B-ACE: An Open Lightweight Beyond Visual Range Air Combat Simulation Environment for Multi-Agent Reinforcement Learning

Andre R. Kuroswiski
Aeronautics Institute of Technology
Sao Jose dos Campos, SP, Brazil
kuroswsiki@ita.br

Annie S. Wu
University of Central Florida
Orlando, FL, USA
aswu@cs.ucf.edu

Angelo Passaro
Institute for Advanced Studies
Sao Jose dos Campos, SP, Brazil
angelo@ieav.cta.br

ABSTRACT

This paper introduces B-ACE (Beyond Visual Range (BVR) - Air Combat Environment), an open-source simulation framework leveraging the Godot game engine to evaluate Multi-Agent Reinforcement Learning (MARL) for military research and development. Traditional military simulations are often restricted, limiting research discussions and comparisons among different groups. B-ACE addresses this by providing an open, accessible environment that can be easily shared and extended within the research community, ensuring reproducibility and flexibility for further studies. Our approach capitalizes on Godot's high performance and script-based development, offering a cost-effective and customizable solution for creating air combat scenarios. This integration allows rapid prototyping and evaluation of autonomous agent behaviors using existing reinforcement learning frameworks. In the developed scenario, agents should learn to engage in BVR Air Combat, defend itself and a position against enemy aircraft. Using integration with state-of-the-art MARL algorithms, we explore advanced techniques in autonomous agent development within complex Beyond Visual Range (BVR) air combat scenarios. The environment simulates key aspects of air combat, including radar detection, weapons engagement, and tactical maneuvers. While not overly realistic, B-ACE provides a valuable testbed for prototyping and evaluating AI development approaches. Through three study cases, we demonstrate B-ACE's capability to support research in BVR air combat scenarios, including the generation of weapon efficiency models, optimizing baseline behaviors, and training agents using MARL.

ABOUT THE AUTHORS

Andre R. Kuroswiski, M.S., is a Lieutenant Colonel in the Brazilian Air Force. As a fighter pilot and electrical engineer, he became a researcher at the Institute for Advanced Studies (IEAv) in 2018. Since then, he has been working on simulation development to support autonomous agents research and military scenario analysis. Currently, he is a visiting researcher at the University of Central Florida, focusing on enhancing deep reinforcement learning techniques to develop cooperative air combat agents. This work is also part of his Ph.D. program, supported by the Brazilian Air Force Postgraduate Program in Operational Applications (PPGAO) at the Aeronautics Institute of Technology (ITA).

Annie S. Wu, Ph.D., is an Associate Professor of Computer Science and the Director of the Evolutionary Computation Lab at the University of Central Florida. Prior to joining UCF, she served as a National Research Council Postdoctoral Research Associate at the Naval Research Laboratory.

Angelo Passaro, Ph.D., is the Head of the Virtual Engineering Laboratory at the Institute for Advanced Studies (IEAv) and a Professor in the Space Science and Technology Graduate Program at the Aeronautics Institute of Technology (ITA) in Brazil. A researcher at IEAv since 1984, he holds a B.Sc. in Physics, an M.Sc. in Nuclear Physics, and a Ph.D. in Electrical Engineering. His research interests include high-performance parallel programming, nanostructured semiconductor devices (quantum wells, wires, and dots), hypersonics, numerical methods, and computational optimization techniques.

B-ACE: An Open Lightweight Beyond Visual Range Air Combat Simulation Environment for Multi-Agent Reinforcement Learning

Andre R. Kuroswiski
Aeronautics Institute of Technology
Sao Jose dos Campos, SP, Brazil
kuroswsiki@ita.br

Annie S. Wu
University of Central Florida
Orlando, FL, USA
aswu@cs.ucf.edu

Angelo Passaro
Institute for Advanced Studies
Sao Jose dos Campos, SP, Brazil
angelo@ieav.cta.br

INTRODUCTION

This paper introduces the B-ACE (Beyond Visual Range (BVR) - Air Combat Environment), an open-source environment designed to facilitate the experimentation and evaluation of autonomous agents in BVR air combat scenarios, focusing on Multi-Agent Reinforcement Learning (MARL) research. The advancement of autonomous air combat systems has become a major focus in military research, aiming to enhance the performance of Unmanned Combat Aerial Vehicles (UCAVs) in complex environments. To address this, DARPA created the Air Combat Evolution (ACE) program to increase trust in combat autonomy. In 2024, ACE successfully conducted in-flight tests of AI algorithms, enabling an F-16 to autonomously engage in Within Visual Range (WVR) combat against a human-piloted aircraft (DARPA, 2023). These advancements are largely related to the progress in Reinforcement Learning (RL) algorithms, which allow an AI model to continuously learn from its own errors to achieve desired behaviors (Pope et al., 2021). While ACE has been focusing on WVR air combat development for individual aircraft, many recent academic works are already exploring more complex scenarios involving multiple agents in both WVR and BVR engagements (P. R. Gorton et al., 2024).

BVR air combat, characterized by engagements beyond visual contact, presents distinct challenges where coordinated tactical decisions are crucial. These engagements occur in dynamic scenarios with states determined by partial and uncertain information from onboard sensors and allies (D. Hu et al., 2021). The complexity of BVR air combat necessitates sophisticated AI techniques to enhance learning efficiency, with MARL proving to be a feasible alternative (D. Hu et al., 2021; Wang & Wang, 2024). Despite recent advances, MARL research, even for general approaches, still faces significant challenges due to a lack of standardization, which limits fair comparisons (Gorsane et al., 2022). Additionally, developing MARL-based agents typically requires thousands or even millions of simulation runs to achieve reasonable results, making their effectiveness heavily dependent on the capabilities of the simulation frameworks. These challenges are further compounded in air combat research, as in many other military scenarios, by the reliance on restricted data. This reliance limits reproducibility and makes it difficult to build upon existing solutions. In this context, although academic research on UCAV agents has grown significantly in recent years, it is often conducted using proprietary governmental platforms or unique solutions (P. R. Gorton et al., 2024) those lack standardization, hindering broader collaborations.

B-ACE aims to balance realism and accessibility by providing a simplified yet representative simulation of BVR air combat dynamics avoiding the use of restricted military data and focusing on key aspects of the decisions challenges. Built on an open-source framework, the environment leverages the high-performance capabilities of this game engine and integrates off-the-shelf implementations of the most common MARL algorithms, allowing rapid prototyping for researchers using state-of-the-art solutions. The main contributions of B-ACE are as follows:

1. B-ACE is an open BVR air combat simulation, allowing researchers to easily access, modify, and extend the environment to suit their research needs.
2. B-ACE adheres to the Gymnasium and PettingZoo standards, which are widely used in the reinforcement learning community to standardize environments for single and multi-agent learning. This adherence ensures compatibility with existing MARL frameworks and facilitates the comparison and reproduction of results.
3. By abstracting away certain complexities and focusing on essential BVR air combat dynamics, B-ACE enables rapid prototyping and experimentation of MARL algorithms without compromising the core challenges of the domain.
4. B-ACE demonstrates the capabilities of Godot Engine (Godot Foundation, 2024) as a competitive alternative for military simulation research.

BACKGROUND AND RELATED WORK

Simulation Environments for BVR Air Combat

The simulation of BVR air combat scenarios has been a crucial aspect of pilot training for several decades and is now essential for developing autonomous agents. Recent efforts in Machine Learning (ML) approaches for agent development have employed various simulation engines, divided between proprietary platforms and custom solutions (P. R. Gorton et al., 2024). Government-owned platforms are commonly chosen due to research support from defense institutions, and adapting established air combat training simulations is a straightforward approach. Since these simulations are built for real-time training, however, they can be inefficient for ML applications, as training autonomous agents often requires thousands or even millions of simulation runs. For instance, the Next Generation Threat System (NGTS) is a government-owned synthetic environment generator sustained by the Naval Air Warfare Center Aircraft Division (NAWCAD). NGTS serves as the backbone of most Naval Aviation ground-based simulators. It has been explored for some ML studies like (Abbott et al., 2010; P. R. Gorton et al., 2023), but presents limitations for broader research due to challenges in running multiple simulations. The Advanced Framework for Simulation, Integration, and Modeling (AFSIM), a more recent project from the Air Force Research Lab (AFRL), offers more flexibility for ML research and has been used to develop air combat agents (Floyd et al., 2017) , but it is only accessible to DoD partners. TACSI (Tactical Simulation) is another simulator used as a tactical environment for manned simulators at Saab Aerosystems that has been adapted for ML research (Johansson, 2018), however with restricted access.

To address the need for higher performance simulations for ML research, many studies rely on custom solutions built from scratch or based on existing libraries and development frameworks. JSBSim, for instance, is an open-source flight dynamics framework frequently used in air combat research (P. R. Gorton et al., 2024). While it does not address combat directly, it offers well-established aerodynamic models for real aircraft such as the F-16. IAGSim leverages JSBSim and integrates it with the Unity game engine (Juliani, Berges, et al., 2018) to represent BVR air combat scenarios (Qian et al., 2024). This approach has achieved high performance in deep reinforcement learning research, yielding promising results (Li et al., 2024; Qian et al., 2024), but the solution is not publicly available. The Mixed Reality Simulation Platform (MIXR), an open-source alternative from AFRL, also uses JSBSim and provides some off-the-shelf models for BVR combat simulations, though significant development effort is still required for efficient utilization in MARL applications. The Aerospace Simulation Environment (Ambiente de Simulação Aeroespacial - ASA in Portuguese) project by the Brazilian Air Force (Dantas et al., 2022) leverages MIXR models to create its own air combat simulation, supporting many works in air combat research (Dantas et al., 2021; Kuroswiski et al., 2023; Lima Filho et al., 2022), but this solution, including the BVR engagements, is also not publicly accessible. Similarly, the WUKONG platform has supported several studies exploring MARL for BVR air combat (Li et al., 2024; Qian et al., 2024) but remains a proprietary solution. To our knowledge, BVRGym (Scukins et al., 2024) is the only environment for BVR engagement simulations that is open to the public, developed specifically for reinforcement learning research. BVRGym is the closest solution to our proposed B-ACE, but it aims to support the development of low-level control of the aircraft, while we aim to learn at a higher level with multi-agent tactics decision-making.

The B-ACE environment aims to address the limitations of existing solutions, such as restricted accessibility, lack of standardization, and the high complexity of simulations, by providing an open and flexible simulation platform specifically designed for MARL research in BVR air combat scenarios. It does not aim to replace proprietary solutions, which contain much of the organizational doctrine and restricted knowledge, but serves as a lightweight alternative for initial research, allowing the evaluation of experimental settings in a simplified manner, exploring innovative approaches, and easily comparing previous benchmarks.

Multi-Agent Reinforcement Learning Approach

MARL is a framework in which agents learn to make decisions through interactions with an environment and each other. Formally, MARL can be modeled as a Markov Decision Process (MDP), extended to accommodate multiple agents. In this setting, the environment is described by a set of states S, a set of actions A_i for each agent i, a state transition function $\mathrm{P}: \mathrm{S} \times \mathrm{A}_1 \times \mathrm{A}_2 \times \ldots \times \mathrm{A}_\mathrm{n} \rightarrow \Delta(\mathrm{S})$, where $\Delta(S)$ denotes the probability distribution over states, and a reward function $R_i: S \times A_i \rightarrow R$ for each agent.

In the context of BVR air combat, the application of MARL is particularly challenging. Each aircraft (agent) must make decisions based on limited information and learn to cooperate or compete with other agents in a highly dynamic and adversarial scenario (D. Hu et al., 2021). The state space includes possible configurations of aircraft positions, velocities, weapons, sensors, and other relevant air combat parameters. The action space A_i includes maneuvers, weapon deployment, and other tactical decisions available to each aircraft. The transition function P in a BVR scenario is influenced by the physics of flight, weapon dynamics, and interactions between different aircraft and sensors (Scukins et al., 2024). The reward function R_i is designed to reflect the mission objectives, such as minimizing risks of being hit, maximizing enemy damage, and achieving strategic positions. In a MARL framework, each agent aims to learn an optimal policy $\pi_i: S \rightarrow A_i$ that collectively maximize their expected cumulative reward. This learning process involves exploring different actions and updating policies based on observed shared rewards and state transitions.

The BVR air combat problem is both adversarial and cooperative. It is adversarial because active enemies react to the agents' behaviors, creating a constantly changing threat landscape. It is cooperative because allied agents need to work together, sharing information and coordinating their actions to achieve better results. Agents communicate with each other to share their states and detected targets. In real-world scenarios, this communication typically occurs via radio transmissions involving human interactions and data sent through datalink. Autonomous agents can enhance the utilization of shared data among allies, with MARL studies providing insights into how the amount and frequency of data can improve or limit the development of such systems.

The challenges in a BVR scenario, such as balancing adversarial and cooperative dynamics, managing partial observability, and ensuring effective coordination among agents, are similar to the common challenges addressed in MARL research. This alignment allows researchers to leverage state-of-the-art MARL solutions and lessons learned to tackle BVR air combat problems. Developing and comparing MARL applications, however, still face significant challenges due to the lack of standardization (Gorsane et al., 2022). To address these challenges and promote MARL development, there is a growing trend toward standardization and the use of open-source solutions. OpenAI Gym, now rebranded as Gymnasium (Towers et al., 2023), is a protocol designed to standardize the interface between RL environments and algorithms, ensuring consistent input and output formats. PettingZoo (Terry et al., 2021) extends this concept to multi-agent environments, allowing multiple agents to interact within the same environment. Both protocols facilitate the training and evaluation of RL and MARL algorithms by providing a unified structure for environment interaction, making it easier to implement, compare, and benchmark different algorithms across a variety of tasks. Additionally, BenchMARL (Bettini et al., 2023), Tianshou (Weng et al., 2022), and MARLLib (S. Hu et al., 2023) are recent alternatives that simplify the development of benchmarks for MARL problems, offering flexible implementations of state-of-the-art algorithms. Integrating military research scenarios into these solutions can be an efficient way to increase collaboration and advance the field.

The Godot Game Engine as a Simulation Framework

With the advances in game engines such as Unity and Unreal, their use for military simulation has become a common approach due to their high performance, maturity, and simplified implementation (Goecks et al., 2023). As commercial frameworks, however, they bring intrinsic limitations to broadening the academic benefits of research based on these solutions. Godot (Godot Foundation, 2024), an open-source alternative, is gaining increasing relevance among game developers and has the potential to be a viable alternative for military simulations and serious games.

The selection of the Godot game engine for developing the B-ACE environment is motivated by its open-source nature, performance capabilities and ease of use. Godot's open-source nature provides complete access to the source code, enabling deep customization, bug fixing, and feature development. This makes it easier to reproduce results, reduces costs, and offers flexibility for academic military simulations (Mohd et al., 2023). Godot achieves strong performance capabilities comparable to traditional engines like Unity or Unreal. Its optimized scene system and lightweight architecture support complex simulations with minimal computational overhead (de la Torre et al., 2024; van Rozen, 2023). Although the Godot community is smaller than that of Unity or Unreal, it has shown significant growth and potential (Holfeld, 2023). The community-driven development model ensures continuous improvement in response to user feedback, making it advantageous for simulation research that benefits from constant enhancements.

The Godot engine's intuitive interface and flexible scripting language (GDScript) facilitate rapid prototyping and iteration. Its user-friendly design lowers the barrier to entry for new developers and researchers (van Rozen, 2023). Additionally, it supports multiple alternative languages, including C#, VisualScript, and C++, which can enhance flexibility and performance. Godot is completely free under the MIT license, allowing unrestricted access, modification, and distribution of its source code (Godot Foundation, 2024). This contrasts with commercial engines, which impose costs and restrictions, especially for military and government applications. Godot's integration with reinforcement learning frameworks, such as the Godot RL Agents (Beeching et al., 2021) enables the development of environments and facilitates agent behavior learning using deep reinforcement learning algorithms. This integration with Python-based libraries accelerates the development and testing of autonomous agents. These characteristics demonstrate how the Godot Engine can be a compelling alternative for military simulation projects and academic research. The authors inform that they have no partnership with the Godot project; our endorsement is based solely on our experiences and needs in developing a simulation prototype for reinforcement learning.

B-ACE ENVIRONMENT

The B-ACE environment is a simulation solution designed to represent BVR Air Combat scenarios, facilitating the research and development of autonomous agents for academic and training purposes. We built the B-ACE environment using the Godot framework, leveraging Godot-RL (Beeching et al., 2021) to simplify the implementation of MARL solutions. This integration allows the use of Python scripts to train and evaluate models for BVR agents' behaviors using the standard Gymnasium protocol, ensuring compatibility with PettingZoo and state-of-the-art MARL frameworks as BenchMARL, MARLLib and Tianshou (Bettini et al., 2023; S. Hu et al., 2023; Weng et al., 2022).

Model Definitions

The primary objective of the B-ACE environment is to serve as a tool for evaluating MARL solutions aimed at developing efficient cooperative behaviors. Our model design prioritizes simplicity while accurately representing BVR air combat scenarios to necessitate realistic decision-making processes. Most studies on BVR agents utilize highly detailed models for aircraft and sensors, often controlling the fighters at the flight command surfaces (Zhou et al., 2023). This level of detail, however, can be excessive and complicate BVR behavior research by requiring the model to learn basic aircraft control, which can be effectively managed by established control theories or lower-level ML models. In our study, we focus on higher-level decision-making, presuming that basic flight operations are already addressed. Therefore, the physical models do not need to be highly detailed; they only need to perform adequately when required. For example, executing a 180-degree turn at maximum performance is a typical maneuver in BVR combat. If actions control the basic aircraft commands, the policy must first learn to fly the aircraft before executing tactical actions. By bypassing this phase and assuming the aircraft can fly in a determined direction while maintaining the desired g-force, the policy can concentrate on learning higher-level decisions, which are more relevant to our objectives.

Aircraft, Sensors, and Weapons

The fighter aircraft model in the B-ACE environment is implemented using Godot's CharacterBody3D class (Godot Foundation, 2024). This model encapsulates general properties and behaviors to simulate air combat dynamics relevant to BVR air combat engagement. Key properties include position, rotation, velocity, radar specifications, and missile capabilities. Inputs for heading, flight level, and desired G-force govern the aircraft's movements, assuming an effective control system to achieve the desired conditions. In the B-ACE simulation, fuel consumption is not accounted for, and aircraft operate at maximum available thrust for each altitude, eliminating the need for learning engine operation. This simplification is reasonable for specific combat conditions and is considered sufficient to evaluate the agents' capability to learn general tactical behavior. We also assume that communication systems are always effective, ensuring that agents always receive complete information about their allies. This simplification works for initial evaluations; however, as the models evolve, potential limitations in communication systems could become a crucial aspect to analyze.

Another factor that influences the pilots' decisions is how each aircraft's performance is affected by flight altitude. For instance, higher altitudes result in decreased air density, which reduces aerodynamic forces. Reduced aerodynamic forces can limit available G-force, impacting maneuverability, but can also lead to higher ground speeds due to reduced

drag. Jet engine performance can be more efficient at higher altitudes due to cooler temperatures, but may produce less thrust because of reduced oxygen availability (Anderson & Bowden, 2005). These factors create a complex pattern for overall aircraft performance that must be considered when engaging in a BVR air combat scenario. In B-ACE, we incorporate simplified linear effects of altitude on speed and available G-force. While this approach does not necessarily represent a specific aircraft model, it generates conditions that require agents to adapt their decisions to simulated altitude-related changes in aircraft performance.

In BVR air combat, the onboard radar's capability to detect enemies is a crucial factor, as it becomes the main reference for engagement. Detailed radar simulation can be computationally expensive and dependent on the sensor and target characteristics. We simplify the model by defining three main parameters: maximum detection range, horizontal field of view, and vertical field of view. These parameters define a volume in front of the aircraft where an enemy would be detected. Despite being simplistic compared to real models, this representation captures the essential characteristics of the radar and the basic limitations that the agent needs to consider when making decisions.

The primary weapon in BVR air combat is a long-range air-to-air missile, capable of hitting an enemy aircraft from dozens of miles away. The missile model, though potentially complex, is simplified to focus on the main characteristics that affect decision-making. The key configurable parameters are the maximum time of flight and maximum speed, which determine the missile's capability to hit targets at different ranges. The simulated missile performance is also influenced by altitude and depends on the aircraft's radar. After launch, the aircraft needs to keep tracking the enemy with the radar until the missile can pursue the target independently, defined as 10 nautical miles away. In BVR air combat, successful missile operation is highly dependent on the Weapon Engagement Zone (WEZ) (Dantas et al., 2021). The WEZ continuously calculates the missile's capability to hit the target given the current state, providing crucial information to support the pilot's decision on when to launch the weapon. The WEZ for the default model in B-ACE is integrated into the simulation, and agents have access to this information.

For the default configuration in B-ACE, the aircraft has a maximum speed of 650 knots (1203.8 km/h) and can withstand a maximum g-force of 9g at 25,000 feet (7620 meters), with performance varying based on the altitude of operation, which ranges from 1,000 to 50,000 feet (304.8 to 15,240 meters). The radar has a maximum detection range of 50 nautical miles (approximately 92.6 kilometers) and a field of view of 60 degrees both horizontally and vertically. The aircraft can carry up to six missiles, each capable of flying for 50 seconds at a maximum speed of 3,600 km/h. All models operate with a 20 Hz update rate, which is sufficient given that the simulation does not involve detailed low-level physics. Radar and behavior processing are set to 1 Hz, meaning MARL algorithms interact with the simulation once every second in simulation time.

State Space

The state space is a critical aspect of the MARL process, as it carries all available information about the scenario for the agents when learning and making decisions. In common approaches, the state space variables are the input for AI models such as Neural Networks, which should learn the desired policy. The B-ACE environment defines a comprehensive state space to provide detailed information about the aircraft, allied units, and tracked enemies. This extensive state space is essential for enabling effective decision-making by autonomous agents and allows researchers to explore variable possibilities using all or part of it. Table 1 summarizes the various state variables available for the simulation. The environment outputs normalized values to simplify data processing in external algorithms, maintaining the values within a range of -1.0 to 1.0.

Table 1. B-ACE default Space State variables

Aircraft State (5)	Allies Info (9)	Enemies Info (10)
Aircraft's X position	Allied aircraft's X position	Altitude difference with tracked enemy
Aircraft's Z position	Allied aircraft's Z position	Aspect angle to tracked enemy
Aircraft's altitude	Allied aircraft's altitude	Angle off to tracked enemy
Current heading	Allied aircraft's distance to target	Distance to tracked enemy
Current speed	Allied aircraft's aspect angle to target	Tracked enemy's distance to target
Mission / Navigation State (2)	Allied aircraft's current heading	Own missile RMax against tracked enemy
Distance to target	Allied aircraft's current speed	Own missile Nez against tracked enemy
Aspect angle to target	Allied aircraft's number of missiles	Enemy missile RMax against own aircraft
Missile Status (2)	Allied in-flight missile status	Enemy missile Nez against own aircraft
Number of missiles		Tracked enemy detection status
In-flight missile status		

Action Space

The action space is the only mechanism that allows agents to interact with entities in the simulation. The policy must essentially learn which action to select at every step. The B-ACE environment supports both continuous and discrete types of actions, enabling the use of a wider range of MARL algorithms. In the continuous action space, agents provide continuous values for heading, flight level, desired g-force, and missile firing. These continuous action inputs allow agents to make precise adjustments to the aircraft's heading, altitude, and g-force, as well as to decide the exact moment to fire a missile:

1. **Heading Input**: The desired heading change from the current heading, represented as a continuous input value ranging from -180 to 180 degrees, normalized by 180.
2. **Flight Level Input**: The desired flight level change from the current level, with the input normalized by 25,000 feet.
3. **Desired G-Force Input**: The desired g-force, determined using a continuous input value, scaled to the range between 1g (for -1.0 input) and the maximum g-force capability of the aircraft (for +1.0 input).
4. **Fire Input**: An input value of +1.0 indicates the desire to fire a missile, while any other value means no action is taken.

The discrete action space is a quantized version of the continuous action space. In this space, continuous values for heading, flight level, desired g-force, and missile firing are discretized using predefined conversion tables. These tables map discrete input values to specific continuous values, simplifying the action space while maintaining effective control. Although this quantization can increase compatibility with certain algorithms, it may limit the agent's flexibility depending on the learning strategy and scenario conditions.

Rewards and Penalties

Rewards and penalties provide essential feedback to autonomous agents in a MARL, guiding their learning process. Positive values (rewards) reinforce desired behaviors, while negative values (penalties) discourage undesired behaviors. The environment outputs these feedback values at each simulation step, helping agents adapt their decisions to maximize cumulative rewards while minimizing penalties. The proposed rewards are based on missile operation, radar detection, aircraft status, and overall mission accomplishment. While the primary rewards and penalties are tied to critical events such as hitting an enemy or completing a mission, we also include shaping feedback rewards to guide learning, such as maintaining radar contact with an enemy or stay close to the mission target position. Table 2 summarizes the feedback factors and their default values.

Table 2. Reward and Penalty Factors in B-ACE Environment

Explanation	Default Value
Reward for successfully completing a mission	10.00
Shaping reward related to mission objectives	0.001
Penalty for firing a missile	-0.100
Shaping penalty for firing when not possible	-0.001
Penalty for missing a target with a missile	-0.500
Shaping penalty for losing track of an enemy	-0.100
Shaping reward for maintaining track of an enemy	0.001
Reward for hitting an enemy	3.000
Penalty for being hit	-5.000

Scenario Mission

To define the roles of the agents in the simulation, we allow two types of missions: Defensive Counter Air (DCA) and Airstrike operations (Joint Chiefs of Staff, 2020). For DCA, the agents need to prevent enemies from reaching a critical position. They must learn to keep the enemies away by threatening or neutralizing them while defending themselves. In Airstrike operations, the agents must achieve a target position while avoiding enemy aircraft. Although the air combat engagement behavior with Airstrike role should be similar to DCA, the primary goal is to advance into the field rather than merely holding a defensive position.

In Airstrike missions, the mission is considered accomplished when one aircraft reaches the target position. In contrast, for DCA missions, the critical position must remain safe (i.e., no enemy aircraft reaching it) for a specified time window or until all enemies are neutralized. For both missions, agents do not necessarily need to hit the enemy, as deterring the enemies can also be an effective strategy that reduces the risk of being hit and conserves resources. The simulation is limited to 36,000 steps by default, representing 30 minutes in real time, with agents taking actions every 20 steps (equivalent to 1 second, considering the 20 Hz update rate).

Finite State Machine Baseline Agent

One of the primary goals of B-ACE is to establish a reasonable baseline behavior for evaluating multiple alternative solutions. To achieve this, we developed a Finite State Machine (FSM) based solution that enables agents to make general expected decisions during BVR air combat. The primary decisions of the B-ACE baseline agent include initial defense, last-minute defense, and missile firing moments. By varying these conditions, it is possible to create agents with different combat characteristics, balancing offensiveness and defensiveness (Kuroswiski et al., 2023). The agent takes into consideration WEZ predictions of missile effectiveness to base its decisions, allowing for more conservative or aggressive actions depending on their proximity to critical points (Dantas et al., 2021). Additionally, we created a steady baseline agent, referred to as Duck. This agent maintains a steady flight until it reaches its target position or is neutralized. The Duck provides an alternative baseline that offers a simpler engagement scenario for initial algorithm evaluations. Despite the recent focus on AI-based agents, FSM-based solutions continue to play a crucial role in military simulations due to their explainability and simplicity. Using them as a baseline is an effective starting point for evaluating AI-based models.

Integration with MARL Standards

For B-ACE, we developed a custom wrapper to enable compatibility between Godot-RL (Beeching et al., 2021) and the PettingZoo (Terry et al., 2021) standard. This wrapper facilitates seamless interaction between the Godot environment and MARL algorithms, such as BenchMARL (Bettini et al., 2023) and Tianshou (Weng et al., 2022), supporting parallel execution of multiple environments and agents for comprehensive algorithm evaluation. By adhering to the PettingZoo standard, which is a multi-agent extension of the Gymnasium (Towers et al., 2023), the environment provides users with standardized outputs of states and rewards and accepts standardized inputs for actions in multi-agent simulations. Although it is not mandatory to interact with agents in B-ACE, following these standards offers significant advantages. It increases the possibilities for evaluations and comparisons with state-of-the-art MARL solutions, making it easier to integrate and test various algorithms.

Visualization

As a critical resource to support the verification and validation process, the visualization capabilities of a simulation are crucial. As a game engine, Godot natively allows for the straightforward creation of 3D visualizations and visual debugging resources to analyze behaviors. Figure 1 (left) illustrates a typical BVR scenario with blue agents learning to protect a critical position. As highlighted in the figure, the current visualization includes essential resources to observe a BVR engagement, such as the radar detection volume, aircraft flight trails, critical positions, and the simulation components themselves. Despite its simplicity, the current B-ACE visualization has proven sufficient for initial evaluation. Depending on specific needs, however, additional visual debugging resources can be easily added using Godot's tools. This flexibility ensures that visualization can evolve to meet more complex requirements, enhancing the analysis to support the MARL research.

Experiment Mode

To simplify the development of additional resources, B-ACE includes an Experiment Mode designed to run multiple simulations in parallel. This mode facilitates environment initialization, agent setup, and experiment parameter management from external scripts. Key features include configuring the environment with global engine parameters such as rendering options, seed, and speedup, along with detailed agent and experiment configurations. This mode ensures seamless integration with the Godot engine, enabling efficient base experiments to develop and evaluate models before being integrated into the MARL process. Figure 1 (right) presents the visualization of an experiment running 10 simulations of a BVR 2x2 Air Combat scenario in parallel using the Experiment Mode.

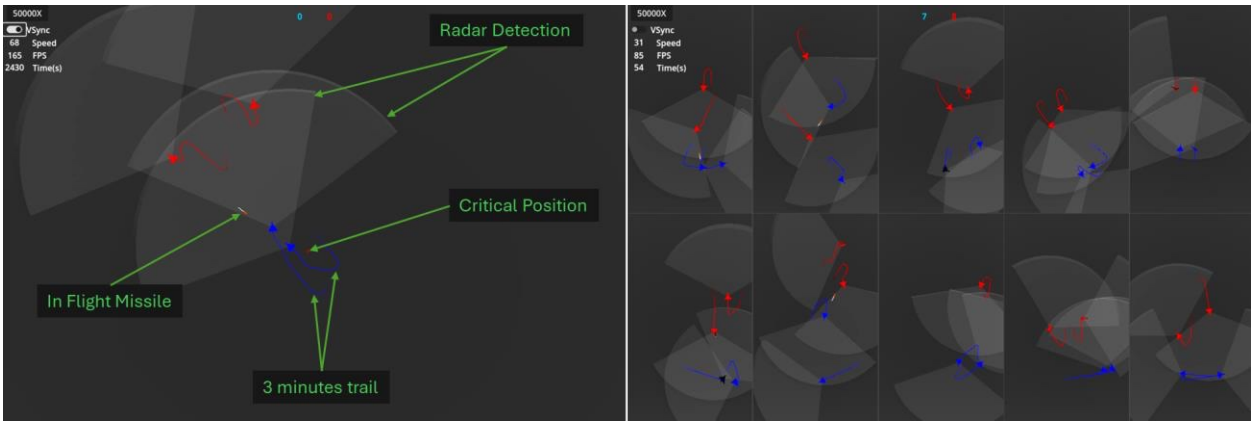

Figure 1. Visualization of BVR 2x2 Air Combat simulations in B-ACE. Snapshot of evaluation in a MARL experiment (left). Multiple simulation running in parallel in the Experiment Mode (right).

PRACTICAL STUDY CASES

The studies presented in this paper serve as a demonstration of the capabilities and possibilities using B-ACE. These preliminary studies showcase potential applications and are ongoing research that will be further elaborated and evaluated in our future publications.

Study Case 1: Missile Weapon Engagement Zone Determination

The first set of experiments demonstrates the capabilities of the B-ACE simulation environment in supporting analysis and model development. A critical aspect of BVR air combat is determining the efficiency of weapons, which guides both offensive and defensive maneuvers. This efficiency is encapsulated in the WEZ model, which predicts the effectiveness of a missile launch at each combat moment. To refine WEZ model predictions, a possible strategy involves running thousands of simulations under varying conditions and using regression algorithms to generate accurate predictions (Dantas et al., 2021). In B-ACE's experiment mode, we run parallel simulations starting with different distances between the shooter and the enemy. From these results, the maximum effective missile distance is determined for multiple conditions. Following the methodology proposed by Cheng et al., 2019, we then generate a comprehensive WEZ model based on a fourth-degree polynomial regression. With the WEZ model as a polynomial

expression, it is possible to use Godot's built-in text-based equation system to integrate the predictive model into the simulation, supporting engagement decisions. Beyond generating the desired information for the agents, analyzing the model results also serves as a tool to evaluate whether the simulated missile behaves as expected. Figure 3 exemplifies the WEZ model predictions for different altitudes and aspects between the two aircraft. This case study highlights B-ACE's ability to efficiently run multiple simulations, generating sufficient data to understand and optimize models for each study's requirements.

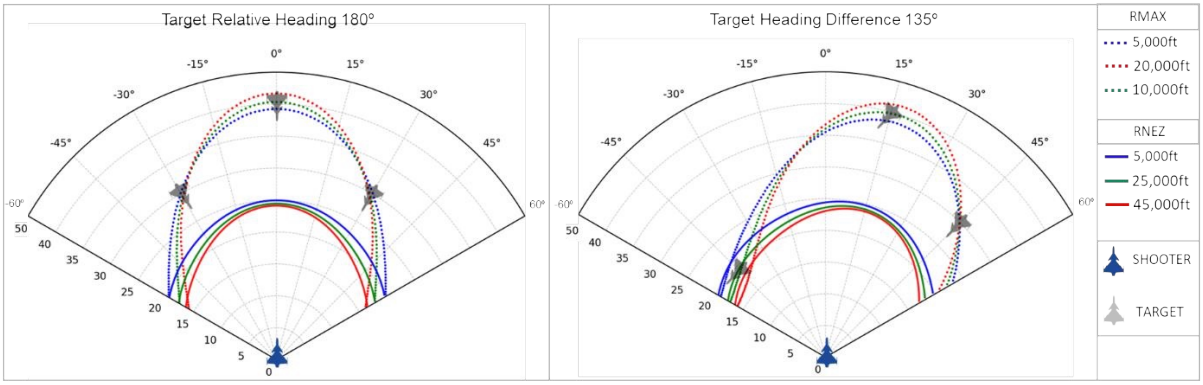

Figure 2. Example of the prediction generated by the WEZ model created based on the default B-ACE Missile performance, evaluated across different altitudes and aspects among agents.

Study Case 2: Optimizing FSM Behavior

The FSM baseline behavior in B-ACE aims to serve as a static reference for evaluating evolving agents. Despite its simplicity, by adjusting the parameters that define the movement and state changes, it is possible to develop a diverse set of agents, ranging from conservative to aggressive. This allows for a variety of baseline enemies, making the learning process more challenging for MARL-based agents. To identify a diverse and strong group of baseline agents, we employed an evolutionary optimization process to search for the best sets of parameters. Our strategy involves running groups of simulations in parallel to evaluate agents against a diverse set of enemies. The goal is to ultimately select a group of ten agents that can be considered diverse and robust adversaries.

Initially, we select ten random agents, each representing a unique set of behavioral parameters. We then optimize a population of 50 agents based on their performance against this initial enemy's group. The evaluation score is the mean difference between the number of times the agent neutralized the enemy and the number of times it was destroyed over 30 simulations. Additionally, every three generations, we update the enemies list by adding the three best new agents and removing the three weakest. This approach ensures that the enemies list continually evolved to include the strongest agents found, with the expectation that the optimization process will yield increasingly capable agents. Given that BVR engagement is an adversarial scenario, the score value is directly dependent on the selected group of enemies. Therefore, agents that perform well initially against the initial random group may not perform as well as the enemies evolve. After 100 optimization iterations, we take all enemies selected during the evolutionary process to define a group of the possibly best agents. Since this group still large, with 87 unique agents, we apply a clustering algorithm to reduce it to the desired group of ten, selecting only the agent with the best score in each cluster.

As an example of the results, the agent with the best score was able to destroy the enemies in 74% of the engagements while being hit in 30% of them. Another agent, although less effective in destroying the enemy with an average success rate of 35%, was only hit 8% of the time. These two agents demonstrate different characteristics that generate different challenges for the adversaries: the more aggressive agent is more effective at destroying enemies but also takes significantly more risks, whereas the other agent is more cautious and still manages to be effective in its own way.

The final ten agents become references for future evaluations in B-ACE. This allows for the selection of baseline enemies ranging from one to ten. This experiment aims to demonstrate a potential strategy for enhancing the use of FSM behaviors to support MARL development. By incorporating diverse agent behaviors, researchers can create more challenging and realistic scenarios, ultimately improving the robustness and effectiveness of MARL agents.

Study Case 3: Development of Autonomous Agents using Reinforcement Learning

In the third study case, we leverage the integration with the BenchMARL solution to evaluate the development of autonomous agents using MARL. We set up a BVR 2x2 scenario, using either a "duck" behavior or a baseline behavior for the enemy agents. The objective is to enhance the capabilities of the allied agents to accomplish their DCA mission of holding a strategic position. For the training process, we apply the multi-agent versions of Proximal Policy Optimization (PPO) as an on-policy alternative and Deep Deterministic Policy Gradient (DDPG) as the off-policy alternative, both of which are common choices for these types of problems (P. R. Gorton et al., 2024). The BenchMARL framework serves as the foundation for the training process and comparison.

The results in Figure 3 illustrate the mean agents' rewards over the training iterations for scenarios against the Duck (left) and Baseline (right) behaviors. The increasing trend in the results demonstrates that the agents are learning to avoid being eliminated by enemies and to maintain control of the critical position. Against the Duck behavior, as expected, since the enemy does not react, the model is able to learn efficient behaviors to prevent the enemies from reaching the critical position. Observing the simulations, it is evident that the agents learn to split up and engage the enemies as soon as possible to eliminate them. The reward values close to 16 are expected to represent this, as eliminating both enemy agents provides a reward of 6.0, plus an additional 10.0 for mission accomplishment, with variations due to penalties and rewards from shaping values. The results against the FSM baseline behavior show how the challenge increases, making the problem much more dynamic and resulting in greater fluctuations during training. In both cases, PPO achieves higher performance and stability. These tests, however, were initial experiments to evaluate the B-ACE infrastructure and are not sufficient to conclusively determine the advantage of PPO without further evaluations.

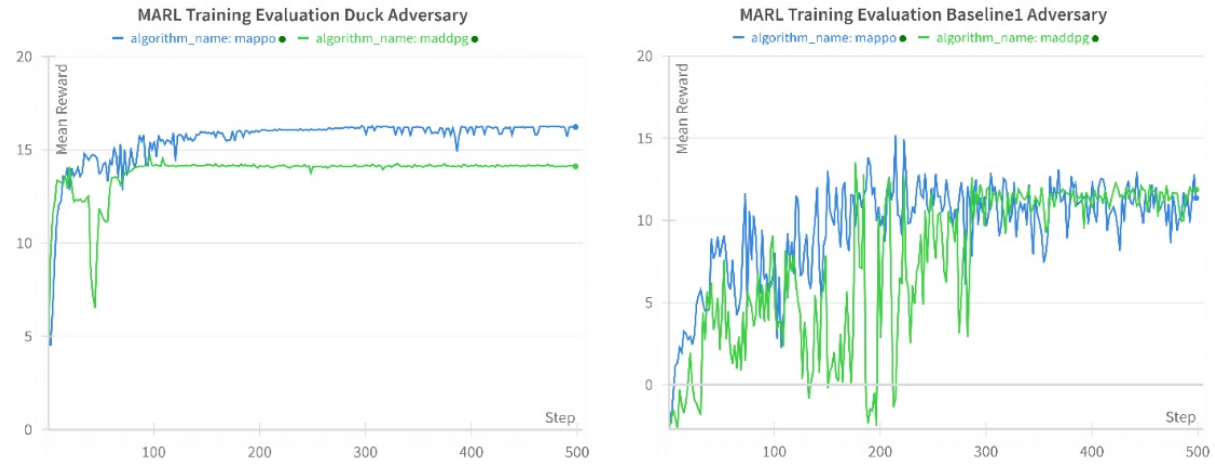

Figure 3. Mean Reward for MARL training in a BVR 2x2 Air Combat Scenario using Multi-Agent PPO and Multi-Agent DDPG

CONCLUSION

In this paper, we introduced B-ACE, an open-source simulation framework built on the Godot game engine designed to support the development and evaluation of autonomous agents using MARL in BVR air combat scenarios. By balancing realism and flexibility, B-ACE provides a simplified yet representative simulation of BVR air combat dynamics without relying on restricted military data.

The B-ACE environment provides a solution that can be easily modified and extended by researchers. It adheres to Gymnasium and PettingZoo standards, ensuring compatibility with existing MARL frameworks and facilitating result comparison and reproducibility. By abstracting certain complexities while maintaining core BVR air combat challenges, B-ACE enables efficient prototyping and experimentation with MARL algorithms. Additionally, the use of the Godot engine demonstrates its suitability for military simulation development.

Through practical study cases, we showcase B-ACE's capabilities in supporting model development and optimizing agent behaviors. The WEZ experiment highlights the B-ACE's efficiency in running multiple simulations to generate comprehensive data for model refinement. The optimization of the FSM baseline behavior provides a stronger benchmark for evaluating MARL solutions. Finally, the development of autonomous agents using MARL demonstrates the framework's potential in training agents to adapt to various enemy behaviors and mission objectives.

B-ACE advances research in autonomous air combat systems by providing a versatile platform for MARL experimentation. Its design supports academic research and serves as a foundation for exploring innovative solutions before applying them to practical systems. Future work will focus on further enhancing the environment's flexibility and realism by adding new models and details of BVR air combat that can be relevant for the agents' decision-making processes.

SOURCE CODE

For further insight and access to the source code of this research, please visit our repository at https://github.com/andrekuros/B-ACE

ACKNOWLEDGEMENTS

This work was supported by the Brazilian Air Force Postgraduate Program in Operational Applications (PPGAO).

REFERENCES

Abbott, R. G., Basilico, J. D., Glickman, M. R., & Whetzel, J. (2010). Trainable automated forces. *Proceedings of the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC) 2010*. https://www.researchgate.net/publication/241970920_Trainable_automated_forces

Anderson, J. D., & Bowden, M. L. (2005). *Introduction to Flight* (Vol. 582). McGraw-Hill Education.

Beeching, E., Debangoye, J., Simonin, O., & Wolf, C. (2021). Godot Reinforcement Learning Agents. *ArXiv Preprint ArXiv:2112.03636*. http://arxiv.org/abs/2112.03636

Bettini, M., Prorok, A., & Moens, V. (2023). BenchMARL: Benchmarking Multi-Agent Reinforcement Learning. *ArXiv Preprint ArXiv:2006.07869*. http://arxiv.org/abs/2312.01472

Cheng, X., Khomtchouk, B., Matloff, N., & Mohanty, P. (2018). Polynomial Regression As an Alternative to Neural Nets. *ArXiv Preprint ArXiv:1806.06850*. http://arxiv.org/abs/1806.06850

Dantas, J. P. A., Costa, A. N., Geraldo, D., Maximo, M. R. O. A., & Yoneyama, T. (2021). Weapon engagement zone maximum launch range estimation using a deep neural network. *Brazilian Conference on Intelligent Systems*, 193–207.

Dantas, J. P. A., Costa, A. N., Gomes, V. C. F., Kuroswiski, A. R., Medeiros, F. L. L., & Geraldo, D. (2022). ASA: A Simulation Environment for Evaluating Military Operational Scenarios. *ArXiv Preprint ArXiv:2207.12084*. http://arxiv.org/abs/2207.12084

DARPA. (2023). *ACE Program's AI Agents Transition from Simulation to Live Flight*. https://www.darpa.mil/news-events/2023-02-13

de la Torre, J. C., Aragón-Jurado, J. M., Crespo-Álvarez, A., & Bárcena-González, G. (2024). GAGI: Game engine for Artificial General Intelligence experimentation. *SoftwareX*, *26*, 101665. https://doi.org/10.1016/j.softx.2024.101665

Floyd, M. W., Karneeb, J., Moore, P., & Aha, D. W. (2017). A goal reasoning agent for controlling UAVs in beyond-visual-range air combat. *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 4714–4721. https://www.ijcai.org/proceedings/2017/0657.pdf

Godot Foundation. (2024, July 8). *Godot Engine - Free and open source 2D and 3D game engine*. https://godotengine.org/

Goecks, V. G., Waytowich, N., Asher, D. E., Jun Park, S., Mittrick, M., Richardson, J., Vindiola, M., Logie, A., Dennison, M., Trout, T., Narayanan, P., & Kott, A. (2023). On games and simulators as a platform for development of artificial intelligence for command and control. *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, *20*(4), 495–508. https://doi.org/10.1177/15485129221083278

Gorsane, R., Mahjoub, O., de Kock, R. J., Dubb, R., Singh, S., & Pretorius, A. (2022). Towards a standardised performance evaluation protocol for cooperative MARL. *Advances in Neural Information Processing Systems*, *35*, 5510–5521. https://doi.org/10.48550/arXiv.2209.10485

Gorton, P. R., Asprusten, M., & Bråthen, K. (2023). Imitation learning for modelling air combat behaviour-an exploratory study. *Norwegian Defence Research Establishment (FFI)*, *22/02423*. https://ffi-publikasjoner.archive.knowledgearc.net/bitstream/handle/20.500.12242/3136/22-02423.pdf

Gorton, P. R., Strand, A., & Brathen, K. (2024). A survey of air combat behavior modeling using machine learning. *ArXiv Preprint ArXiv:2404.13954*. http://arxiv.org/abs/2404.13954

Holfeld, J. (2023). On the relevance of the Godot Engine in the indie game development industry. *ArXiv Preprint ArXiv:2401.01909*. http://arxiv.org/abs/2401.01909

Hu, D., Yang, R., Zuo, J., Zhang, Z., Wu, J., & Wang, Y. (2021). Application of Deep Reinforcement Learning in Maneuver Planning of Beyond-Visual-Range Air Combat. *IEEE Access*, *9*, 32282–32297. https://doi.org/10.1109/ACCESS.2021.3060426

Hu, S., Zhong, Y., Gao, M., Wang, W., Dong, H., Liang, X., Li, Z., Chang, X., Yang, Y., & Fuxi Lab, N. A. (2023). MARLlib: A Scalable and Efficient Library For Multi-agent Reinforcement Learning. *Journal of Machine Learning Research*, *24*, 1–23. http://jmlr.org/papers/v24/23-0378.html.

Johansson, T. (2018). *Tactical Simulation in Air-to-Air Combat: Evolutionary Algorithms and Behavior Tree Framework* [Luleå University of Technology]. http://www.diva-portal.se/smash/get/diva2:1258472/FULLTEXT01.pdf

Juliani, A., Berges, V.-P., Teng, E., Cohen, A., Harper, J., Elion, C., Goy, C., Gao, Y., Henry, H., Mattar, M., & Lange, D. (2018). Unity: A General Platform for Intelligent Agents. *ArXiv Preprint ArXiv:1809.02627*. http://arxiv.org/abs/1809.02627

Kuroswiski, A. R., Medeiros, F. L. L., De Marchi, M. M., & Passaro, A. (2023). Beyond visual range air combat simulations: validation methods and analysis using agent-based models. *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*. https://doi.org/10.1177/15485129231211915

Li, L., Zhang, X., Qian, C., Zhao, M., & Wang, R. (2024). Cross coordination of behavior clone and reinforcement learning for autonomous within-visual-range air combat. *Neurocomputing*, *584*, 127591. https://doi.org/10.1016/j.neucom.2024.127591

Lima Filho, G. M., Kuroswiski, A. R., Medeiros, F. L. L., Voskuijl, M., Monsuur, H., & Passaro, A. (2022). Optimization of Unmanned Air Vehicle Tactical Formation in War Games. *IEEE Access*, 1–1. https://doi.org/10.1109/ACCESS.2022.3152768

Mohd, T. K., Bravo-Garcia, F., & Love, L. (2023). Analyzing strengths and weaknesses of Modern Game Engines. *Journal of Computer*. https://www.researchgate.net/profile/Tauheed-Khan-Mohd/publication/368590412_Analyzing_Strengths_and_Weaknesses_of_Modern_Game_Engines/links/656a0c70ce88b8703127f517/Analyzing-Strengths-and-Weaknesses-of-Modern-Game-Engines.pdf

Joint Chiefs of Staff. (2020). Joint Air Operations. In *U.S. Department of Defense*. U.S. Department of Defense. https://www.jcs.mil/Portals/36/Documents/Doctrine/pubs/jp3_30.pdf

Pope, A. P., Ide, J. S., Micovic, D., Diaz, H., Rosenbluth, D., Ritholtz, L., Twedt, J. C., Walker, T. T., Alcedo, K., & Javorsek, D. (2021). Hierarchical Reinforcement Learning for Air-to-Air Combat. *2021 International Conference on Unmanned Aircraft Systems, ICUAS 2021*, 275–284. https://doi.org/10.1109/ICUAS51884.2021.9476700

Qian, C., Zhang, X., Li, L., Zhao, M., & Fang, Y. (2024). H3E: Learning air combat with a three-level hierarchical framework embedding expert knowledge. *Expert Systems with Applications*, *245*, 123084. https://doi.org/10.1016/J.ESWA.2023.123084

Scukins, E., Klein, M., Kroon, L., & Ögren, P. (2024). BVR Gym: A Reinforcement Learning Environment for Beyond-Visual-Range Air Combat. *ArXiv Preprint ArXiv:2403.17533*. http://arxiv.org/abs/2403.17533

Terry, J., Black, B., Grammel, N., Jayakumar, M., Hari, A., Sullivan, R., Santos, L. S., Dieffendahl, C., Horsch, C., Perez-Vicente, R., & others. (2021). Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, *34*, 15032–15043. https://openreview.net/pdf?id=fLnsj7fpbPI

Towers, M., Terry, J. K., Kwiatkowski, A., Balis, J. U., Cola, G. de, Deleu, T., Goulão, M., Kallinteris, A., KG, A., Krimmel, M., Perez-Vicente, R., Pierré, A., Schulhoff, S., Tai, J. J., Shen, A. T. J., & Younis, O. G. (2023). *Gymnasium*. Zenodo. https://doi.org/10.5281/zenodo.8127026

van Rozen, R. (2023). Game Engine Wizardry for Programming Mischief. *Proceedings of the 2nd ACM SIGPLAN International*. https://dl.acm.org/doi/abs/10.1145/3623504.3623570

Wang, H., & Wang, J. (2024). Enhancing multi-UAV air combat decision making via hierarchical reinforcement learning. *Scientific Reports*, *14*(1), 4458. https://doi.org/10.1038/s41598-024-54938-5

Weng, J., Chen, H., Yan, D., You, K., Duburcq, A., Zhang, M., Su, H., Su, H., & Zhu, J. (2022). Tianshou: A Highly Modularized Deep Reinforcement Learning Library. *Journal of Machine Learning Research*, *23*, 1–6. https://github.com/thu-ml/tianshou/.

Zhou, R., Zhao, P., & Wang, H. (2023). Deep reinforcement learning-based air combat maneuver decision-making: literature review, implementation tutorial, and future direction. *IEEE Access*, *11*, 34855–34869. https://doi.org/10.1007/s10462-023-10620-2