ARL-TR-9192 • MAY 2021

DEVCOM ARMY RESEARCH LABORATORY

First-Year Report of ARL Director's Strategic Initiative (FY20–23): Artificial Intelligence (AI) for Command and Control (C2) of Multi-Domain Operations (MDO)

by Priya Narayanan, Manuel Vindiola, Song Park, Anne Logie, Nick Waytowich, Mark Mittrick, John Richardson, Derrik Asher, and Alexander Kott

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-9192 ● MAY 2021

First-Year Report of ARL Director's Strategic Initiative (FY20–23): Artificial Intelligence (AI) for Command and Control (C2) of Multi-Domain Operations (MDO)

Priya Narayanan, Manuel Vindiola, Song Park, Anne Logie, Mark Mittrick, John Richardson, and Derrik Asher
Computational and Information Sciences Directorate,
DEVCOM Army Research Laboratory

Nick Waytowich
Human Research and Engineering Directorate,
DEVCOM Army Research Laboratory

Alexander Kott
Office of the Director, DEVCOM Army Research Laboratory

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
May 2021	Technical Report	October 2019–October 2020

4. TITLE AND SUBTITLE
First-Year Report of ARL Director's Strategic Initiative (FY20–23): Artificial Intelligence (AI) for Command and Control (C2) of Multi-Domain Operations (MDO)

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Priya Narayanan, Manuel Vindiola, Song Park, Anne Logie, Nick Waytowich, Mark Mittrick, John Richardson, Derrik Asher, and Alexander Kott

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
DEVCOM Army Research Laboratory
ATTN: FCDD-RLC-CI
Adelphi MD 20783-1138

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-9192

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES
ORCID ID(s): Derrick Asher, 0000-0012-1217-8581; Alexander Kott, 0000-0003-1147-9726

14. ABSTRACT
This report describes efforts conducted in fiscal year 2020 under the US Army Combat Capabilities Development Command Army Research Laboratory's Director's Strategic Initiative (DSI) project "Artificial Intelligence (AI) for Command and Control (C2) of Multi-Domain Operations (MDO)". The speed and complexity of MDO against a near-peer adversary in hyperactive environments calls for high-speed decision-making and execution that may often exceed human cognitive ability. Recently, emerging AI techniques such as deep reinforcement learning (DRL) have outperformed human world champions in complex, relatively unstructured, partial-information, strategic games such as Dota 2 and StarCraft II. This suggests the potential of such AI to contribute to C2 of MDO. However, many questions about the behaviors and limits of such new AI techniques remain unanswered. As part of this DSI, we are investigating whether DRL might support future agile and adaptive C2 of multi-domain forces that would enable the commander and staff to exploit rapidly and effectively fleeting windows of superiority. In our first year, we have developed two novel C2 testbeds and DRL-based learning within these testbeds. This report includes an overview of the project and demonstrates preliminary research results where an "artificial commander" executes an integrated planning-execution process in a simulated brigade-scale battle.

15. SUBJECT TERMS
artificial intelligence, command and control, deep reinforcement learning, machine learning, gaming

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Priya Narayanan
Unclassified	Unclassified	Unclassified	UU	32	19b. TELEPHONE NUMBER (Include area code) (301) 394-2376

Contents

List of Figures

List of Tables

1. Introduction

The speed and complexity of Multi-Domain Operations (MDO) against a peer adversary are likely to exceed the cognitive abilities of a human command staff in conventional, largely manual command and control (C2) processes. At the same time, emerging findings in artificial intelligence (AI) techniques such as deep reinforcement learning (DRL)[1] begin to suggest the potential to support C2 of MDO. Discoveries that occurred within the last two years showed that DRL-based algorithms could outperform human world champions in complex, relatively unstructured, partial-information, strategic games such as Dota 2 and StarCraft II.[2,3] Through these breakthroughs, reinforcement learning (RL) has demonstrated the potential for AI to develop and implement multilayered strategies followed by the control of multiple agents in complex games. Battle command of future MDO is characterized by a high level of complexity within unstructured task domains, which shares some similarities with complex game simulation environments. Extending the use of AI-based approaches to the military domain, therefore, may offer important possibilities for enhancing capabilities in battle command.

The long-term intent of the project described in this report is not new. The past few decades have seen a number of ideas and corresponding research toward developing automated or semi-automated tools that could support decision-making in planning and executing military operations. What follows are a few past efforts in this space to which some authors of this report contributed to personally.

The Defense Advanced Research Projects Agency's (DARPA's) Joint Force Air Component Commander (JFACC) program took place in the late 1990s[4] and developed a number concepts and prototypes for agile management of a joint air battle. Most of the approaches considered at that time involved continuous real-time optimization and re-optimization (as the situation continually changes) of routes and activities of various air assets. Also in the mid-to-late 1990s, the Army funded the Course of Action Development and Evaluation Tool (CADET) project,[5] which explored potential utility of classical hierarchical planning, adapted for adversarial environments, for transforming a high-level battle sketch into a detailed synchronization matrix—a key product of the doctrinal Military Decision-Making Process (MDMP). In the early 2000s, DARPA initiated the Real-time Adversarial Intelligence and Decision-making (RAID) project,[6] which explored a number of technologies for anticipating enemy battle plans, as well as dynamically proposing friendly tactical actions. Game-solving algorithms emerged as the most successful among the technological approaches explored.

The role of multiple domains and their extremely complex interactions—beyond the traditional kinetic fights but also political, economic, and social effects—were explored in the late 2000s in DARPA's Conflict Modeling, Planning and Outcome Experimentation (COMPOEX) program.[7] This program investigated the use of interconnected simulation submodels, mostly system-dynamic models, in order to assist senior military and civilian leaders in planning and executing large-scale campaigns in complex operational environments. The importance of non-traditional warfighting domains such as the cyber domain has been recognized and was studied in the mid-2010s by a NATO research group[8] that looked into simulation approaches to assessing mission impacts of cyberattacks and highlighted the strong nonlinear effects of interactions among cyber, human, and traditional physical domains.

All approaches taken in the research efforts mentioned previously—and many other similar ones—have major and somewhat common weaknesses. They tend to require a rigid, precise formulation of the problem domain. Once such a formulation is constructed, they tend to produce effective results. However, as soon as a new element needs to be incorporated into the formulation (e.g., a new type of a military asset or a new tactic), a difficult, expensive, manual, and long effort is required to "rewire" the problem formulation and fine-tune the solution mechanism. And the real world presents an endless stream of new elements that must be taken into account.

In the rule-based systems of the 1980s, a system would become un-maintainable as more and more rules (with often unpredictable interactions between them) had to be added to represent the real-world intricacies of the domain. In optimization-based approaches, similarly, an endless number of relations between significant variables and a variety of constraints had to be continually manually added (a maintenance nightmare) to represent the real-world intricacies of the domain. In game-based approaches, the rules governing the legal moves and effects of moves of each piece would gradually become hopelessly convoluted as more and more realities of the domain had to be manually contrived and added to the game formulation.

In short, such approaches are unaffordable in their representation-building and maintenance. Ideally, we would like to see a system that "learns" (i.e., self-programs) its problem formulation and solution algorithm directly from its experiences in a real or simulated world, without any (or with very little) manual programming. Machine learning, particularly RL, offers exactly that promise. This is a major motivation behind our project.

1.1 Army Relevance and Problem Domain

The US Army currently does not have an AI-based, partly autonomous mission command tool that operates at high operational tempo (OPTEMPO) at the tactical or operational level.[9,10] The modus operandi entails hand-crafted strategic and operational planning that is relatively inefficient and slow. Often, life and death decisions are made by few individuals, working with imperfect information under time constraints. Tools currently available to the planners (e.g., Advanced Field Artillery Tactical Data System [AFATDS],[11] Blue Force Tracker,[12] and so on) are generally limited to rudimentary decision aids for analyzing terrain of the battlefield and automation tools to record decisions. Commanders experience information overload while providing guidance at a rapid OPTEMPO to lower echelons.[13,14] Battle Damage Assessment (BDA) is slow and is not synchronized with unit movement/sensor-to-shooter linkage, nor does it allow for the exploitation of windows of superiority.[15,16] Course of action (CoA) analysis largely focuses on the assessment of friendly plans with little emphasis on the complexity of an adversary's objectives and capabilities.[17]

MDO exponentially increases the complexity of C2 with the inclusion of space, cyber-electromagnetic activities (CEMA), and robotic assets,[18] which are likely to drive the OPTEMPO even higher than in the past. Additionally, it will be intractable for human commanders to provide highly detailed instructions[19] using currently available decision aids.[20] There are credible reports[21] that US peer and near-peer competitors, especially China, are vigorously pursuing AI in military applications that include command decision-making and military deduction (i.e., wargaming). Therefore, there is substantial risk associated with failure in the pursuit of an AI-enabled C2 system that can only be overcome by continuous progress toward this goal and a continued effort toward the actualization of an AI system capable of performing C2 in MDO.

1.2 Long-Term Goal

By 2035, we envision the need to develop agile and adaptive AI-enabled C2 systems for operational planning and decision support in complex, high OPTEMPO, hyperactive MDO. These systems will continuously integrate several domains of future warfare. The envisioned system will be capable of analyzing enemy activities; and continuously planning, preparing, executing, and assessing campaigns to enable the rapid response of Army capabilities by continually sensing, identifying, and quickly exploiting emerging windows of superiority. These windows of superiority will emerge during operations across the MDO framework at different echelons, but recognizing and exploiting them requires capabilities that

rely less on deliberate planning cycles and more on continuous, integrated planning. AI-enabled C2 systems have the potential for rapid synchronization of multiple actions across echelons, domains and, multiple simultaneously operating assets to exploit windows of superiority. Forces will predominantly consists of robotic assets (ground, aerial), and the AI-enabled C2 system will collect and process data from intelligent sensors and platforms, evaluate emerging trends in the operational environment, and recommend actions that reduce cognitive burden to allow human commanders to act quickly and effectively. The AI-enabled processes will also provide quantitative analysis, predictive analytics, and other salient data that is tailorable for efficient use by humans. This will ultimately allow the US Army to respond with the ability to reallocate, reorganize, and employ capabilities based on an understanding of enemy vulnerabilities and detailed friendly estimates across functions and domains during an armed conflict, and will produce specific, detailed commands to control autonomous assets.

The DEVCOM Army Research Laboratory has active research programs in robotics, autonomy, AI, and machine learning. The authors of this report have led research and integration activities for large collaborative robotic research efforts among government, academic, and industry partners,[22] conducted path-breaking research in scene understanding,[23] human–AI teaming,[24,25] RL,[26] multi-agent RL,[27] and multi-agent co-operative systems.[28] Further, ARL also has extensive infrastructure for conducting research in the abovementioned areas. This include ground and aerial platforms for robotic research; the Robotics Research Collaboration Campus (R2C2) for scenario-driven research capable of hosting live, scalable, multi-domain experimentation; a containerized supercomputer designed to support emerging requirements for AI and machine learning applications; to name a few. We believe these expertise and resources can be leveraged to building a success program that incorporates AI for C2 applications.

1.3 Objective of the DSI

The ARL Director's Strategic Initiative (DSI) program is a mechanism for cross-disciplinary basic and applied research, with successful proposals reaching across boundaries of scientific and technical disciplines. The program identifies topic areas representing strategic research opportunities of very high potential payoff to the Army mission, to expand existing programs or establish new core competencies and build up in-house expertise in these areas.

As part of the "AI for C2 of MDO" DSI project that was awarded in FY20, we explore the degree to which DRL-based algorithms can be used for estimating the state of the Red Force, assessing Red and Blue battle losses (attrition), predicting

Red's strategy and upcoming actions as the battle unfolds, and formulating Blue plans based on all this information. This approach has the potential to generate novel plans for the Blue Forces that exploit potential windows of opportunities at a speed much faster than an expert planner. The recent success of DRL in unstructured, strategic games has provided significant suggestive evidence that AI approaches may be capable of discovering appropriate tactical concepts essentially "from scratch", and selecting, applying, and executing strategies at faster than human speeds.

In this DSI, we explore the use of DRL for developing detailed plans prior to combat operations and generating real-time plans and suggestions during execution of ongoing operations. We plan to advance the state of the art in two key areas: 1) conceptualizing, designing, and implementing DRL-based agents for generating plans that are as good as or better than plans generated by an expert planner and 2) incorporating humans into the command and learning loops and evaluating these AI-human (human-in/on-the-loop) solutions. While developing a pathway for this AI-enabled C2, several research questions will need to be answered. In this DSI, we try to answer three specific questions:

- What are the training and data requirements for the DRL C2 agents to learn accurately and sufficiently fast?
- How can we make the DRL agents generalizable so that they perform reasonably, as judged by human experts, especially when previously unseen details are introduced into a situation?
- What is the effect of human intervention in an AI-enabled C2 system?

The first year of this project focused on developing essential building blocks for the research including 1) developing simulation capabilities and advanced interfaces by adapting and using StarCraft II- and OpSim-based environments, 2) developing initial end-to-end AI that executes the C2 function, 3) developing computational capabilities by integration with a high-performance computing (HPC) environment, and 4) initial characterization of the volume of data and training requirements. This report provides the details of each of these tasks.

2. Experimental Capabilities

As part of this project, we develop C2 simulation and experimentation capabilities that include simulated battlespaces with interfaces to DRL-based AI algorithms and capabilities for scalable RL on a DOD HPC system (Fig. 1). We use two simulation environments for generating C2 scenarios: the StarCraft II Learning Environment (SC2LE)[29] and OpSim.[30] Tiger Claw, a scenario developed by Maneuver Center of

Excellence (Fort Benning, Georgia) generates realistic combat environments within the simulation environments. Finally, we use RLlib,[31] a library that provides scalable software primitives for RL to scale the learning on an HPC system.

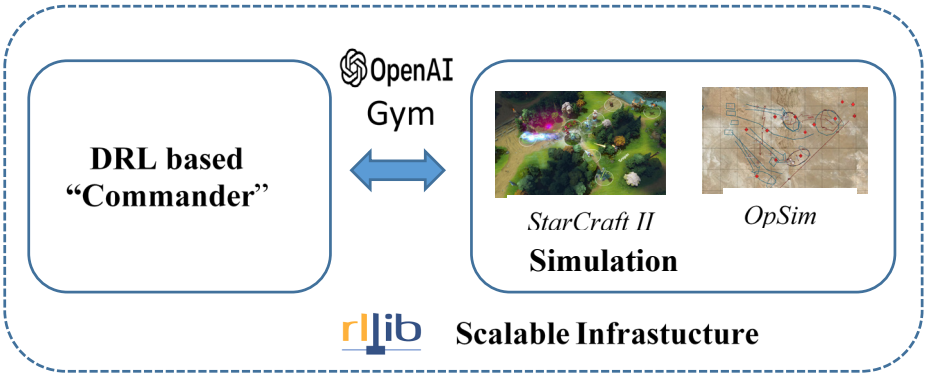

Fig. 1 Overview of the C2 infrastructure

2.1 Tiger Claw

Operation Tiger Claw is a predefined combat scenario consisting of Red and Blue Forces developed by military subject-matter experts (SMEs) at the Captain's Career Course, Fort Benning, Georgia. This hypothetical scenario shows Task Force (1–12 CAV) attack in zone to seize OBJ Lion in order to pass the division's decisive operation (DO) east. The goal of the Task Force is to cross the Thar Thar Wadi, destroy the Red Force, and seize OBJ Lion (Fig. 2). The Task Force consist of combat armor with M1A2 Abrams, Infantry Fighting Vehicles with Bradleys, field artillery howitzer and mortar, armored recon cavalry with Bradley, combat aviation, air defense, and unmanned aerial vehicles. The Red Force consist of mechanized infantry with BMP-2M, combat armor with T-90 tanks, field artillery howitzer, armored recon cavalry BMP-2M, combat aviation, anti-armor, and combat infantry. The Tiger Claw scenario also includes probable plans by the Blue and Red Forces developed by expert military SMEs. These plan are generated using doctrinal force employment in accordance with the Operation Order (OPORD) and corresponding threat tactics. The Tiger Claw scenario has been incorporated into both OpSim and StarCraft II, and is serving as a benchmarking baseline for comparison across different neural network architectures and reward-driving attributes.

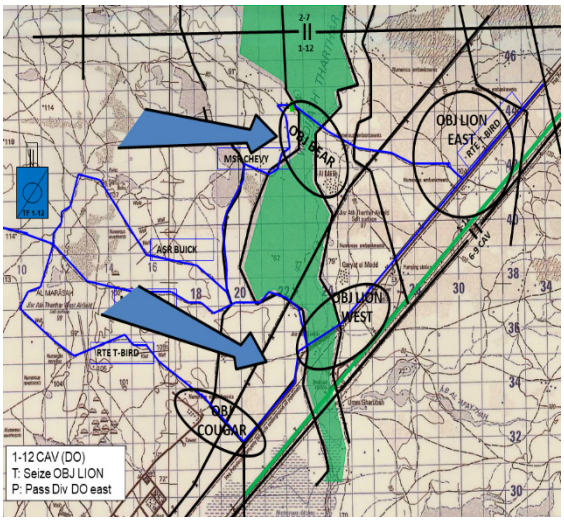

Fig. 2 TF 1–12 CAV area of operations (AO) in Tiger Claw

2.2 StarCraft II Simulation Environment

StarCraft II is a complex real-time strategy game in which players balance high-level economic decisions with low-level individual control of potentially hundreds of units in order to overpower and defeat an opponent force. StarCraft II has a number of difficult challenges for AI that make it a suitable simulation environment for C2 in MDO. For example, the game has complex state and action spaces, and can last tens of thousands of time steps with thousands of actions selected in real time and captures uncertainty due to the partial observability or "fog of war" from the game. Further, the game has heterogeneous assets available for MDO emulation, an inherent C2 architecture, embedded military (kinetic) objectives, and a shallow learning curve for implementation/modification compared to more robust simulations (e.g., One Semi-Automated Force [OneSAF]). DeepMind's SC2LE framework exposes Blizzard Entertainment's StarCraft II machine learning application programming interface as an RL environment. This tool provides access to StarCraft II and associated map editor and an interface for RL agents to interact with StarCraft II, getting observations and sending actions.

As part of the DSI, a SC2LE map was developed based on the Tiger Claw OPORD and supporting documentation (Fig. 3). The game was militarized by re-skinning the icons to incorporate 2525B military symbologies and unit parameters (weapons, range, scaling) associated with the Tiger Claw scenario. The internal scoring system was repurposed to calculate a reward function for RL, which includes convergence of mission objective (cross the wadi), minimization of Blue attrition, and maximization of Red attrition.

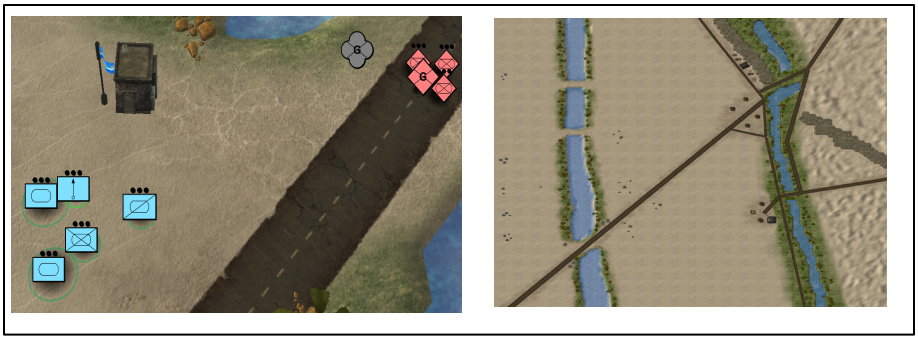

Fig. 3 Tiger Claw map in StarCraft II

2.2.1 StarCraft II Editor

The Tiger Claw scenario is re-created in StarCraft II using its Editor. This Editor is included with the free download of StarCraft II from Blizzard Entertainment and has many capabilities for creating custom content. A good resource for mastering the capabilities are the online community forums dedicated to the Editor. In the following sections, the development of the map, units, and rewards using the Editor are discussed in detail.

2.2.1.1 StarCraft II Map Development

We created a new Melee Map for the Tiger Claw scenario using the StarCraft II Editor. The map size was the largest available in the Editor (256 by 256), using the StarCraft II coordinate system. A wasteland tile set was used as the default surface of the map since it visually resembled a desert region in the AO in Tiger Claw (Fig. 4).

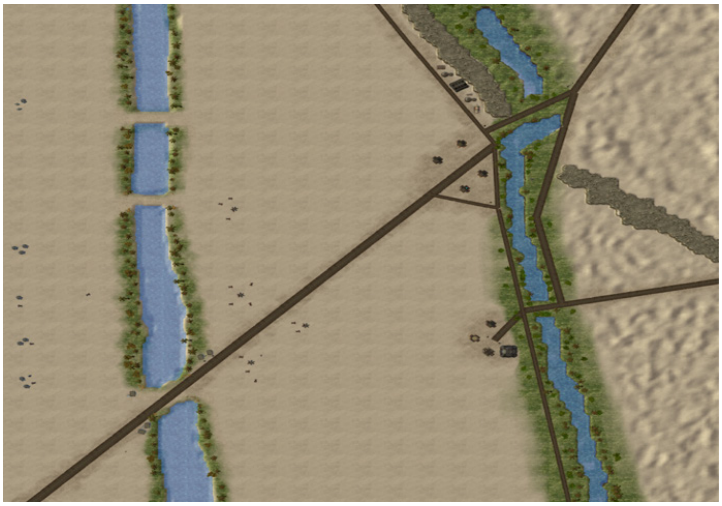

Fig. 4 Initial Tiger Claw map in the StarCraft II Editor

After the initial setup, we used the Terrain tools to modify the map to loosely approximate the AO. The key terrain feature was the impassable wadi with limited crossing points.

Distance scaling was an important factor for the scenario creation. In the initial map, we used the known distance between landmarks to translate the StarCraft II distance, using its internal coordinate system, into kilometers. This translation is important for adjusting weapons range during unit modification (Fig. 5).

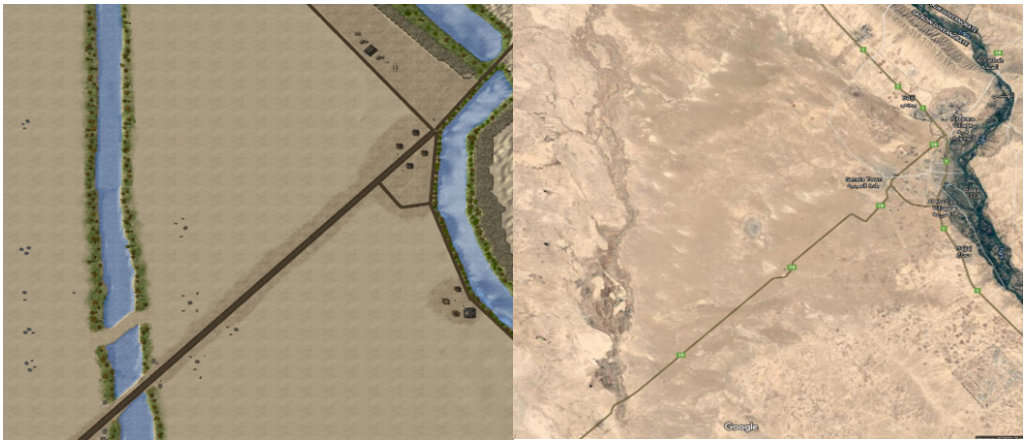

Fig. 5 Modified StarCraft II Editor map

Initial experiments used StarCraft II to visualize the simulation replays. The game-like feel of these replays became a noted distraction. To remedy this issue, other methods of visualization were desired, specifically Aurora, a mixed-reality environment developed by ARL. The new visualization uses geographic maps of the AO. As a result, it was necessary to modify the StarCraft II map to align to the latitude and longitude of the AO. In the modified map, distance scaling was determined by translating the StarCraft II coordinates to latitude and longitude.

2.2.1.2 StarCraft II Unit Modification

To simulate the Tiger Claw scenario, we selected StarCraft II units that approximated the capabilities of military units. The StarCraft II units were duplicated and their attributes modified in the Editor to support the scenario.

First, we modified the appearance of the units and replaced it with an appropriate MIL-STD-2525 symbol (Table 1). In StarCraft II, each unit is associated with multiple actors, which control the appearance of the unit in the game. We were able to unlink the actors from their default renderings, effectively making the units invisible. Next, we imported images of the required military symbols into the Editor. Finally, we used the “rr Sprite Engine” (LGPL 2.1 license) library posted at SCMapster.com to link the units to their military symbol.

Table 1 Mapping of Tiger Claw units to StarCraft II units

Tiger Claw unit	StarCraft II unit
Armor	Siege tank (tank mode)
Mechanized infantry	Hellion
Mortar	Marauder
Aviation	Banshee
Artillery	Siege tank (siege mode)
Anti-Armor	Reaper
Infantry	Marine

Other attributes modified for the scenario included weapon range, weapon damage, unit speed, and unit life (how much damage it can sustain). Weapon ranges were discerned from open-source materials and scaled to the map's dimensions. Unit speed was established in the Tiger Claw OPORD and fixed at that value. The attributes for damage and life were estimated, with the guiding principle of maintaining a balanced conflict. Each StarCraft II unit usually had only one weapon, making it challenging to simulate the variety of armaments available to a company-sized unit. Additional effort to increase the accuracy of unit modifications will require wargaming SMEs.

After modifications, the units were placed on the map to approximate the Tiger Claw scenario (Fig. 6). During experimentations, the Blue Force would be controlled by an intelligent learning agent developed using PySC2 (DeepMind's Python component of the SC2LE). Additionally, the Blue Force units were modified to have no innate aggressiveness. In other words, they would not engage offensively or defensively unless specifically commanded by the agent. To control the Red Force, we used two different strategies. The first strategy was to include a scripted CoA for Red Force movements, which is executed in every simulation. The units default aggressiveness attributes controlled how it engaged Blue. The second strategy was to let a StarCraft II bot AI control the Red Force to execute an all-out attack, or suicide as it is termed in the Editor. The built-in StarCraft II bot has several difficulty levels (1–10), which dictate the proficiency of the bot, where a level 1 is a fairly rudimentary bot that can be easily defeated and level 10 is a very sophisticated bot that uses information not available to players (i.e., a cheating bot). Finally, environmental factors, such as fog of war, were toggled across experiments to investigate their impact.

Fig. 6 StarCraft II using MILSTD2525 symbols

2.2.1.3 StarCraft II Rewards Implementation

The reward function is an important component of RL and it controls how the agent reacts to environmental changes by giving them a positive or negative reward for each situation. We incorporated the reward function for the Tiger Claw scenario in SC2LE and our implementation overrode the internal SC2LE scoring system. The original scoring system rewarded players for the resource value of their units and structures. Our new scoring system focused only on the military aspect of the game of gaining and occupying new territory as well as destroying the enemy.

Our reward function awarded +10 points for the Blue Force crossing the wadi (river) and –10 points for retreating back. In addition, we awarded +10 points for destroying a Red Force unit and –10 points if a Blue Force unit was destroyed.

To implement the reward function, it was necessary to first use the SC2LE Editor to define the various regions and objectives of the map. Regions are areas, defined by the user, which are utilized by triggers (Fig. 7).

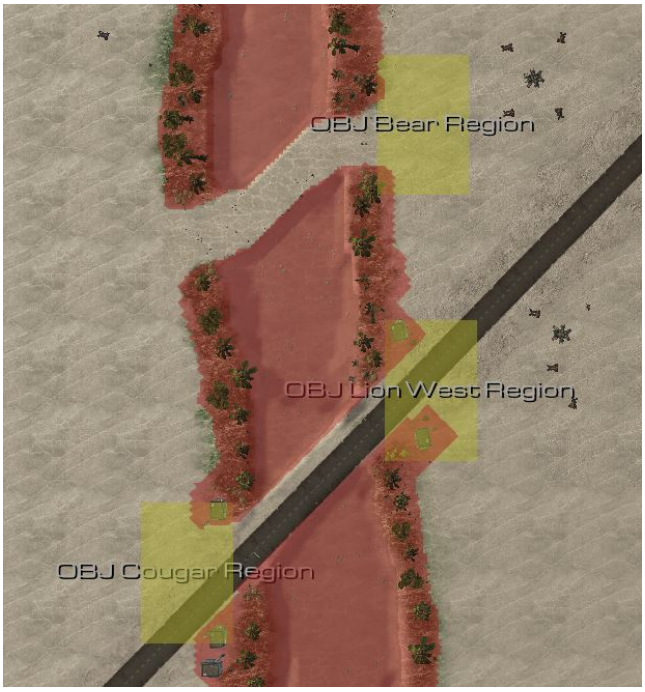

Fig. 7 Regions and objectives in StarCraft II

Triggers are templates for creating a set of instructions that allows the user to program effects into the simulation related to specific events (Fig. 8). In general, a trigger is composed of the following:

- **Events:** Initiates the trigger (i.e., a unit enters a region).
- **Variables:** Stores information. (i.e., BlueForceScore, the score for the Blue Force).
- **Conditions:** A restriction on the action that needs to be true for the action to take place. (i.e., unit is member of Blue Force).
- **Actions:** The result or outcome of the event (i.e., unit gains points).

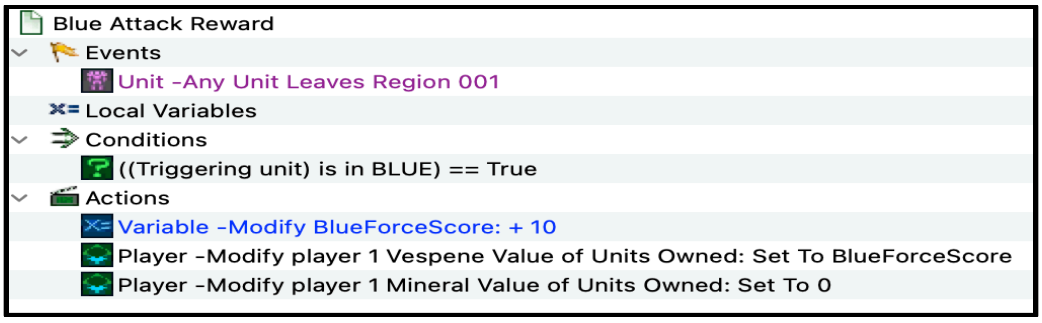

Fig. 8 Example trigger for the Tiger Claw scenario in StarCraft II

As part of future work, we plan to incorporate additional rewards based on the specific team goals defined by the commander's intent in the Tiger Claw Warning Order (WARNORD). The reward function will attempt to train the agent to maintain units as teams, engage the intended objective together as teams, and create optimal behavior that is reasonable to a military SME.

2.3 OpSim Simulation Environment

OpSim is a decision support tool developed by Cole Engineering Services Inc. (CESI) that provides planning support, mission execution monitoring, mission rehearsal, embedded training, and mission execution monitoring and re-planning. OpSim integrates with SitaWare Command, Control, Communications, Computers and Intelligence (C4I), a critical component of the Command Post Computing Environment (CPCE) fielded by the Program Executive Office Command Control Communications-Tactical (PEOC3T), allowing all levels of command to have shared situational awareness and coordinate operational actions, thus making it an embedded simulation that connects directly to operational mission command. It is fundamentally constructed as an extensible service-oriented architecture (SOA)–based simulation and is capable of running faster than current state-of-the-art simulation environments such as OneSAF and the MAGTF Tactical Warfare Simulator (MTWS). While traditional constructive simulations run at most 1–20 times wall clock time, OpSim can run 30 replications of Tiger Claw—which would take 240 h if run serially in real time. Output of a simulation plan in OpSim include an overall ranking of Blue Force plans based on criteria such as ammunition expenditure, casualties, equipment loss, fuel usage, and so on. The OpSim tool, however, was not designed for AI applications and had to be adapted by incorporating interfaces to run DRL-based algorithms. An OpenAI Gym interface was developed to expose simulation state and offer simulation control to external agents with the ability to supply altered actions for select entities within the simulation, as well as the amount of time to simulate before responding back to the interface.

2.4 Deep Reinforcement Learning with OpenAI Gym and the RLlib Interface

RL can be formalized as a Markov decision process consisting of a set of actions, a transition probability function, a reward signal, and the state of an environment.[32] In RL, the goal is to find an optimal action that maximizes the expected, cumulative sum of discounted rewards. Combining deep neural networks with RL, DRL integrates a deep neural network architecture with a RL framework for approximating optimal actions for states within an environment. Design of DRL

entails the following components: state space (environment state representation), action space (set of actions), reward signal(s), and a deep neural network.

For access to the environment states, the RL framework uses an OpenAI Gym-like interface[33] with the OpSim and StarCraft II simulators, providing an abstraction of the environment for RL (Fig. 9). OpenAI Gym is an open-source software package that provides a collection of environments with a common interface for RL development and testing. OpenAI Gym focuses on the abstraction of an environment for RL, thereby keeping an agent development flexible. The specific action, state space, and the reward signal used in the two simulation environments will be discussed in detail in the subsequent sections.

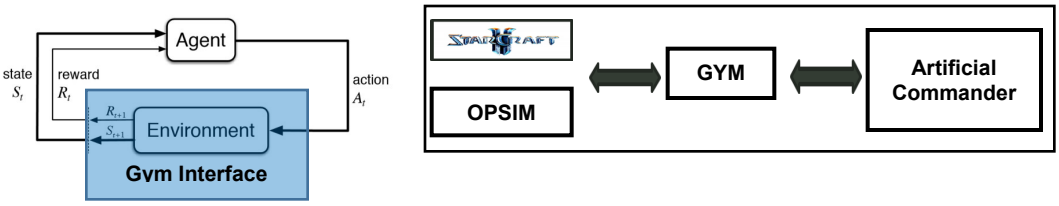

Fig. 9 RL framework using OpenAI Gym with the OpSim and StarCraft II simulators

DRL requires many episodes of an agent interacting with an environment for collecting experiences, and a standard approach is to scale through parallel data collection. In this project, HPC is being leveraged to scale DRL algorithms to support a population of agents learning from many thousands of parallel instances to solve the action space complexity of C2. ARL's FOB system was initially utilized for distributed training, then ported to the DOD Supercomputing Resource Center's (DSRC's) latest SCOUT system. The FOB system is an experimental heterogeneous cluster of 64 nodes, each with an Intel 8-core Xeon CPU and 64 GB of memory. SCOUT is an unclassified HPC-in-a-container system located at the ARL DSRC with 22 training nodes and 128 inference nodes. Each of the compute nodes in SCOUT is equipped with an IBM Power9 40-core processor and 256 GB of memory for inference nodes and 700 GB of memory for training nodes.

In conjunction, RLlib, an open-source library for a scalable RL framework developed at the University of California, Berkeley RISELab, was used for executing distributed learning. RLlib provides a framework agnostic mechanism to efficiently scale training of the DRL neural network architecture on both OpSim and StarCraft II. The framework is deployed on the HPC system to demonstrate scaling of RLlib algorithms across multiple nodes of the system and provides flexibility of customizable neural network models and simulation environment.

3. Results and Discussions

Using the infrastructure described in Section 2, we developed an end-to-end DRL framework for both StarCraft II and OpSim environments and initial experimentations were conducted. In this section, we describe the network architecture, implementation, and some initial experimentation results.

3.1 Deep Reinforcement Learning Using StarCraft

We trained a multi-input and multi-output deep reinforcement neural network using the tactical version of StarCraft II that is described in Section 2.2. We used the Asynchronous Advantage Actor Critic (A3C) algorithm,[29] a state-input processing approach consisting of multilayer convolution nets with a long short-term memory (LSTM) recurrent layer adding memory to the network.

3.1.1 Asynchronous Advantage Actor Critic Architecture

In StarCraft II, the state space consists of 7 mini-map feature layers of size 64x64 and 13 screen feature layer maps of size 64x64 for a total of 20 64x64 2-D images (left panel of Fig. 9). Additionally, it also consists of 13 non-spatial features containing information such as player resources and build queues. These game features were processed using an input processing pipeline, as shown in Fig. 10. The actions in StarCraft II are compound actions in the form of functions that require arguments and specifications about where that action is intended to take place on the screen. For example, an action such as “attack” is represented as a function that would require the x-y attack locations on the screen. The action space consists of the action identifier (i.e., which action to run) and two spatial actions (x and y) that are represented as two vectors of length 64 real-valued entries between 0 and 1. Table 2 delineates the observation space, action space, and rewards for the StarCraft II simulations.

Figure 10 provides an overview of the state input processing pipeline for the mutual-embedding model and the A3C agent for the StarCraft II task. StarCraft II provides three primary streams of state information: mini-map layers, screen layers, and non-spatial features (such as resources, available actions, and build queues). The mini-map and screen features were processed by identical two-layer convolutional neural networks (CNNs) (top two rows) in order to extract visual feature representations of the global and local states of the map, respectively. The non-spatial features were processed through a fully connected layer with a nonlinear activation. These three outputs were then concatenated to form the full state-space representation for the agent, as well as for the state-based portion of the mutual-embedding model.

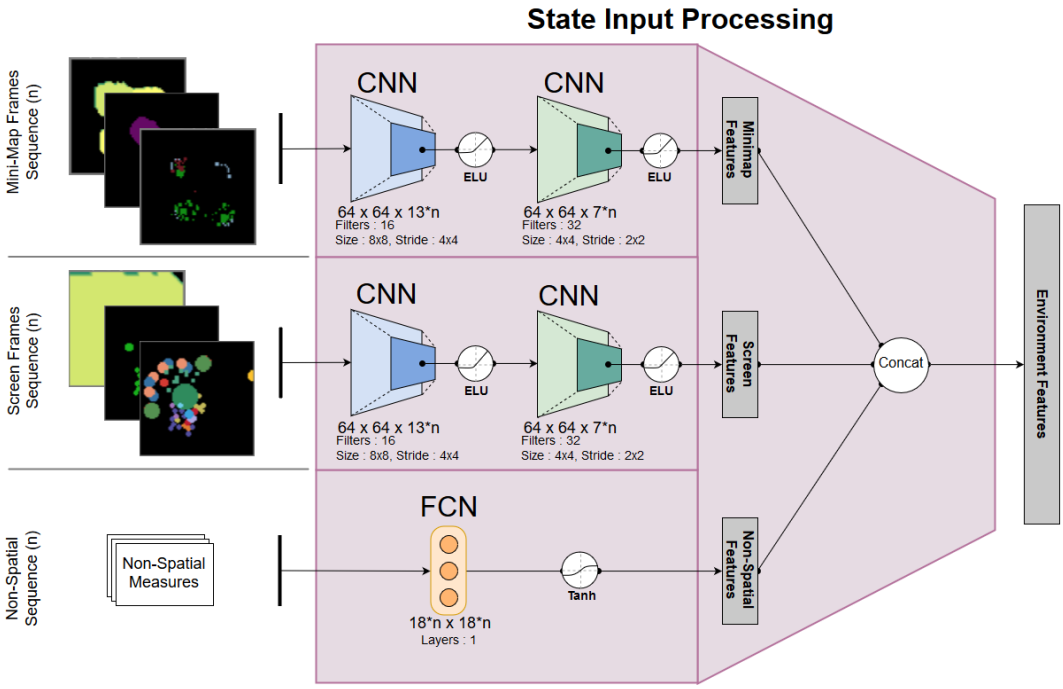

Fig. 10 State input processing for StarCraft II

Table 2 Observation space, action space, and rewards for the StarCraft II simulations

Observation space	Screen images, mini-map images, non-image features
Action space	No-op, select point, attack position, move, patrol, select Army
Rewards	OPFOR platoon destroyed (10), BLUEFOR platoon destroyed (–10), BLUEFOR platoon crosses the Wadi to the east (+10)

Note: OPFOR = Opposing Force, BLUEFOR = Blue Force.

The A3C is a distributed version of the advantage actor-critic algorithm in which multiple, parallel copies of the actor are created to execute actions and collect experience simultaneously. Having multiple actors collect experience increases the exploration efficiency and thus improves learning. The architecture of the A3C agent we use is similar to the Atari-net agent from Mnih et al.,[34] which is an A3C agent adapted from Atari to operate on the SC2LE state and actions space. We make one slight modification to this agent and add an LSTM layer as it was shown in Mnih et al.,[34] that adding memory to the model to improve performance. The architecture of our A3C agent is shown in Fig 11.

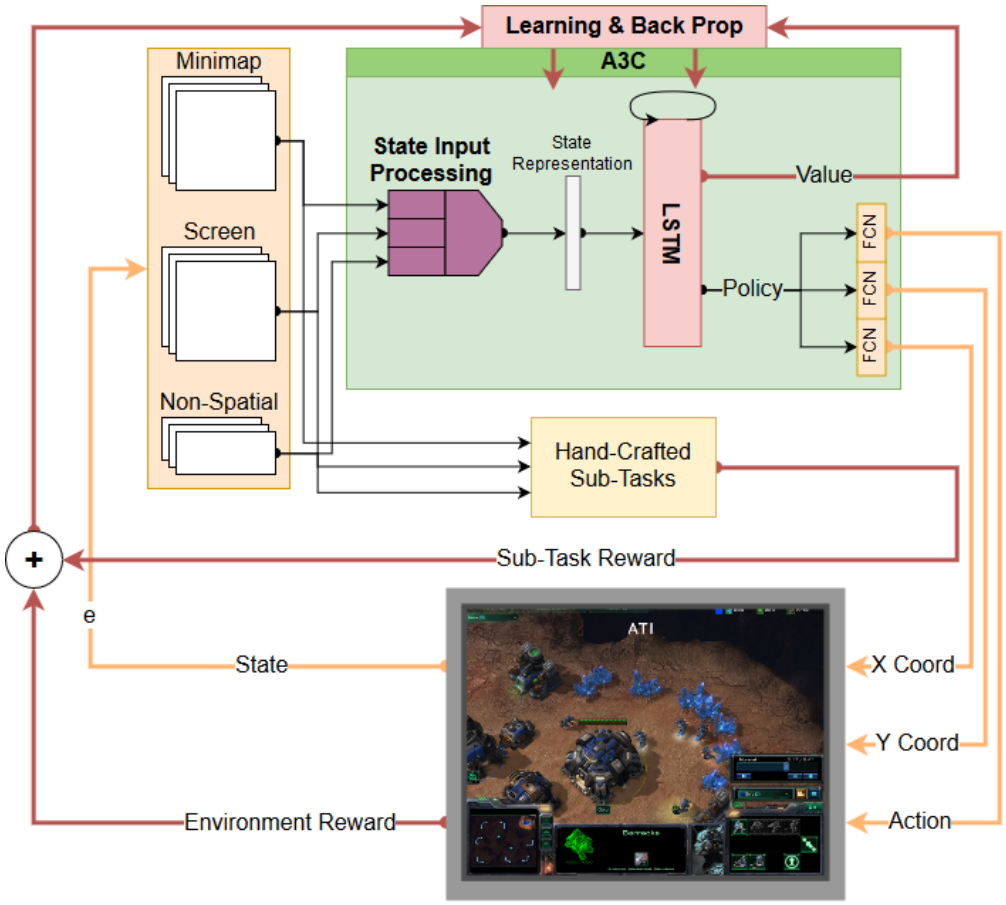

Fig. 11 Architecture of the A3C agent. Shown here is a schematic diagram of the full RL agent and its connection to StarCraft II. As typical of an on-policy agent, the A3C agent here (in green) takes in states and reward information from the task environment and uses the information to compute actions for the next time step, as well as compute gradients to increment reward maximization.

3.1.2 Experiments and Results

We trained the A3C models with 20 parallel actor-learners using 8,000 simulated battles against StarCraft II bot operating on hand-crafted rules developed by DeepMind. A positive reinforcement of +10 was provided if the BLUEFOR crosses the wadi or OPFOR platoons are destroyed and a negative reinforcement –10 is provided if the BLUEFOR is destroyed.

We tested the trained A3C model on 100 rollouts of the agent on the StarCraft II Tiger Claw scenario. The models were compared against a random baseline with randomized actions as well as a human player playing 10 simulated battles against the StarCraft II bot. Summary plots of the metrics collected including the total episode reward and number of Blue Force casualties are provided in Fig. 12. We see that the AI commander has not only achieved comparable performance

compared to a human player, but has also performed slightly better at the task while also reducing Blue Force casualties.

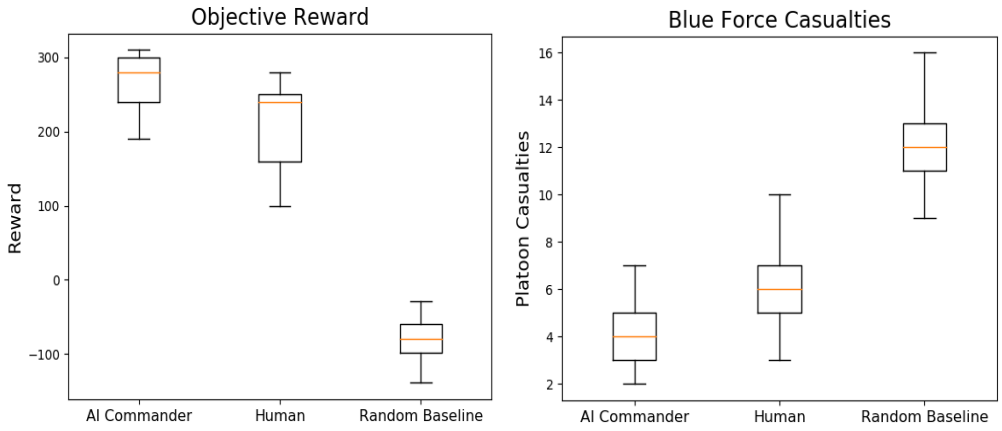

Fig. 12 Total reward and BLUEFOR casualties of the trained AI commander (A3C agent) compared to human and random agent baselines. The AI commander is able to achieve a reward that is comparable (and slightly better) than the human baseline while taking a reduced number of Blue Force casualties.

3.2 Deep Reinforcement Learning with OpSim

Two types of commanders were developed for the OpSim simulation environment. The first is based on an expert designed rule engine developed by military SMEs at Ft Benning, Georgia, using doctrinal rules. The second is a DRL-trained neural network with a multi-input and multi-output LSTM neural network trained with the A2C algorithm. A2C is similar to A3C but without the asynchronous part. OpSim's RL interface supports multi-agent training where each force can be either rule-based or an AI commander.

The policy network was first trained on 15 nodes on FOB with 75 parallel workers collecting 482k simulated battles taking 36 h. This is a significant performance speedup from a serialized version that would take 78 days to simulate 482k battles. Furthermore, local tangent plane position and no goal reward updates were applied and trained on the SCOUT system. With the updated observations and rewards, 39 parallel workers collected 175k battle experiences, which took 37 h.

The observation space consists of 17 features vector where the observation space is partially observable based on each entity's equipment sensors. Unlike S2CLE, OpSim currently does not use image inputs or spatial features from the screen images. The action space primarily consists of simple movements and engagement attacks (Table 3).

Table 3 Observation space, action space, and rewards for OpSim simulations

Observation space	Damage state, x location, y location, equipment loss, weapon range, sensor range, fuel consumed, ammunition consumed, ammunition total, equipment category, max speed, perceived opposition entities, goal distance, goal direction, fire support, taking fire, engaging targets
Action space	No op, move forward, move back, move right, move left, speed up, slow down, orient to goal, halt, fire weapon, call for fire, react to contact,
Rewards	Friendly damaged (–0.5), friendly destroyed (–1.0), enemy damaged (0.5), enemy destroyed (1.0) , –0.01×km from goal destination

3.2.1 Experiments and Results

Trained models are evaluated with 100 rollout simulation results using the frozen policy at a checkpoint with the highest mean reward for BLUFOR. On SCOUT, checkpoint 4510 reached a rolling average of 200 for the BLUFOR policy mean reward and –322 for the OPFOR policy mean reward. Analysis of 100 rollout show that the DRL trained BLUFOR agent minimizes loss from about 4 to 0.5 and increases OPFOR's losses (Fig. 13). This outcome is reached by employing a strategy to engage using only combat armor companies and fighting infantry company. It has learned a strategy to utilize BLUFOR's most lethal units with Abrams and Bradleys while protecting vulnerable assets from engaging with OPFOR (Fig. 14).

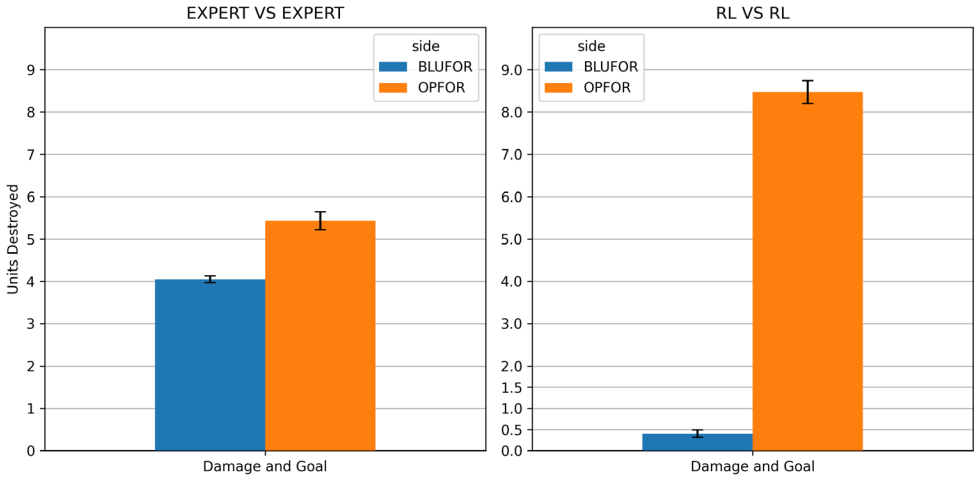

Fig. 13 Entity loss comparison between SME and AI commanders

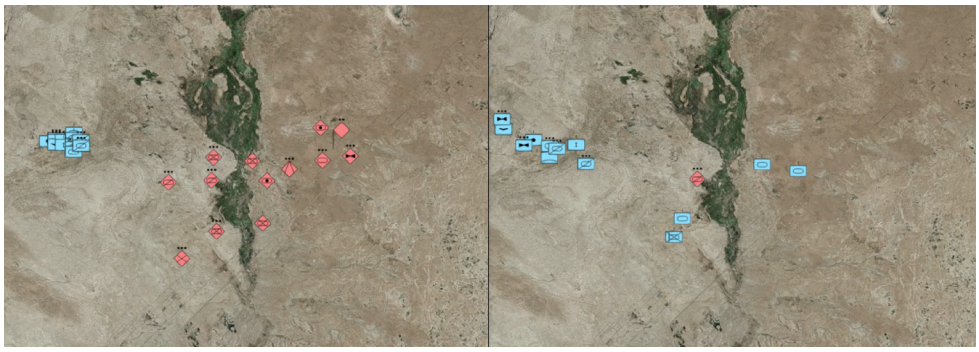

Fig. 14 Snapshot of beginning and end of one rollout

4. Conclusion

As part of the DSI, two novel testbeds for DRL of C2 were developed: StarCraft II-based and OpSim-based. End-to-end DRL methods were developed using these state-of-the-art testbeds. The infrastructure is ported to the DOD's HPC systems to scale the training for parallel data collection.

Initial experimentation results showed preliminary observations that DRL achieved effective and reasonable C2 without pre-coded knowledge, and a DRL-based "artificial commander" could execute an integrated planning-execution process in a simulated brigade-scale battle. Some of the results, especially in the StarCraft II environment, have shown that the strategies taken by the AI are comparable to that of a competent human player. It has also showed that computational resources are not a showstopper for AI in C2; we see adequate speed of learning using the HPC system with convergence in 37 h. Overall, the first year of the DSI has provided adequate evidence that learning-based AI has potential to be used as a key technology for future C2 of military operations.

5. References

1. Mousavi SS, Schukat M, Howley E. Deep reinforcement learning: an overview. In: Proceedings of SAI Intelligent Systems Conference (IntelliSys); 2016. p. 426–440.

2. Berner C, Brockman G, Chan B, Cheung V, Dębiak P, Dennison C, Farhi D, Fischer Q, Hasme S, Hesse C, et al. Dota 2 with large scale deep reinforcement learning. arXiv preprint; 2019. arXiv:1912.06680 A-6.

3. Vinyals O, Babuschkin I, Czarnecki WM, Mathieu M, Dudzik A, Chung J, Choi DH, Powell R, Ewalds T, Georgiev P, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature. 2019;575:350–354.

4. Kott A, ed. Advanced technology concepts for command and control. Xlibris Corporation; 2004.

5. Kott A, Ground L, Budd R, Rebbapragada L, Langston J. Toward practical knowledge-based tools for battle planning and scheduling. AAAI/IAAI. 2002:894–899.

6. Ownby M, Kott A. Reading the mind of the enemy: predictive analysis and command effectiveness. Solers Inc.; 2006.

7. Kott A, Corpac PS. COMPOEX technology to assist leaders in planning and executing campaigns in complex operational environments. Defense Advanced Research Projects Agency; 2007.

8. Kott A, Ludwig J, Lange M. Assessing mission impact of cyberattacks: Toward a model-driven paradigm. IEEE Security & Privacy. 2017;15(5):65–74.

9. Bolton J. Overkill: Army mission command systems inhibit mission command. Small Wars Journal. 2017. https://smallwarsjournal.com/jrnl/art/overkill-army-mission-command-systems-inhibit-mission-command.

10. Ballanco E. We need an AI-based enemy analysis tool…Now! Army War College; 2019. https://warroom.armywarcollege.edu/articles/enemy-analysis-tool-now/.

11. Advanced Field Artillery Tactical Data System (AFATDS). Raytheon; 2021 [accessed 2021]. https://www.raytheon.com/capabilities/products/afatds.

12. Osborn K. Superfast blue-force tracking: new gear to exchange data up to 45 times faster. Defense News; 2008 Oct 13.

13. Fuller JV. Information overload and the operational commander. Naval War College, Joint Military Operations Department; 2000.

14. Curts, RJ, Campbell DE. Avoiding information overload through the understanding of OODA loops, a cognitive hierarchy and object-oriented analysis and design. C4ISR Cooperative Research Program (CCRP); 2001.

15. Sgro JD. Less is more: changing the battle damage assessment paradigm. Air Command and Staff College; 2017 Oct 1. https://apps.dtic.mil/docs/citations/AD1047440.

16. Curry H. The current battle damage assessment paradigm is obsolete. Air & Space Power Journal. 2003;18(4):13.

17. Hanna J, Reaper J, Cox T, Walter M. Course of action simulation analysis. Science Applications International Corp (SAIC); 2005.

18. Kott A, Stump E. Intelligent autonomous things on the battlefield. In: Artificial intelligence for the internet of everything. Academic Press; 2019. p. 47–65.

19. Kott A, Alberts DS. How do you command an Army of intelligent things? Computer. 2017;(12):96–100.

20. Hoehn JR. Joint All Domain Command and Control (JADC2). Congressional Research SVC; 2020.

21. Burke, EJ, Gunness KA, Cooper CA, Cozad MR. People's Liberation Army operational concepts. RAND; 2020.

22. Narayanan P, Yeh B, Holmes E, Martucci S, Schmeckpeper K, Mertz C, Osteen P, Wigness M. An integrated perception pipeline for robot mission execution in unstructured environments. Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II. 2020;11413:1141318. International Society for Optics and Photonics.

23. Wu Z, Suresh K, Narayanan P, Xu H, Kwon H, Wang Z. Delving into robust object detection from unmanned aerial vehicles: a deep nuisance disentanglement approach. Proc IEEE/CVF International Conference on Computer Vision. 2019;1201–1210.

24. Waytowich NR, Goecks VG, Lawhern VJ. Cycle-of-learning for autonomous systems from human interaction. arXiv preprint; 2018. arXiv:1808.09572.

25. Goecks VG, Gremillion GM, Lehman HC, Nothwang WD. Cyber-human approach for learning human intention and shape robotic behavior based on task demonstration. 2018 International Joint Conference on Neural Networks (IJCNN); 2018. p. 1–7. IEEE.

26. Waytowich N, Barton SL, Lawhern V, Stump E, Warnell G. Grounding natural language commands to StarCraft II game states for narration-guided reinforcement learning. Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications. 2019;11006:110060S. International Society for Optics and Photonics.

27. Zhou Y, Asher DE, Waytowich NR, Shah JA. On memory mechanism in multi-agent reinforcement learning. arXiv preprint; 2019. arXiv:1909.05232.

28. Asher DE, Barton SL, Zaroukian E, Waytowich NR. Effect of cooperative team size on coordination in adaptive multi-agent systems. Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications. 2019 May;11006:110060Z. International Society for Optics and Photonics.

29. Vinyals O, Ewalds T, Bartunov S, Petko, Vezhnevets AS, Yeo M, Makhzani A, Küttler H, Agapiou J, Schrittwieser J, et al. StarCraft ii: A new challenge for reinforcement learning. arXiv preprint; 2017. arXiv:1708.04782.

30. Surdu, JR, Haines GD, Pooch UW. OpSim: a purpose-built distributed simulation for the mission operational environment. Simulation Series. 1999;31:69–74.

31. Liang E, Liaw R, Mortiz P, Nishihara R, Fox R, Goldberg K, Gonzalez JE, Jordan MI, Stoica I. RLlib: abstractions for distributed reinforcement learning. arXiv preprint; 2017. arXiv: 1712.09381.

32. Sutton, RS, Barto AG. Introduction to reinforcement learning. MIT Press; 1998. p. 342.

33. Brockman G, Sutskever I, Altman S, D'Angelo A, Karnofsky H, Hoffman R, Zilis S, McCauley T. OpenAI Gym. 2016. [accessed 2021]. https://gym.openai.com/.

34. Mnih V, Badia AP, Mirza M, Graves A, Lillicrap T, Harley T, Silver D, Kavukcuoglu K. Asynchronous methods for deep reinforcement learning. Proc 33rd International Conference on Machine Learning, PMLR. 20016;48:1928–1937.

List of Symbols, Abbreviations, and Acronyms

A3C	Asynchronous Advantage Actor Critic
AFATDS	Advanced Field Artillery Data System
AI	artificial intelligence
AO	area of operations
ARL	Army Research Laboratory
BDA	Battle Damage Assessment
BLUEFOR	Blue Force
C2	command and control
C4I	Command, Control, Communications, Computers and Intelligence
CADET	Course of Action Development and Evaluation Tool
CEMA	cyber-electromagnetic activities
CESI	Cole Engineering Services Inc.
CNN	convolutional neural network
CoA	course of action
COMPOEX	Conflict Modeling, Planning and Outcome Experimentation
CPCE	Command Post Computing Environment
CPU	central processing unit
DARPA	Defense Advanced Research Projects Agency
DRL	deep reinforcement learning
DO	decisive operation
DSI	Director's Strategic Initiative
DSRC	DOD Supercomputing Resource Center
HPC	high-performance computing
JFACC	Joint Force Air Component Commander
LSTM	long short-term memory

MDMP	Military Decision-Making Process
MDO	Multi-Domain Operations
MTWS	MAGTF Tactical Warfare Simulator
NATO	North Atlantic Treaty Organization
OneSAF	One Semi-Automated Force
OPFOR	Opposing Force
OPORD	Operation Order
OPTEMPO	operational tempo
PEOC3T	Program Executive Office Command Control Communications-Tactical
R2C2	Robotics Research Collaboration Campus
RAID	Real-time Adversarial Intelligence and Decision-making
RL	reinforcement learning
SC2LE	StarCraft II Learning Environment
SME	subject-matter expert
SOA	service-oriented architecture
WARNORD	Warning Order

1 DEFENSE TECHNICAL
(PDF) INFORMATION CTR
DTIC OCA

1 DEVCOM ARL
(PDF) FCDD RLD DCI
TECH LIB

9 DEVCOM ARL
(PDF) FCDD RLC CI
P NARAYANANA
FCDD RLC NC
M VINDIOLA
S PARK
A LOGIE
FCDD RLC IT
M MITTRICK
J RICHARDSON
D ASHER
FCDD RLH FC
N WAYTOWICH
FCDD RLD
A KOTT