Explainability in Multi-Agent Reinforcement Learning for Air Combat Tactics

Ardian Selmonaj **Alessandro Antonucci**
Dalle Molle Institute for Artificial Intelligence (IDSIA) - SUPSI/USI
SWITZERLAND

ardian.selmonaj@idsia.ch alessandro.antonucci@idsia.ch

Adrian Schneider **Michael Rüegsegger** **Matthias Sommer**
armasuisse Science and Technology
SWITZERLAND

adrian.schneider@armasuisse.ch michael.rueegsegger@armasuisse.ch matthias.sommer@armasuisse.ch

ABSTRACT

Artificial intelligence (AI) is pivotal in shaping the future technological landscape. Multi-Agent Reinforcement Learning (MARL) has emerged as a significant AI technology for simulating complex dynamics across various domains, enabling novel potentials for advanced strategic planning and coordination among autonomous agents. However, its practical deployment in sensitive military contexts is constrained by the lack of explainability: a critical factor for reliability, safety, strategic validation, and human-machine interaction. This paper reviews the latest advancements in explainability within MARL and presents novel use cases, emphasizing its indispensability for examining agents' decision-making processes. We first critically assess existing techniques and associate them with the domain of military strategies, focusing on simulated air combat scenarios. We then introduce the concept of a novel information-theoretic explainability descriptor to analyze agents' cooperation capabilities. Through our research, we aim to highlight the necessity of precisely understanding AI decisions and aligning these artificially generated tactics with human understanding and strategic military doctrines, thereby enhancing the transparency and reliability of AI systems. By illuminating the crucial importance of explainability in advancing MARL for operational defense, our work supports not only strategic planning but also the training of military personnel with insightful and comprehensible analyses.

1.0 INTRODUCTION

1.1 The Potential of AI

Artificial Intelligence (AI) has become a transformative force across various domains, with significant achievements in both general applications and specialized areas like wargames. In recent years, AI has demonstrated remarkable capabilities in strategic decision-making, adaptability, and handling complex environments, which are essential qualities in wargaming scenarios. A sub-category of AI is *Reinforcement Learning* (RL), in which an agent learns to act correctly in its surrounding environment through trial-and-error interactions, therefore not necessarily relying on human expert data to find strong *Courses of Action* (CoAs). By utilizing RL techniques, autonomous agents can adapt and devise new strategies in response to evolving battlefield conditions. RL's success in games like chess [1] has showcased its potential to excel in high-level reasoning, planning, and execution under uncertainty. *Multi-Agent Reinforcement Learning* (MARL) [2] extends the concept of RL to multiple, simultaneously interacting, agents within a shared environment. MARL is particularly well-suited for wargames because it allows for the modeling of complex interactions between multiple agents, reflecting the collaborative and competitive dynamics typical in military conflicts. It enables the simulation of coordinated strategies, real-time adaptation, and the exploration of emergent behaviors, making it an ideal framework for developing and testing sophisticated tactics in wargaming scenarios. Overall,

the integration of MARL into the simulation of military scenarios serves as a tool for training and operational planning, offering valuable insights that can inform real-world military decisions.

In this work, we particularly focus on air combat scenarios where agents are trained with a MARL algorithm. Within these scenarios, dogfighting is of particular interest, which refers to a close-range, highly maneuverable aerial battle between fighter aircraft. It typically involves intense, dynamic maneuvers as agents attempt to gain a positional advantage over their opponents to achieve a successful attack. Our goal is to have trained agent pilots, whose behavior is then examined.

1.2 Understanding Decisions

Deep RL involves neural networks for decision-making in complex and real-world environments like wargames. However, these networks are frequently viewed as black-box models due to the difficulty in interpreting their outcomes. *Explainable Reinforcement Learning* (XRL) refers to the ability to interpret and understand the decision-making processes of reinforcement learning models, providing insights into why certain actions are taken in specific situations. Challenges in XRL include risks related to scientific evaluation and operational reliability, the absence of universally accepted evaluation metrics, and the difficulty of providing comprehensive explanations for complex tasks [3]. Despite these challenges, employing effective explainability methods to understand model outputs is especially crucial in military operations for diagnosing errors, enhancing model performance, and comprehending intricate agent behaviors. These methods play a critical role in building trust among military personnel, ensuring transparency in safety-critical missions, and facilitating compliance with stringent operational and regulatory standards. In the context of complex and sensitive military scenarios, XRL enables commanders and decision-makers to interpret and justify AI-driven strategies and actions, leading to more informed and accountable decisions. Furthermore, precise explainability (i.e. correct and reliable explanations) contributes to better risk assessment and management, improves coordination between human and AI agents, and supports the integration of advanced AI systems into existing military frameworks while maintaining operational reliability and effectiveness. Air combat simulations involve complex decision-making processes where agents must make split-second decisions to achieve strategic objectives. These simulations often involve numerous factors, including maneuvering, targeting, threat avoidance, fuel management, and coordination with other units. As an example, consider the following scenario: the agent detects an incoming missile from hostile forces. To counter, it quickly releases flares and performs a barrel roll to confuse the missile's heat sensors and evade the enemy's targeting. In this scenario, the observation of the missile acted as a significant feature to perform the action of flare release and barrel roll.

This paper reviews the latest advancements in explainability within MARL and introduces novel use cases that highlight its crucial role in analyzing agents' decision-making processes within simulated air combat scenarios (Figure 1-1). By examining these advancements, we underscore the importance of explainability in understanding and improving agent behavior, particularly when applied in complex environments like military simulations. Our paper is more than a survey as it explores how explainability can enhance strategic planning, facilitate human-AI collaboration, and ensure the trustworthiness of AI-driven decisions in mission-critical operations. Through these insights, we aim to demonstrate the urgency of explainable MARL for both research and practical deployments in high-stakes scenarios.

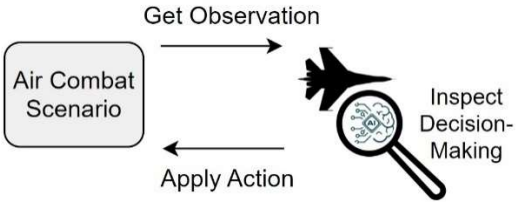

Figure 1-1: Overview of inspecting decision-making process.

2.0 LITERATURE REVIEW

In this section, we go through existing approaches involving RL and MARL in air combat and drone swarms, as both these wargames involve similar dynamics. We then review general explainable AI methods.

2.1 RL for Air Combat

There are various approaches incorporating RL and MARL for training agents in air combat scenarios. These approaches are not only limited to dogfighting maneuvers of fighter aircraft, but also include swarms of *Unmanned Aircraft Vehicles* (UAVs) and different types of aircraft (heterogeneous agents).

Aerial combat on small engagements usually focuses on *controlling* the aircraft through RL for gaining advantageous positions against opponents with little risk of return fire. Early works to control the aircraft include expert systems [4-5] or hybrid systems with learning classifiers [6-7], whereas newer methods rely on RL [8-9]. For learning stronger CoAs, simulated air combat approaches using RL methods rely on more advanced techniques such as *Deep Q-Networks* (DQN) [10], *Deep Deterministic Policy Gradients* (DDPG) [11], curriculum learning methods [12] or methods incorporating self-play, where agents play against a copy of itself [13].

On the other hand, larger engagements focus on high-level tactical decisions [14] or weapon-target assignment [15], i.e. on *planning* of CoAs. In this case and under consideration of the course of dimensionality, MARL approaches are especially well suited by exploiting symmetries within individual agents. There are advanced methods in this field using multi-agent DDPG [16], hierarchical RL [17] or attention-based neural networks [18]. One of our previous works [19] includes a hierarchical MARL model with an attention mechanism that is trained using *Proximal Policy Optimization* (PPO) [28]. In our work, we additionally considered heterogeneous agents, which seems rarely to be the case in literature. Incorporating heterogeneous agents can increase the complexity of coordination, as agents may be unaware of each other's skills and capabilities. Further approaches for air combat simulations using RL methods can be found in [20].

2.2 Multi-Agent Reinforcement Learning

The primary challenge in XRL lies in balancing efficiency with reliability, i.e. the need to deliver accurate and dependable explanations while maintaining efficiency [3]. XRL methods [21] can broadly be classified as follows, which we further examine in the next section: 1) Policy simplification to track decision steps; 2) Reward influence to decompose the reward signal into sub-components; 3) Feature contribution to identify most significant information of input data; 4) Causal models to identify cause-effect relationship; 5) hierarchical models to inspect sub-policy selection behavior. A reward decomposition method is presented in [22]. The approach uses semantically grouped reward components to debug and evaluate agent behavior. The work in [23] assesses agents' competency by combating against identical RL opponents but with altered initial conditions, thereby falling in the category of feature contribution.

Our work critically assesses existing XRL techniques to be used post-hoc, i.e. when training of agents is finished. Since explainability methods primarily focus on single-agent RL, we adapt them for the multi-agent domain, as was done in similar works [30]. Interpreting simulated combat trajectories provides valuable insights that enhance strategic planning and military training.

3.0 METHODS

The work in this paper primarily involves explainability techniques for air combat simulations trained through a MARL framework. We first introduce the concept of MARL and afterwards present some XRL techniques in detail and associate them to the domain of air combat scenarios.

3.1 Multi-Agent Reinforcement Learning

The MARL framework involves a set of N agents learning to cooperate or compete through trial and error within a shared environment, which is shown in Figure 3-1. These interactions are typically modeled by a Markov Game, originating from strategic decision-making processes in game theory. The joint policy $\boldsymbol{\pi}(\boldsymbol{A}|S) = \{\pi_1(A_1|S), \dots, \pi_N(A_N|S)\}$ represents a probability distribution over the joint action space $\boldsymbol{A}$, given the state S, and describes how all agents make decisions. At time t in state $s_t \in S$, each agent $i \in N$ selects an action $a_i \in A_i$ based on its policy π_i and receives a reward $R_i(\boldsymbol{a}, s_t)$ after all agents take the joint action $\boldsymbol{a} = \{a_1, \dots, a_N\}$. The main objective for each agent is to learn a policy π_i^* that maximizes the expected cumulative reward, with a discount factor $\gamma \in (0,1)$ to balance short- and long-term rewards:

$$\pi_i^* = \underset{\pi_i}{\operatorname{argmax}}\, \mathbb{E}_{\pi_i}\left[\sum\nolimits_t \gamma^t R_i(\boldsymbol{a}, s_t)\right].$$

We adopt a partially observable environment, where each agent observes a limited information $O \subset S$, yielding a partially observable Markov Game. In our training framework, we use the *Centralized Training and Decentralized Execution* (CTDE) [27] scheme. Its popularity is attributed to its capability to address non-stationarity by sharing information amongst agents during training, enhance coordination, and preserve each agent's ability to act independently during execution.

To approach the decision structure of defense organization, we extend our MARL model with *Hierarchical Reinforcement Learning* (HRL), which improves learning efficiency by using temporal abstraction to break down tasks into a hierarchy of subtasks. High level policies issue abstract commands to activate low-level policies over a specified time span. The low-level policies manage specific actions within a sub-task. This greatly facilitates the training process by exploiting policy symmetries of individual agents and by separating control from command tasks. Combining HRL with MARL yields a *Hierarchical Multi-Agent Reinforcement Learning* (HMARL) framework, which is modelled by a *partially observable semi Markov Decision Process* (POSMDP). This model was also employed in our previous work [19]. We give specific details about our model and the training process in Section 4.2.

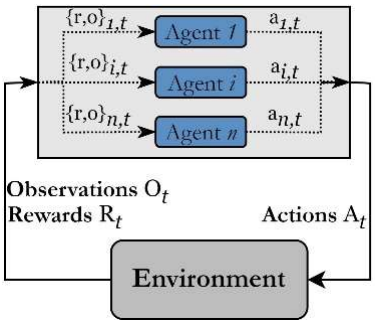

Figure 3-2: MARL interaction cycle

3.2 Explainability Approaches for Air Combat

We now go through the XRL categories as reviewed in the related work section and subsequently associate them with the multi-agent domain of air combat scenarios to highlight the benefits and indispensability of understanding AI tactics.

The first three methods (policy simplification, reward decomposition, and feature contribution) fall under the category of *reactive* explanations. This type of explanation focuses on a short time horizon, providing feedback based on immediate behaviors. For example, a question like "Why did the aircraft fire a missile?" could be answered by an immediate incentive such as "an opponent entered the *Weapon Engagement Zone* (WEZ)." These explanations tend to focus on individual actions rather than on broader strategic considerations.

In contrast, *proactive* explanations consider longer time horizons and are more suited for explaining strategic decisions. For example, they could explain why certain agents with specific skills were set to defensive mode in a particular situation, while other agents adopted aggressive tactics. Causal and hierarchical RL models can offer this type of explanation, providing insights into longer-term strategies and coordinated maneuvers in air combat.

3.2.1 Policy Simplification

In deep RL, neural networks are used as function approximators to learn a decision-making function, either the policy or the Q-function, where in our analysis, we focus on the former. Policy simplification refers to the process of reducing the complexity of a policy so that it becomes interpretable by humans. This can be achieved by learning policies in the form of a decision tree to trace each decision step, casting the learned policy as an "if-then" ruleset (e.g., fuzzy rules), using state abstraction to group similar states and reduce the dimensionality of the state space, or representing the learned policy with high-level, human-readable programming languages. The simplicity achieved by these methods is the key advantage, as it facilitates generating explanations and fostering trust in the system.

In environments with relatively simple dynamics and few agents, such methods may generalize sufficiently and scale to extract meaningful explanations, even in unforeseen (and simple) air combat scenarios. However, this may not hold in more complex environments, where there are numerous mission objectives and agents with different skill sets, as the explanations tend to be static. The primary disadvantage of this approach is the trade-off between model performance and explainability: as the level of explainability increases, the accuracy of the model often decreases. In the context of simulated air combat scenarios, where realism is crucial for generating valuable insights, maintaining high model accuracy is important. This typically necessitates complex models involving sophisticated neural networks, extensive hyperparameter tuning, advanced training algorithms, and highly dynamic environments. While policy simplification constrains the types of policy representations and, consequently, the overall performance, it can serve as a practical and efficient starting point. Simplified policies can effectively train and explain basic controlling maneuvers for air combat agents, providing a foundation for future iterations that balance explainability and accuracy as the complexity of the scenarios increases.

3.2.2 Reward Decomposition

During RL training, the total reward is typically summed to learn a Q-function (or policy), making it difficult to pinpoint the specific positive and negative factors influencing decision-making. To address this, reward decomposition breaks the total reward R into a sum of semantically meaningful components R_i, each corresponding to a distinct aspect of the environment. When an agent selects an action a in a given state s, the expected reward R for that action is a combination of these individual components R_i. Reward decomposition provides insights into how each component contributes to the overall expected reward, thereby explaining the

agent's action preferences. In general, the reward components are the same for all agents. This changes if each agent has individual rewards, e.g. for individual tasks to solve.

For this, the Q-function, defined as $Q(s,a) = \mathbb{E}_\pi[R_i|s,a]$, must also be decomposed with respect to the reward components R_i, such that $Q(s,a) = \sum_i Q_i(s,a)$. This is typically achieved by learning separate Q-networks for each Q_i. By comparing the differences in Q-values for all actions across each reward component R_i, one can gain a deeper understanding of the agent's decision-making process. This approach allows for two types of analysis:

1) For a given state-action pair (s,a), one can examine the contribution of each component R_i to the overall Q-value. This helps identify which components are driving the agent's action preferences.

2) For a given state and a specific reward component, one can assess which action offers more advantage compared to others. In the case of two actions, computing $\Delta_i(s,a_1,a_2) = Q_i(s,a_1) - Q_i(s,a_2)$ determines the preference of a_1 over a_2, and vice versa.

The feasibility of reward decomposition relies on having a naturally decomposable reward function with components that are meaningful within the context of the environment and task. However, reward engineering can be challenging, as it involves carefully balancing the need to incentivize desired behaviors while avoiding unintended consequences. Improper reward design can lead to the agent exploiting the reward function in ways that deviate from the intended objectives.

In air combat simulations involving multiple interacting agents with various skill sets and dynamic mission objectives, defining appropriate reward components for each agent (or for groups of agents with shared tasks) is crucial but non-trivial. Reward components can conflict, such as when minimizing exposure to enemy radar contradicts the need to engage an opponent aircraft. Such conflicts can lead to misalignments (social dilemmas) between individual and global objectives and complicate the capture of interactions between agents.

Despite these challenges, reward decomposition offers valuable clarity by breaking down the reasoning behind an AI agent's preference for certain actions or tactics. This can be particularly useful in understanding critical components such as enemy engagement, threat avoidance, and mission completion. Commanders can leverage this method to better understand, adjust, and validate the agent's decisions to align with overall mission goals.

Additionally, this approach allows for fine-tuning priorities, such as balancing offensive versus defensive maneuvers, by adjusting reward weights, making it adaptable to different tactical scenarios. As an example, suppose the agent's reward consists of components like position, distance from the enemy, fuel consumption, and hit success. The agent might retreat rather than attack because the position and fuel rewards outweigh the potential reward for hitting the enemy. The decomposition clarifies that the agent's decision balances factors like fuel efficiency and strategic positioning, not just immediate combat success. A further important insight provided by reward decomposition is the ability to balance trade-offs between individual agent success (e.g., surviving the mission) and collective team objectives (e.g., achieving air superiority), leading to more cohesive and effective team strategies. Visualization techniques that provide detailed explanations of air combat tactics, as proposed in [22], further enhance the interpretability of the AI's decision-making process.

3.2.3 Feature Contribution

Closely related to reward decomposition is the method of feature contribution. The objective of this approach is to identify the input features with the greatest influence on decision-making and use them to generate explanations. Techniques in this category include analyzing model outputs with modified (contrastive) input data, importance ranking through methods such as saliency maps, assigning Shapley values [24] to each feature (see Section 3.2.6), or using LIME [25] to approximate the model with a simpler, interpretable model like

linear regression. These methods focus on specific input data or scenarios, but they often fail to explain the complete decision-making process, leaving the model as a "black box."

Nevertheless, in critical air combat scenarios, these methods can enhance transparency by identifying key environmental features that contribute to tactical success. For instance, in situations where agents are collaborating to attack an enemy aircraft, features like enemy speed or proximity to obstacles can be identified as having the greatest influence on the decision to engage or disengage. By pinpointing such influential features, the training process can be optimized to prioritize the most important tactical considerations, as demonstrated in [23]. In Human-AI collaborative training scenarios, feature contribution methods can help provide and explain recommendations for specific in-game situations, further improving transparency and trust in the AI system.

However, caution must be exercised when interpreting explanations gained from feature contribution or reward decomposition. There is a risk of drawing misguided conclusions. For example, consider a scenario where an agent's decision to retreat is primarily based on the proximity of friendly forces. While this feature may be highlighted as highly influential, it could be only one aspect of a more complex decision-making process. Overemphasizing a single feature can obscure other critical factors, especially in complex air combat simulations with a rapidly changing operational environment. Additionally, scalability and computational overhead are potential limitations of this approach, particularly when applied to large-scale or real-time simulations. A more reliable approach in such cases could be causal methods (Section 3.2.5) which are gaining popularity. To conclude, while feature contribution methods offer valuable insights, they can either enhance or complicate the understanding and performance of MARL systems in dynamic air combat environments. The interpretability provided by these methods must be balanced against their limitations, particularly in highly complex and fluid scenarios.

3.2.4 Hierarchical Model

HRL breaks down a task into a hierarchy of subtasks, with each subtask being managed by a low-level policy. These low-level policies can be examined independently, making it easier to understand how specific decisions are made within the context of a subtask. Symmetries at the lower levels allow the same commands to be applied across similar subtasks, such as controlling similar airplanes. Once a sub-policy is well understood in one context, its reuse in another makes the new behavior easier to interpret, as it builds on previously acquired knowledge. Hierarchical policy structures can also be visualized across different levels of the hierarchy, for instance, using decision trees or diagrams that show the frequency of low-level policy selection, which helps gain insight into the temporal sequence of decisions.

However, one drawback of this approach is that subtasks and low-level control policies need to be pre-specified. This can limit their ability to generalize or adapt to new environmental conditions, as the system may struggle to learn entirely new skills. Nonetheless, this hierarchical structure closely mirrors military frameworks where multiple layers of decision-making (ranging from high-level mission planning to low-level maneuvers) enable a structured approach to managing complex tasks. In air combat scenarios, where engagements can be prolonged, decisions must account for future states and potential outcomes. The temporal abstraction in HRL allows the agent to plan and execute action sequences over time, aligning with (and potentially enhancing) human planning capabilities.

Subtasks in HRL can be aligned with specific mission objectives. For example, a high-level goal like "achieve air superiority" can be decomposed into subtasks such as "engage opponent aircraft," "evade missile," and "maintain formation." These subtasks can be further divided into simpler actions, making the overall decision-making process more transparent to human operators. Certain maneuvers, like "missile evasion," can be reused across different missions, allowing sub-policies to be applied in various scenarios. Additionally, enforcing the condition that a subtask can only be instantiated once another subtask is completed (such as only allowing "fire

missile" after "engage opponent" has been successfully completed) further enhances interpretability by providing a clear structure for task completion.

However, designing effective hierarchical structures for air combat scenarios is a complex challenge that requires deep domain expertise. Striking a balance between performance and simplicity is critical: while deeper hierarchical structures may enable more advanced tactics and maneuvers, they can also make it harder to provide clear explanations. Conversely, a rigid hierarchy may struggle to adapt to unexpected situations, which could also complicate the explanation of decision-making processes. HRL's emphasis on long-term planning may result in decisions that are not immediately intuitive. For instance, an agent may choose a seemingly suboptimal maneuver as part of a long-term strategy that is difficult to explain in the short term. Moreover, as HRL involves policies as in single- and multi-agent RL, the previously discussed methods can be integrated to enhance explainability.

3.2.5 Causal Model

A promising but complex approach in XRL is the use of *Structural Causal Models* (SCM). This method captures the cause-effect relationships between variables in a system. The interactions between these variables are represented through a set of structural equations. The system includes endogenous variables X (internal states, such as the fuel level of an aircraft) and exogenous variables Y (external states, such as the positions of obstacles in the environment), which represent distinct components. The structural equations define how each endogenous variable X_i is generated based on the parent variables $PA(X_i)$ and exogenous states Y_i: $X_i = f(PA(X_i), Y_i)$.

The final component of SCMs is a *Directed Acyclic Graph* (DAG) that visually represents the dependencies between variables, with each node representing a variable and each edge representing a causal relationship. In MARL, the interactions between various components are inherently complex. One could develop an SCM by identifying the relationship of the state to the action selection in the joint policy of agents $\boldsymbol{\pi}(\boldsymbol{a}|s)$, or the effect of the current state-action pair on the future state s' in the transition dynamics function $p(s'|s, \boldsymbol{a})$. An example of learning a DAG in MARL can be found in [29], where the mutual information is used to measure the influence of the current action to future states given the current state: $I(s_{t+1}; a_t|s_t)$. If $I > 0$, it means that there is a causal realtionship between these states when this action is applied. However, as the number of components and agents increases, and with the non-stationary nature of the environment, modeling these relationships becomes increasingly difficult, especially in continuous action and state spaces. For this reason, SCMs are best suited to smaller air combat configurations (e.g., 2-vs-2 scenarios) where meaningful cause-effect relationships can be derived between informational data, agents, and environmental factors.

A significant drawback of SCMs is their inefficiency in providing real-time causal relationships, a critical requirement in the fast-evolving dynamics of aerial warfare. Moreover, causal relationships in complex air combat scenarios may not always be clear or easy to determine. For example, the outcome of a particular maneuver could depend on various factors such as pilot skill, enemy capabilities, and environmental conditions, all of which are difficult to model accurately.

To enhance the predictive accuracy of SCMs in multi-agent air combat simulations, individual SCMs can be developed for each agent. This approach allows for a detailed analysis of each agent's interactions with its co-players or environmental factors. For instance, an agent's SCM could highlight its preferences towards cooperation: a strong influence on a co-player's decisions may suggest a high inclination to cooperate, while a lack of influence might indicate that the agent tends to disregard that co-player. Another possibility is to create an SCM for hierarchical policies, capturing the effects of tactical commands on maneuvers and skills in specific combat situations. This could provide insights into the conditions that trigger a commander to activate certain control policies or into the situations where an agent chooses to apply specific skills.

Additionally, SCMs support counterfactual analysis, which is similar to feature contribution methods. Questions like "What would have happened if the state at that moment had been different?" can be explored by querying the causal model. For instance, a specific air combat scenario could involve examining how enemy strategies would have changed if the commander had engaged them in a different formation. Although learning SCMs in a multi-agent setting is highly complex, once accurately modeled, they can offer military personnel valuable insights for developing and adjusting tactics in defense operations. Since our primary focus is on emphasizing the need for advanced XRL techniques in the military domain and given that our air combat simulation operates in a continuous domain, we omit doing experiments involving causal models. Their complexity is significant and represents a distinct area of research on its own.

3.2.6 Game Theory

The five explainability approaches discussed represent the state-of-the-art for RL models and can also be applied to MARL models. Multi-agent systems are categorized as cooperative, competitive, or mixed-motive [26], with the latter involving social dilemmas where agents must choose between cooperation and self-interest. In an air combat, this can create tactical and ethical conflicts, such as whether an agent should help an ally or prioritize their own survival. Various evaluation metrics exist to assess cooperation, including *Social Welfare* (mean rewards for all agents), *Reward Shaping* (adding intrinsic and social motivations), and *Sustainability* (sacrificing immediate rewards for long-term group benefit). Most of these methods rely on reward signals, but these signals can be delayed or even absent for extended periods. This phenomenon is called “sparse rewards” and occurs in RL tasks with limited feedback from the environment. Such effects also apply to air combat scenarios, where aircraft may circle each other in an attempt to gain a tactical advantage, such as positioning behind the enemy to reduce the risk of being fired upon. In such typical dogfighting situations, reward feedback may not be immediately available, especially if no reward-shaping method is in use. These and similar scenarios are challenging to evaluate due to the lack of immediate feedback. We therefore provide a concept to evaluate the level of cooperation during gameplay. Shapley values [24] are a method for cooperative games to fairly distribute a payoff amongst the players N. The payoff value is defined by the characteristic function $v(N)\colon 2^N \to \mathbb{R}$. For a coalition $C \subset N$, the marginal contribution of player i is defined as:

$$\Delta v(C, i) = v(C \cup i) - v(C)$$

The weighted average of Δv across all coalitions gives the Shapley value $\varphi_i(v)$ for player i. The characteristic function v can be freely defined. As we consider the scenario with no or only limited reward signals, we evaluate the agents based on their ability to provide options to their co-players. We define options here in the number of available actions per time step, as some actions might be restricted in certain scenarios. For example, agent A can't fire a missile because the opponent is out of range. Agent B swoops in and forces the opponent to change direction, pushing them closer to agent A. This maneuver brings the opponent within the range of agent A, allowing A to fire the missile. In this scenario, B acted as an action enabler for A. To measure this type of contributions during gameplay, one could define the characteristic function as the size of the action space A_C of coalition $C\colon v(C) = |A_C|$. As the interaction of multiple agents within a shared environment exhibits non-stationary transition dynamics, it is a reasonable choice to measure the contribution of an agent towards the number of available actions its co-players have. The presented explainability method can in fact be applied to any multi-agent system with sparse rewards. In our case of simulated air combat scenarios, it is particularly useful to examine agent behavior *while* achieving a specified goal (and not the goal itself). We intentionally leave this as a solution concept for future research directions within XRL.

4.0 EXPERIMENTS

4.1 Overview

To explain the tactical decision-making process, we first train homogeneous air combat agents, including a high-level commander. Once the combat policies have been trained and demonstrate good performance, we use them to generate explanatory insights from various situations. This is achieved by combining different approaches discussed in previous sections. Given that we employed a hierarchical model, we can utilize the techniques described in Section 3.2.4, that analyzes the selection behavior of low-level policies by the upper-level instance. Additionally, we apply the method of feature contributions (Section 3.2.3) by varying initial conditions (different inputs) to observe the resulting outcomes. Finally, we adopt the reward decomposition approach, as outlined in Section 3.2.2 and similarly applied in [22]. Specifically, we cast reward types to low-level control modes and analyze which low-level control policy has the most significant impact on specific battlefield conditions, as directed by the high-level commander policy. This analysis can be done in two ways: 1) *Global*, by examining the overall decisions of all interacting agents, and 2) *Local*, by focusing on the decisions of a single aircraft in particular situations. Since our primary interest lies in air combat tactics rather than maneuver control, we focus on the hierarchical commander policy's behavior. We gain explanatory insights by analyzing the different low-level activations of each agent based on modified input data and combat configurations. It is worth noting that the user is free to select which features to alter for outcome inspection. For visualization purposes, we limit the selection to three features (dimensions).

4.2 Air Combat Training

We base our modeling of aircraft on the dynamics of Dassault Rafale[1]. There are beyond and within visual range air combat scenarios, where we focus here on the latter. The attacking mechanism is shown in Figure 4-1, where the blue aircraft is our AI-agent. The agent can attack using either its cannon, which hits the opponent when they enter the *Weapon Engagement Zone* (WEZ), or by firing a missile.

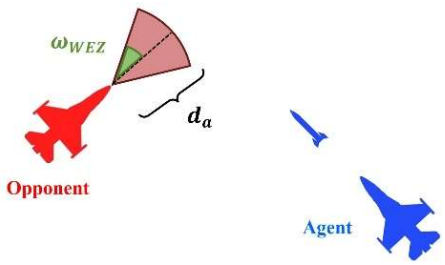

Figure 4-1: Aircraft attacking mechanism.

A visualization of the high- and low-level policies is shown in Figure 4-4. There are three low-level control modes that are learned by the respective policies:

1) Attack (π_a): aggressive behavior to fight and destroy opponent
2) Engage (π_e): reach and maintain position to face tail of opponent
3) Defend (π_d): evade and keep a large distance from opponent

To accelerate learning and since we are using homogeneous agents with identical skills, we do not train separate policies for each agent. Instead, we develop a shared policy for each maneuver mode, which is used

[1] https://dassault-aviation.com/en/defense/rafale.

by all agents. Specifically, every agent $i \in N$ utilizes π_a in attack mode, π_e in engage mode, and π_d in defense mode. Each agent receives information about its own position, orientation, remaining ammunition, as well as that of its closest friendly aircraft and nearest opponent aircraft. Each low-level policy is trained using different reward functions. We consider 5-vs-5 air combat scenarios for training and testing our model. However, by employing shared policies, we emphasize the flexibility of our approach, allowing any combat configuration to perform inference once training is complete, such as a 4-vs-2 configuration.

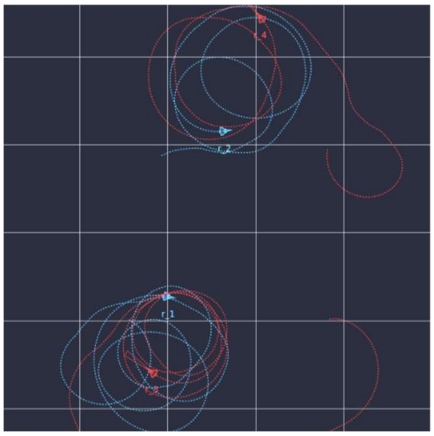

Figure 4-2: Visualization of simulation environment.

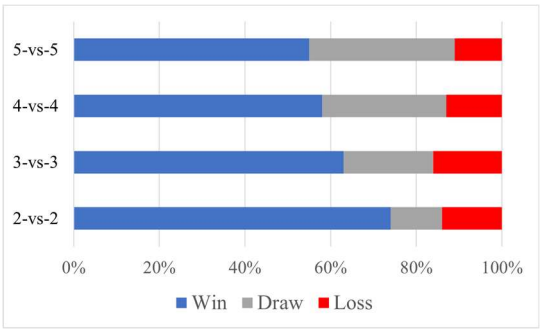

Figure 4-3: Combat performance.

The agents are trained on a dedicated 2D simulation platform developed in Python, with a visualization of the environment shown in Figure 4-2. We use PPO [28] for training the agents, as in our previous work [19]. The first step is to train the three low-level control policies. To ensure the opponents are competitive, we implement a league-based self-play mechanism, which incorporates curriculum learning with increasing levels of complexity. Initially, our agents are trained against random opponent maneuvers. Then, we copy the learned policies to the opponents and continue training the agents until the learning rate converges. Once this is achieved, we proceed to train the high-level commander policy, while the low-level policies remain fixed (i.e., no further training).

We trained five distinct commander policies, each with a different sensing capability $m \in [1,5]$. The commander observes the m closest enemy and m closest friendly aircraft for each agent and decides which control policy to apply for a specified duration. For inference, agents act according to the commander's decisions, while opponents are assigned low-level policies without a commander. The combat results (with the commander having a sensing range of $m = 3$) are shown in Figure 4-3, demonstrating that incorporating high-level decision-making significantly improves air combat performance. A win/loss is recorded when all opponents/agents are destroyed, while a draw occurs if at least one aircraft per team remains alive at the end of an episode.

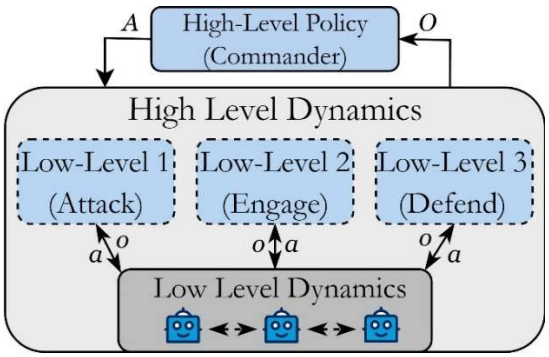

Figure 4-4: High- and Low-level Policies.

4.3 Global Explanations

To gain an explanatory insight of the decisions within multi-agent systems, one typically needs to consider the non-stationary environment dynamics, which arise due to the concurrently interacting agents within the same environment. As this is a highly complex task, we instead opt for a summarized approach that reveals the most significant decision across all agents involved. We do so by focusing on three features:

1) Opponent strategy: setting the enemy aircraft purely to the three distinct modes (attack, engage and defend) or to a mixed variant, where one of the three modes is randomly selected per aircraft.

2) Combat Difference: Based on the 5-vs-5 combat scene, we increase or decrease the number of agents, e.g. -2 means a 3-vs-5 configuration, with 3 agents and 5 opponents. Similarly, 2 means a 7-vs-5 configuration. The ranges are 1-vs-5 up to 10-vs-5.

3) Sensing capability: alter the radar range m of commander to change the number of detected opponents near an agent.

With these features we inspect the outcomes of the commander to gain an understanding what the crucial inputs are for the decision-making process. As the commander is responsible to decide which low-level policy to activate per agent, we plot the control mode with the highest activation frequency across all feature configurations in Figure 4-5. Each feature corresponds to an axis. To have consistent results, we run 100 simulation episodes per feature combination, where one episode has up to 500 low-level timesteps and up to 25 high-level timesteps (decisions). There can be less timesteps, as the episode ends when all agents of one team are destroyed.

From the results in Figure 4-5 we can infer interesting insights. With a short-sighted sensing capability ($m \leq 3$), the commander tends to neglect long term consequences as it always perceives a restricted number of the total number of opponents. It can therefore less reliably make decisions with respect to the overall game scenario, as it primarily decides for attacking CoAs even in disadvantageous scenarios with less agents than opponents (negative combat difference). This, however, immediately changes when increasing the radar range m, which reveals a more cautious behavior. Especially in the opponent modes attack and mixed, the commander rather tends to apply defensive maneuvers when opponents are in numerical superiority. This pattern also applies to the other opponent modes (engage and defend), although less pronounced. We may conclude that the sensing capability m of the commander has a high impact on the decision-making process. However, an increasing radar range also led to an overall weaker commander performance. This is because there might be situations where some opponents are far away s.th. they do not influence the current decision significantly and may act as noise to the neural network during training. From the current analysis however, it might still be beneficial to accept a performance decrease to gain an understanding of the model behavior. Striking the balance between model performance and explainability is a general issue in XRL [3].

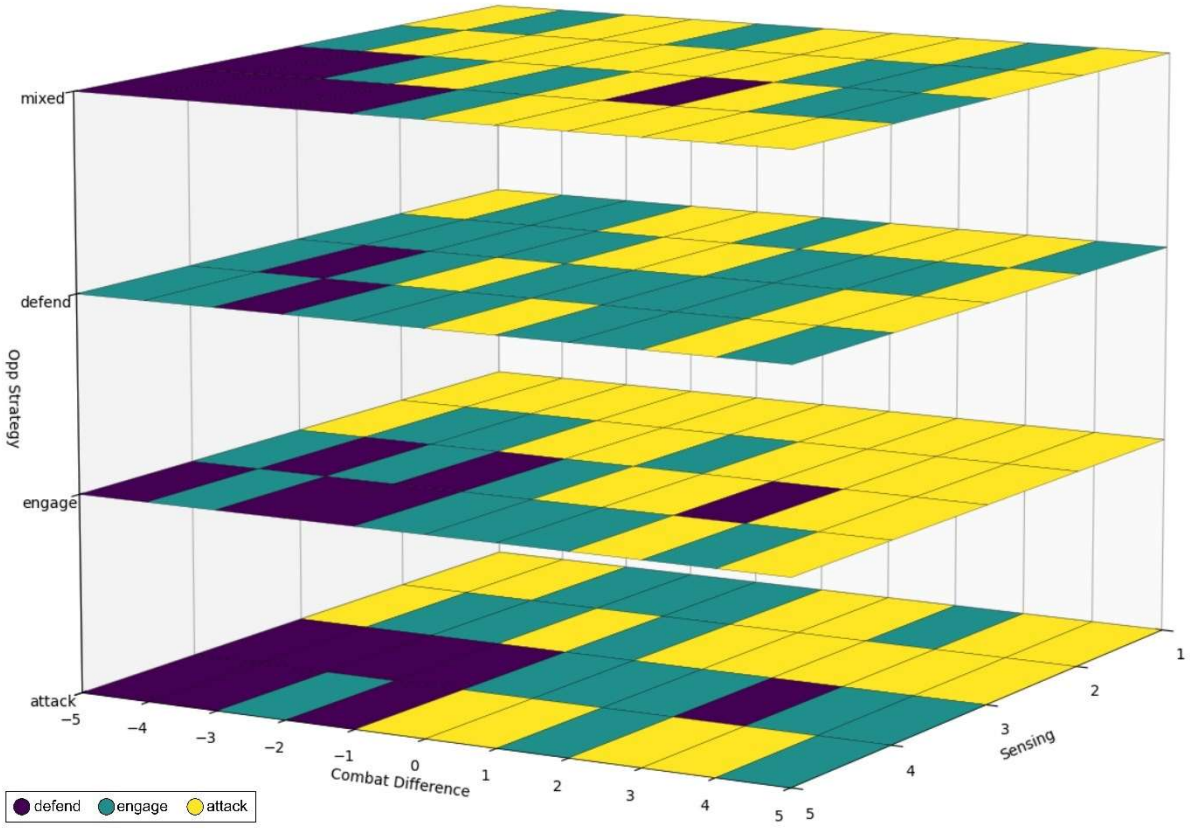

Figure 4-5: Hierarchy activations *per group*. The mode with the highest frequency is shown.

4.4 Local Explanations

We now focus on local explanations by considering the hierarchical decisions on one specific aircraft. In this setting, we do not need to run simulation episodes anymore, as we can just modify the observation values to the commander input. We selected the commander with the best radar range performance ($m = 3$). Once again, we choose three input features to modify:

1) Distance to closest opponent in kilometers.

2) Antenna Train Angle to opponent (0°=agent faces opponent, 180°=facing away from opponent).

3) Aspect Angle to opponent (0°=opponent faces away from agent, 180°=opponent faces agent).

As the commander observes the three closest hostile planes and the three closest friendly aircraft per agent, we only alter the input features of the *closest* enemy aircraft and set the remaining inputs to random observations (that are still valid). We again run 100 experiments per feature combination and plot the low-level mode with the highest activation frequency in Figure 4-6. It is worth noting that one can individually select which features to inspect, even with more than three features to infer the most influential inputs.

When analyzing the local commander decisions for each agent, we observe a consistent pattern across the distance dimension. Specifically, when the agent is in an advantageous position (ATA < 60° and AA < 60°), the commander consistently favors attacking maneuvers. However, this behavior shifts notably when the opponent is facing the agent (AA > 120°), prompting the commander to opt for more evasive actions.

From this analysis, we can infer that the Aspect Angle plays a significant role in the commander's decision-making process, as the plots reveal the highest variation along this dimension. The commander expectedly engages opponents when they are at a greater distance, as there is less risk of return fire. This suggests that the decision-making algorithm is effectively balancing offensive and defensive maneuvers based on both position and proximity.

While the results are influenced by the random values generated for other aircraft in the observation, the analysis of 100 simulations per feature combination offers a reliable understanding of the commander's tactical preferences. The consistent behavior across different configurations reinforces the importance of both Aspect Angle and distance in shaping the commander's decisions, as these features appear to have the highest impact on strategic choices during combat scenarios. This insight highlights the ability of the hierarchical model to adapt its decisions based on evolving battlefield conditions, leading to more effective tactical responses.

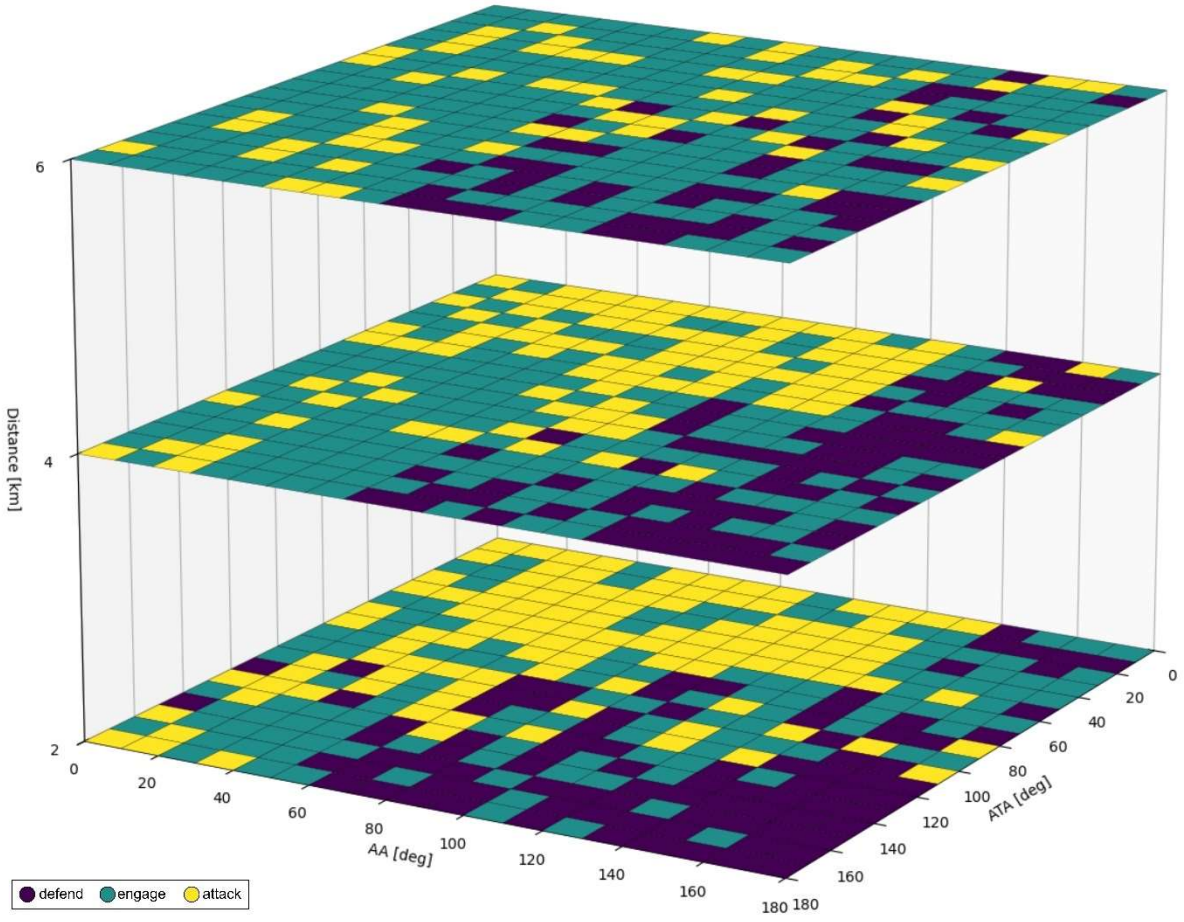

Figure 4-6: Hierarchy activations *per agent*. The mode with the highest frequency is shown.

5.0 CONCLUCSION

In conclusion, while HMARL systems demonstrate significant potential in enhancing decision-making processes in air combat scenarios, there is still much to be improved in terms of explainability. A key area for improvement lies in developing advanced methods to better interpret the internal processes of these AI systems, enabling military personnel to understand the reasoning behind complex decisions. This is particularly important for ensuring trust and transparency when these systems are deployed in critical, high-risk environments like air combat.

One major difference between the AI system and human pilots is the nature of decision-making. AI systems optimize their decision function (policy) based on large data inputs and pre-defined objectives, processing information at speeds and scales beyond human capability. However, AI systems may lack the intuition and adaptability that human pilots bring to complex, dynamic scenarios, as human pilots incorporate their experience, intuition, and contextual awareness to adapt to unforeseen circumstances. This difference raises important safety concerns, as AI systems might not fully comprehend the nuances of each situation, potentially leading to decisions that are technically optimal but operationally unsafe. Ensuring the explainability of the AI's decision-making process is critical, as it allows human operators to trust the system and intervene when required.

Furthermore, the key inputs to the AI's decision process, such as sensor data, radar signals, and environmental variables, may differ from what human pilots prioritize. For instance, AI systems may rely heavily on data-driven metrics like optimal attack angles or velocity, whereas human pilots often incorporate instinctual or experiential knowledge. Identifying and making transparent these differences is vital in integrating AI systems with human teams, ensuring that both humans and AI can collaborate effectively and safely in real-world air combat operations.

In summary, advancing explainable AI in hierarchical MARL for air combat scenarios will not only enhance system performance but will also improve safety, transparency, and the trustworthiness of AI systems, which are crucial for deployment in critical applications.

REFRENCES

[1] Silver D, Hubert T, Schrittweiser J, et al. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. arXiv preprint arXiv:1712.01815, 2017.

[2] Albrecht S, Christianos F, and Schäfer L. Multi-Agent Reinforcement Learning: Foundations and Modern Approaches, MIT Press, 2024.

[3] Kranja A, Brcic M, Tomislav L, et al. Explainability in reinforcement learning: perspective and position. arXiv preprint arXiv:2203.11547, 2022.

[4] Burgin GH, Fogel LJ, and Phelps JP. An adaptive maneuvering logic computer program for the simulation of one-on-one air-to-air combat. Volume 1: General description, No. NASA-CR-2582, NASA, 1975.

[5] Jones RM, Laird JE, Nielsen PE, et al. Automated Intelligent Pilots for Combat Flight Simulation. AI magazine 20.1: 27-27, 1999.

[6] Smith RE, Dike BA, Ravichandran B, et al. Discovering Novel Fighter Combat Maneuvers: Simulating Test Pilot Creativity. Creative evolutionary systems, Morgan Kaufmann, 2002.

[7] Smith RE, Dike BA, Mehra RK, et al. Classifier Systems in Combat: Two-Sided Learning of Maneuvers for Advanced Fighter Aircraft. Computer Methods in Applied Mechanics and Engineering 186.2-4: 421-437, 2000.

[8] Ma X, Xia L, Zhao Q. Air-Combat Strategy Using Deep Q-Learning. Chinese Automation Congress (CAC), IEEE, 2018.

[9] Vlahov B, Squires E, Strickland L, et al. On Developing a UAV Pursuit-Evasion Policy Using Reinforcement Learning. International Conference on Machine Learning and Applications (ICMLA), IEEE, 2018.

[10] Zhang J, Yu Y, Zheng L, et al. Situational Continuity-Based Air Combat Autonomous Maneuvering Decision-Making. Defence Technology 29: 66-79, 2023.

[11] Guo J, Wang Z, Lan J, et al. Maneuver decision of UAV in Air Combat based on deterministic Policy Gradient. International Conference on Control & Automation (ICCA), IEEE, 2022.

[12] Bae JH, Jung H, Kim S, et al. Deep Reinforcement Learning-Based Air-to-Air Combat Maneuver Generation in a Realistic Environment. IEEE Access 11: 26427-26440, 2023.

[13] Wang Z, Li H, Wu H, et al. Improving Maneuver Strategy in Air Combat by alternate Freeze Games with a Deep Reinforcement Learning Algorithm. Mathematical Problems in Engineering, 2020.

[14] Day M. Multi-Agent Task Negotiation among UAVs to defend against Swarm Attacks. Diss. Monterey, California. Naval Postgraduate School, 2012.

[15] DenBroeder GG, and Ellison RE. On optimum Target Assignments. Operations Research 7.3: 322-326, 1959.

[16] Wang L, Hu J, Xu Z, et al. Autonomous Maneuver Strategy of Swarm Air Combat based on DDPG. Autonomous Intelligent Systems 1.1: 15, 2021.

[17] Kong W, Zhou D, Du Y, et al. Hierarchical Multi-Agent Reinforcement Learning for Multi-Aircraft Close-Range Air Combat. IET Control Theory & Applications 17.13: 1840-1862, 2023.

[18] Zhang T, Qiu T, Liu Z, et al. Multi-UAV cooperative short-range Combat via Attention-based Reinforcement Learning using individual Reward Shaping. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2022.

[19] Selmonaj A, Szehr O, Rio GD, et al. Hierarchical Multi-Agent Reinforcement Learning for Air Combat Maneuvering. International Conference on Machine Learning and Applications (ICMLA), volume 22, IEEE, 2023.

[20] Gorton PR, Strand A, and Brathen K. A Survey of Air Combat Behavior Modeling using Machine Learning. arXiv preprint arXiv:2404.13954, 2024.

[21] Glanois C, Weng P, Zimmer M, et al. A Survey on Interpretable Reinforcement Learning. Machine Learning: 1-44, 2024.

[22] Saldiran E, Hasanzade M, Inalhan G, et al. Explainability of AI-Driven Air Combat Agent. Conference on Artificial Intelligence (CAI), IEEE, 2023.

[23] Hasanzade M, Saldiran E, Guner G, et al. Analyzing RL Agent Competency in Air Combat: A Tool for Comprehensive Performance Evaluation. 42nd Digital Avionics Systems Conference (DASC), IEEE, 2023.

[24] Shapley LS. A value for n-person games. Contributions of the Theory of Games vol.2: 307-317, 1953.

[25] Ribeiro MT, Singh S, and Guestrin C. "Why should I trust you?" Explaining the Predictions of any Classifier. International Conference on Knowledge Discovery and Data mining, 2016.

[26] Du Y, Leibo JZ, Islam U, et al. A Review of Cooperation in Multi-Agent Learning. arXiv preprint arXiv:2312.05162, 2023.

[27] Lowe R, Wu YI, Harb J, et al. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. Advances in Neural Information Processing Systems (NeurIPS), Vol. 30, 2017.

[28] Schulman J, Wolski F, Dhariwal P, et al. Proximal Policy Optimization Algorithms. arXiv preprint arXiv:1707.06347, 2017.

[29] Du X, Ye Y, Zhang P, et al. Situation-Dependent Causal Influence-Based Cooperative Multi-Agent Reinforcement Learning. Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 38, No. 16, 2024.

[30] Kraus S, Azaria A, Fiosina J, et al. AI for explaining Decisions in Multi-Agent Environments. Proceedings of the AAAI conference on artificial intelligence, Vol. 34, No. 9, 2020.