academic, government, and commercial sectors [20]. Russel and Norvig [20] explain RL as simply learning through a series of rewards and punishments. Within the last decade, AI agents became competitive in RTS games like StarCraft II [12] and DotA [13], while others mastered the difficult game of Go [14], [26].

3. RL in Wargaming

Recently, researchers applied RL methods for military wargames. Jonathan Boron [17] extended the achievements of RL agents from competitive RTS games to the Atlatl wargame environment. Using a simple neural network, Boron was able get RL agents to exhibit ideal combat behavior, specifically mass and economy of force [17].[3] Christopher Cannon and Stefan Goericke [18] expanded on Boron's work by using convolutional neural networks (CNNs) to develop combat capable RL agents with spatial invariance. Spatial invariance meant these agents could effectively manage a variety of unit and terrain types by passing the agents' observations through a CNN to extract the most relevant information about the environment [18]. Details about the Atlatl wargame environment and CNNs are discussed further in Chapter III. Finally, the U.S. Army Research Lab ported a military training scenario into the StarCraft II simulation environment and trained several RL agents successfully [11]. This research shows that RL can operate in the wargame environment, at a small scale.

C. REINFORCEMENT LEARNING

This section discusses RL fundamentals, the types of RL algorithms used for this thesis, the basics of RL for multiple agents, and a brief introduction into HRL.

[3] Mass is the military principle of concentrating combat power to achieve decisive results, while economy of force (EoF) is the principle of using the minimum essential amount of combat power to secondary objectives to ensure mass can be applied to the primary objective [JP 3-0].

1. RL Basics

The basic components of RL is the state of the environment (observation space), possible actions (action space), and a reward function [23]. Figure 3 shows the relationship between the agent's actions and its environment and rewards. RL can be described by a Markov decision process (MDP), a 5-tuple that Richard Bellman developed in 1957 to describe the relationship between the state and action space, and reward function [20]. The MDP equation is

$$MDP =< S, A, P, R, \gamma > \tag{1}$$

where S and A is the state and action space, respectively; $P: S \times A \rightarrow [0,1]$ is the state transition function, describing the probability of arriving in state, S_{i+1} from taking action, A_i, in state, S_i; $R: S \times A \times S \rightarrow \mathbb{R}$ is the reward function based on the actions taken at a given state; and $\gamma \in [0,1)$ is the discount factor [28]. The complete mapping of state to action is called a policy [23], as seen in Figure 4. The goal of RL is to find the optimal policy, that is, the policy that achieves the greatest total reward [29]. The key to the MDP is the Markov assumption; the only information needed to make a decision is encoded in the current state. The Markov assumption overcomes the sequential decision-making problem because the results of past decisions becomes part of the state.

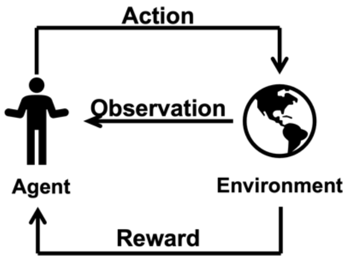


Figure 3. Interaction of an RL Agent and Its Environment. Adapted from [30].

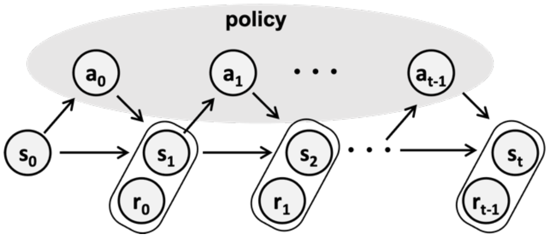


Figure 4. A Policy Mapping All States to Actions.
Adapted from [31].

The design of the reward function, *R*, significantly effects the performance [32] and behavior [33] of an RL agent. By changing the reward function, Tampuu et al. [33] caused RL agents playing Pong to go from cooperative to competitive behavior. We can generalize the reward value, R_t, for a given action as the immediate reward, r_{t+1}, and all subsequent rewards as the following:

$$R_t = \sum_{i=0}^{T-t-1} \gamma^i r_{t+i+1} \tag{2}$$

where T is the total timesteps remaining and γ is the discount factor [34]. Changes in the discount factor, γ, also changes the performance of an RL agent. In their seminal textbook on RL, Richard Sutton and Andrew Barto [23] explains that $\gamma=0$ causes the agent to only consider the immediate reward and behaves in a myopic manner. As γ approaches 1, then the agent behavior becomes more far-sighted [23].

2. Types of RL

Some RL methods perform better than others in wargame environments due to the games' properties. RL methods for wargaming rely on model-free methods, since models of the game are unavailable [5]. Sutton and Barto summarize the differences sufficiently, "model-based methods rely on *planning*, while model-free methods primary rely on *learning*" [23]. The two model-free RL algorithms used throughout this research were proximal policy optimization (PPO) [35] and Deep Q-Network (DQN) [36]. With model-free approaches, RL agents learn the state transition function, *P*, from Equation (1) by interacting with their environment. Both approaches rely on neural networks to approximate the state transition function. While PPO is a policy optimization approach, DQN is a Q-Learning approach.

a. DQN: Q-Learning

Q-Learning, particularly DQN, has demonstrated human-level performance in a variety of tasks but is restricted in its application. Q-Learning is based on the concept that, through trial and error, the agent learns the approximation of the action-value function (q_*), of the different states in the environment, that is, which actions lead to the greatest reward from a given state [23]. Equation 3 defines the action-value function, q_π,, for a given policy π, as:

$$q_\pi(s,a) = \mathbb{E}_\pi[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t = s, A_t = a] \qquad (3)$$

which is the expected discounted reward for following policyπafter taking action a at state s. If an agent follows the optimal policy, π_*, the policy with the greatest reward, then q_*, is it action-value function.

The most popular Q-learning approach is DQN, which Mnih et al. [36] used to reach human level performance on several Atari video games. In DQN, as an agent explores its environment it populates its experience into its replay buffer [36]. It samples from its replay buffer and uses deep neural networks to calculate Q-values for a given state [36]. With updated Q-values it is able to formulate an optimal policy through iteration [23]. While DQN develops a policy by iteratively improving its estimate of the action-value function, PPO develops its policy directly through policy optimization.

b. PPO: Policy Optimization

Policy optimization provides more flexibility since, unlike DQN, it can handle large continuous action-spaces. Policy optimization, or policy gradient method, works by increasing the probability of the actions that return high rewards based on the logic that these actions lead to optimal policy [23]. Instead of calculating an action-value, a better estimate of the quality of the action can be obtained by checking if returns are better than expected, called an advantage function [35]. Sutton and Barto [23] assert that working directly with policies is crucial when operating in a continuous environment (or a high-dimensional action space in a discrete environment) because the model can learn a probability distribution of actions. Probability distribution is preferred for large action spaces, particularly continuous spaces, like those found in many constructive simulations, since it precludes the need to learn probabilities for each action [23].

Several policy optimization algorithms have been successfully used in military constructive simulations [17], [18], [11]. The main agent of this research used a policy optimization method due to the agent's large action space. For this thesis, PPO was used due to its stability through gradient clipping. Gradient clipping ensures that the model does not make drastic changes to its policy during each update [35].

3. Multi-Agent Reinforcement Learning

Since large military wargames are essentially multi-agent environments, any RL method used should address the problems inherent to a multi-agent environment. In a survey of MARL research, [28] identified five challenges to learning in a multi-agent environment: 1) non-stationarity, 2) credit assignment, 3) communication, 4) coordination, and 5) scalability. Each challenge is examined with a deeper discussion of scalability in the next section.

a. Non-stationarity

With multi-agent environments, a current effective strategy may become ineffective in the future. A constantly changing environment is commonly referred to as a nonstationary problem (or non-stationarity) [20]. Non-stationarity occurs when the same action fails to generate the same result because the changing nature of the problem [23]. Non-stationarity is different from stochasticity because a probabilistic model that maps actions to results can be developed for stochastic environments [20]. Non-stationarity occurs in multi-agent competitive environments, like wargames. As one agent learns an effective strategy to defeat its opponent, the opposing agent is improving its own strategy [20]. Non-stationarity also develops in cooperative environments. As each agent's strategy evolves, partners cannot anticipate each other's actions anymore [28]. In a survey of non-stationarity within multi-agent environments, [37] reported that Q-learning, like DQN, cannot converge to an optimal solution because the Markov assumption is no longer true.

b. Credit Assignment

Developing an optimal strategy is hard in multi-agent environments because it is difficult for evaluating strategies. It is difficult to assess which agents made good moves and which did not. In a complex problem, involving a multitude of decisions, it is difficult to assign positive or negative credit to each decision, which [38] refers to as the credit assignment problem. For multi-agent environments, the credit assignment problem extends to knowing which of the individual agents' actions were positive or negative [28]. In their survey of MARL research, [28] noted that "an additional challenge arises when agents have only access to local observations of the environment." Its reasonable, therefore, to limit an agent's rewards to only its actions and its observations of the environment.

c. Communication and Coordination

With multiple agents, communication and coordination is an integral component to developing an effective strategy. Military history has shown that units who coordinate their actions[4] are the most successful in combat [4], [39]. How agents train together and how they choose (or are assigned) their actions impacts their level of coordination [28]. Oftentimes, learning how to properly communicate critical information and coordinate actions is an important objective of a wargame [40], [2]. How MARL agents train and execute drives their level of coordination.

d. MARL Training and Execution Spectrum

The training and execution scheme for an RL agent within a wargame is a critical factor since there are multiple friendly units within the game competing against an opposing agent. How the agents train determines how well they manage the problems of non-stationarity and credit assignment and how they learn to coordinate and communicate. The various training and execution schemes can be organized into three categories:

[4] Coordination is viewed as the synchronizing of efforts (or capabilities) to achieve the greatest effect [39].

centralized training, centralized execution (CTCE), distributed training, decentralized execution (DTDE), and centralized training, decentralized execution (CTDE) [28], which can be seen in Figure 5. On the spectrum of MARL training and execution schemes there is, on one end, a single policy which controls the action of all agents in the environment (CTCE) and on the other end, multiple agents learning and executing their own policies (DTDE) [30].

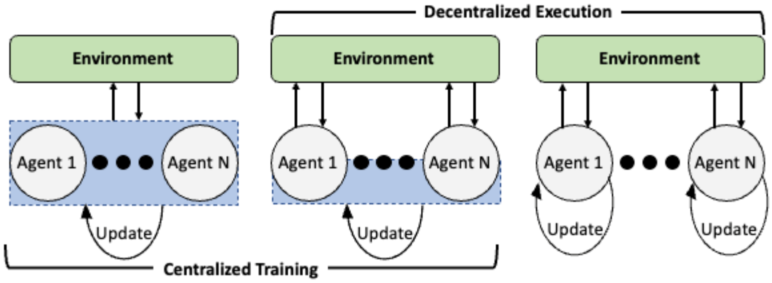


Figure 5. Types of the MARL Training and Execution Schemes. Adapted from [28].

(1) CTCE: Centralized Training, Centralized Execution

CTCE is when a single policy is updated during training and executed by a central unit that determines the action for all subordinate agents [28]. CTCE was used for previous RL wargaming research [17], [18], [11], as well as DeepMind's AlphaStar [12]. In these cases, despite the number of units within the simulation, it is a single decisionmaker that is trained and deployed. A significant problem with CTCE is that the action space grows exponentially with the number of agents [28]. Additionally, CTCE relies on global rewards, "which represents the entire group's performance" vice a local reward unique to each agent [37]. Global rewards can lead to a credit assignment problem without additional means to determine which agent's actions contributed to the global reward [37]. Therefore, other training and execution strategies should be explored.

(2) DTDE: Decentralized Training, Decentralized Execution

DTDE is the opposite end of the spectrum: each agent develops an individual policy and determines its actions from its own policy [28]. A non-stationary environment is a significant problem for DTDE. As each agent explores its environment, the other agents appear as part of the environment. As the agent develops a policy for the environment, the other agents are also adapting their policies, therefore changing the environment. In addition to the non-stationary problem, [41] has shown that decentralized execution can have difficultly scaling to larger number of agents.

(3) CTDE: Centralized Training, Decentralized Execution

CTDE overcomes non-stationarity because the agents learn together and the consequences of an action can be attributed to the appropriate agent [28]. CTDE works best when the agents are homogeneous, that is, when they share a common neural architecture. Homogeneous agents in CTDE can share the updates to their policy parameters and accelerate their learning. Homogeneous agents can also employ centralized value functions for RL methods that utilize them, (e.g., DQN). Lin et al. [42] showed that CTDE can provide flexibility and efficiency to large-scale MARL problems by employing a modified version of combined Q-Learning/Policy Optimization algorithm with a single centralized critic.

4. Hierarchical Reinforcement Learning

HRL can be seen as a unique subset of MARL and has been explored as a means to improve coordination among agents. Peter Dayan and Geoffery Hinton originally develop HRL in 1993 for single agent application. HRL has since been applied to MARL [28]. HRL is when “multiple layers of policies are trained to perform decision-making and control at successively higher levels of temporal and behavioral abstraction” [43]. In a recent survey of HRL approaches, Pateria et al. [44] developed a taxonomy of six HRL categories along 3-dimensions. One category of HRL uses a single agent to solve a single task without subtask discovery. Pateria et al. [44] call this category “Learning Hierarchical Policy” and can be broken into two divisions: Feudal Hierarchy and Policy Tree approaches. This thesis explores the Feudal Hierarchy approached that had been extended to MARL.

5. Feudal Multi-Agent Hierarchies

In feudal hierarchy approaches, there is a hierarchy of policies. The highest policy consists of subgoals which the lower policies try to achieve, which may be further subgoals. The lowest policy is composed of primitive actions. In developing the concept of feudal networks for HRL, Vezhnevets et al. [19] of Google DeepMind, considered these different policy levels as sub-agents. A higher policy sub-agent is considered the "Manager" while the lower policy sub-agent is the "Worker" [19]. The Worker operates at a faster time scale and "produces primitive actions, conditioned on the goals it receives from the Manager" [19]. In a wargame, the primitive actions are shooting and moving. The higher policies are the decisions of where to move and who to attack.

Sanjeevan Ahilan and Dayan, inspired by [45] and [19], extended HRL to the multi-agent environment with feudal multi-agent hierarchies (FMH) [46]. FMH used a CTDE approach for developing and deploying agents to account for non-stationarity [46]. The Manager is responsible for coordinating the actions of the Workers [46]. In their experiments [46], concluded that FMH sufficiently scaled as the number of Workers increased. Ahilan and Dayan [46] concluded that FMH successfully addressed multi-agent problems of non-stationarity, coordination, and scalability; critical aspects of designing an agent for large military wargames. Figure 6 shows the relationship between the Manager, Worker, and the environment. The Manager issues goals, g, to the Workers who take action, a. Both Manager and Workers receive observations, o, from the environment. The Manager also receives a reward, r, from the environment, while the Workers receive locally calculated rewards.

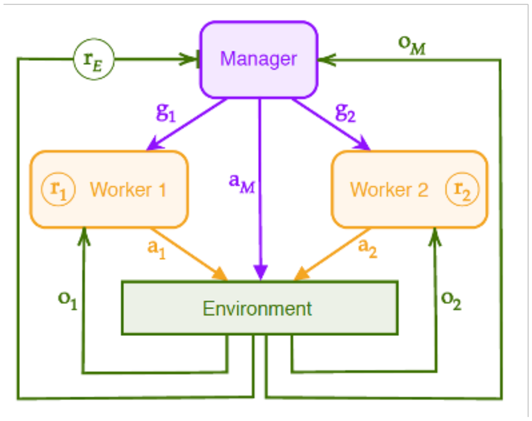


Figure 6. The FMH Architecture. Source [46].

D. SCALABILITY

There is a gap in research in how to scale AI implementations to support large wargames [15], [28]. Failing to account for scale may make some of the prior RL approaches in wargames impractical. There are three metrics to measure scalability: performance, latency, and accuracy. For performance the metric is whether the system can simulate the number of units, or entities, (and their interactions) as advertised. It should be "measured as the number of independent entities that can be simulated concurrently while meeting acceptable latency and accuracy metrics" [15]. Latency is important because some wargames rely on modeling and simulation (M&S) systems with specific latency guarantees in order to federate into a larger system [15]. Accuracy is the degree of realism; a comparison of the real world to the simulation. Does the simulation offer the required accuracy as it scales. For example, properly adjusting from individual tactics to large unit maneuvers as the scale of the simulation increases.

As the number of units increase, the demands on the RL algorithm and MARL training and execution scheme should reasonably scale. MARL research has shown that as the number of agents increases, the size of the joint action-space increases exponentially. An exponentially increasing action space can be partially mitigated by utilizing the exponential decay property (also known as correlation decay or spatial decay) [29]. The exponential decay property refers to the fact that the influence of an agent decays exponential from its graph distance to another agent. In other words, the behavior of a subordinate unit impacts the other agents within the same hierarchy more than it does with agents in a different hierarchy.

E. SUMMARY

RL agents have achieved human level performance in the types of environments found in wargames: partially observable, stochastic, dynamic, sequential, and continuous environments. Wargames are inherently multi-agent and suffer from the following problems: nonstationary, communication and coordination, credit assignment, and scalability. HRL, through the use of abstractions, offers a potential solution to the problem of scale. A specific type of HRL, FMH, uses Managers and Workers to divide high-level tasks like combat behavior from low-level primitive actions like shoot and move. The next chapter will discuss the experimental design for testing the feasibility of FMH.

THIS PAGE INTENTIONALLY LEFT BLANK

III. METHODOLOGY

This thesis conducted a qualitative analysis of the feasibility and acceptability of FMH to scale to large wargames. This chapter explains the investigation, starting with the training and evaluation environment. The chapter then describes how the wargame environmental properties drove the FMH agent design and how the particulars of the training and evaluation environment, Atlatl, drove the experimental design and procedures employed.

A. TRAINING AND EVALUATION ENVIRONMENT DETAILS

Training and evaluation of agents were conducted in the Atlatl wargaming system designed by the MOVES Institute at the Naval Postgraduate School for AI research. Atlatl is a two-sided, turn-based, wargame played on a hexagon-based map that utilizes a simple combat model and software infrastructure to support experimentation [47]. Significant components of the wargame are the units, terrain, combat modeling, and scoring.

1. Units within Atlatl

The two opposing sides are represented as either "Red" or "Blue" forces. For this thesis, every tested agent was observed while controlling the Blue forces. There are four unit types in Atlatl: infantry, mechanized infantry (referred to as "mechinf"), armor, and artillery units. The units are displayed using MIL-STD-2525D as seen in Figure 7. The different unit types have different movement speeds depending on the terrain.

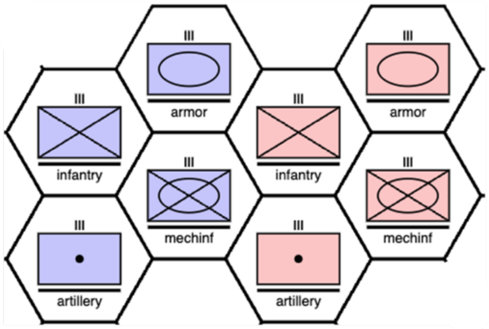


Figure 7. Blue and Red Unit Types in Atlatl

2. Terrain and Terrain Effects within Atlatl

There are six terrain options: clear, rough, marsh, urban, unused, and water as shown in Figure 8. Each terrain type effects the various unit types differently [47] as shown in Table 1. Generally, a unit can only move one hex per phase, except for clear terrain where mechinf, armor, and artillery units can move two hexes per phase. Artillery is unable to move into a marsh hex and no unit can enter an unused or water hex.

clear
urban
rough
marsh
water
unused

Figure 8. Terrain Types in Atlatl

Table 1. Atlatl Terrain Types and Effects by Unit Type. Adapted from [47].

		Movement Cost to Enter Hex			
		Clear	**Urban**	**Rough**	**Marsh**
Unit Types	**infantry**	1	1	1	1
	mechinf	1/2	1	1	1
	armor	1/2	1	1	1
	artillery	1/2	1	1	N/A

3. Combat within Atlatl

Combat within Atlatl is deterministic; the only contributing factors in combat are unit strength, unit type, and terrain. The amount of combat power (or strength) a defender loses is based on its unit type, the terrain it occupies, the attacker’s type and the attacker’s combat power in accordance with following equation:

$$Loss\ =\frac{1}{2}Cbt\ Pwr_{Attacker}\ x\ FT(Attacker, Def)\ x\ DT(Def, Terrain) \qquad (4)$$

where Loss is the amount of combat power the defender losses, $Cbt\ Pwr_{Attacker}$ is the attacker's current strength, FT is the modifier found in the Attacker-Target Firepower Table found in Table 2, and DT is the modifier found in the Defender-Terrain Table found in Table 3.

Table 2. Attacker-Target Firepower Table (*FT*). Adapted from [47].

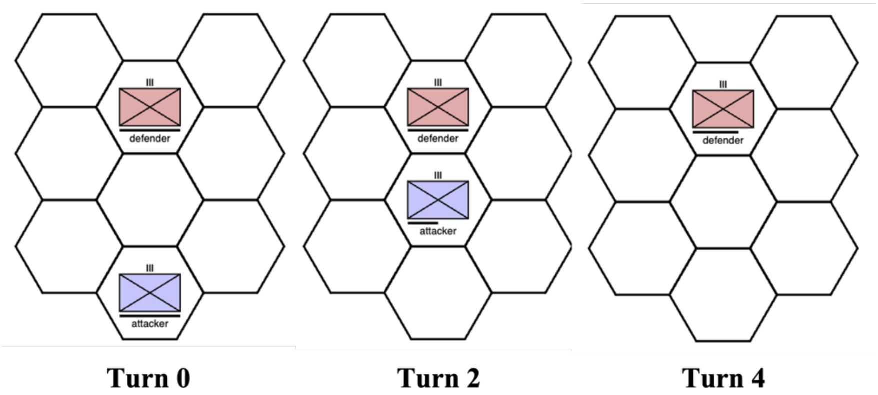
		Target Unit Type			
		infantry	**mechinf**	**armor**	**artillery**
Attacker Unit Type	**infantry**	1	1	0.5	1.5
	mechinf	1	1	1	1.5
	armor	0.5	0.75	1.0	1.0
	artillery	1	0.75	0.5	1.5

Table 3. Defender-Terrain Table (*DT*). Adapted from [47].

		Target's Terrain Hex			
		clear	**rough**	**marsh**	**urban**
Defender Unit Type	**infantry**	1	0.5	1	0.5
	mechinf	1	1	2	1
	armor	1	1	2	1
	artillery	1	1	2	1

Anytime an entity's strength is less than 50%, the entity is considered combat ineffective (or "killed") and removed from the wargame [47]. All units start with a strength of 100 points in standard scenarios. So, based on Equation (4), one Red infantry unit attacking a Blue infantry unit that is occupying a clear hex, inflicts 50 points of damage. The Blue unit could be eliminated in two moves. Similarly, a Red armor unit of 100 points, attacking a Blue infantry unit in rough terrain would deal 25 points of damage. The Blue unit could be eliminated in three moves.

Atlatl closely replicates U.S. military doctrine [48] that the defender has the advantage in land combat. When two equal units (same unit types, terrain, and strength) attack each other, the first unit to shoot will kill its opponent first. Except for artillery, units in Atlatl can only attack units in adjacent hexes. Since a unit can only move or shoot in a given turn, an attacking unit must move to a hex adjacent to its target first, then wait till its next turn to attack. This required move gives the defender the opportunity to shoot first. Figure 9 illustrates the advantage the defense holds. Artillery units can shoot at targets two hexes away. Combat plays a significant role in Atlatl because it is often used for scoring.



Blue unit moves to attack the Red unit (left), Red shoots first and reduces Blue strength by 50 points (middle), Blue can return fire once but is eliminated by Red on the next turn (right).

Figure 9. Defenders Have an Advantage in Atlatl

4. Scoring within Atlatl

There are two ways to score points in Atlatl: combat and the control of key terrain (urban hexes). The amount of combat damage dealt to the defender is awarded to the attacking unit [47]. If the unit is eliminated, all of its remaining combat is awarded to the attacking unit, regardless of how much damage was dealt [47]. If a player occupies an urban hex, that player now controls that city and receives the number of points associated with that city at the end of every phase [47]. Even if the unit moves out of the urban hex, it is still in control of that city until an opposing side occupies it. Users can load multiple cities per scenario.

Users can pre-build scenarios to meet their needs. Options for a scenario are size, units, scoring, and length of game. Users can set the size of the map (rows x columns) and terrain type of each hex (to include the number of cities). Users can choose the number of Blue and Red units, their type, starting strength, and starting positions. Additionally, users can modify the combat scoring system. Users can modify how many points Blue loses for every loss in Blue combat power (Blue can only gain[5] an equal number of points for every loss in Red combat power). Users can also change the city scoring system, that is the points awarded for control of a city. Finally, users specify the number of phases (or turns) in a game, which starts on Phase 0.

B. IMPLEMENTATION

While Atlatl is the training and evaluation environment, two more "wrappers" are needed to integrate RL into the wargame: OpenAI Gym and Stable-Baselines3. These wrappers are Python library packages used to help users implement basic RL tasks and complex algorithms. Understanding the benefits and limitations of these wrappers is necessary for understanding the design of the FMH approach used in this thesis.

1. OpenAI Gym Environment

The OpenAI Gym environment allows RL researchers to employ the basics of RL without having to develop the codes themselves. OpenAI developed and released the OpenAI Gym package in 2016 because they believed the greater RL community needed "good benchmarks on which to compare algorithms" [49]. The OpenAI gym package is focused on providing a common environment and provides researchers a common set of observation spaces, action spaces, reward method, and timestep method [49]. OpenAI allows researchers to focus on developing and integrating agents into these environments, instead of building environments for their agents. Having an open source RL environment ensures that performance of the agent is based on the agent's algorithm and not its environment, enabling experimental results to be repeatable.

2. Stable-Baselines3

Similar to OpenAI, Stable-Baselines3 (SB3) provides researchers a means to implement popular RL algorithms without having to develop the codes themselves. The creators of SB3 wanted to provide "an open-source framework implementing seven

[5] There is only a single score for each game, so points in favor of the Blue side are positive, while points in favor of the Red side are negative. For example, if control of a city is worth 5 points, then every phase that Blue controls the city is +5 point to the total score, while every phase that Red controls the city is -5 points to the total score.

commonly used model-free deep RL algorithms" to ensure RL researchers could compare algorithm performance to a common baseline [50]. Comparable to OpenAI's intentions, SB3 ensures agent performance is repeatable.

3. Constraints of Training and Evaluation Environment

The combination of the Atlatl wargame, with OpenAI Gym environment and the SB3 RL package places constraints on the design of the FMH and the experiments themselves. Figure 10 shows the relationship formed by using this combination. Replicating past thesis research on RL within wargaming is the overriding factor for using this arrangement. Using this combination also made it easier to build the FMH and test it. That being said, the limitations of the OpenAI Gym action and observation spaces and available RL algorithms within SB3 resulted in a FMH different than the one designed by Ahilan and Dayan [46].

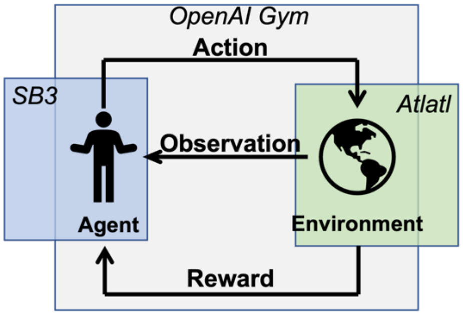


Figure 10. Relationship Between Atlatl, OpenAI Gym, and SB3

C. DESIGN OF FMH FOR ATLATL

FMH consists of a Manager sub-agent and multiple Worker sub-agents. This section explains the design choices for both the Manager and the Worker sub-agents for a FMH agent within the Atlatl wargame.

1. Design of the Manager

The Manager sub-agent was designed to exhibit combat behavior similar to past thesis research [17], [18] into RL agents within wargames. The Manager produces combat behavior through goals the Manager establishes for its subordinate Workers. The Manager has a shallow but wide observation space, a multi-discrete action space to handle the large number of goals it can generate, and the same reward function used in past Atlatl RL agents.

This thesis designed the observation space for the Manager sub-agent to process select elements of the entire state of the wargame to give the Manager broad information to support decision-making across space and time. The Manager's observation space is a tensor of size <4, C, R> where C and R is the number of columns and rows in the loaded scenario, respectively. All values are represented as real numbers. The first layer consists of the location of urban hexes, the next layer contains the location and percent strength of all the friendly forces, the third layer consists of all the hostile forces' locations and percent strength, and the last layer is the percentage of the game remaining.

The Manager's actions space are just hexes on the board, which the Workers translate as goals. The Manager's action space is an array of discrete numbers, limited by the size of the map. The size of the array is twice the number of subordinate Workers. The Manager's actions space is $A \in [C_1, R_1, \ldots, C_n, R_n]$, where $C \in [0, number\ of\ Columns\ -\ 1], R \in [0, number\ of\ row\ -\ 1]$ and n is the number of Workers.

The Manager's action space is based on a balance of benefits and costs of using multi-discrete values. This action space simplified the mapping from action space to map location. Figure 11 shows the relationship between the Manager's action space and the Workers' targets. Since Q-learning cannot predict multiple values, this thesis used the PPO algorithm provided by SB3. Using a Q-learning approach, like DQN, which predicts a single integer value, would require an additional function to map a single discrete value to hexes and workers, potentially introducing errors. Additionally, the Manager's action space is size $R\ x\ C\ x\ N$, so the action space grows linearly with the size map and number of workers which is good for scaling. Unfortunately, this action space does not allow the Manager to change the movement order of the Workers as described later in the sub-section on sequencing units.

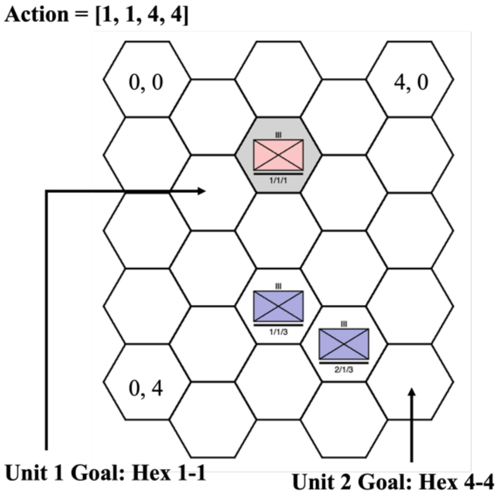


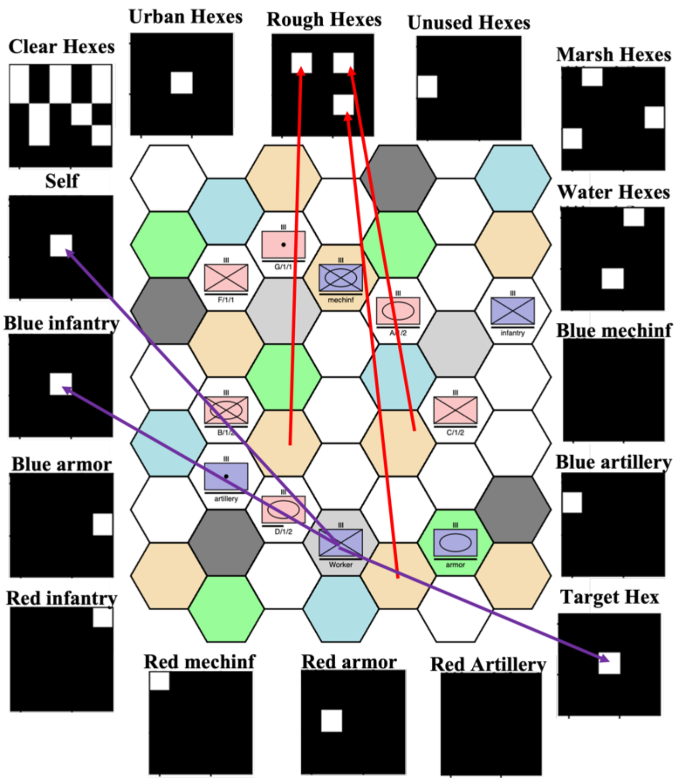
Figure 11. Translation of Manager's Goal to Atlatl Hexes

The same reward function was used for all top-level agents in the experiment, which is based on a reward function made specifically for Atlatl. The reward function is the same as the BoronRewArt reward function from [18]'s research in Atlatl and is based on the work by [17]. As seen in Equation (5), the BoronRewArt function can only be positive. If the difference in the Atlatl game score is greater than the previous timestep, then the difference is multiplied by the fraction of the friendly units' current total strength divided by the friendly units' starting total strength. The reward function is designed to encourage combat while conserving strength and has proven to generate combat behavior [18].

$$reward \ = \ max[0, (difference\ in\ score\ x\ \frac{current\ total\ strength}{starting\ total\ strength}] \qquad (5)$$

2. Design of the Worker

The Worker's observation space is designed to be more narrow but deeper than the Manager's observation space, while also map size independent. The observation space is a tensor of size [16, 5, 5]. The first 14 layers are one-hot encoding of the terrain types and friendly/hostile unit types. Each layer is dedicated to a terrain or unit type (by affiliation). For the terrain layer, if the hex matches the specified terrain, then the hex is represented as a 1, otherwise 0. For the unit layer, if the hex contains the specified unit type and color, then the hex is represented as a percent of the unit's strength, otherwise 0. The next layer represents all the hexes that the unit can move to or attack. The final layer represents where the Worker's target hex is located, that is the goal established by the Manager. Figure 12 shows the mapping from Atlatl to the Worker's observation space.



The Worker is a Blue infantry unit and is occupying an urban hex which is also its target. Since the observations are centered on the Worker, the Self, Urban hex, Blue infantry, and Target hex layers have a highlighted (white) center. Any observations beyond the map's boundaries are padded "0" and shown as black.

Figure 12. Mapping of Worker's Observation Space

The Worker's action space is designed to be translated into the primitive actions of Atlatl: shoot, move, or pass. The infantry Worker's action space is a discrete value from 0 to 6, inclusive. The mechinf, armor, and artillery Workers' actions spaces are discrete values from 0 to 18, inclusive. The values represent the hex number that a Worker can move to. If an opposing unit occupies the hex, the unit attacks, otherwise it moves. 0 is take no action. Figure 13 illustrates the mapping of action space to Atlatl actions. Since the

Worker's action space is discrete values, which works well with Q-learning, this thesis used SB3's DQN algorithm.

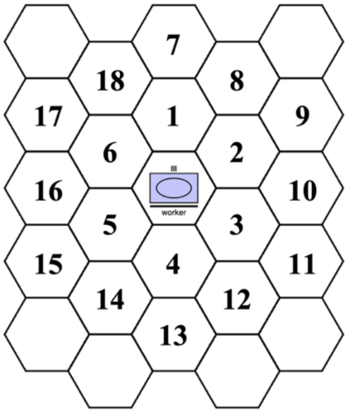
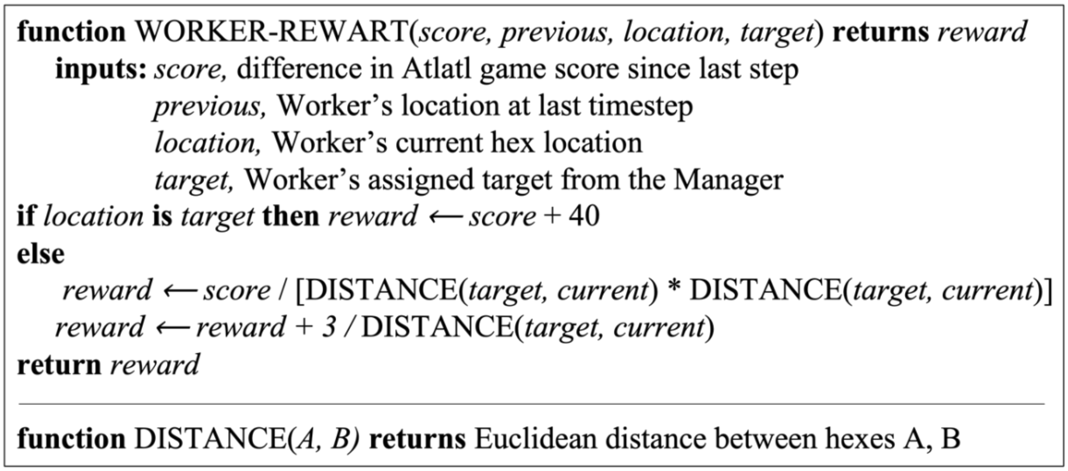


Figure 13. Mapping of Worker's Action to Nearby Hexes

The Worker's reward function is similar to the BoronRewArt function while also encouraging movement to the target. Figure 14 shows the pseudocode for the Worker's reward function. For each time step, the Worker receives a reward for reaching the target. The Worker is rewarded if it successfully engages in combat, based on how close it is to its target. Otherwise the Worker receives increasingly larger rewards for getting closer to its target.

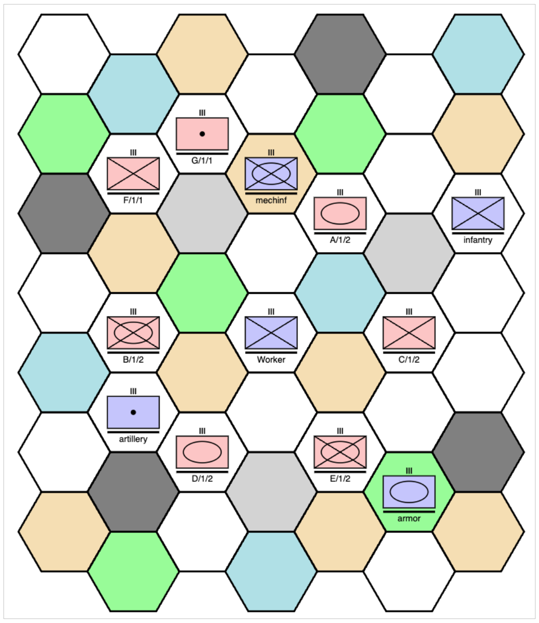


```
function WORKER-REWART(score, previous, location, target) returns reward
    inputs: score, difference in Atlatl game score since last step
            previous, Worker's location at last timestep
            location, Worker's current hex location
            target, Worker's assigned target from the Manager
if location is target then reward ⟵ score + 40
else
     reward ⟵ score / [DISTANCE(target, current) * DISTANCE(target, current)]
    reward ⟵ reward + 3 / DISTANCE(target, current)
return reward

function DISTANCE(A, B) returns Euclidean distance between hexes A, B
```

Figure 14. Pseudocode for Worker's Reward Function

The Worker's training environment is almost as important as the Worker's reward function. During training, an RL agent is learning the optimal policy, i.e., the mapping of state to actions that returns the greatest reward [23]. If the RL agent encounters a state during evaluation that is completely dissimilar from any state transitions the agent encountered during training, then the agent's behavior will be unpredictable. Therefore, the training environment should be robust enough to present state transitions similar to the evaluation environment. The Worker's training environment is shown in Figure 15. The environment contains a mixture of terrain types and friendly and hostile units. A Worker neural network was trained for each type of unit, so that the Worker's actions, for that unit type, were optimized for the strength and weaknesses of each unit type.[6] The Worker received a randomized target during each episode and trained for 1.5 million steps.



While the current Worker depicted is an infantry unit, the same environment is used for all unit types.

Figure 15. Worker's Training Environment

[6] For example, while the artillery unit has greater movement and combat range than the infantry unit, artillery units cannot enter marsh hexes.

D. DESIGN OF EXPERIMENT

This thesis explores the ability of FMH to continue to exhibit combat behavior while the size and complexity of the scenario grew. The thesis also investigates scalability. Specifically, the thesis evaluates training and execution requirements as the size of the scenarios increases. Other agents and a variety of scenarios were required to investigate the issues of feasibility and scalability. FHM is compared against two different agents. The performance of the three agents are tested in increasingly complex environments.

1. Comparison Agents

FMH performance in a series of complex scenarios is compare against a rules-based agent and a single RL agent. This section explains the details of the two comparison agents.

a. Rules Base: pass-agg

The first comparison agent is a conventional rules-based agent referred to as *pass-agg*. The *pass-agg* agent follows a decision tree to determine which action to take. At the start of each turn, the agent assumes an "attack" or "defense" posture based on a comparison of friendly to hostile unit strengths. If the total current friendly unit strength is greater than or equal to the total current hostile strength, then the *pass-agg* agent assumes an "attack" posture. The agent will direct a subordinate unit to attack a hostile unit, if possible, regardless of posture, when a hostile unit is within range. If there are no hostile units to attack, then the agent will move units based on rules given its current posture.

b. Single RL Agent: AI16

The second comparison agent is an RL agent similar to the one used by [18]. *AI16* is a single agent that observes the entire wargame and determines an action for each subordinate unit. This agent is an example of CTCE since only a single neural network is trained and deployed, regardless of the number of subordinate units. As with any RL agent, *AI16*'s observation space, action space, and reward function are critical aspects of the agent's design.

AI16's observation space is similar to the FMH Worker's observation space. Both observation spaces consist of 16 layers with the first 14 layers using one-hot encoding of the terrain and unit types (by affiliation). The difference is in the last two layers and overall size of the observation space. *AI16* has a layer for all the legal moves its subordinate unit can make. *AI16's* final layer is like the FMH Manager's final layer and represents the percentage of the time remaining in the game.

AI16's action space is similar to the FMH Worker's action space with one critical difference. Like, the FMH Worker, *AI16*'s actions are the primitive actions of Atlatl: shoot, move, or pass. A user can set the action space as a set of either discrete values from 0 to 6, inclusive, or discrete values from 0 to 18, inclusive. If the training and evaluation environment is purely infantry units, then the user can select the smaller action space. Like the FMH Worker, the values translate to hexes to move to, or if occupied, to attack. The critical difference is that *AI16* has to take an action for each of its subordinate units, therefore its action space grows asymptotically 18^n, where n is the number of units in the scenario.

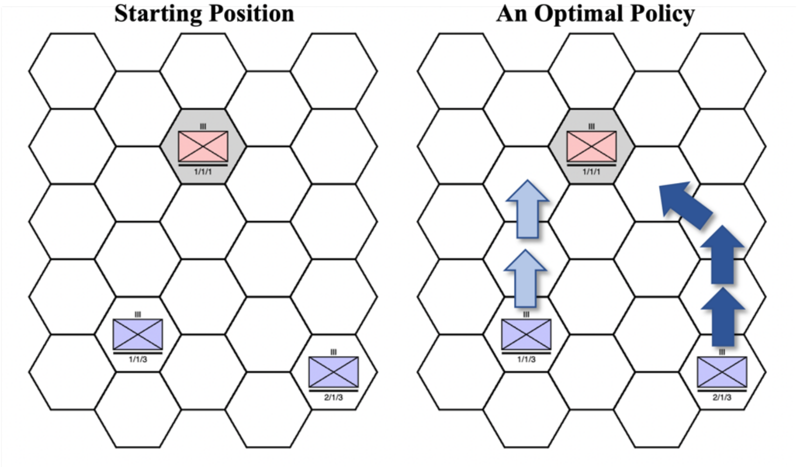
AI16 uses the same reward function as FMH Manager—the BoronRewArt. Using the same reward function ensures that any difference in performance between agents is from their designs. Specifically, differences in how these agents manage their observation and action space as a wargame gets increasingly larger.

2. Scenarios

This thesis utilizes five scenarios for its investigation. FMH's performance is compared against *pass-agg* and *AI16* in a series of four increasingly complex scenarios. A fifth scenario, a maze, serves as a proof of concept for the FMH Manager.

a. 2v1 Scenario—Principle of Mass

The first scenario tests the ability of RL agents to develop the military principle of mass. This scenario is based on the work of [18], where they demonstrated that a basic RL agent can learn to attack with two subordinate units simultaneously; referred to as massing fires [39]. Figure 16 shows the starting position and optimal policy for this scenario. The scenario consists of two Blue infantry units versus a single Red infantry unit that is occupying an urban hex. The Red unit is controlled by a *shootback* AI. A *shootback* AI will not move from its starting position but will attack any unit with range (chosen randomly if multiple units are within range).



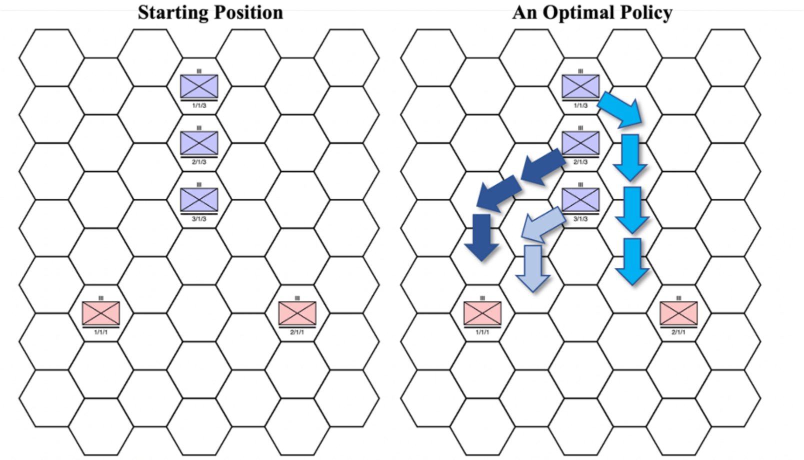
Several optimal policies are possible in the 2v1 scenario, one is shown. Optimal policy occurs when both Blue units can attack Red at the same time.

Figure 16. Starting Position (left) and an Optimal Policy (right) for the 2v1 Scenario

An optimal policy for this simple scenario can be derived from the deterministic rules of the wargame. The rules of Atlatl combat results in Blue units only being able to deal damage equal to a quarter of the Blue attacker's combat strength since all units are infantry units with the Red unit defending from an urban hex. Similarly, the Red unit can deal damage equal to half of its current strength since the Blue unit can only occupy clear hexes. Therefore, the Red unit can eliminate a Blue unit unless both Blue units attack at the same time. Blue 1/1/3 starts two hexes away from the Red unit, while Blue 2/1/3 is three hexes away. Accordingly, an optimal policy can only occur if Blue 1/1/3 takes no action for one turn so both units can then attack the single Red unit in their fourth turn.

b. 3v2 Scenario—Principle of Economy of Force

The second scenario tests the ability of RL agents to develop the military principle of economy of force. This scenario is also based on the work of [18], where they demonstrated that a basic RL agent can use only the minimum amount of combat power to accomplish a secondary task. Thus, the maximum combat power is applied to the main objective by the RL agent. Figure 17 shows the starting position and an optimal policy for this scenario. The scenario consists of three Blue infantry units versus two Red infantry units. One of the Red units starts at 25 points of combat strength, vice the normal 100 points. Again, the *shootback* AI controls both Red units, so the Red units will not move but will attack Blue units within range.


Starting Position

An Optimal Policy

Several optimal policies are possible in the 3v2 scenario, one is shown. Optimal policy occurs when two Blue units mass on the stronger (left) Red unit while only one Blue unit attacks the weaker (right) Red unit.

Figure 17. Starting Position (left) and an Optimal Policy (right) for the 3v2 Scenario

Using the rules of Atlatl, an optimal policy can be predicted for this 3v2 scenario. A suboptimal policy occurs if all three Blue units attack the stronger Red unit, ignoring the smaller Red unit. Likewise, a suboptimal policy occurs if two or more units attack the smaller Red unit. Therefore, the optimal policy is a simultaneous attack by two Blue units on the stronger Red unit with the third Blue unit attacking the weaker Red unit.

c. *6v4 Scenario—Intermediate Complexity*

The third scenario is designed to test the ability of agents to handle a larger, more complex scenario. This scenario is the first to use all six terrain types and various friendly and hostile units. Figure 18 shows the starting position for the scenario. The scenario consists of six Blue units (two mechinf, two armor, and two artillery units) against four Red units (one of each type) in control of an urban hex. For this scenario, *pass-agg* controlled all the Red units.

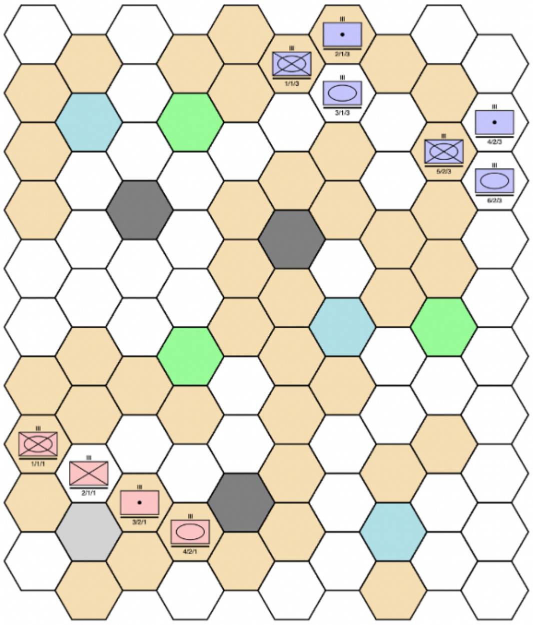


Figure 18. Starting Position for the 6v4 Scenario

This scenario is too complicated to calculate the optimal policy but a near optimal policy can be detected after playing the scenario as the Blue side several times. Red will stay near the urban hex based on the rules coded for *pass-agg*. The fastest way to get the Blue units to attack the Red unit (and gain control of the urban hex) is to move in columns along the clear hexes (since all Blue units can move two hexes per turn along clear hexes). Additionally, Blue should use the two artillery units to attack Red units before the other Blue units attack.

d. 12v6 Scenario—High Complexity

The fourth scenario, the largest scenario of the experiment in both the size of the map and number of units involved, is also designed to measure the ability of the agents to scale without loss in combat behavior. Figure 19 shows the starting position for the 12v6 scenario. Blue starts with three divisions. Each division consists of two mechinf, one armor, and one artillery unit. Similar to the previous scenario, Red is controlled by *pass-agg* and defends an urban hex with six units. Red starts from good defensive terrain, i.e., terrain reduces Blue's attack strength. Also, the artillery unit behind the urban hex makes Blue's task of taking the city more difficult, due to Red artillery's attack range.

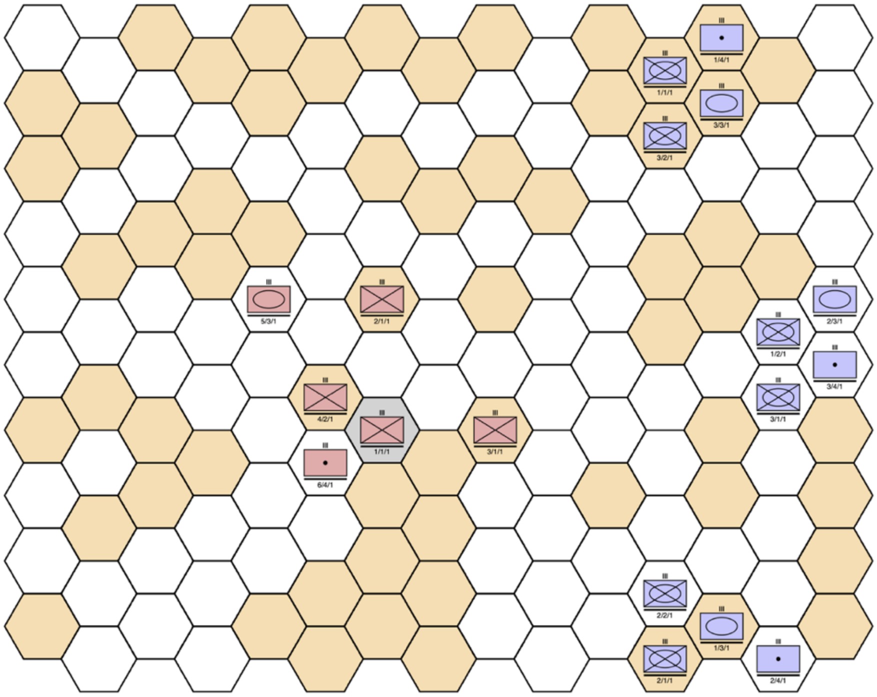
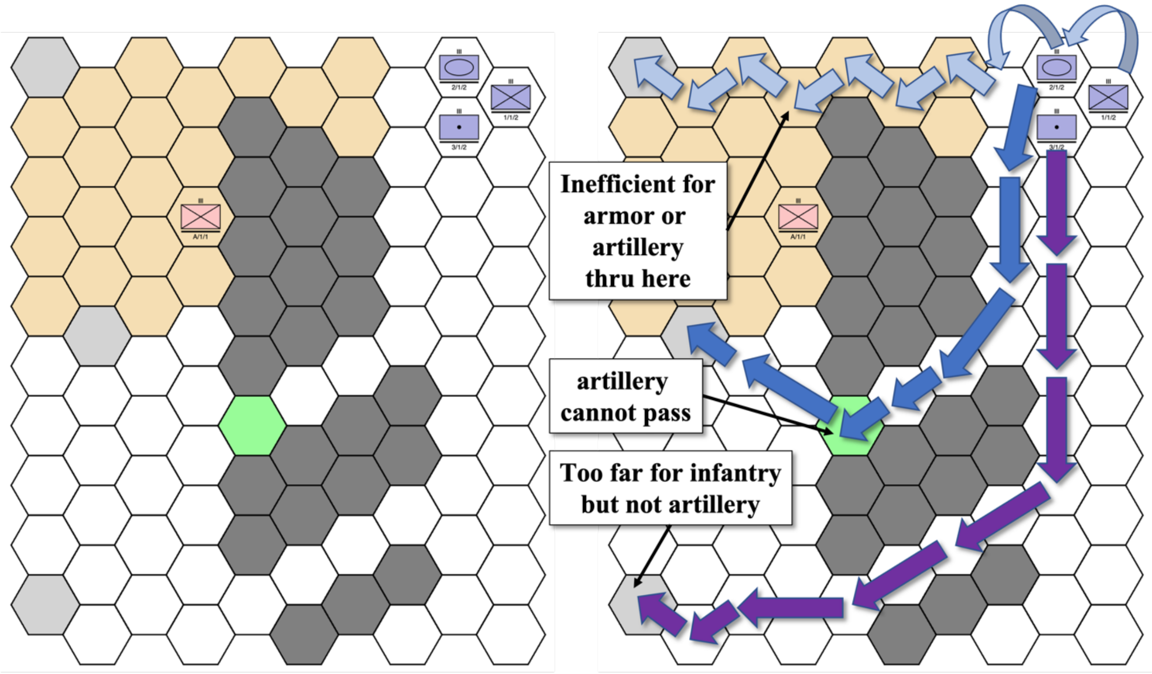


Figure 19. Starting Position for 12v6 Scenario

e. Three City Maze—Sequencing of Units

A final scenario tested the ability of FMH to sequence which units to move. The Atlatl interface allows players to select which unit the player wants to move first. Unfortunately, previous agents, from rules-based agents to standard RL agents, all moved units in numerical order. Failing to take advantage of Atlatl’s unit sequencing capabilities results in situations where the agent choses to not move a unit because that unit is blocked by other friendlies. The FMH Manager can be modified so that in addition to ordering goals to subordinates, the Manager choses the order that subordinates pursue their goals. Figure 20 shows the starting position for the final scenario consisting of three Blue units trying to reach three urban hexes. The starting position and terrain setup requires the agent to determine which path to send each of the Blue units and in what sequence. This scenario also serves as an initial proof of concept for the FMH Manager with rules-based Workers as its subordinates. The Workers are programmed to move to the hex closest to its assigned target.


2/1/2
1/1/2
3/1/2
A/1/1
2/1/2
1/1/2
3/1/2
A/1/1
Inefficient for armor or artillery thru here
artillery cannot pass
Too far for infantry but not artillery

The optimal strategy requires the infantry unit to move toward the top urban hex, armor unit to the center urban hex, and the artillery unit to the bottom urban hex.

Figure 20. Three-City Maze Scenario Starting Position (left) and Optimal Policy (right)

f. Summary of Scenarios

The scenarios are designed to measure the ability of FMH's ability to develop and maintain combat behavior in increasingly larger and more complex scenarios. RL agents have demonstrated combat behavior in past research [18] for the first two scenarios (2v1 and 3v2). These scenarios are used to see if FMH can replicate the same behavior and with the same resources (as measured by the number of training steps required) as previous RL agents. The third and fourth scenarios measure FMH performance in comparison to the past agents in increasingly complex scenarios with the same resources. The final scenario tested if FMH could develop a new capability for RL agents by sequencing its order of agents for optimal performance.

E. PROCEDURES FOR EXPERIMENT

1. Common Feature Extractors among Agents

Every RL agent that received observations from the Atlatl wargame environment used the same feature extractor. RL researchers [12] use feature extraction as a method of dimensionality reduction, that is, to "simplify the data without losing too much information [51]. SB3 provides a feature extraction function, which is employed in this thesis. The feature extractor, along with the HexagDLy Python package [52], allows for the use of Convolutions Neural Networks (CNNs) with Atlatl's hexagon-based map to obtain the most relevant information from the agent's observations. Figure 21 illustrates the relationship between the Atlatl observation that OpenAI Gym provides and SB3's Features Extractor and RL architecture.

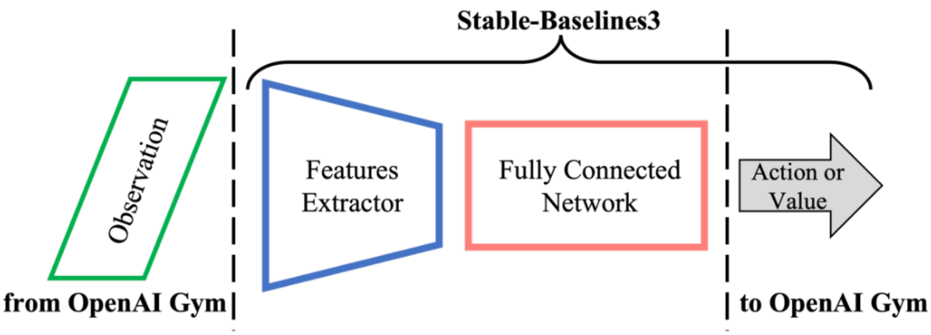


Figure 21. SB3's Features Extractor in Relations to OpenAI Gym. Adapted from [50].

The CNNs used with SB3's feature extractor function enables the agent's observation space, which is a multi-dimensional tensor to be transformed into a vector which can then be processed by the agent's learning algorithm (i.e., DQN or PPO). CNNs provide two additional capabilities beyond dimensionality reduction: spatial invariance and equivariance. Spatial invariance means that small spatial changes to the input (e.g., vertical shift or small rotation of an image) will generate the same output [51]. Equivariance means that changes in the input will generate an equal change in the output [53]. These capabilities are achieved through kernels, or a function applied over the input which generates an output, commonly called a feature map [53]. Figure 22 depicts a kernel applied to a 3x32x32 tensor generating a single output feature.

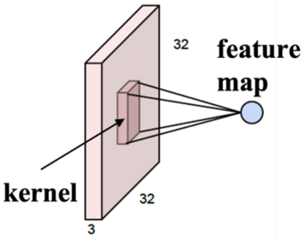
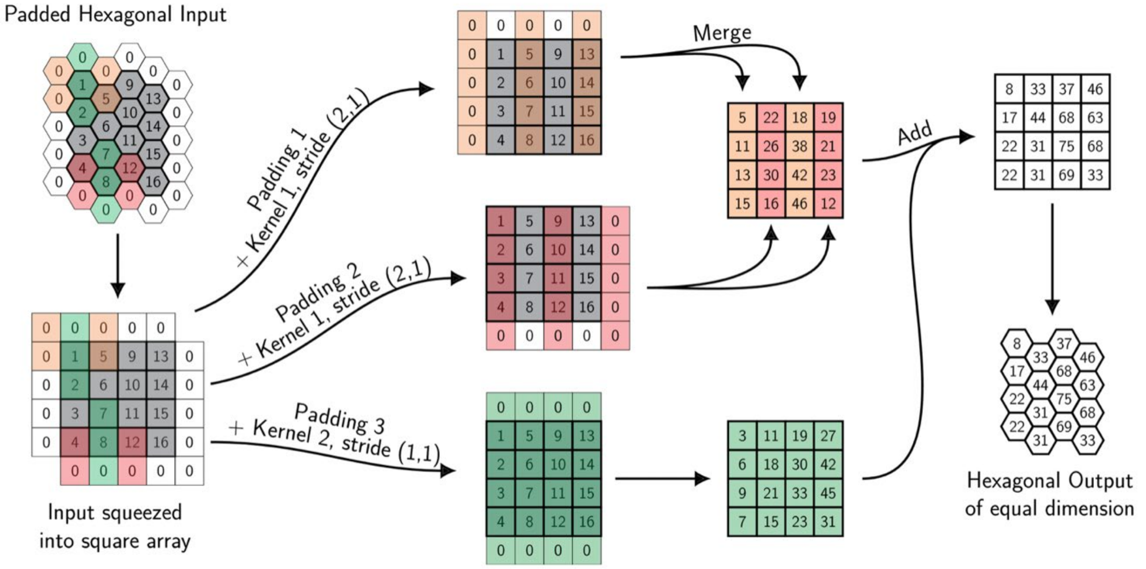


Figure 22. Convolutional Kernel Over an Input with a Single Output. Adapted from [54].

The HexagDLy Python package [52] enables the use of convolutional kernels over the hexagon-based observations the agent receives. Since Atlatl is a hexagon-based game, preprocessing must "translate the information from the hexagonal grid to a square grid tensor" before the feature extractor is applied [18]. Figure 23 shows how a hexagonal input is transformed into a square matrix for a convolutional kernel before being transformed back into a hexagonal output, referred to as hexConv2d.


Padded Hexagonal Input
Input squeezed into square array
Padding 1 + Kernel 1, stride (2,1)
Padding 2 + Kernel 1, stride (2,1)
Padding 3 + Kernel 2, stride (1,1)
Merge
Add
Hexagonal Output of equal dimension

0	0	0	0	0	
0	1	5	9	13	0
0	2	6	10	14	0
0	3	7	11	15	0
0	4	8	12	16	0
	0	0	0	0	0

0	0	0	0	0
0	1	5	9	13
0	2	6	10	14
0	3	7	11	15
0	4	8	12	16

1	5	9	13	0
2	6	10	14	0
3	7	11	15	0
4	8	12	16	0
0	0	0	0	0

0	0	0	0
1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16
0	0	0	0

5	22	18	19
11	26	38	21
13	30	42	23
15	16	46	12

3	11	19	27
6	18	30	42
9	21	33	45
7	15	23	31

8	33	37	46
17	44	68	63
22	31	75	68
22	31	69	33

An "input tensor is convolved with hexagonal size 1 kernel (all weights set to 1, i.e., the convolution adds up all data points covered by the kernel)" [52].

Figure 23. Example of a Size 1, Stride 1 Convolutional Kernel (hexConv2d). Source: [52].

The feature extractor used by all the RL agents is four layers deep and transforms the agent's observation into a vector of size 64, representing the 64 most important features of the observation. Figure 24 illustrates the CNN used for the feature extractor. The first layer, called the *conv* layer utilizes a hexConv2d kernel of size 1 and stride 1 with a depth of 32. The second layer, *resid1*, is hexConv2d kernel of size 1, stride 1, depth 32 but also employs a residual layer. The residual layer adds the output of *conv* to the output of *resid1*'s hexConv2d layer before applying the activation function. The third layer is a Flatten Layer which transforms the tensor into a single dimensional vector. The final layer is a Linear layer with an output layer of size 64. Layers 1, 2, and 4 all used Rectified Linear Units

(ReLU) for their activation functions. The output of the CNN is used by the agent's learning algorithm during training and execution.

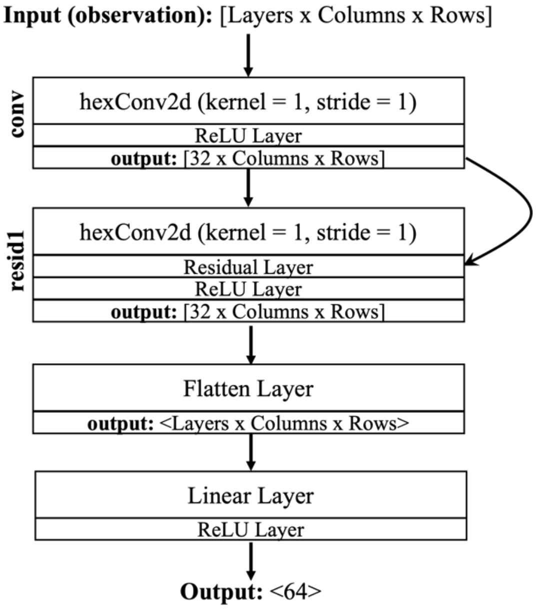

Input (observation): [Layers x Columns x Rows]
conv
hexConv2d (kernel = 1, stride = 1)
ReLU Layer
output: [32 x Columns x Rows]
resid1
hexConv2d (kernel = 1, stride = 1)
Residual Layer
ReLU Layer
output: [32 x Columns x Rows]
Flatten Layer
output: <Layers x Columns x Rows>
Linear Layer
ReLU Layer
Output: <64>

Figure 24. CNN Used for the Feature Extractor of All Agents

2. Training

For each scenario, the *AI16* and FMH agents train for the same number of timesteps. The number of timesteps used for each scenario varies and are based on how long it took for *AI16* to either reach an optimal policy or failed to improve its learning for that scenario. Training environments are used for each scenario prior to evaluation. Training environments generally have more phases than the evaluation scenarios to support exploration by the agents. *AI16* and FMH Manager are both trained using PPO with the same hyperparameters. FMH Workers are trained using DQN.

3. Evaluating

The Atlatl game score is used as the performance measure for each scenario. The optimal policy is the one that achieves the highest possible score for the game. The highest possible scores could be manually calculated for the 2v1, 3v2, and Three City Maze scenarios. The near optimum score for the other two scenarios, 6v4 and 12v6, can be approximated through human play.

F. SUMMARY

The constraints of Atlatl and support RL packages influenced the design of the FMH agent and the experiments. The experiments are designed to access FMH's ability to exhibit combat behavior in increasingly complex environments. FMH's performance is compared to a rules-based agent and a standard RL agent to assess FMH's feasibility. The resources needed to train each RL agents are used to assess acceptability. The experiment is built to ensure that only the design of FMH is assessed and not extraneous factors like its reward function or feature extractor. The next chapter presents the results of the experiments.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. RESULTS

This chapter presents and discusses the results of the four Atlatl scenarios used to assess the feasibility and acceptability of using a FMH agent for large scale wargaming. First the chapter begins with a discussion on the results of training the FMH Worker sub-agent. Then the chapter explains the possible scores in each scenario and assesses each agent's score. Finally, the chapter presents an analysis of the FMH's performance.

A. WORKER TRAINING RESULTS

The first step in developing a complete FMH agent is a functional Worker sub-agent. Several iterations and a significant redesign of the agent's observation space and training scenario, as well as hyperparameter tuning, specifically the discount factor were needed to create a Worker that provides consistent behavior.

1. Desired Worker Behavior

The Worker's observation space, action space, and training were designed to provide consistent behavior based on the goals issued by the Manager. Desirable Worker behavior can be understood as moving advantageously to the target hex assigned by the Manager. If the hex is occupied by an opposing force, then the Worker should attack the opposing unit until its eliminated, then move to the target hex. The Worker should stay on the target hex until it receives a new target from the Manager and defend the target hex from any opposing force. The Worker's observation space was modified to achieve this behavior.

2. Worker's Observation Space

Initially, the Worker's observation space was 16 layers of size 5x5, centered on the hex that the Worker occupied. This observation space design allows the Worker to be usable for any map size and reduce training time for each scenario. Unfortunately, this approach fails to converge to an optimal policy. Figure 25 shows that a Worker with only local observations stagnates at approximately 20 reward points during training, while the Worker with global observations plateaus at approximately 140 reward points. Both types of Workers have 16 layers but Worker-Local's observations are limited to the 5x5 hexes centered on the Worker (as described in the previous chapter), while Worker-Global receives input from every hex (similar to *AI16*). Table 4 shows the performance of each variety of Worker after 1.5 million training steps. An additional benefit of using a global observation space is that it ensures the Worker's target is always within its observation space thereby improving the Worker's ability to locate and move to its target.

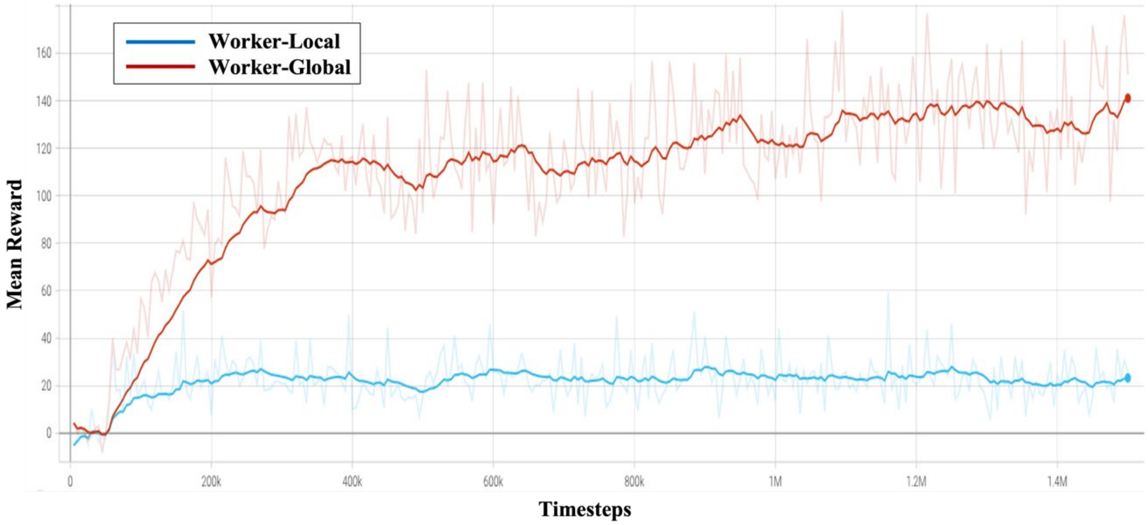


Figure 25. Mean Reward During Worker Training, n=50

Table 4. Mean Reward of Worker after 1.5 Million Steps, n=100

	Mean	Standard Deviation
Worker-Local	165.47	162.46
Worker-Global	27.51	55.35

Unfortunately, using global observations means that each Worker type (infantry, mechinf, artillery, and armor) needs to be trained for each scenario that uses a different map size. One of the reasons for designing the Worker to be scenario map size independent is to support scaling by reducing the amount of time required to train a complete FMH agent for a scenario. Table 5 shows new training requirements based on the modification of the FMH Worker to receive observations from the entire wargame. The Workers needed for the 6v4 and 12v6 scenarios are trained using the 6v4 and 12v6 scenarios instead of creating independent training environments like the 2v1 and 3v2 Workers used. In other words, for the 6v4 and 12v6 scenarios, the Workers and the Manager train using the same environment.

Table 5. Worker Training Requirements for Each Scenario

Scenario	Size (Columns x Rows)	Training Requirement
2v1	7x7	Initial training
3v2	7x7	Reuse 2v1 Scenario Workers
6v4	10x10	Re-training required
12v6	15x10	Re-training required

3. Worker's Training Scenario

Workers need to train with other friendly units to deconflict their movements. Initially, Worker training consisted of only a single friendly unit in a scenario of mix terrain types and opposing units. Even though the Worker's observation space had layers to detect other friendly units, the Worker failed to react to other friendly units during Manager training since the Worker never encountered friendly units during Worker training.

4. Worker's Discount Factor

Hyperparameter tuning for the Worker showed that a discount factor of 0.6 during training results in the best performance. Figure 26 shows results of hyperparameter tuning for an infantry Worker. While a discount factor of 0.6 provides the best results, values between 0.6 and 0.85 provide near comparable results. The upper and lower extremes for a discount factor, i.e., 0 or 0.99, show the worse performance. Therefore, all the Workers are trained using a discount factor of 0.6.

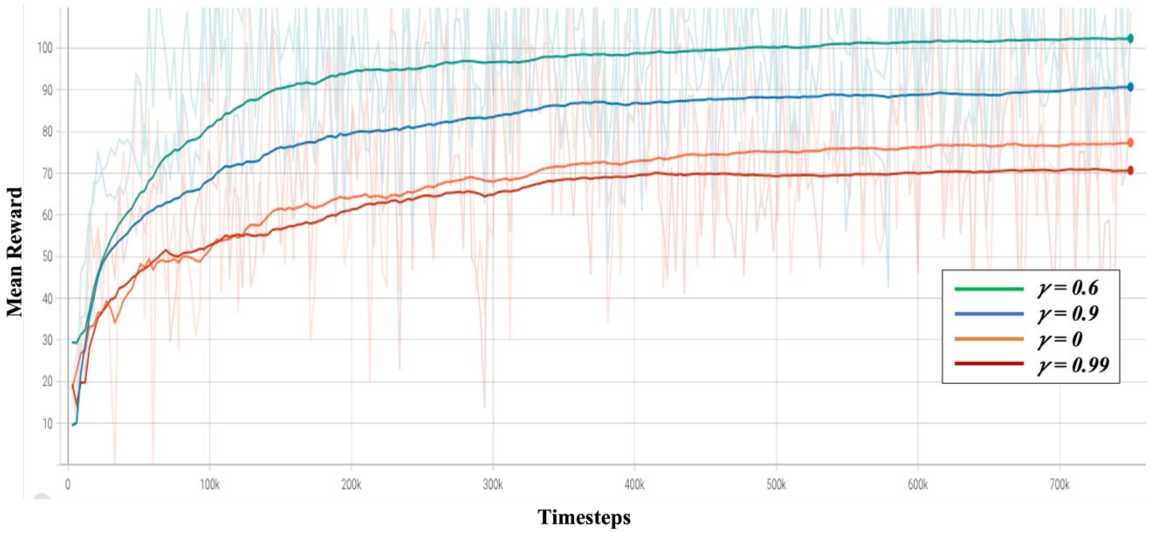


Figure 26. Effect of Discount Factor on Mean Reward During Worker Training

5. Variance between Differing Worker Types

Each Worker type trains independently but in similar environments, which results in different mean rewards during training. It is expected that mechinf, armor, and artillery would score more reward points than infantry since the other unit types can move twice as far as an infantry unit in clear terrain (and reach their target quicker). Also, it is expected that mechinf and armor would score more reward points than artillery, since artillery cannot reach any targets that are marsh hexes. Surprisingly, there is greater variation in reward points between armor and the other units as seen in Figure 27.

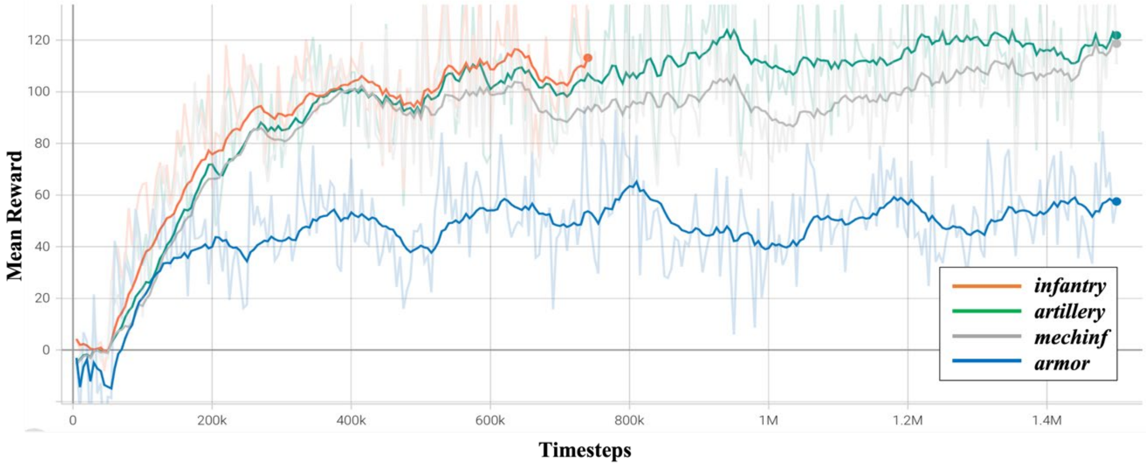


Figure 27. Differences in Mean Reward During Worker Training by Type

B. FMH RESULTS

The FMH Manager's performance is compared to a rules-based agent and a single RL agent in four different scenarios: 2v1, 3v2, 6v4, and 12v6. In all of these scenarios, FMH fails to demonstrate better performance than a single RL agent or a rules-based agent. The FMH agent repeatedly scores the same as doing nothing.

1. 2v1 Scenario – Mass

During training for the 2v1 scenario, FMH fails to demonstrate any learning, as measured by mean reward during training as compared to the single RL agent, *AI16*. Figure 28 shows that *AI16* continues to learn until about 160,000 training steps while FMH never improves its learning, even when FMH trains against a *passive* agent, an agent that does not move or shoot back. FMH receives zero rewards when training against the *shootback* AI but did occasionally attack when training against a *passive* AI.

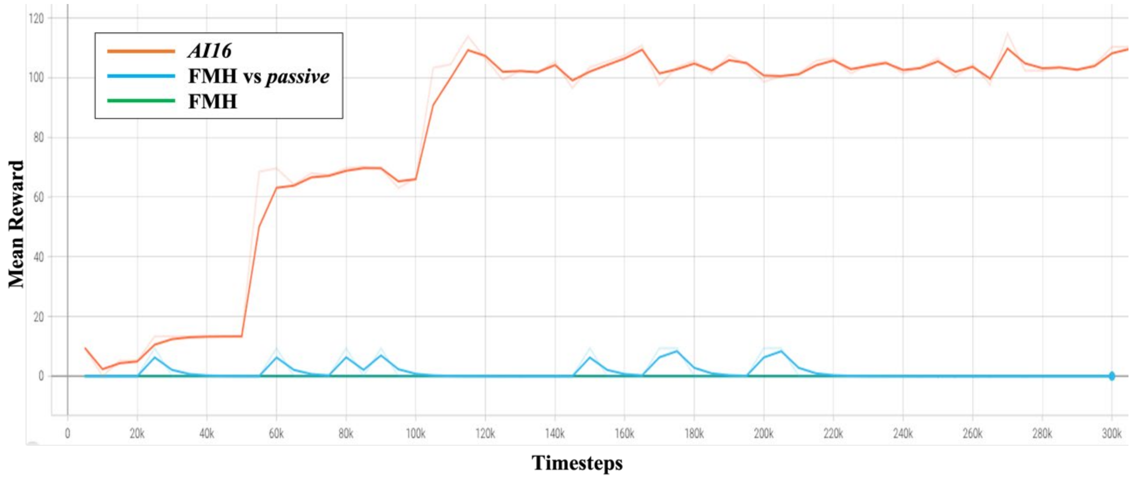


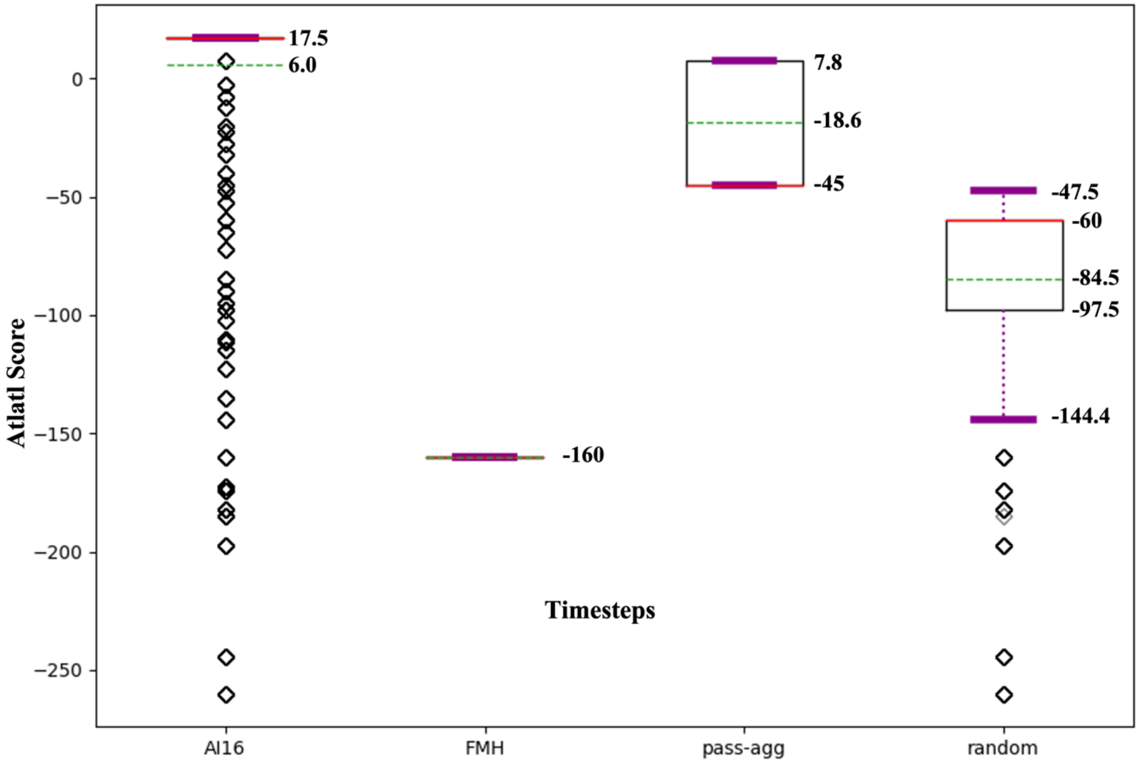
Figure 28. Mean Reward per Evaluation During 2v1 Training, n=50

In the 2v1 scenario, the FMH and rules-based agent fail to demonstrate the combat behavior of massing while the single RL agent did mass. The optimal policy can be calculated since the scenario is small enough. Optimal policy occurs when the two subordinate units simultaneously attack the single opposing unit, resulting in an Atlatl score of 17.5 points. An agent that does nothing[7] will score -160 points, while an agent that chooses random actions will score -84.5 points. Figure 29 and Table 6 illustrates that *AI16* repeatedly demonstrates optimal behavior, while FMH fails to take any effective actions. While *pass-agg* was more effective than FMH, it only achieves sub-optimum performance.

Table 6. Performance of Various Agents in the 2v1 Scenario, n=100,000

Agent	Mean Score	Standard Deviation
AI16	6.01	38.29
FMH	-160	0
FMH vs *passive*	-160	0
pass-agg	-18.6	26.4
random	-84.51	39.16

[7] For this thesis, an agent that does nothing includes moving around the map without ever getting close enough to engage, or be engaged, by an opposing unit.



87% of AI16's games resulted in an optimal score of 17.5 points, which is why the upper and lower interquartile are centered on 17.5. FMH had zero variation, therefore it lacks an upper and lower interquartile plot.

Figure 29. Box Plot of 2v1 Scenario Agent Scores, n=100,000

2. 3v2 Scenario – Economy of Force

FMH continues to show a lack of training during the 3v2 scenario. Figure 30 shows that *AI16* continues to learn for 450,000 training steps while FMH never improves its learning, even when FMH trained against a *passive* agent. This scenario utilizes the same Worker that was used for the 2v1 scenario since both scenarios use the same map size.

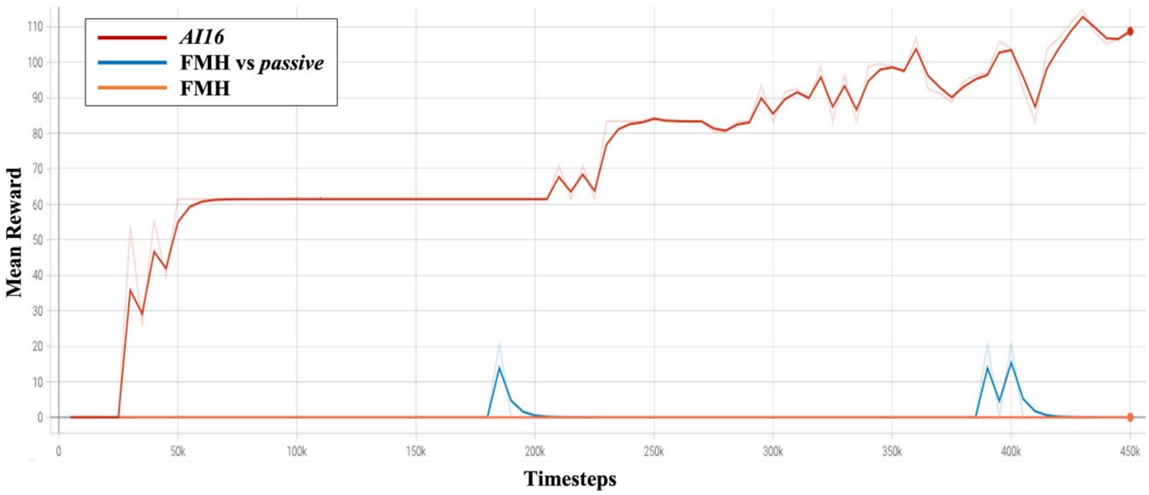
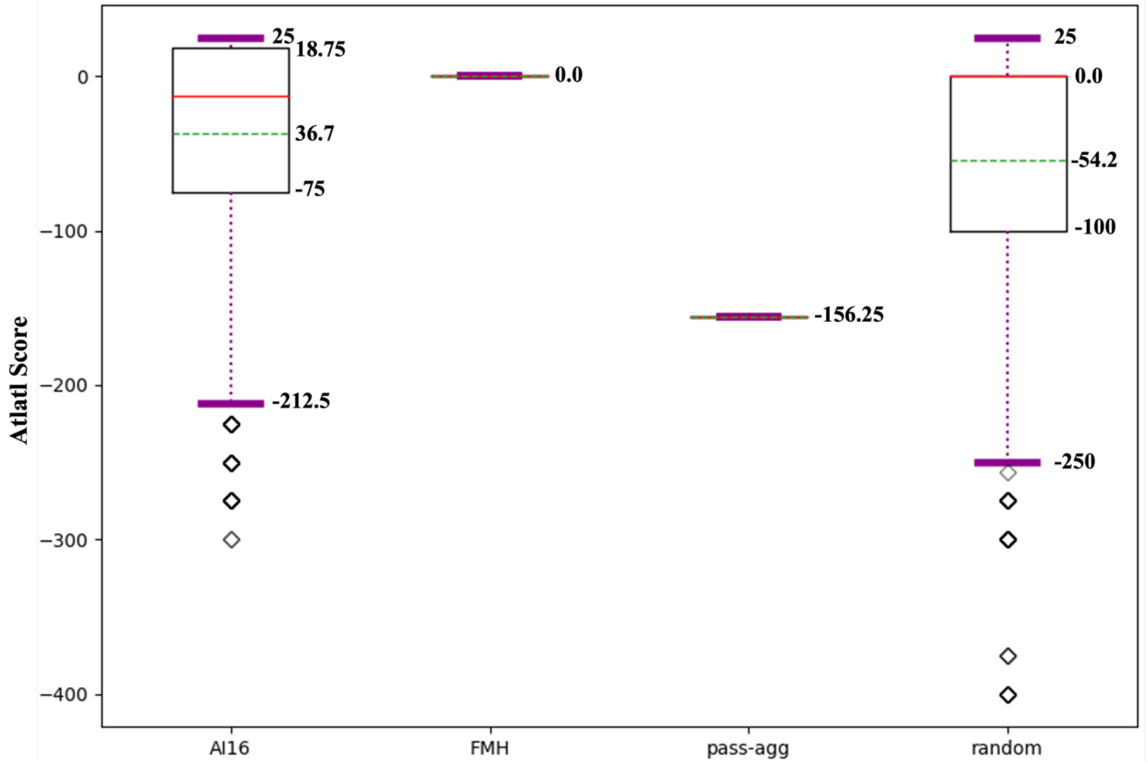


Figure 30. Mean Reward During 3v2 Training, n=50

In the 3v2 scenario, the FMH and rules-based agent fail to demonstrate the combat behavior of economy of force (EoF) while the single RL agent did. The scenario is small enough to calculate the optimal policy. Optimal policy, i.e., EoF, occurs when two subordinate units simultaneously attack the opposing unit with the most combat power, while a single subordinate unit attacks the "weaker" unit. Agents that demonstrate EoF behavior results in an Atlatl score of 18.75 points. An agent that does nothing will score 0 points, while an agent that choses random actions will score -54.2 points. Figure 31 and Table 7 illustrates that *AI16* repeatedly demonstrates optimal behavior, while FMH fails to take any effective actions. Remarkably, *pass-agg* is less effective than taking random actions.

Table 7. Performance of Various Agents in 3v2 Scenario, n=100,000

Agent	**Mean**	**Standard Deviation**
AI16	-36.71	60.24
FMH	0	0
pass-agg	-156.25	0
random	-54.2	77.21



AI16 achieved the optimal score of 18.75 points in 32% of the games evaluated. More training could have increased the percentage of optimal scores but was not within the scope of the thesis.

Figure 31. Box Plot of 3v2 Scenario Agent Scores, n=100,000

3. 6v4 Scenario – Increasing Complexity

Both FMH and the single RL agent show a lack of training during the 6v4 scenario. Figure 32 shows that neither FMH or AI16 demonstrate consistent learning during its 2 million training steps. These results are as expected due to the size and complexity of the scenario and initial sparseness of rewards. This scenario requires new Workers to be trained since this scenario uses a larger map.

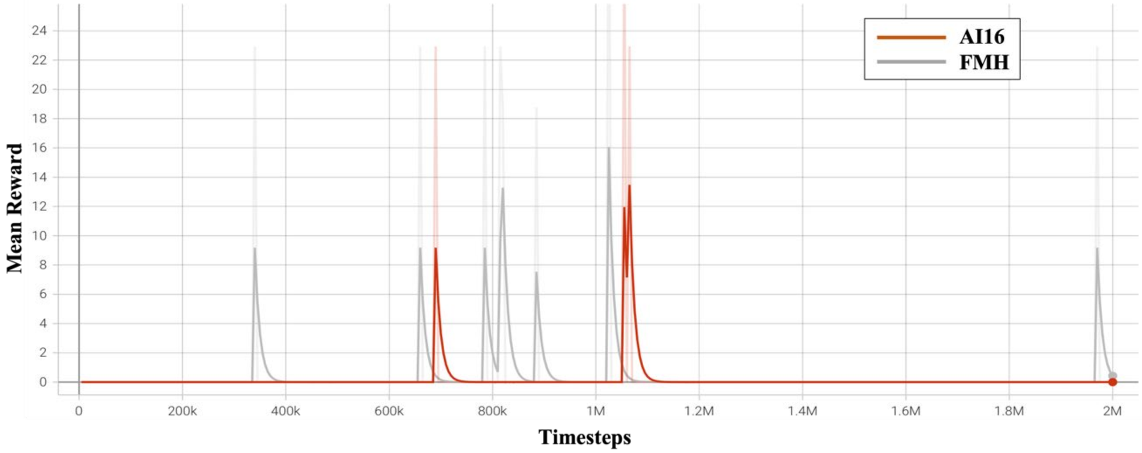


Figure 32. Mean Reward During 6v4 Training, n=50

In the 6v4 scenario, all agents fail to demonstrate acceptable performance with the rules-based agent scoring the best. The scenario is too large to calculate what an optimal policy would score. After playing several iterations, an "acceptable" score can be deduced. Blue can score approximately +125 points when employing a near optimal policy.[8] An agent that does nothing will score -435 points, while an agent that choses random actions will score an average of -459 points.

The size of the scenario and the distance between opposing sides is large enough that random actions likely result in opposing units not engaging each other. Both AI16 and a randomly acting agent score -435 points in 90% of the evaluations (i.e., failed to engage in any combat 90% of the time). Figure 33 and Table 8 illustrates that while *pass-agg* could score the highest (max score was -60 points), it is far from near optimal and fails to achieve a positive score. FMH continues to take actions that fail to engage opposing units, even when trained against *passive* agents.

[8] Near optimal policy requires Blue units to move along clear hexes to quickly reach the opposing units and use the Blue artillery units to attack opposing units, starting with the Red artillery unit. Place the other friendly units within range of Red artillery, when Blue artillery attacks to decrease the odds that Red artillery attacks Blue artillery. Prevent Red from reoccupying the urban hex and eliminate remaining units.

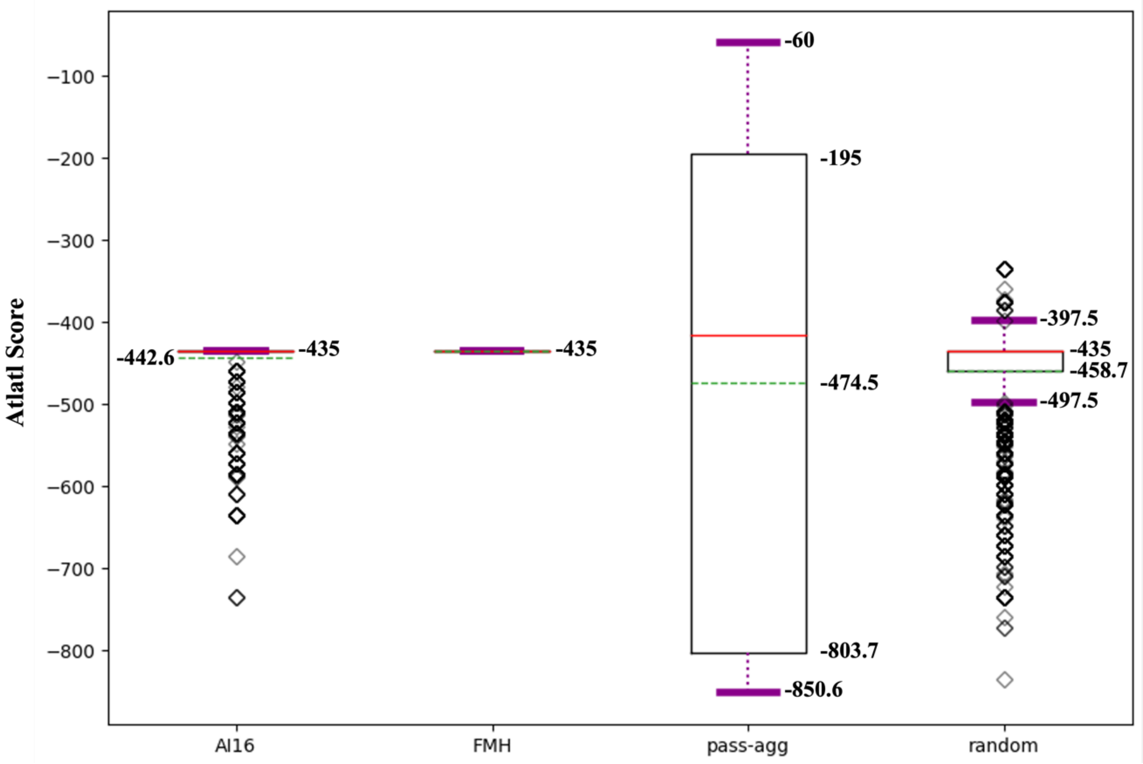


Figure 33. Box Plot of 6v4 Scenario Agent Scores, n=100,000

Table 8. Performance of Various Agents in 6v4 Scenario, n=100,000

Agent	Mean	Standard Deviation
AI16	-442.57	24.97
FMH	-435	0
pass-agg	-474.5	292.84
random	-458.65	44.14

4. 12v6 Scenario – Largest Complexity

FMH finally started showing potential for learning but none of the agents showed near optimal performance in the 12v6 scenario. Figure 34 shows that FMH, when training against a *passive* agent, achieves a higher reward than AI16 over the course of 3.5 million training steps. It is difficult to assess if either agent would improve with more training time based on the large variation in mean rewards throughout the training period. This scenario also uses Workers specifically trained for the 12v6 scenario.

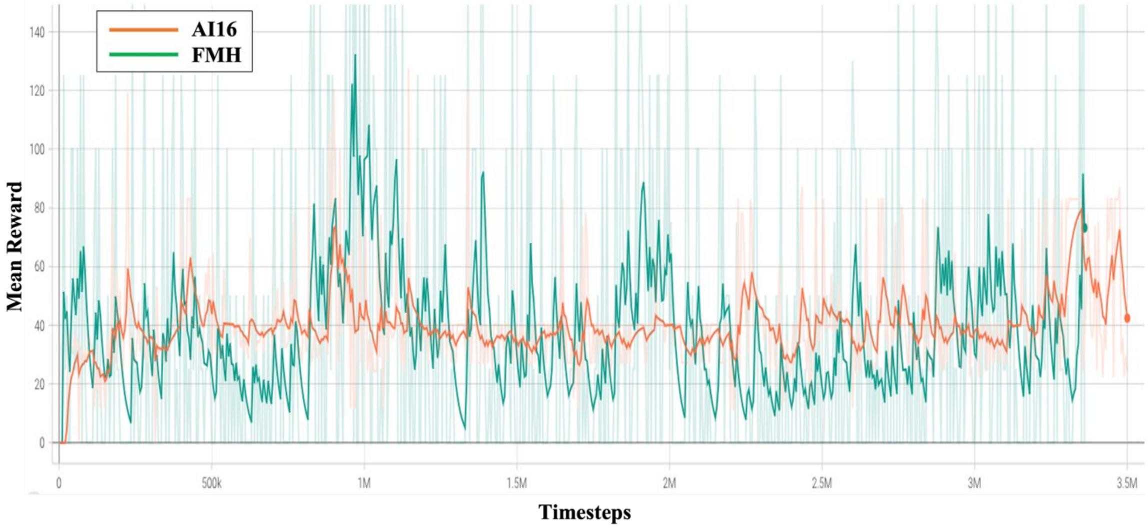


Figure 34. Mean Reward During 12v6 Training, n=50

During the 12v6 scenario, all agents barely performed better than random actions, with the rules-based agent occasionally achieving the highest score amongst all agents, as seen in Figure 35 and Table 9. Similar to the 6v4 Scenario, after playing several iterations, an "acceptable" score of +180 can be deduced.[9] Both AI16 and FMH fail to mass units for attacking the defending Red units, attacking with only single units at a time. Additionally, FMH only employs its mechinf units.

Table 9. Performance of Various Agents in 12v6 Scenario, n=100,000

Agent	**Mean**	**Standard Deviation**
AI16	-885.3	170
FMH	-680	0
pass-agg	-1065.6	252
random	-864.2	208

[9] Near optimal score requires the massing of artillery units to reduce or eliminate opposing units occupying or in front of the urban hex. Then use the remaining friendly units to attack the sole Red artillery unit. And finally, prevent Red units from reoccupying the urban center once the defending unit is eliminated by artillery.

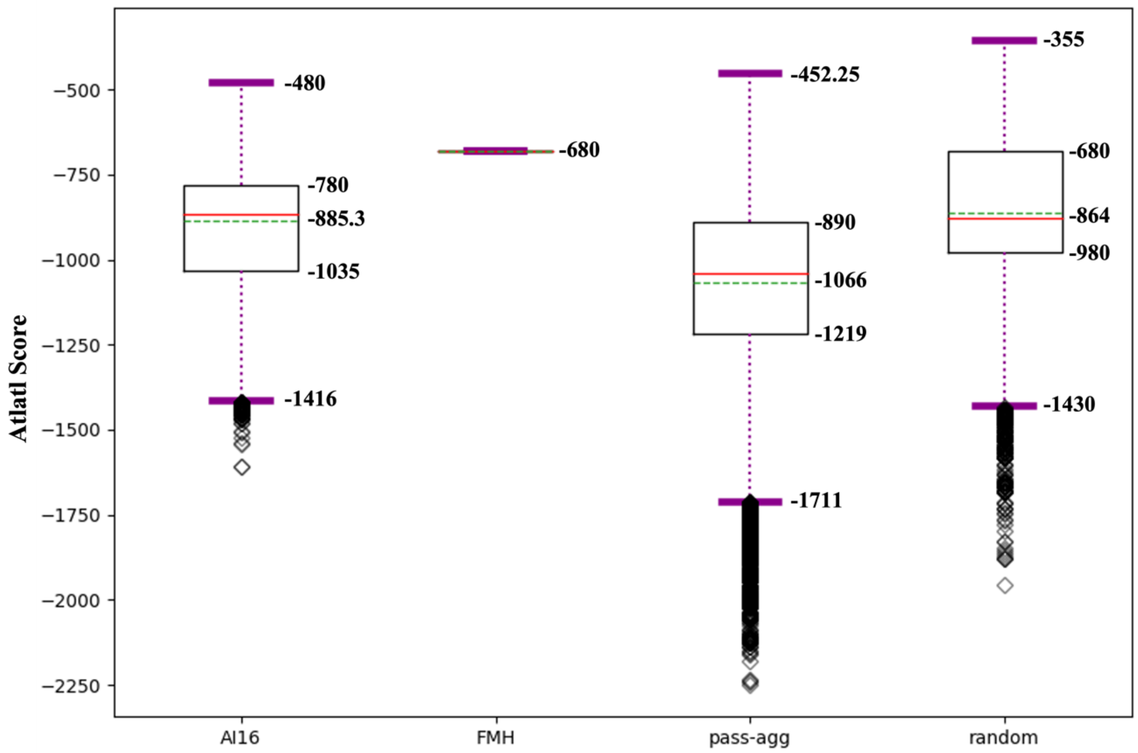
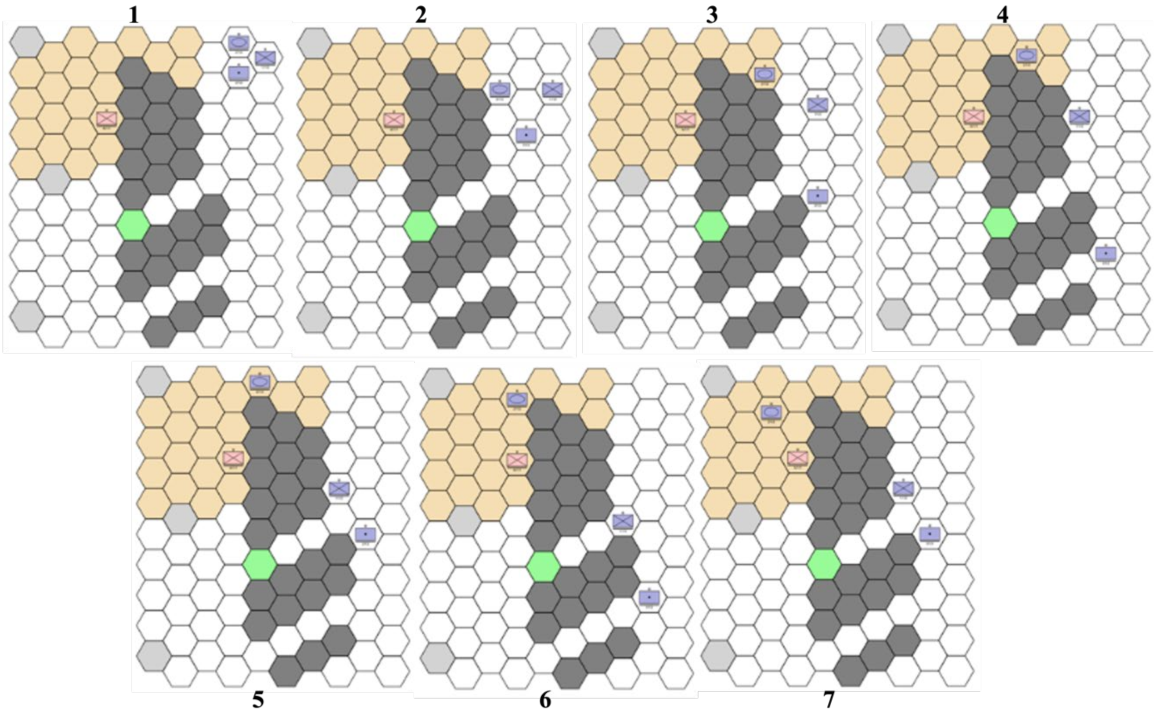


Figure 35. Box Plot of 12v6 Scenario Agent Scores, n=100,000

5. 3-City Maze

The 3-city maze is the initial proof of concept for the FMH Manager and also a test of the Manager's ability to sequence Worker movements using rules-based agents. The FMH Manager successfully learns to move its rules-based Workers towards the city. The Manager does not learn the proper sequencing and fails to all occupy all three urban hexes in the fastest time possible. The Manager's ability to move the Workers towards the urban hex serves as proof that the Manager is prepared to direct fully trained RL Workers. Figure 36 shows the FMH Manger directing the Workers towards the urban hex. The Manager learns to move the armor unit quickly towards the urban hex and for the artillery unit to avoid the marsh hex (which is impassable for artillery).



Each image is a single phase consisting of three Worker movements. Each image is in order of movements from left to right, top to bottom.

Figure 36. FMH Manager with Rules-Based Workers in 3-City Maze Scenario

C. ANALYSIS

The FMH Worker’s failure to provide consistent actions based on the Manager’s targets prevents the FMH agent from developing an optimal policy. An analysis of the Manager’s targets and the Worker’s actions shows irregular behavior from the FMH Worker. For the 2v1 Scenario, a “Good Target” is the hex that the opposing unit occupies or the adjacent hexes as highlighted in green in Figure 37. The Worker is designed to move towards its target and engage any opposing units on or next to the target. Theoretically, “Good Target” hexes should result in combat. Ultimately, the FMH Manager will develop an optimal policy by providing targets to its subordinate Workers so that they attack the opposing unit simultaneously. The FMH Manager designated a “Good Target” hex to either subordinate Worker only 4% of its training time. More importantly, is what the Worker did when it receives a “Good Target.”

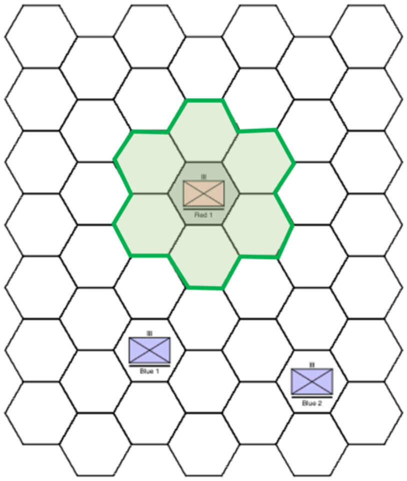


Figure 37. Hexes That Are Good FMH Targets

The FMH Worker consistently fails to act in a positive manner when the FMH Manager issues a "Good Target." Table 10 summarizes all the Workers' actions when it receives a "Good Target." The Manager assigns 93% of its "Good Targets" to the Worker labeled Blue 1. Both Workers pass most times when they receive a "Good Target." Preferably, a Worker would only pass if it was already occupying its target hex and had zero opposing units to attack, which never occurred during training. The Workers chose to pass for an unknown reason. An assessment of the Worker's training shows that occasionally, the Worker would pass even though its target is reachable.

Unexpectedly, a "Good Target" from the Manager never results in a Worker attacking the opposing unit, yet from the training data the Workers did attack the opposing unit (refer to Figure 28, positive rewards only occur from combat). The positive rewards indicate that occasionally, when the Worker is within range of the opposing unit and the Manager issues a goal that should order the Worker to move away from the opposing unit, the Worker choses to attack instead.

The Manager's failure to learn to issue only "Good Targets" is unsurprising since the Worker choses to pass most of the times it receives them. When the Worker is trained in the 2v1 Scenario prior to the Manger's training, the Worker passes less and the Manager improves its learning. Table 11 shows the Worker's improved behavior and its effect on the Manager's training, see Figure 38.

Table 10. FMH Worker's Actions When Issued a "Good Target" Target

Action Taken	Percentage of All Actions
Pass	82.4%
Attack	0%
Failed to Attack	0%
Move to Any "Good Target" Hex	0.07%
Move to Assigned Target Hex	0.035%

The Worker was trained in an environment dissimilar to the one in which the Manager will employ it (i.e., the 2v1 scenario).

Table 11. FMH Worker's Actions When Trained in the 2v1 Scenario

Action Taken	Percentage of All Actions
Pass	1.6%
Attack	0.14%
Failed to Attack	0.04%
Move to Any "Good Target" Hex	25.9%
Move to Assigned Target Hex	25.6%

The Worker was trained in 2v1 scenario (against a *passive* agent) before the Manager trained in the 2v1 scenario (also against a *passive* agent).

There is a measurable difference in actions taken by the Worker when the Worker trains in the 2v1 scenario. The Worker is less likely to pass, more likely to attack (though barely). Additionally, the Worker remains within the good hexes longer. These actions should support the FMH Manager learning a better policy, i.e., the Manager should continue to issue "Good Targets."

While the Manager is able to achieve more reward points during training, it fails to translate into better performance. The Manager continues to develop a policy of ineffective movements resulting in the same score as doing nothing. Again, this sub-optimal is unsurprising. The Manager demonstrates large variation during its training period as seen by the green line in Figure 38. Conversely, *AI16* expresses consistent improvement during its training as seen by the orange line in Figure 38. The differences in training behavior result in *AI16* learning to mass while FMH takes ineffective actions.

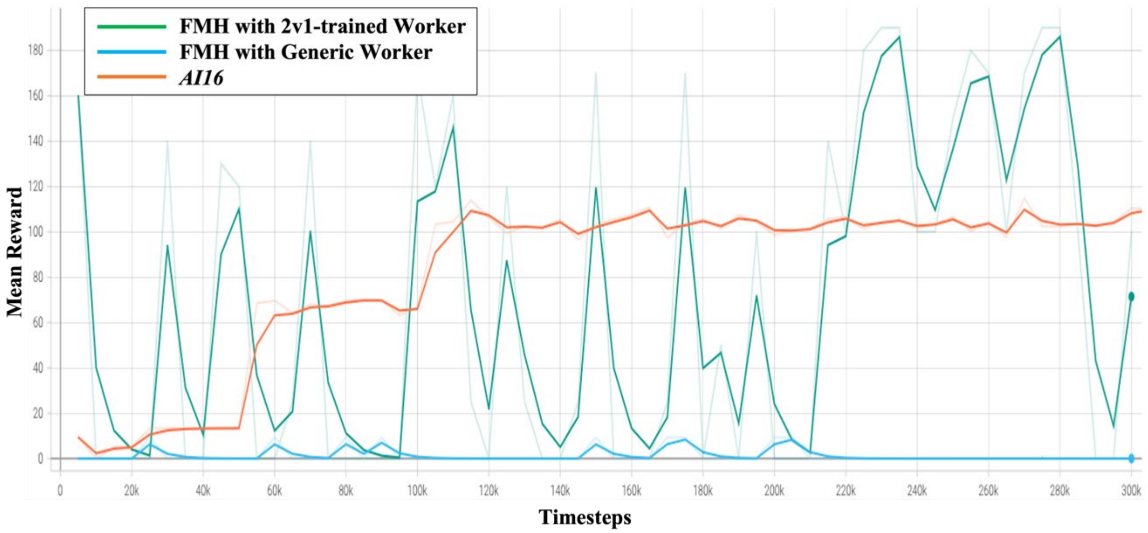


Figure 38. Mean Reward for 2v1 Scenario with Improved Worker

While the Worker's action improves (as measured by the percentage of times the Worker passes), the Worker is still inconsistent. The Worker choses to move to a non-"Good Target" hex 75% of the time the Manager assigns it a "Good Target." The Worker continues to demonstrate the behavior of attacking the opposing unit even though it receives an order to move away. Looking at Table 4, there is a large variation in the Worker's mean reward at the conclusion of its training. This variation in behavior is translating into inconsistent actions and disrupting Manager learning.

D. SUMMARY

FMH fails to develop optimal or near optimal policies in any of the scenarios and only shows potential for learning with significant pre-training. The FMH demonstrates the most potential for learning an optimal policy when it uses Workers trained in the same environment as the Manager.

THIS PAGE INTENTIONALLY LEFT BLANK

V. DESIGNING A SUITABLE WARGAMING AI

This thesis examines not only the feasibility and acceptability scaling HRL for large military wargames, but also assesses the suitability of doing so. This chapter explores the suitability of replacing human operators with intelligent agents by examining trustworthy and responsible AI within wargames. If integrating an intelligent agent into wargaming counters the purpose of wargaming, then the endeavor is not suitable. Therefore, the chapter begins by discussing the purpose of wargaming and how wargames can fail to achieve their purpose. This chapter focuses on one aspect of integrating AI into wargames: replicating the opposing force (OPFOR). While this chapter will focus on OPFOR, its implications can also apply to using AI to replicate any simulation entity: friendly forces, allies, and civilians. The chapter then explains that DOD policy requires researchers and military leaders to assess the suitability of integrating AI. Finally, the chapter provides a recommendation for ensuring that any AI method implemented into wargaming remains suitable.

A. WARGAMING'S PURPOSE

Militaries conduct wargames to learn about themselves. That is, obtain insights into their decision-making process by fighting against tough, realistic opponents. Wargaming achieves its purpose by imparting meaningful experiences in its participants, particularly the decision makers, through their active involvement in the consequences of their decisions [40]. Rubel claims the wargame experience is meaningful because the participants "gain valid and useful knowledge" [2]. Perla and McGrady argue that wargaming provides "participatory narrative experiences" which is fundamental to understanding new or unfamiliar concepts [40]. The opposing force plays a critical role in the wargaming experience because learning comes from the "dynamic interaction of the competing ideas and wills of its players" [1].

How the OPFOR is employed in a wargame is critical to determining how the dynamic interaction emerges. The U.S. Army's concept of an OPFOR is established in Army Regulation 350-2, which defines an OPFOR as "a plausible, flexible military and/or paramilitary force representing a composite of varying capabilities of actual worldwide forces, used in lieu of a specific threat force for training and developing U.S. forces" [55]. The U.S. Army developed the Training Circular 7–100 series to ensure that its OPFOR could provide "a challenging, uncooperative sparring partner capable of stressing any or all warfighting functions and mission-essential tasks of the U.S. force" [56]. Since the U.S. military may find itself in a large variety of situations, the U.S. military requires its OPFOR to be able to replicate a wide spectrum of threats from nation-states to non-state actors and from a basic intelligence to a sophisticated threat [56]. Overall, TC-100.2 requires the baseline OPFOR to be "a flexible, thinking, adaptive [threat] that applies its doctrine with considerable flexibility, adaptability, and initiative" [56].

OPFOR can be adjusted to meet the needs of the wargame which operates across the spectrum of training, analysis, and experimentation. OPFOR in a training environment should be challenging but must ultimately comport to expected doctrine. Otherwise, negative training can set in [1]. Negative training occurs when soldiers learn incorrect techniques due to improper training (e.g., hiding behind a bush to avoid simulated gunfire from a laser training system, which is impractical in actual combat). OPFOR used for analysis must also adhere to expected adversarial doctrine, otherwise the analysis would be invalid. Yet, the OPFOR should also be creative within the bounds of its doctrine, otherwise the analysis would be incomplete.[10] OPFOR used for experimentation can vary from set doctrine to completely free play (i.e., free to try untested techniques). When wargame designers fail to properly apply the OPFOR, the wargame is no longer useful and is potentially counterproductive [2].

B. HOW OPFOR AI (AND WARGAMES) CAN FAIL

Wargames fail to achieve their purpose when the OPFOR generates experiences detached from reality. Weuve et al. explained the different pathologies that can cause a wargame to fail, that is, invalidating the knowledge we gain from wargaming [16]. While they established 21 ways a wargame can fail, this chapter will only look at one: model

[10] During COA analysis, within military planning, the staff should wargame against two OPFOR actions: the enemy's most likely and most dangerous COA [57].

failure. Models can be viewed as "proportional representations of the real world" [22], therefore, an OPFOR intelligent agent is a model, a representation of real-world adversaries. Consequently, that OPFOR intelligent agent needs to be the right model. Weuve et al. [16] states there are many ways that a model can fail: gives the wrong results; is too opaque for proper use; used in the wrong context; or inflexible. This section examines model failure within the application of an OPFOR AI with the additional failure mode: improper implementation.

1. Wrong Results

The simplest model failure is a model that gives incorrect results. An OPFOR agent that does not replicate real world OPFOR behavior will invalidate a wargame [1], [22]. This model failure would be evident in an OPFOR that does not use expected, or anticipated, tactics. Conversely, Perla also argues that an OPFOR AI too committed to doctrine is also improperly modeled because the agent is likely to make clearly incorrect and illogical moves [1]. An OPFOR that makes incorrect moves can be gamed by the other player. The player could trick the OPFOR agent into making moves that do not reflect real world actions. As shown in Figure 1, there is a spectrum of desired OPFOR behavior based on the objective of the wargame. Ultimately, the wargame fails to inform its participants how to think and act against a potential future adversary when the OPFOR actions are counter to the wargame's objective.

2. Too Opaque

A model can also fail if the model is too difficult to understand. Weuve et al. [16] defines an incomprehensible model as being too opaque. An OPFOR agent can be opaque if the wargame designers, players, and sponsors do not understand the agent's decision-making process. Opaqueness invalidates a wargame in two ways: players disbelieve the results or game controllers fail to understand the limitations of the results. If the participants do not understand the model, then they will not trust the model's results [16]. For example, a staff fails to improve its flawed plan because the staff believes the OPFOR fought unfairly. On the other hand, if the controllers and sponsors of the game do not understand the models, they may place too much trust in the results; they fail to understand the limitations of the model and incorrectly extrapolate the results.[11]

3. Wrong Context

Additionally, a well-designed OPFOR agent can be applied to the wrong wargame. A "correct" model can still fail if it is applied in the wrong context [16]. It may not be feasible or acceptable to train an OPFOR AI model that can execute all forms of combat. U.S. joint doctrine [39] specifies three levels of war (strategic, operational, and tactical) conducted in a wide range of military operations. These levels of war and military operations can be executed within and across five domains (air, land, sea, space, cyber), [39]. If the training audience operates at a level of war, within a military operation, or a domain that the AI is not trained to handle, then the training audience would gain an unfair advantage and results would be invalid.

4. Overly Complex

Finally, a suitable OPFOR can easily adjust to meet the needs of the wargame designers and its participants. Models that are "difficult, inflexible, or overly complex" fail to achieve wargame objectives [16]. An OPFOR agent should be able to reset its state or change some of its parameters mid-exercise. If the training audience is struggling or the exercise is failing to achieve its objectives, then the game controllers may require changes from the OPFOR [22]. The game controllers may be required to reset the game to an earlier period or change game parameters (e.g., adjust OPFOR weapon ranges to ensure an exercise remains the proper classification). A suitable OPFOR should adjust accordingly

[11] After a 2015 wargame, RAND researchers [58] claimed that Russia could occupy two Baltic capitals in 60 hours. LTC Alex Vershinin correctly argued in a 2021 essay, "*Feeding the Bear,*" that it would be improper to apply the Russian sustainment model from the 2015 RAND study to other countries [59]. Nevertheless, unnamed Western intelligence agencies predicted that Russia would topple Ukraine in a matter of weeks at the start of the 2022 Russo-Ukrainian War [60]. After six months of fighting, Russia has failed to defeat Ukraine.

5. Improper Implementation

While not identified as a pathology, a similar OPFOR failure is a wargame that employs a “cheating AI.” A “cheating AI” is an agent that has access to information about a game that a normal human would not have access to (e.g., the real time location of all its opponents) [11]. Sometimes agents cheat due to a constraint imposed by the game interface. Sometimes wargame controllers want to provide the OPFOR with extra information about the training audience to drive an exercise objective. A properly implemented OPFOR agent only has information that the agent can acquire through its own assets or knowingly supplied by the game designers.

C. IMPLICATIONS OF IRRESPONSIBLE DESIGN AND INTEGRATION

The U.S. military risks dire consequences if it fails to consider wargame pathologies in the design and integration of an OPFOR AI. Experimental and analytical wargames drive force modernization, doctrine development, and unit training. Today, organizations like the Army’s Futures Command host exercises like Project Convergence and the Joint Warfighter Assessment to evaluate future weapon systems, force designs, and their implications on future doctrine. The decisions made from these wargames assume the information gained from the game is valid.

For training, the U.S. Army envisions a future where multiple echelons integrate their live, virtual, and constructive wargames seamlessly in what U.S. Army calls the Synthetic Training Environment (STE) [61]. The U.S. Army states that integrating AI into STE is a future critical capability by replacing costly human operators and giving training units the capacity to vary the intensity of the training through the AI OPFOR [61]. This capability requires reliable and trustworthy AI. Training units need to trust that the intelligent agent is properly representing real threats and any modification to OPFOR behavior can be controlled. Training units need to understand the OPFOR agent’s decision-making in order to improve performance. Specifically, what OPFOR actions were in response to the training unit’s actions.

Finally, there is increasing discussion about using simulations to generate data for future ML projects. Researchers envision using AI in a simulation to drive the development of ML projects that could be deployed to the battlefield, which in turn generates data to improve the simulation AI in a continuous cycle [62]. Invalid simulations using flawed OPFOR models will lead to flawed deployed AI applications with deadly consequences.

Researchers should acknowledge that they are designing a model of real-world threats when designing an OPFOR agent. DOD policy requires that these design decisions be explicitly expressed. These are not the only policy requirements.

D. DOD POLICY REQUIREMENTS FOR AI

Two policy documents, the Deputy Secretary of Defense memo on Implementing Responsible Artificial Intelligence in the DOD [63] and the Defense Innovation Unit's Responsible AI (RAI) Guidelines [64] inform OPFOR AI integration. These documents call for researchers and military leaders to investigate and think through how to integrate intelligent agents into wargames. The National Security Commission on Artificial Intelligence Final Report makes clear why these policies are important. Schmidt et al. asserts that:

> [AI] systems must be developed and fielded with justified confidence. If AI systems do not work as designed, or are unpredictable in ways that can have significant negative consequences, then leaders will not adopt them, operators will not use them, Congress will not fund them, and the American people will not support them. [9]

Schmidt et al. [9]'s assertion applies to integrating AI into wargaming. If the OPFOR model is wrong, opaque, or inflexible, then the wargaming profession is at risk. Failed wargames lead to sharp criticism from the military profession, Congress, and the public.[12] DOD RAI policies seek to prevent this. The DOD AI Ethical Principles can be applied to the integration of an OPFOR agent into wargames:

1. Responsible Judgement. An appropriate authority needs to be identified to ensure the appropriate OPFOR model is being used for the wargame's

[12] The Millennium Challenge 2002, costing $250 million, is an excellent example of how OPFOR and game controller actions can cause a wargame to fail resulting in public scrutiny [65].

objective. This will prevent “wrong context” and “improper implementation” model failure pathologies.

2. Traceability. Design decisions made in the development of the OPFOR model, including data sources used, need to be documented. This will prevent “wrong results” and “inflexibility” model failures.

3. Reliability. The OPFOR AI needs to be tested intensely and by disinterested third parties. The model’s behavior needs to be protected from malicious attacks. This will prevent “wrong results” and “too opaque” pathologies.

4. Governance. A process should be established to stop using the OPFOR model if it starts to demonstrate unintended, or irrelevant, behavior. And ultimately, governance ensures the other principles are executed.

E. RECOMMENDATION FOR RESPONSIBLE AI WARGAMING

DOD Instruction 5000.61 [66] directs the military to use the verification, validation, & accreditation (VV&A) process to ensure the responsible development and integration of models into DOD applications. Therefore, VV&A should apply to an OPFOR intelligent agent, which is a representation of human OPFOR operators. VV&A consists of three distinct but interrelated processes to “to determine whether a simulation’s capabilities, accuracy, correctness, and usability are sufficient to support its intended uses” [67]. Figure 39 illustrates the DOD’s conceptual framework for VV&A.

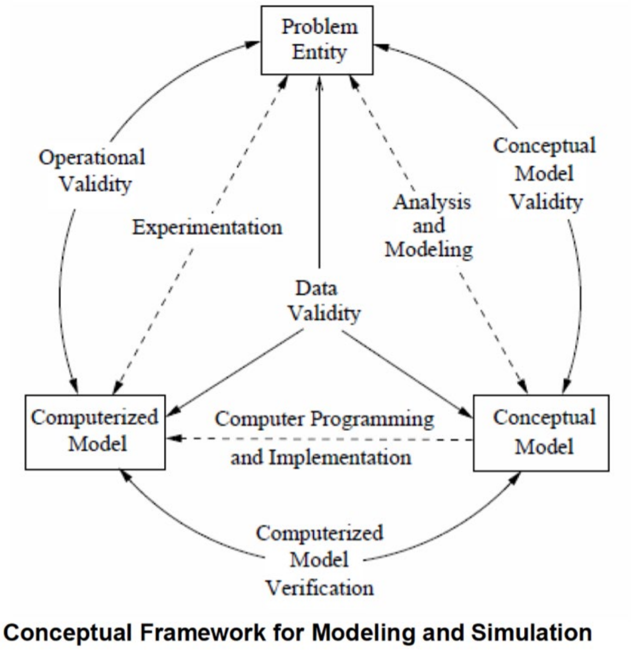

Problem Entity
Operational Validity
Conceptual Model Validity
Experimentation
Analysis and Modeling
Data Validity
Computerized Model
Computer Programming and Implementation
Conceptual Model
Computerized Model Verification
Conceptual Framework for Modeling and Simulation

Figure 39. DOD’s VV&A Framework for M&S. Source [67].

1. Verification, Validation, & Accreditation

The M&S verification process provides reliability to model development. Within the M&S community, verification asks, “Did I build the model right?” This process ensures that the model (and associated data) “accurately represents the developer’s conceptual description and specifications” [67]. Verification is proving that the model is computationally correct [67]. Verification is critical step for ML techniques like RL, because ML models have limits and failure modes [68]. For RL, verification is ensuring that the agent is demonstrating appropriate behavior. Verification prevents [16]‘s “wrong results” model failure pathology. RL verification requires comparing the agent’s actions to the conceptual model of the agent, preferably in a mix of qualitative and quantitative assessments.

The M&S validation process ensures researchers applied the appropriate judgement to their model development. For M&S systems, validation asks a similar but critically different question, “Did I build the *right* model?” This process ensures the model accurately represents the intentions of its user [67]. Validation proves that the model captures the physical phenomena that is of interest to the user. For RL, validation is comparing the agent’s training and evaluation environments against the environment that the agent will be deployed in. Validation prevents [16]‘s “wrong context” model failure pathology or improper implementation. Any differences between environments should be addressed in accreditation.

Finally, the M&S accreditation processes offers governance traceability to model design considerations. The M&S process of accreditation asks, “Is the model believable enough to be used?” This process is the official certification of the model and proving that its acceptable to use [67]. Accreditation for an RL agent should not be any different from the accreditation of any other model or simulation. Accreditation prevents [16]‘s “too complex” and “wrong context” model failure pathologies.

Ultimately, VV&A establishes credibility for a model or simulation that [9] calls for in their final report and is required by [63] and [64]. That credibility is based on identifying the model’s capabilities, accuracy, correctness, and usability [67]. VV&A addresses the improper model pathologies that we identified for an OPFOR model.

2. VV&A for an OPFOR Agent

The VV&A framework operationalizes some of the questions posed by the DOD AI Ethical Principles. The operationalization starts by examining the problem entity from the viewpoint of the DOD AI Ethical Principles. Investigating the conceptual model of the OPFOR is an initial assessment of the proper and necessary representation of our adversaries. The operationalization continues throughout the VV&A process.

The DOD AI Ethical principle of reliability is met through verification. Starting at model input, the process of data validation scrutinizes where the OPFOR agent is deriving its knowledge about our adversaries from. The agent’s behavior is compared to expected behavior of human operators.

An OPFOR AI accreditation process would require identification of who is responsible for integration of an OPFOR agent (i.e., the Responsible Judgement principle). The DOD Instruction 5000.61 assigns the Director, Defense Intelligence Agency as the validation authority for the representation of non-U.S. forces and capabilities [66]. This responsibility should extend to an agent representing an OPFOR.

The VV&A process is traceability. Multiple reports are generated in the process from the VV&A Acceptability Criteria Report to the Verification and Validation Plan to the Accreditation Report. These documents help us understand design decisions, data origin and validity, expected applications, system performance evaluation, etc. VV&A's traceability ensures "that relevant personnel possess an appropriate understanding of [the OPFOR agent's] capabilities" [63].

While the purpose of VV&A is to improve reliability, the current framework needs to look beyond weapon/platforms and focus on agent behavior. Fortunately, this can be addressed within the DOD's Model and Simulations Enterprise's governance system, specifically. the DOD M&S Steering Committee. The Steering Committee is responsible for establishing VV&A standards [66]. There are several documents governing M&S, but the steering committee needs to update them to reflect integrating AI.

F. SUMMARY

While this chapter focused on integrating an intelligent agent to represent OPFOR, this is applicable to any attempt to replace human participants with AI inside a wargame. Vendors are already trying to meet the U.S. military's desire for AI integration into wargaming [69], [70], [71]. The DOD, and specifically, the DOD Model and Simulations community needs to clearly articulate those needs. Otherwise, an improper OPFOR models will invalidate future wargames. VV&A framework offers the best method for ensuring responsible AI implementation for wargaming because replacing human operators with an OPFOR intelligent agent is the equivalent of adding an additional model to the simulation. Like any other model added to a wargaming simulation it must be credible, and that credibility is achieved through VV&A.

VI. CONCLUSIONS AND FUTURE WORK

This thesis examined the feasibility and acceptability of using HRL to support large scale wargames as well as the suitability of using AI agents within wargaming. This thesis assessed feasibility and acceptability by comparing the performance and training requirements of FMH to a standard RL agent. Additionally, this thesis explored the suitability of employing AI agents through the lens of replacing human OPFOR with an intelligent agent. While the FMH agent required more training than a standard RL agent and failed to outperform an RL agent, this thesis did show the foundations needed for using HRL for large wargames. Furthermore, this thesis recommends the DOD's VV&A process as a method to ensure any intelligent agent employed within wargames are suitable. This chapter discusses these conclusions and offers recommendations for future work

A. CONCLUSIONS

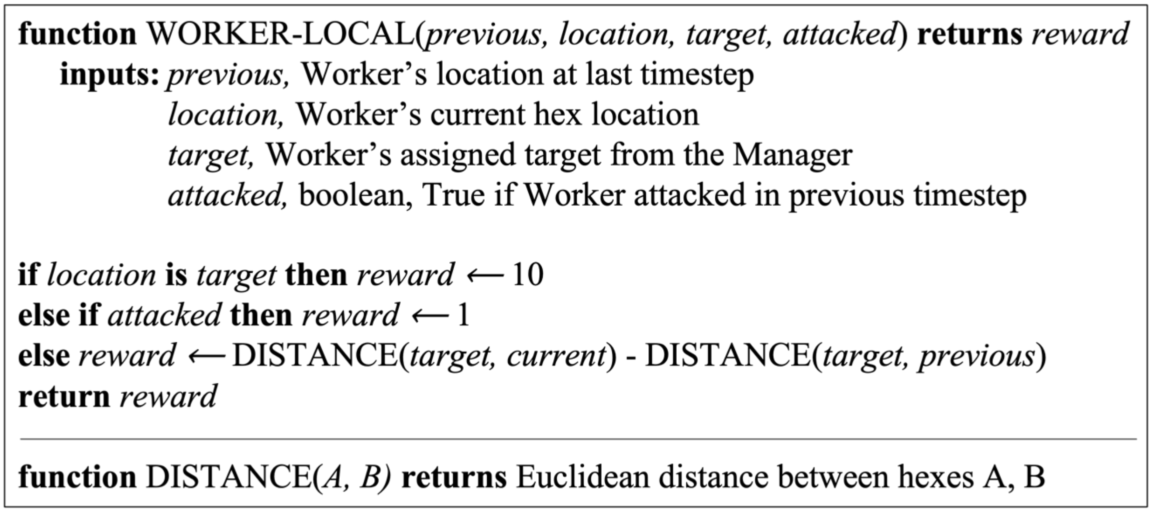
This thesis makes several conclusions about using FMH for large wargames after conducting a series of experiments using increasingly complex scenarios. Additionally, this thesis concludes that using AI for wargaming is suitable when the correct processes are applied.

1. Feasibility and Acceptability

The FMH implemented for this thesis failed to demonstrate combat behavior at any scale and is not a current feasible option for large wargames. FMH failed to demonstrate combat behavior or outperform standard RL and rules-based agents. Inconsistent FMH Worker actions prevented the FMH Manager from developing a policy, resulting in ineffective actions commensurate with taking no actions.

The FMH Worker demonstrated inconsistent actions during Worker and Manager training. The unreliable behavior can be observed in the large variance in mean reward during Worker training and the actions the Worker takes in response to the Manager's targets, specifically the quality targets. The FMH Worker has the potential to improve its performance and demonstrate more consistent actions. A FMH Worker trained using the same environment that the Manager will use performed better than a Worker trained in an independent environment.

Additionally, improving the Worker's reward function will likely improve the Worker's policy for more consistent actions. In [46]'s paper on FMH, the Worker computed its reward locally and did not receive a reward from the environment. The Worker created for this thesis used a combination of local rewards (based on proximity to the Manager's target) and rewards from the environment (through attacking opposing units). Future Worker implementations should remove rewards derived from the environment. Figure 40 shows a possible Worker reward function based on locally derived scores independent of the Atlatl score. The Worker receives the highest reward for reaching the Manager's target. Otherwise, the Worker is rewarded for attacking its way to the target unless the Worker can find a quicker path by bypass opposing units.



```
function WORKER-LOCAL(previous, location, target, attacked) returns reward
   inputs: previous, Worker's location at last timestep
           location, Worker's current hex location
           target, Worker's assigned target from the Manager
           attacked, boolean, True if Worker attacked in previous timestep

if location is target then reward ⟵ 10
else if attacked then reward ⟵ 1
else reward ⟵ DISTANCE(target, current) - DISTANCE(target, previous)
return reward
________________________________________________________________________
function DISTANCE(A, B) returns Euclidean distance between hexes A, B
```

Figure 40. Improved Worker Reward Function Based on Local Information

FMH demonstrated potential application for large wargames. FMH learned to maneuver some of its subordinate units to attack the enemy in the experiment's largest scenario, 12 friendly units versus 6 opposing units. FMH developed a policy that was similar to a standard RL agent's policy. Unfortunately, both the FMH and the standard RL agents' policies were far from optimal. If a better FMH Worker is developed, then it is worth investigating which of the two agents could develop a near optimal policy the fastest.

The FMH training method used for this thesis is not acceptable for large scale wargames. The complete FMH agent required more training time than a standard RL agent and failed to provide better performance. A complete FMH agent required every unit type to train on the scenario before the Manager could train. While multiple Workers could be trained simultaneously to reduce training time, training Workers is a constraint in the development of a complete FMH agent. The Manager must wait until all Workers are trained before beginning its training. Additionally, the more complex a scenario is, in terms of different unit types, the more training resources would be required for Workers.

2. Suitability

Implementing AI applications into wargames is suitable when the DOD's VV&A process is applied. An AI implementation in a wargame should be seen as an additional model, especially if the AI application is replacing human actors. Therefore, the DOD's process for model verification, validation, and accreditation should be applied to all AI programs for wargames. This thesis demonstrates that an improper OPFOR AI can cause model induced wargame pathologies and invalidate the objectives of the wargame. This finding has implications for future training, force design, force modernization, and war plan analysis. The DOD's Model and Simulations Enterprise should develop a governance policy for proper integration of an AI application and a process to remove obsolete intelligence agents.

B. FUTURE WORK

HRL still offers a valid path for scaling large wargames even though this thesis' implementation of FMH did not produce acceptable results. This thesis recommends the following for future research into HRL for wargames: making the agent map size invariant, improving Manager-Worker interaction, simultaneous Manager-Worker training, and exploring HRL levels of abstraction.

1. Map Size Invariance

Further research is needed to investigate the performance of RL agents that are map size invariant. Wargame designers incur a significant cost if they must train new RL agents for difference size wargames. There are two areas worth further exploration: pre-processing map data and utilizing only local observations.

Pre-processing map data is a potential means to make an RL agent map size invariant by ensuring that an RL agent operates with the same observation space regardless of scenario. The RL agents for this thesis utilized CNNs to conduct feature extraction of the most important information in the map data. Standard image analysis utilizes pre-processing procedures to standardize all inputs for ML [51]. The RL agents already conduct pre-processing by converting hex information into one-hot encodings. That information is stored in absolutes, that is, each layer of information for each hex is stored as its own cell. The size of a map can be standardized by storing hex information relatively to map size as shown in Figure 41.

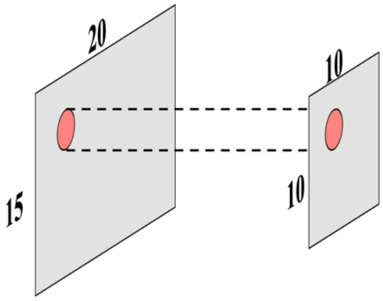


Figure 41. Pre-processing a Layer of Map Information to a Standard Size

This thesis explored the use of local observations as a means to make the Workers map size invariant. Further research is needed to understand the mechanisms that caused the Worker to fail to develop a policy when using only local observations. One possible solution is to introduce uncertainty for Workers. The Worker's observation space is still the same size as the Manager's observation space but the areas of the map outside the Worker's own local area (i.e., sensor range) are uncertain. Local observations would improve replication of real-world entities and can be extended to how the Manager learns in a partially observable environment.

2. Exploring Manager-Worker Relationship

The next area of further research needs to be understanding the relationship between the Manager's target, the Worker's actions, and how the Manager learns. The Manager failed to learn due to the Worker's inconsistent actions. Therefore, the first logical exploration is improving the Worker's policy for more consistent actions by using locally computed rewards that are independent of rewards derived from the Atlatl wargame.

The Manager's performance should improve with better Worker policies because Atlatl is a deterministic environment. Unfortunately, most military wargames are not deterministic, therefore a RL agent for military wargames needs to learn in a stochastic environment. Consequently, more research is needed to understand how an HRL agent can learn when a subordinate agent's actions and the environment's subsequent state and rewards are stochastic.

3. Simultaneous Manager-Worker Training

Another potential research topic is exploring if FMH performance improves and training requirements decrease by simultaneously training the Manager and its Workers through centralized training, then employ the agents decentralized. The original FMH paper [46]simultaneously trained the Manager and Worker using a modified version of single-agent deep deterministic policy gradient (DDPG). In their study [46], the Workers pre-trained for 10 epochs, then continued training as the Manager trained. Identical agents (e.g., the same unit types), shared parameters and contributed to the same experience replay buffer. For a large wargame (e.g., the 12v6 scenario with six mechinf units), [46]'s framework would allow for greater exploration by the different Workers and increased performance. Additionally, training simultaneously reduces the cost of increasing the number of different Workers within a scenario as seen in the training method implemented in this thesis.

Simultaneous Manager-Worker training could improve cooperation amongst the Workers. The training method used for this thesis fails to address the non-stationarity problem found in MARL. The Workers are trained in an environment where its friendly units remain stationary, yet operate in an environment where those same friendly units are employing their own policies. Simultaneous Manager-Worker training overcomes that problem as each Worker is learning its policy in an environment where other Workers are employing theirs [46]. Additionally, Simultaneous Manager-Worker training could be built so that the Manager learns to coordinate Workers so that non-stationarity is mitigated. The Manager could learn to issue zones for the Workers to operate within or issue targets that prevent Worker conflicts.

Simultaneous Manager-Worker training may also support continual learning by allowing an agent to quickly learn new scenarios or sub-agent configurations. The FMH framework employed for this thesis requires the FMH Manager to retrain if new Workers are introduced, specifically new unit types. The FMH Manager may reduce the amount of training required for integrating new Workers by combining continual learning methods with simultaneous Manager-Worker training. Both the Manager and older Workers could use continual learning to integrate the policies of the new Worker without forgetting lessons learned from previous training.

4. HRL Levels of Abstraction

Further research is needed to understand how levels of abstraction can improve AI applications for wargames and other military decision-making processes. This thesis proposes further research into the proper number of levels of abstraction, the temporal abstraction between levels, the role discount factor plays between levels of abstraction, and mixture of model types.

More research is needed to understand how many levels of abstraction for a given wargame application are needed. This thesis only used two levels of abstraction but FMH can operate with an indeterminate number of layers with sub-Managers deployed in the intermediate layers. The 12v6 scenario consisted of 3 groups of Workers consisting of two mechinf, one armor, and one artillery unit. Future research could explore the deployment of sub-Managers for each group or sub-Managers specifically trained for the management of a specific Worker type.

Further research is needed to understand temporal abstraction between hierarchies, specifically in wargames or similar real time strategy games. This thesis initially explored the difference between updating the Manager's observation after every Worker's timestep or every three. Unfortunately, the Worker's inconsistent actions precluded any insights.

Similarly, more research should explore the role of discount factor between layers of hierarchy, specifically with different temporal abstractions. Several different discount factors were used for the Workers but the only effect observed was during Worker training. Intuitively, the Worker should be more myopic than the manager, with the highest level manager utilizing the largest discount factor. It is unknown if operating at different temporal scales negates the need for different discount factors.

Finally, it is unknown if using different algorithms for different layers affected the results. Only a single RL algorithm was used for the Manager, PPO, and a different one was used for the Workers, DQN, while the original FMH framework used a modified version of DDPG. More research is needed to understand if the algorithms used in FMH influence performance and if a consistent or mixed approach matters. This also supports research into using a modular AI approach to wargaming with different RL methods employed for different parts of the wargame problem.

C. SUMMARY

This thesis explored the use of HRL in a wargame environment. The HRL agent was tested in the Atlatl wargame using a fully observable, multi-agent, deterministic, and discrete environment. This thesis used a FMH agent within the distributed training, decentralized execution MARL spectrum in a series of increasingly complex scenarios. The FMH agent failed to demonstrate the required combat behavior and failed to outperform a standard RL agent in every scenario. While the FMH Manager did demonstrate the potential to learn how to employ FMH Workers in complex environments, inconsistent actions from the Worker prevented the Manager from developing an optimal or near-optimal policy. Therefore, this implementation of FMH should not be used for large wargames.

This thesis also explored the suitability of integrating AI into wargames by assessing potential wargame pathologies from employing OPFOR AI. An intelligent agent designed to replace human operators is a model of those human operators. Therefore, employing an intelligent agent within a wargame is only suitable when proper processes are applied to prevent model-induced pathologies. The DOD's VV&A process can be updated to ensure intelligent agents are properly integrated into wargames.

These findings show that the DOD should continue to research integrating AI into wargames. More research is needed to understand how to scale current AI methods into larger wargames. HRL, and specifically FMH, shows promise for supporting large wargames. This thesis recommends several research topics that can help scale current HRL methods. Also, the DOD MSE needs to update its governance policies to anticipate the increase use of AI within wargames.

LIST OF REFERENCES

[1] P. Perla, *The Art of Wargaming: A Guide for Professionals and Hobbyists.* Annapolis, MD, USA: Naval Institute Press, 1990.

[2] R. C. Rubel, "The epistemology of war gaming," *Nav. War Coll. Rev.*, vol. 59, no. 2, pp. 108–128, 2006.

[3] R. C. Rubel, "War-gaming network centric warfare," *Nav. War Coll. Rev.*, vol. 54, no. 2, pp. 61–74, 2001.

[4] C. Von Clausewitz, M. Howard, and P. Paret, *On War*, Rev. ed. Princeton, NJ: Princeton University Press, 1984.

[5] J. Goodman, S. Risi, and S. Lucas, "AI and wargaming," 2020, [Online]. Available: https://arxiv.org/abs/2009.08922

[6] Deputy Secretary of Defense, "Wargaming and innovation," official memorandum, Department of Defense, Washington, DC, USA, 2015. [Online]. Available: https://news.usni.org/2015/03/18/document-memo-to-pentagon-leadership-on-wargaming

[7] E. B. Kania and I. B. McCaslin, "Learning warfare from the laboratory - China's progression in wargaming and opposing force training," Institute for the Study of War, 2021. [Online]. Available: https://www.understandingwar.org/report/learning-warfare-laboratory-china's-progression-wargaming-and-opposing-force-training

[8] D. C. Moffat, "The Creativity of Computers at Play," Kent, UK, 2015. [Online]. Available: https://www.cs.kent.ac.uk/events/2015/AISB2015/proceedings/computationalCreativity/aisb-15-cc.pdf

[9] E. Schmidt et al., "National Security Commission on artificial intelligence," Washington, DC, USA, Final, 2021. [Online]. Available: https://www.nscai.gov

[10] J.E. Whitley and J.P. McConville, "On the posture of the United States Army," Washington, DC, USA, 2021. [Online]. Available: https://www.army.mil/e2/downloads/rv7/aps/aps_2021.pdf

[11] P. Narayanan et al., "First-year report of ARL Director's Strategic Initiative (FY20–23): Artificial intelligence (AI) for command and control (C2) of Multi-Domain Operations (MDO)," DEVCOM Army Research Laboratory, Adelphi, MD, USA, ARL-TR-9192, 2021.

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

[13] OpenAI, “OpenAI Five,” Jun. 25, 2018. https://openai.com/blog/openai-five/

[14] D. Silver et al., “Mastering the game of Go with deep neural networks and tree search,” *Nature*, vol. 529, pp. 484–489, 2016.

[15] C. Newton, J. Singleton, C. Copland, S. Kitchen, and J. Hudack, “Scalability in modeling and simulation systems for multi-agent, AI, and machine learning applications,” 2021, vol. 11746. [Online]. Available: https://doi.org/10.1117/12.2585723

[16] C. A. Weuve et al., “Wargame pathologies,” Naval War College, Newport, RI, USA, ADA596774, 2004. [Online]. Available: https://apps.dtic.mil/sti/citations/ADA596774

[17] J. A. Boron, “Developing combat behavior through reinforcement learning,” M.S. Thesis, Dept. of Comp. Sci., NPS, Monterey, CA, USA, 2020. [Online]. Available: https://calhoun.nps.edu/handle/10945/65414

[18] C. T. Cannon and S. Goericke, “Using convolution neural networks to develop robust combat behaviors through reinforcement learning,” M.S. Thesis, Dept. of Comp. Sci., NPS, Monterey, CA, USA, 2021. [Online]. Available: https://calhoun.nps.edu/handle/10945/67681

[19] A. S. Vezhnevets et al., “Feudal networks for hierarchical reinforcement learning,” Sydney, Australia, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1703.01161

[20] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2010.

[21] J. Boyd and G.T. Hammond, *A Discourse on Winning and Losing*. Maxwell AFB, AL, USA: Air University Press, Curtis E. LeMay Center for Doctrine Development and Education, 2018.

[22] M. B. Caffrey Jr., *On Wargaming: How Wargames Have Shaped History and How They May Shape the Future*, U.S. Government Official edition. Newport, RI, USA: Naval War College Press, 2019. [Online]. Available: https://permanent.fdlp.gov/gpo120656/viewcontent.cgi.pdf

[23] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2d ed. Cambridge, MA, USA: MIT Press, 2020.

[24] N. Bostrom, *Superintelligence: Paths, Dangers, Strategies.* Oxford, UK: Oxford University Press, 2014.

[25] N.J. Nilsson, *Quest for Artificial Intelligence: A History of Ideas and Achievements*. Cambridge, UK: Cambridge University Press, 2010.

[26] D. Silver et al., “Mastering chess and shogi by self-play with a general reinforcement learning algorithm,” *arXiv*, 2017, [Online]. Available: https://arxiv.org/abs/1712.01815

[27] R. S. Sutton, “The Bitter Lesson,” *Incomplete Ideas*, Mar. 13, 2019. http://www.incompleteideas.net/IncIdeas/BitterLesson.html

[28] S. Gronauer and K. Diepold, “Multi-agent deep reinforcement learning: a survey,” *Artif. Intell. Rev.*, 2021, [Online]. Available: https://doi.org/10.1007/s10462-021-09996-w

[29] G. Qu, Y. Lin, A. Wierman, and N. Li, “Scalable multi-agent reinforcement learning for networked systems with average reward,” presented at the 34th Conference on Neural Information Processing Systems, Vancouver, Canada, 2020. [Online]. Available: https://papers.nips.cc/paper/2020/file/168efc366c449fab9c2843e9b54e2a18-Paper.pdf

[30] P. K. Sharma, R. Fernandez, E. Zaroukian, M. Dorothy, A. Basak, and D. E. Asher, “Survey of recent multi-agent reinforcement learning algorithms utilizing centralized training,” 2021. [Online]. Available: https://doi.org/10.1117/12.2585808

[31] J. Schulman, “Optimizing expectations: From deep reinforcement learning to stoachastic computation graphs,” Ph.D. dissertation, University of California, Berkeley, Berkely, CA, 2016. [Online]. Available: http://joschu.net/docs/thesis.pdf

[32] A. Irpan, “Deep reinforcement learning doesn’t work yet,” *Sorta Insightful*, Feb. 14, 2018. https://www.alexirpan.com/2018/02/14/rl-hard.html

[33] A. Tampuu et al., “Multiagent cooperation and competition with deep reinforcement learning,” *PLoS ONE*, vol. 12, no. 4, 2017, [Online]. Available: https://doi.org/10.1371/journal. pone.0172395

[34] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, “Deep reinforcement learning for multi-agent systems: A review of challenges, solutions and applications,” *IEEE Trans. Cybern.*, vol. 50, pp. 3826–3839, 2020.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

[36] V. Mnih et al., “Playing Atari with deep reinforcement learning,” *arXiv*, 2013, [Online]. Available: https://arxiv.org/pdf/1312.5602.pdf

[37] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multiagent deep reinforcement learning," *Auton. Agents Multi-Agent Syst.*, vol. 33, no. 6, pp. 750–797, 2019.

[38] M. Minsky, "Steps toward artificial intelligence," *Proc. IRE*, vol. 49, no. 1, pp. 8–30, 1961.

[39] *Joint Operations,* JP 3-0, Joint Chiefs of Staff. Washington, DC, USA, 2018.

[40] P. P. Perla and E.D. McGrady, "Why wargaming works," *Nav. War Coll. Rev.*, vol. 64, no. 3, pp. 111–130, 2011.

[41] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *International Conference on Autonomous Agents and Multiagent Systems*, Cham, 2017, pp. 66–83. [Online]. Available: https://ala2017.it.nuigalway.ie/papers/ALA2017_Gupta.pdf

[42] K. Lin, R. Zhao, Z. Xu, and J. Zhou, "Efficient large-scale fleet management via multi-agent deep reinforcement learning," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, 2018, pp. 1774–1783. [Online]. Available: https://doi.org/10.1145/3219819.3219993

[43] O. Nachum, S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning," presented at the 32nd Conference on Neural Information Processing Systems, Montreal, Canada, 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/file/e6384711491713d29bc63fc5eeb5ba4f-Paper.pdf

[44] S. Pateria, B. Subagdja, A. Tan, and C. Quek, "Hierarchical reinforcement learning: A comprehensive survey.," *ACM Comput. Surv.*, vol. 54, no. 5, 2021, [Online]. Available: https://doi.org/10.1145/3453160

[45] P. Dayan and G. E. Hinton, "Feudal reinforcement learning," in *Proceedings of NIPS*, 1992, vol. 5. [Online]. Available: https://proceedings.neurips.cc/paper/1992/file/d14220ee66aeec73c49038385428ec4c-Paper.pdf

[46] S. Ahilan and P. Dayan, "Feudal multi-agent hierarchies for cooperative reinforcement learning," *arXiv*, 2019, [Online]. Available: https://doi.org/10.48550/arXiv.1901.08492

[47] C. Darken, "Atlatl." Monterey, CA, USA. [Online]. Available: https://gitlab.nps.edu/cjdarken/atlatl

[48] *Operations,* FM 3-0, Department of the Army. Washington, DC, USA, 2017. [Online]. Available: https://armypubs.army.mil/epubs/DR_pubs/DR_a/ARN6503-FM_3-0-001-WEB-8.pdf

[49] G. Brockman et al., "OpenAI Gym," *arXiv*, 2016, [Online]. Available: https://arxiv.org/abs/1606.01540

[50] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-Baselines3: Reliable reinforcement learning implementations," *J. Mach. Learn. Res.*, vol. 22, pp. 1–8, 2021.

[51] A. Geron, *Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems*, Second edition. Beijing, China: O'Reilly Media, Inc., 2019.

[52] C. Steppa and T. L. Holch, "HexagDLy—processing hexagonally sampled data with CNNs in PyTorch," *SoftwareX*, vol. 9, no. 2352–7110, pp. 193–198, 2019.

[53] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2017.

[54] "CS231n convolutional neural networks for visual recognition." https://cs231n.github.io/convolutional-networks/ (accessed Jun. 21, 2022).

[55] *Operational Environment and Opposing Force Program,* AR 350-2, Department of the Army . Washington, DC, USA, 2015. [Online]. Available: https://armypubs.army.mil/epubs/DR_pubs/DR_a/pdf/web/r350_2.pdf

[56] *Opposing Force Tactics,* TC 7–100.2, Department of the Army . Washington, DC, USA, 2011. [Online]. Available: https://armypubs.army.mil/epubs/DR_pubs/DR_a/pdf/web/tc7_100x2.pdf

[57] *Joint Operations Planning,* JP 5-0, Joint Chiefs of Staff. Washington, DC, USA, 2011.

[58] D. A. Shlapak and M. Johnson, "Reinforcing deterrence on NATO's eastern flank: Wargaming the defense of the Baltics," RAND Corp., Santa Monica, CA, USA, 2016. [Online]. Available: https://www.rand.org/pubs/research_reports/RR1253.html

[59] A. Vershinin, "Feeding the bear: A closer look at Russian army logistics and the fait accompli," *War On the Rocks*, Nov. 23, 2021. https://warontherocks.com/2021/11/feeding-the-bear-a-closer-look-at-russian-army-logistics/

[60] N. Drozdiak and M. Champion, "Western allies see Kyiv falling to Russian army within hours," *Bloomberg*, Feb. 24, 2022. https://www.bloomberg.com/news/articles/2022-02-24/western-allies-see-kyiv-falling-to-russian-forces-within-hours

[61] "Synthetic Training Environment (STE) White Paper," Combined Arms Center - Training, 2018. [Online]. Available: https://usacac.army.mil/sites/default/files/documents/cact/STE_White_Paper.pdf

[62] P. M. zu Drewer and H. Schmitz, “Enhancing operations by applying constructive simulation and artificial intelligence,” presented at the Interservice/Industry Training, Simulation, and Education Conference, Orlando, FL, USA, 2021. [Online]. Available: https://s3.amazonaws.com/amz.xcdsystem.com/44ECEE4F-033C-295C-BAE73278B7F9CA1D_abstract_File15115/PaperUpload_21143_0827075333.pdf

[63] Deputy Secretary of Defense, “Implementing responsible artificial intelligence in the Department of Defense,” official memorandum, Department of Defense, Washington, DC, USA, 2021. [Online]. Available: https://media.defense.gov/2021/May/27/2002730593/-1/-1/0/IMPLEMENTING-RESPONSIBLE-ARTIFICIAL-INTELLIGENCE-IN-THE-DEPARTMENT-OF-DEFENSE.PDF

[64] J. Dunnmon, B. Goodman, P. Kirechu, C. Smith, and A. Van Deusen, “Responsible AI guidelines in practice,” Defense Innovation Unit, 2021. [Online]. Available: https://www.diu.mil/responsible-ai-guidelines

[65] M. Zenko, *Red Team: How to Succeed By Thinking Like the Enemy*. New York, NY, USA: Basic Books, 2015.

[66] *DOD Modeling and Simulation (M&S) Verification, Validation, and Accreditation (VV&A),* DOD Instruction 5000.61, Department of Defense. Washington, DC, USA, 2018.

[67] Defense Modeling and Simulation Enterprise, “VV&A recommended practices guide,” May 18, 2011. https://vva.msco.mil

[68] M. Malik, “A hierarchy of limitations in machine learning,” *arXiv*, 2020, [Online]. Available: https://doi.org/10.48550/arXiv.2002.05193

[69] DARPA, “Gamebreaker AI effort gets under way,” May 13, 2020. https://www.darpa.mil/news-events/2020-05-13

[70] Booz Allen Hamilton, “AI-powered analytics for Indo-Pacific wargaming.” https://www.boozallen.com/markets/defense/indo-pacific/ai-powered-analytics-for-indo-pacific-wargaming.html (accessed Jun. 17, 2022).

[71] A. McKeon, “Can artificial intelligence apply gaming to military strategy?,” Northrop Grumman. https://www.northropgrumman.com/what-we-do/can-artificial-intelligence-apply-gaming-to-military-strategy/ (accessed Jun. 17, 2022).

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California