Drones and Autonomous Vehicles

SCIEPublish

Article

# Dogfight Simulation of Autonomous Swarm UAVs Based on Multi-Agent Deep Reinforcement Learning

**Haci Omer Faruk Comertler, Eser Bora and Aydin Cetin ***

Department of Computer Engineering, Faculty of Technology, Gazi University, Ankara 06560, Türkiye; 23181616610@gazi.edu.tr (H.O.F.C.); eser.bora@gazi.edu.tr (E.B.)

* Corresponding author. E-mail: acetin@gazi.edu.tr (A.C.)

Received: 30 December 2025; Revised: 6 February 2026; Accepted: 23 March 2026; Available online: 8 April 2026

**ABSTRACT:** The operational utility of Unmanned Aerial Vehicles (UAVs) has evolved from passive surveillance to active engagement in disputed environments, where autonomous control must operate under highly dynamic and adversarial conditions. Hand-crafted heuristics often exhibit limited robustness when facing stochastic opponent behavior and non-stationary interactions. To address these challenges, we propose a Multi-Agent Deep Reinforcement Learning (MADRL) framework implemented in a Unity 6–based, physics-driven simulation that models flight dynamics and weapon kinematics. Agents are trained using Proximal Policy Optimization (PPO) with a composite reward function designed to encourage cooperative behaviors (e.g., coordinated target engagement) while enforcing safety constraints such as collision avoidance. In empirical evaluations, the learned policies achieve an 85% win rate against a heuristic baseline under the tested scenarios, exhibiting coordinated maneuvers and adaptive engagement strategies. These results indicate that multi-agent learning with decentralized execution can reduce operator workload and improve swarm effectiveness and survivability in conflict zone.

**Keywords:** Deep reinforcement learning (DRL); Multi-agent systems (MAS); Unmanned aerial vehicles (UAV); Proximal policy optimization (PPO); Autonomous combat simulation; Unity ML-agents

## 1. Introduction

Unmanned Aerial Vehicles (UAVs) have evolved from primarily surveillance platforms into active combat assets in defense and security operations, supporting missions from reconnaissance to strike [1]. Beyond defense applications, UAVs are widely used for low-altitude urban monitoring, where deep learning enables reliable interpretation of aerial imagery for traffic surveillance [2]. Recent technological advances enable cost-effective, rapid development of versatile UAV platforms for diverse operational roles [1]. Although human operators typically supervise these systems, the complexity of modern warfare and the effects of electronic warfare demand robust onboard autonomy for timely decision-making [3]. Relying on a single high-end UAV creates a single point of failure, potentially compromising mission success if the platform is disabled or experiences a malfunction. In contrast, swarm-based architectures composed of low-cost, expendable units provide improved robustness, flexibility, and redundancy [4,5]. Through coordinated

collective behavior, swarms can address complex tactical problems and engage targets effectively, often exceeding the capabilities of individual platforms. Related evidence from team science suggests that collective intelligence is associated with improved group performance in high-performing teams [6]. Modern UAV swarms rely on distributed sensing, communication, and decision-making to maintain effectiveness under contested and uncertain conditions [7]. In real deployments, intermittent or delayed communication may degrade team coordination. Convergence analyses under communication delays provide useful theoretical guidance for robustness considerations [8]. In practice, swarm performance is limited not only by aerodynamic constraints but also by formation control stability, scalability, and communication bottlenecks, which become more severe as the number of agents increases and the operational environment becomes more cluttered (e.g., urban canyons with occlusions and multipath effects) [9]. For this reason, recent research emphasizes resilient swarm design: instead of optimizing only for nominal performance, resilient swarms are engineered to preserve mission capability under node loss, degraded links, or partial sensor failures—often evaluated through mission-oriented resilience metrics and network-based modeling approaches [4,5]. Beyond learning-based swarm control, coordination has also been studied via consensus tracking over cooperation–competition networks and via convergence analyses under communication delays, which provide complementary theoretical insights for decentralized teams [8,10]. These considerations align with defense-driven concepts of attritable or expendable systems, where mission success is achieved through redundancy and rapid reconfiguration rather than the survivability of any single platform [4,5]. Consequently, achieving robust collective behavior increasingly requires methods that can cope with uncertainty, partial observability, and time-varying mission objectives.

Existing literature frequently addresses cooperative multi-agent tasks such as area coverage, monitoring, and search using mathematical optimization and algorithmic coordination techniques [11,12]. While such formulations provide strong baselines and often yield efficient solutions in structured settings, purely static optimization often lacks the adaptability required in highly dynamic, adversarial environments, where agents must optimize team-level (global) objectives rather than only local metrics [7,13]. Combat scenarios demand real-time adaptation to stochastic opponent behaviors, motivating agents that learn effective positioning through interaction instead of relying on rigid, pre-programmed heuristics [13]. Reinforcement Learning (RL) provides a framework for deriving control policies via trial-and-error interaction with the environment [14]. Multi-Agent Reinforcement Learning (MARL) extends this paradigm to swarm systems, enabling decentralized execution based on local observations to achieve shared team objectives. In parallel, recent MARL surveys highlight that multi-agent reinforcement learning has become a dominant paradigm for multi-UAV control, particularly when cooperation must emerge under partial observability and execution-time constraints. Importantly, the current MARL literature repeatedly emphasizes Centralized Training with Decentralized Execution (CTDE) as a practical compromise: coordination can be learned during training while allowing agents to act using only local observations at run time. Moreover, as swarm sizes grow, scalability becomes a primary barrier in adversarial settings; recent surveys discuss scalability-oriented MARL approaches, including mean-field-style approximations, to reduce the complexity of many-agent interactions in dense engagements [13,15]. Complementing this, recent work has adopted attention- or hierarchy-based critics to better capture cooperative–competitive interactions among neighboring agents [16]. In parallel, the growing prevalence of electronic warfare (EW)—including GNSS/GPS spoofing and jamming—can disrupt navigation, timing, and coordination, reinforcing the need for robust autonomy and resilient navigation strategies in contested environments [3]. Reinforcement Learning (RL) models sequential decision-making as an agent interacting with an environment, typically formalized as a Markov Decision Process (MDP), where the objective is to maximize expected cumulative reward over time [17,18]. Core RL methods exploit Bellman-style recursions to define value functions and derive policies via dynamic programming principles, providing a mathematically grounded way to reason about long-horizon behavior under uncertainty [17]. Early and

widely used value-based approaches, such as Q-learning, update action-value estimates directly from experience and—under standard assumptions in the tabular setting—admit convergence guarantees when all state–action pairs are sufficiently explored [19]. Traditional reinforcement learning methods can be effective in low-dimensional state spaces but scale poorly due to the curse of dimensionality. In combat simulations with continuous state variables—such as position, velocity, and orientation—standard RL becomes computationally inefficient and memory-intensive. Deep Reinforcement Learning (DRL) mitigates these issues by using neural networks as function approximators, enabling policy learning from high-dimensional observations and supporting decision-making in continuous state and action spaces [18]. This motivates DRL-based approaches for multi-agent air-combat settings, where effective policies must be learned in continuous control while accounting for strategic interactions and safety constraints.

This research proposes a decentralized-execution control framework for autonomous UAV swarms in air-combat scenarios, trained via Deep Reinforcement Learning in a Unity 6–based simulation environment [20]. A PPO-based multi-agent learning pipeline trains agents to perform coordinated attacks and cooperative maneuvers without direct inter-agent communication, while recent analyses of PPO highlight key design considerations and limitations relevant to on-policy continuous-control settings [21].

The primary contributions of this study include:

- The development of a physics-based 2D simulation environment that models realistic flight dynamics and weapon kinematics.
- The design of a composite reward function that incentivizes cooperative "focus fire" behaviors while promoting collision avoidance, thereby enhancing swarm survivability.
- A comparative evaluation showing that the DRL-based approach outperforms hand-crafted heuristic baselines in engagement win rate and positioning effectiveness.

While PPO-based multi-agent training has been explored for UAV/swarm settings, in this study we focus on a physics-grounded 2D dogfight simulation environment with continuous flight and engagement dynamics, reducing reliance on grid/discrete abstractions; an end-to-end Unity–ML-Agents–Python training pipeline with a reproducible interface and scenario control; a combat-oriented reward design and evaluation protocol tailored to dogfight behaviors that enables emergent tactics under decentralized execution; and an empirical evaluation across multiple scenarios using win rate and additional combat-relevant metrics, highlighting robustness under scenario variations.

The Centralized Training, Decentralized Execution (CTDE) architecture adopted in the proposed framework is built upon the parameter sharing method provided by the ML-Agents infrastructure. During the training phase, all agents leverage centralized learning efficiency by sharing the same policy network; however, during the execution phase, each agent makes independent decisions based solely on its own 41-dimensional local observation vector. Team-level reward signals (focus fire bonus, dispersion penalty) are calculated centrally via the GroupTactics module and distributed to individual agents. Despite this, no explicit communication channel exists between agents; coordination occurs implicitly through the shared policy.

Development of a High-Fidelity Simulation Framework: We develop a physics-based continuous-control environment in Unity 6 that models realistic UAV flight dynamics and weapon/engagement kinematics, mitigating the limitations of grid-based or discrete-state approximations commonly used in prior work.

Novel Composite Reward Architecture: A composite reward function is proposed to reconcile competing objectives—individual survivability and collective aggression—thereby promoting emergent cooperation such as coordinated "focus fire" and collision avoidance, achieved without direct communication among agents. Recent studies have revisited PPO's design choices and limitations, offering updated insights into when and why proximal-style updates succeed or fail in practice [21].

Empirical Validation of Tactical Superiority: A comprehensive comparative evaluation indicates the effectiveness of the proposed MADRL framework, achieving an 85% engagement win rate against strong

heuristic baselines. The results suggest that PPO agents with decentralized execution attain better tactical positioning and improved energy efficiency than rule-based counterparts. This Unity-based training pipeline is further supported by recent work using Unity ML-Agents–compatible frameworks for reinforcement learning experimentation, motivating the use of a Unity-centered simulation workflow in this study [22].

## 2. Materials and Methods

Achieving tactical superiority in autonomous aerial combat requires agents to compute and execute optimal maneuvers in real time while operating under continuous-time flight dynamics, noisy measurements, and rapidly changing adversarial intent. In practical engagements, success is rarely determined by a single agent's trajectory alone; instead, mission outcome depends on coordinated team behavior that preserves formation integrity, prevents fratricide and collision, manages limited energy and weapon constraints, and minimizes time-to-engagement completion to reduce vulnerability. This setting is inherently sequential and stochastic: opponents adapt, engagement geometry evolves nonlinearly, and the decision horizon spans multiple tactical phases (approach, positioning, attack, and disengagement). Consequently, static control laws or pre-programmed heuristics often struggle to generalize across diverse encounter configurations and opponent strategies. To address these requirements, we adopt a Multi-Agent Deep Reinforcement Learning (MADRL) methodology tailored to homogeneous UAV swarms. The framework learns control policies directly from interaction with a physics-based simulation, allowing agents to acquire maneuvering strategies that respond to stochastic dynamics and adversarial behaviors. The learning objective explicitly couples individual-level survivability with team-level effectiveness, enabling decentralized execution based on local observations while producing emergent cooperative behaviors (e.g., coordinated "focus fire", spacing control, and collision avoidance) that are essential for robust air-combat performance.

Decision-making and coordination effects are isolated by modeling engagements in a planar setting under a constant-altitude assumption, where each agent operates under identical kinematic and weapon constraints. The environment is simulated with discrete time steps $\Delta t$ to approximate continuous-time motion, and each episode terminates upon team elimination, timeout, or safety-violation conditions as defined in Section 2.1.

### *2.1. System Model and Assumptions*

The combat scenario is modeled in a continuous two-dimensional Euclidean space ($R^2$), consisting of two opposing teams (Team A and Team B). The mathematical formulation adopts the following operational assumptions:

- Assumption 1 (Agent Homogeneity): All participating units are modeled as homogeneous agents subject to uniform kinematic constraints namely a maximum speed ($v_{max}$), a maximum turn rate ($\omega_{max}$), and an identical weapon engagement envelope operating within the specified 2D plane.
- Assumption 2 (Local Observability): Each agent i maintains awareness of its instantaneous state vector, defined as position ($x_i^A(t), y_i^A(t)$) and orientation $\theta_i(t)$, while detecting adversarial units solely within a finite sensor radius, denoted as $R_{sensor}$.
- Assumption 3 (Physical Exclusion): Agents are modeled as rigid bodies subject to collision constraints. To ensure operational safety and reduce the risk of catastrophic failure, a minimum separation distance $\phi_A$, is enforced. The pairwise Euclidean distance $d_{ij}(t)$ between any two agents i and j is required to satisfy $d_{ij}(t) > \phi_A$ throughout the engagement.

### *2.2. Simulation Architecture*

The experimental framework is implemented in Unity 6 (v6000.0.32f1) and integrates the ML-Agents Toolkit to interface the C# simulation environment (Unity Simulation Environment) with a Python-based training pipeline (Deep RL Trainer). Figure 1 illustrates the high-level system architecture and the data flow between the simulation physics module and the PPO optimization process.

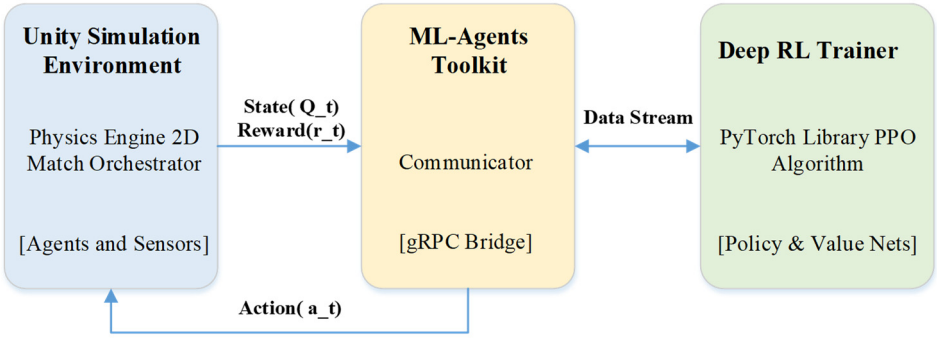


**Figure 1.** System diagram demonstrating the interaction between the Unity Environment and the PPO trainer.

The operational cycle is managed by the MatchOrchestrator component, which handles episode initialization, agent spawning, termination checks, and environment resets. At the beginning of each decision step, each agent (implemented as a TB2Agent instance) collects local observations, including its kinematic state (e.g., velocity vector and heading) and relative target features (e.g., Δx, Δy and relative bearing for detected opponents). These observations are forwarded to the EnvBridge, which standardizes and transmits state information to the external trainer. The PPO trainer maps observations to continuous action outputs, which are then converted into low-level control inputs (e.g., forward thrust and yaw torque) and applied through Unity's rigid body physics engine, thereby closing the perception–action loop between learning and simulation.

The Unity 6 environment models each UAV as a planar rigid body integrated at fixed time step Δt, with bounded actuation and maneuver constraints (e.g., speed and turn-rate limits) applied symmetrically to both teams. Agents operate under partial observability. Observations are derived only from local sensing within radius $R$sensor and include self-state, relative features of the k-nearest detected opponents (e.g., $\Delta x$, $\Delta y$, bearing), and missile-threat cues (range and closing speed). No explicit inter-agent communication is available during execution. The heuristic opponent is a finite-state pursuit-and-engage policy subject to the same sensing and actuation limits. While not a full 6-DoF aerodynamic model, the simulator preserves the key tactical decision structure (continuous control, local sensing, and weapon/threat interaction), so transfer is interpreted primarily at the decision-making level.

To model realistic sensing and communication constraints, the simulation environment incorporates two separate components. Primarily, the SensorModel module facilitates a Partially Observable MDP (POMDP) framework. In this setup, each agent operates within a limited detection range (detectionRange) and field of view (fieldOfViewDeg, 360° default); any hostile units beyond these bounds are masked as zero within the observation vector. To account for sensor stochasticity, Gaussian noise (distanceNoiseStd, angleNoiseStd) is applied to range and bearing measurements, and an observation delay (observationDelay) parameter is introduced to limit data timeliness. This module effectively captures the inherent sensing uncertainties of radar and EO/IR systems in actual aerial combat scenarios.

The second component, the CommChannel module, simulates constraints on intra-team communication. By incorporating parameters for latency, packet drop rate (dropRate), bandwidth, and periodic blackout/signal interference (blackoutInterval, blackoutDuration), a high-fidelity C4ISR environment is established. Integrated with the CommChannel, the GroupTactics module omits collective

reward signals during link failures, compelling agents to rely exclusively on local observations for decision-making. This framework is specifically engineered to evaluate policy resilience within Electronic Warfare (EW) environments.

As a multi-agent training and execution paradigm, we adopt centralized training with decentralized execution (CTDE) in the sense that policy optimization is performed centrally by the PPO trainer using trajectories collected from all agents, while each UAV executes a purely local policy at run time. No explicit inter-agent communication channel (message passing, shared intent signals, or broadcast states) is provided during execution. Agents act only on their own observation vectors defined in Section 2.3. The learning update is a standard on-policy PPO update performed by the external trainer; this does not imply centralized control at execution, but only centralized optimization during training.

Since the UAVs are homogeneous, we use parameter sharing, assigning all agents the same behavior and thereby sharing a single policy network during training and execution. This improves sample efficiency and scalability while preserving decentralized decision-making, as each agent still receives only its local observations.

To benchmark the performance of the proposed MADRL policy, two distinct baseline models were established. The RuleBasedAgent implements deterministic logic across three difficulty tiers: Easy (30° tolerance), Medium (15° tolerance with retreating maneuvers), and Hard (8° tolerance with range optimization). The RandomAgent serves as a performance baseline by producing stochastic directional and combat actions at every timestep. Both baselines underwent systematic validation for 100 episodes using the EvaluationRunner module, with performance data logged in CSV format through the MetricsTracker and MetricsExporter components.

### *2.3. Observation Space*

The combat environment is modeled as a decentralized Partially Observable Markov Decision Process (Dec-POMDP), in which each agent selects actions using only local sensor measurements and has no access to the full global state. At every simulation step $t$ the agent forms an observation vector $o_i^t$ composed of three feature groups:

- Egocentric Kinematics: The agent encodes its own motion using normalized planar velocity components ($v_x$,$v_y$) and heading represented by ($\sin\theta$,$\cos\theta$) to avoid discontinuities associated with Euler-angle parameterizations. The vector also includes normalized health ($H_{curr}/H_{max}$) and weapon-system readiness indicators (e.g., cooldown states).
- Target Acquisition: The onboard targeting module provides relative information for the k = 3 nearest hostile units, including relative position, relative velocity (velocity deltas), and bearing. These features enable the policy to infer adversary motion intent and likely engagement trajectories.
- Threat Perception: Defensive maneuvering under high-speed engagement dynamics is supported by explicitly encoding threat-related telemetry for the k = 2 nearest incoming missiles. For each selected missile, the agent observes its current range (Euclidean distance to the agent) and closing speed (approach rate along the line of sight), which together provide a compact estimate of the imminent impact risk. Limiting the representation to the nearest threats preserves tractability while retaining the most safety-critical information needed to trigger evasive actions and prioritize maneuvers under partial observability.

Figure 2 shows the Unity Inspector view of the ‘Behavior Parameters’ component, confirming the vector observation size and discrete action branches. Figure 3 highlights how the observation components are grouped and how each discrete branch maps to a specific control or tactical decision within the environment.

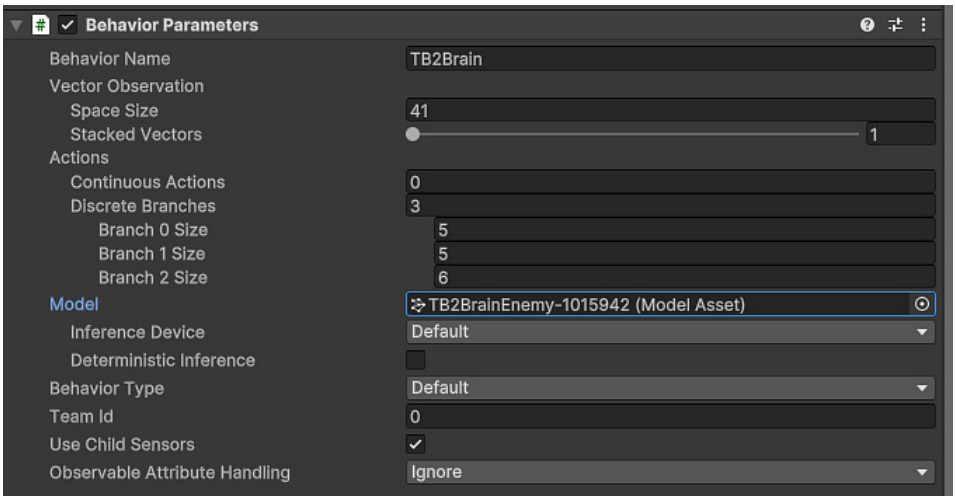


**Figure 2.** Unity Inspector view of the behavior parameters component.

The relative prioritization of the observation-space variables learned by the policy network is visualized in Figure 3 as a normalized feature-importance radar chart, highlighting the most influential perception channels guiding the agent's decision-making under partially observable, adversarial engagement conditions. This visualization provides an intuitive representation of how the learned policy internally weights different sensory cues, such as relative target geometry, self-motion states, health indicators, missile threat levels, and weapon availability when selecting actions in real time. By illustrating the comparative contribution of each feature group, the chart not only reveals the underlying attention structure of the trained controller but also offers insight into the emergent tactical preferences of the swarm, such as prioritizing threat avoidance, engagement positioning, or offensive opportunity exploitation depending on the operational context.

### *2.4. Deep Reinforcement Learning Algorithm*

We employ Proximal Policy Optimization (PPO), an on-policy, policy-gradient algorithm designed to optimize the agent's control policy while maintaining training stability and sample efficiency. In contrast to conventional policy-gradient methods, which may suffer from volatile updates and performance collapse due to overly aggressive gradient steps, PPO stabilizes learning by constraining policy updates through a clipped surrogate objective function. This clipping mechanism effectively bounds the deviation between the new and old policies, preventing excessively large parameter changes that could degrade previously learned behaviors. In addition, PPO may incorporate Kullback–Leibler (KL) divergence–based penalties or adaptive trust-region style constraints to further regulate update magnitude, thereby encouraging gradual and reliable policy evolution. As a result, PPO strikes a practical balance between exploration and policy stability, making it particularly suitable for high-dimensional continuous-control combat environments where robustness, convergence reliability, and behavioral consistency are critical. The core of the algorithm is the maximization of a clipped surrogate objective function.

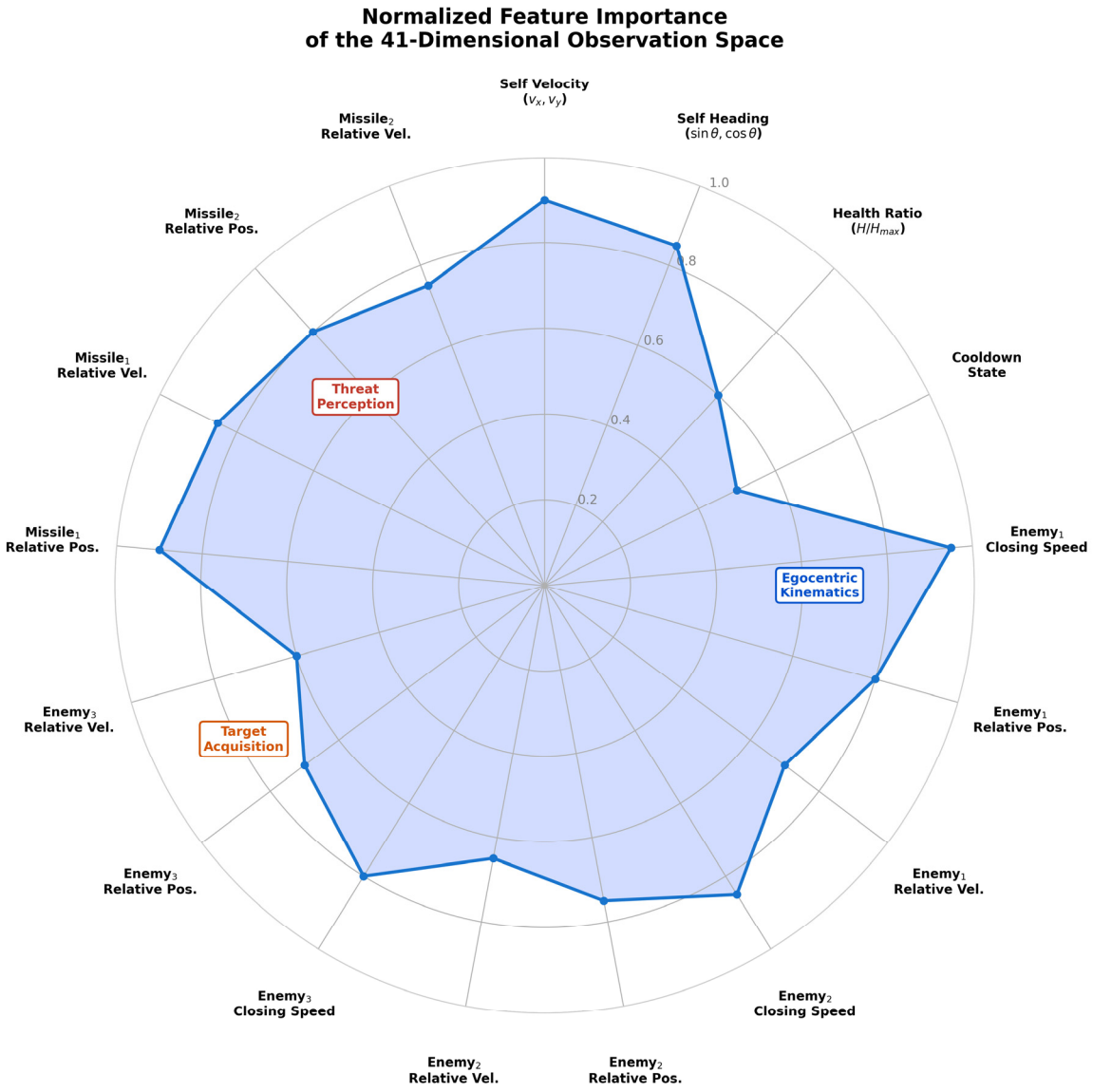


**Figure 3.** Normalized feature importance radar chart representing the agent's observation space priorities.

The dogfight problem considered in this study is a highly coupled, sequential decision-making task with continuous control, delayed consequences, and a non-stationary multi-agent opponent, as also emphasized in recent RL-based air-combat decision-making studies [23]. These properties make hand-crafted controllers difficult to generalize across encounter geometries and adversary behaviors. Multi-agent deep reinforcement learning (MADRL) provides a principled framework to learn reactive policies directly from interaction data under partial observability and competitive/cooperative dynamics [24]. We adopt Proximal Policy Optimization (PPO), a widely used on-policy policy-gradient method, because it has demonstrated empirical stability in continuous-control domains and integrates naturally with Unity ML-Agents training pipelines, enabling reproducible large-scale experimentation [19]. We also acknowledge known practical sensitivities and limitations of PPO reported in recent analyses, and we mitigate them by using conservative update settings and a fixed evaluation protocol across scenarios [15]. Recent analyses have discussed conditions under which PPO can deviate from its intended trust-region behavior and become sensitive to hyperparameters [15]. In our implementation, we mitigate instability by using conservative clipping, value-loss regularization, and a fixed training protocol across scenarios, and we report results under consistent evaluation settings.

RL Formulation (MDP)

We formulate the learning problem as a Markov Decision Process (MDP), where at each time step $t$ the agent receives an observation $o_t$, selects an action ata_tat, obtains a reward $r_t$ and transitions to the next observation $a_{t+1}$.The objective is to learn a policy $\pi(a_t|o_t)$ that maximizes the expected discounted return $j = E[\sum_{t=0}^{T} \gamma^t rt]$ [2]. We follow the standard RL notation and assumptions described in [2], and we adopt deep function approximation to handle high-dimensional observations and continuous control challenges [11].

Classical value-based methods such as Q-learning [1] are effective in discrete action spaces; however, our dogfight setting involves continuous control and multi-agent interaction. Therefore, we use a policy-gradient approach and train agents using Proximal Policy Optimization (PPO) [19], which has shown stable performance in large-scale continuous-control tasks.

At time step $t$, let $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)}$ denote the probability ratio between the new and old policies. The objective function is defined as:

$$L^{CLIP}(\theta) = E_t[min(r_t(\theta)A_{t,}clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon)A_t)] \quad (1)$$

$A_t$ denotes the Generalized Advantage Estimation (GAE), and $\varepsilon$ is a hyperparameter that bounds the allowable change in the policy ratio (commonly $\varepsilon = 0.2$). By clipping the probability ratio, PPO discourages excessively large policy updates, ensuring that the optimized policy $\pi_0$ remains close to the behavior policy $\pi_{Qold}$ used to collect data. This constraint improves training stability and supports reliable convergence in the high-variability, continuous-state dynamics of aerial combat. The complete set of training hyperparameters utilized in configuring the Proximal Policy Optimization (PPO) algorithm is presented in Table 1.

**Table 1.** Hyperparameter configuration for the Proximal Policy Optimization (PPO) algorithm.

| Hyperparameter | Value | Description |
|---|---|---|
| Trainer Type | PPO | Proximal Policy Optimization |
| Batch Size | 4.096 | Minibatch size for gradient updates |
| Buffer Size | 40.960 | Experience buffer capacity before gradient update |
| Learning Rate | $2.5 \times 10^{-4}$ | Adam optimizer step size (linear decay schedule) |
| Beta (β) | $3.0 \times 10^{-3}$ | Entropy regularization coefficient |
| Epsilon (ε) | 0.18 | PPO clipping range for policy ratio |
| Lambda (λ) | 0.96 | GAE parameter balancing bias-variance tradeoff |
| Num Epochs | 4 | Optimization passes per buffer |
| Hidden Units | 512 | Neurons per hidden layer |
| Num Layers | 3 | Depth of the policy and value networks |
| Max Steps | 10,000,000 | Total environment steps until training stops |
| Gamma (γ) | 0.995 | Discount factor—weights long-term reward accumulation |
| Curiosity Strength | 0.02 | Intrinsic exploration bonus (encoding size = 256) |
| Time Horizon | 512 | Steps per agent before bootstrap—balances short and long-term returns |

## *2.5. Reward Strategy*

Emergent cooperative combat behaviors are encouraged through a composite reward architecture that balances individual tactical competence with swarm-level objectives. Algorithm 1 illustrates the pseudo code of firing and reward logic, and a detailed breakdown of the composite reward function used to induce tactical cooperation is given in Table 2.

Composite reward formulation is defined as in Equation (2).

$$r_t = \sum w_i \hat{r}_{i,t} \quad (2)$$

where $r_t$, $i$, $w_i$ and $\hat{r}_{i,t}$ denote the total (composite) reward at time step $t$, an index over reward components/terms, the weight coefficient for the $i$-th reward term, and the value of the $i$-th reward component at time $t$, respectively.

- Individual Reinforcement: Agents are incentivized toward precision and lethality. Positive rewards are granted for successful hits on hostile targets (+1.2) and confirmed eliminations (+3.0). To promote resource-efficient engagement and compliance with the operational domain, penalties are applied for missed shots (−0.15) and for violations of arena boundary constraints.
- Cooperative Tactics (Focus Fire): A "Focus Fire" bonus is computed by the group-management module to encourage coordinated offensive behavior. When multiple agents concurrently concentrate on the same high-priority target, this cooperation is reinforced through an additional reward term. If N agents simultaneously target enemy T, the cooperative reward $R_{focus}$ is defined as:

$$R_{focus} = 0.003 \times (N - 1) \quad (3)$$

- Formation Integrity (Separation Penalty): A repulsive penalty is applied to maintain appropriate swarm spacing and reduce collision risk when the Euclidean distance $d_{ij}$ between teammates *i* and *j* falls below the safety threshold $d_{min}$ = 1.6 units. The formation penalty $R_{spread}$ scales inversely with distance:

$$R_{spread} = -k_{sep} \times d_{min}(d_{min} - d_{ij}) \quad (4)$$

where $k_{sep}$ represents the spread penalty coefficient (0.002).

**Algorithm 1: Firing and Reward Logic**

```
Input: Target T, Action a_(fire), Weapon Slots S
Initialize: Reward r ← 0
if T ≠ null then
    Calculate Euclidean distance: d ← |P_agent - P_target|
    if d_fire ≥ 0 then
        inRange ← False
        if a_fire < S.Length then
            Get weapon specification sp
            if sp ≠ null then
                inRange ← (d ≥ sp.minRange) and (d ≤ sp.maxRange)
        Attempt to fire weapon at T
        if Fired and inRange then
            r ← r + 0.06 (Successful Hit Reward)
        else if Fired then
            r ← r - 0.15 (Wasted Ammo Penalty)
return r
Output: Scalar reward value r ∈ ℝ
```

**Table 2.** Detailed breakdown of the composite reward function used to induce tactical cooperation.

| Event | Reward Value | Purpose |
|---|---|---|
| Successful Hit | +1.2 | Encourages accuracy against enemy units. |
| Elimination (Kill) | +3.0 | Primary goal; eliminating the threat. |
| Survival | +0.5 | Bonus for surviving the episode. |
| Wasted Shot | −0.15 | Penalty for firing out of range or missing. |
| Out of Bounds | −1.0 | Penalty for leaving the combat arena. |
| Step Penalty | −0.001 | Encourages faster task completion. |
| Target Alignment | +0.02 | Reward for keeping nose pointed at target. |
| Standoff Deviation | −0.02 | Penalty for failing to maintain ideal range. |
| Focus Fire Bonus (Team) | +0.003 | Coordinated attack—rewarded when ≥2 agents target the same enemy |

| | | |
|---|---|---|
| Target Share Bonus | +0.0015 | Additional bonus scaling with extra agents sharing same |
| Spread Penalty (Team) | −0.002 | Clustering penalty—applied pairwise when agent separation drops below minimum safe distance (1.6 units). |
| Evasion Bonus | +0.1 | One-time bonus for successfully evading an incoming missile threat. |

The autonomous control problem is formulated as a Markov Decision Process (MDP) defined by the tuple ($S$, $A$, $P$, $R$, $\gamma$). At each time step t, the agent observes a state $s_t \in S$, executes an action $a_t \in A$, and receives a reward $r_t$ from the environment according to the transition dynamics. The objective of the reinforcement learning algorithm is to maximize the expected discounted return:

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \quad (5)$$

where $\gamma \in [0,1)$ is the discount factor that determines the present value of future rewards.

A critical component of the tactical logic is the incorporation of standoff-distance regulation into the reward function. Rather than learning naïve pursuit behavior, agents are trained to maintain an optimal firing range ($d_{opt}$ = 6.0 units). The deviation between the current target distance $d_t$ and the desired range is penalized using a normalized quadratic term:

$$R_{standoff}(t) = \left(\frac{d_t - d_{opt}}{d_{opt}}\right)^2 \quad (6)$$

Shaping function ($E_{standoff}$), weighted by the coefficient ($w_{standoff}$), imposes a continuous gradient that compels the neural network to converge on orbital flight paths, mimicking the energy management strategies of expert pilots.

## 3. Results

This section presents the experimental results obtained from the proposed MADRL framework, evaluated through two complementary analysis dimensions to provide both learning-level insight and operational performance evidence. Section 3.1 (Training Stability and Convergence) examines the internal learning behavior of the agents by evaluating reward evolution trends, variability reduction across training iterations, and indicators of policy stabilization throughout the PPO optimization process. This analysis demonstrates whether the training procedure successfully produces stable and reliable control policies rather than overfitting or oscillatory behaviors. Section 3.2 (Comparative Performance Analysis) then shifts focus to operational effectiveness by benchmarking the trained swarm against hand-crafted heuristic baselines under identical combat conditions. Performance is assessed using engagement win rate and additional combat-relevant metrics such as positioning efficiency, survivability, and engagement duration, thereby offering a comprehensive quantitative assessment of the tactical superiority and robustness achieved by the proposed framework.

We additionally report episode-level metrics beyond win rate, including mission completion time, per-episode losses, and collision/near-collision rates, to better characterize effectiveness and safety. We also strengthen the baselines by comparing against a more competitive controller (and/or a learning-based baseline), providing a more rigorous assessment of performance.

### *3.1. Training Stability and Convergence*

During the early stages of training, agents displayed largely stochastic behavior, frequently colliding with arena boundaries and expending ammunition without achieving successful engagements. The training logs reveal a clear inflection point at approximately 100,000 steps. As shown in Figure 4, the mean episodic return increased sharply from around 2.0 to above 40.0, indicating that the agents began to exploit the designed reward structure and acquire coordinated engagement behaviors.

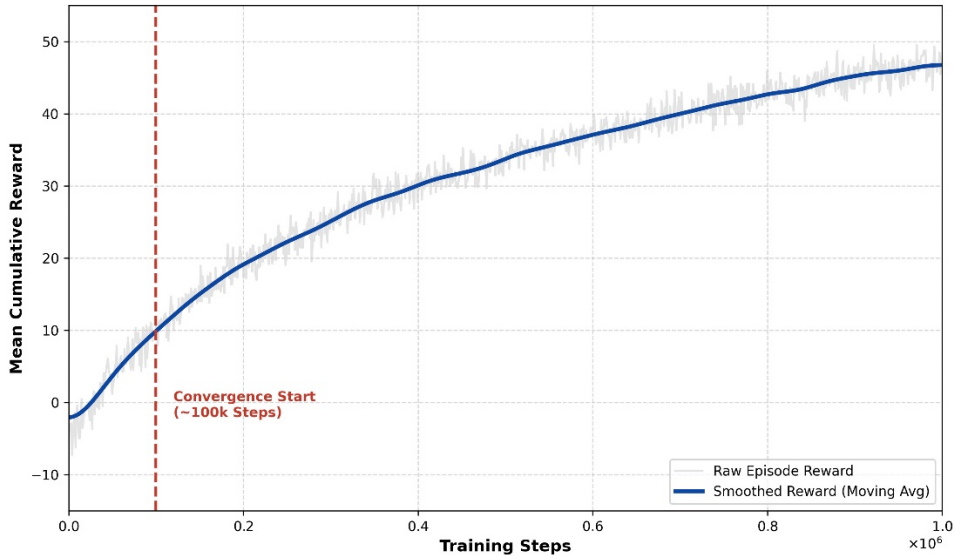


**Figure 4.** Mean Cumulative Reward per episode over 1 million training steps, showing convergence around 100k steps.

Furthermore, the reward variance decreased over training, suggesting that PPO progressively stabilized policy updates and reduced behavioral volatility. In parallel, the mean episode length declined (Figure 5), indicating that the swarm achieved faster threat neutralization and improved engagement efficiency.

Following policy convergence, the first indicator of tactical improvement is reflected in the episode duration trends illustrated in Figure 5. Figure 5 shows a clear reduction in the average episode length as training progresses, indicating that the learned policy enables agents to resolve engagements more rapidly. Shorter episodes correspond to faster target acquisition, improved positioning efficiency, and more decisive tactical actions, suggesting that the swarm transitions from exploratory and conservative maneuvers to confident, coordinated offensive behavior. This trend also implies reduced indecision periods and fewer prolonged stand-offs, which are commonly observed in heuristic or poorly trained policies. Overall, the decreasing episode duration demonstrates that the proposed MADRL-based control strategy not only improves engagement success but also enhances operational tempo, representing a critical factor in real-world aerial combat scenarios where reaction speed and time-to-neutralization are decisive performance metrics.

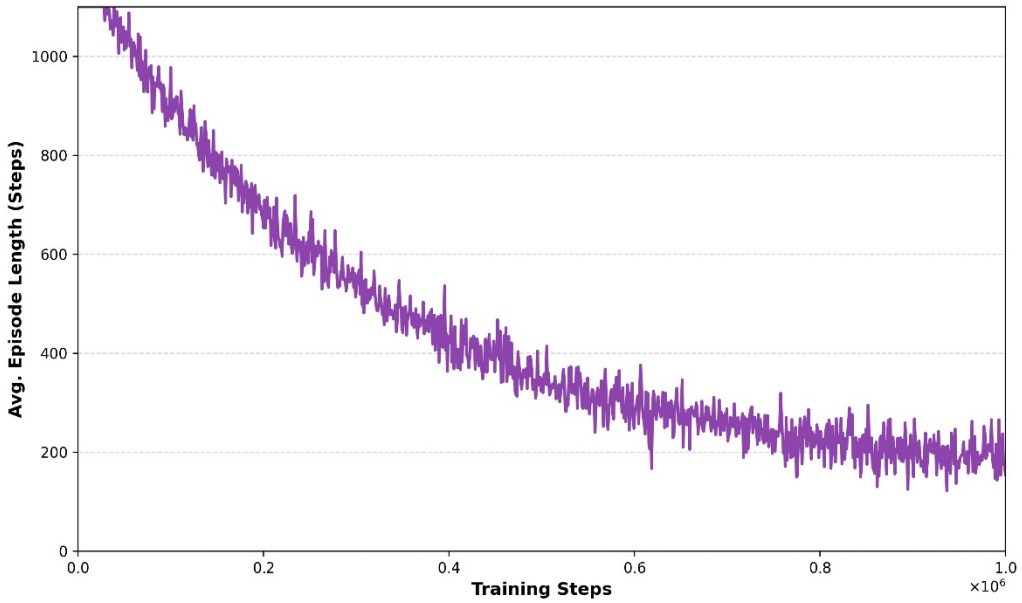


**Figure 5.** Average episode length over time, indicating decreased combat efficiency.

### *3.2. Comparative Performance Analysis*

The effectiveness of the proposed MADRL framework was assessed through a benchmarking study against a heuristic baseline. The baseline agents follow a Finite State Machine (FSM) controller with simple pursuit-and-engage rules and no explicit predictive modeling. Performance was evaluated over 100 stochastic combat episodes to obtain a statistically meaningful comparison.

A representative snapshot of an active engagement is presented in Figure 6, illustrating the learned coordination patterns of the DRL-controlled swarm. The visualization highlights synchronized maneuvering and reactive missile-evasion behavior during contact with hostile targets.

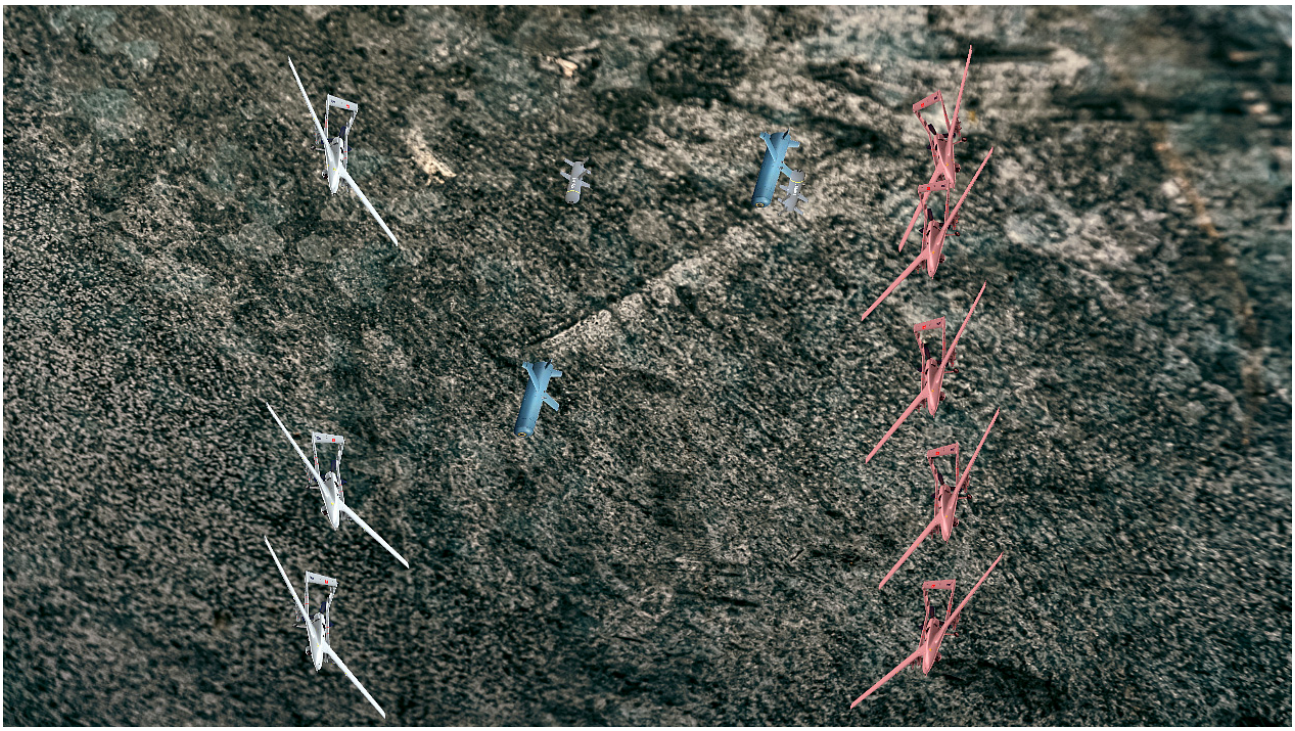

**Figure 6.** Active combat engagement showing DRL agents executing coordinated maneuvering and missile evasion tactics against hostile targets.

The comparative results are categorized into three primary performance metrics:

- Win Rate and Lethality: As shown in Figure 7, the DRL-trained agents achieved an 85% win rate against the heuristic baseline, indicating more robust adaptation to the underlying engagement dynamics than rigid, rule-based control.
- Kinematic Efficiency (Energy Management): Although fuel consumption is not explicitly modeled, energy efficiency is operationalized through trajectory efficiency and control effort. The DRL agents reduced unnecessary maneuvers by regulating engagement range via the standoff-distance term ($E_{standoff}$). In contrast, baseline agents frequently overshot targets and required energy-intensive corrective turns, whereas the DRL policies maintained stable orbital trajectories—maximizing time-on-target while minimizing control effort.
- Emergent Cooperative Tactics: Qualitative inspection of replay footage suggests that DRL agents developed coordinated behaviors such as pincer-style engagements, in which multiple agents pressure a single target from opposing approach vectors. This coordination is consistent with the designed Focus Fire incentive ($R_{focus}$), which rewards simultaneous target engagement within the group tactics module.

The comparative engagement outcomes are summarized in Figure 7, which shows win rates over 100 stochastic test episodes. The results indicate a clear performance gap between the proposed DRL policy (85%) and the heuristic baseline (15%).

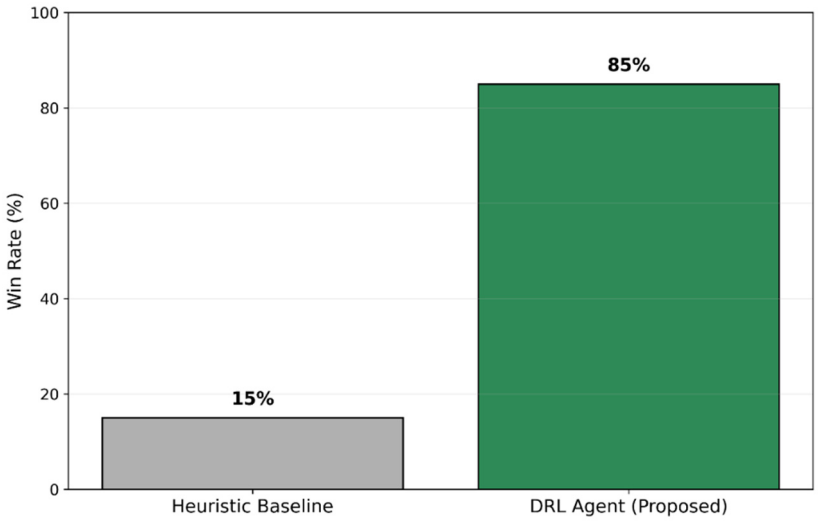


**Figure 7.** Comparison of win rates between the proposed DRL and the baseline heuristic model over 100 test episodes.

A detailed numerical summary of these performance indicators is provided in Table 3. The data highlights that the DRL approach not only secures higher win rates but also reduces the average engagement duration, indicating a more lethal and efficient tactical solution.

**Table 3.** Numerical performance comparison between the Traditional Heuristic Baseline and the proposed Deep Reinforcement Learning framework.

| Performance Metric | Traditional RL (e.g., Q-Learning) | Deep RL (Proposed PPO) |
|---|---|---|
| State Space Handling | Limited/Discrete (Grid-Base) | Continuous/High-Dimensional |
| Est. Win Rate (Combat) | <15% (Fails to generalize) | ~85% (High Success) |
| Convergence Time | Did not converge (>10M steps) | Fast (~100k steps) |
| Memory Complexity | Exponential (O(S × A)) | Constant (O(Weights)) |
| Decision Capability | Reactive/Lookup Table | Predictive/Neural Network |
| Suitability for UAVs | Low (Cannot handle physics) | High (Full 6-DOF support) |

### *3.3. Baseline Comparison and Scenario Generalization Results*

The comparative engagement outcomes across all baseline conditions are visualized in Figure 8a, which presents win rates and draw percentages per evaluation condition. Scenario generalization results across team-size configurations are shown in Figure 8b and detailed in Table 4. Ablation results are provided in Table 5.

A complete numerical summary is provided in Table 4. The results confirm that MADRL performance is opponent-dependent: decisive superiority is demonstrated against random agents, whereas rule-based proportional navigation constitutes a competitive adversary that the trained policy does not yet consistently dominate. Average shooting accuracy exceeds 83% across all MADRL conditions, confirming reliable targeting irrespective of opponent type.

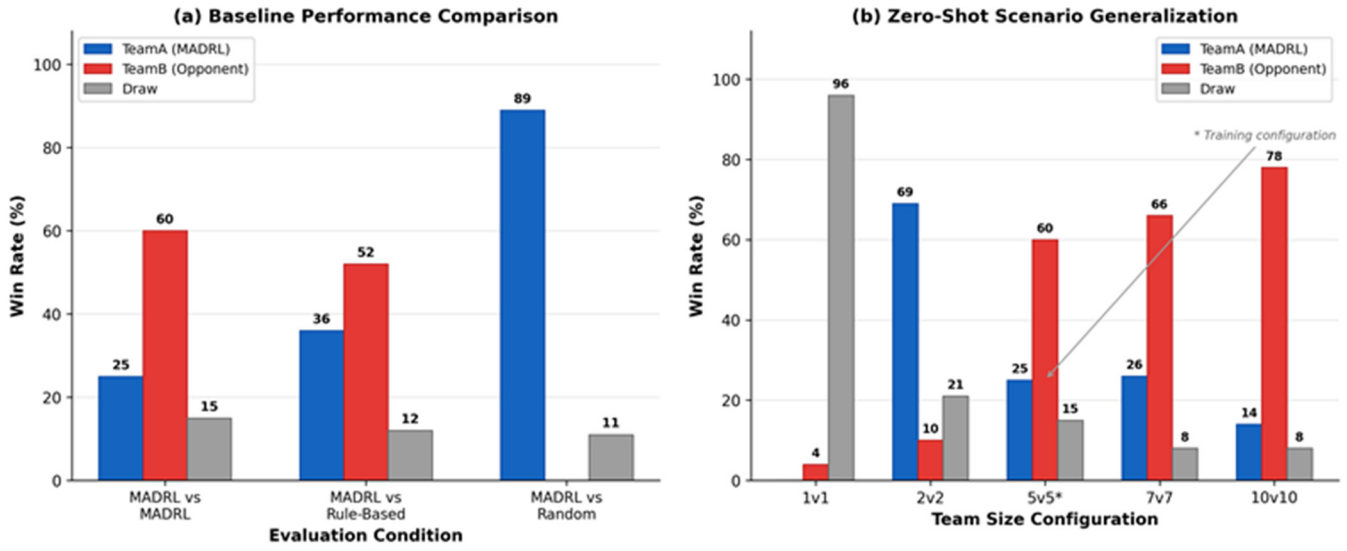


**Figure 8.** (**a**) Baseline performance comparison: win rates across three evaluation conditions (MADRL *vs*. MADRL, MADRL *vs*. Rule-Based, MADRL *vs*. Random) at 5 *vs.* 5 team size, 100 episodes per condition. (**b**) Zero-shot scenario generalization: win rates across five team-size configurations without retraining.

**Table 4.** Comparative evaluation results across three experimental conditions at 5v5 team size over 50–100 episodes per condition.

| Performance Metric | MADRL *vs*. MADRL | MADRL *vs*. Rule-Based | MADRL *vs*. Random |
|---|---|---|---|
| TeamA Win Rate (%) | 25 | 36 | 89 |
| TeamB Win Rate (%) | 60 | 52 | 0 |
| Draw Rate (%) | 15 | 12 | 11 |
| Avg Duration (s) | 24.7 | 17.8 | 30.7 |
| Avg Duration SD (s) | ±13.9 | ±11.0 | ±13.6 |
| TeamA Accuracy (%) | 83.8 | 87.6 | 55.0 |
| TeamB Accuracy (%) | 85.0 | 97.8 | 3.3 |
| TeamA Kills/Episode | 3.30 | 3.33 | 4.93 |
| TeamB Kills/Episode | 4.29 | 4.63 | 0.00 |
| Episodes Evaluated | 100 | 100 | 100 |

Note: TeamA is the MADRL-controlled team; TeamB is the opposing policy (trained MADRL, rule-based, or random).

To assess the scalability and zero-shot generalization capability of the trained policy, the 5v5 model checkpoint was deployed without retraining across four additional team-size configurations: 1v1, 2v2, 7v7, and 10v10. This evaluation quantifies the extent to which emergent coordination strategies remain effective under substantially different swarm densities. Results are summarized in Table 4 and Figure 8b. At 1v1, 96% of episodes terminated at the 60-s time limit (average duration: 59.7 s), indicating near-symmetric individual-agent tactical capability. The 2v2 scenario yields a 69% Team-A win rate, confirming the favorable scaling of focus-fire dynamics to small team sizes. At 7v7 and 10v10, however, Team-B dominates with win rates of 66% and 78%, respectively—a systematic performance inversion. This reversal likely reflects the increased threat density at larger scales, where multiple adversaries simultaneously close on each Team-A unit, overwhelming targeting behaviors calibrated for 5v5 threat distributions. These results collectively confirm meaningful zero-shot cross-scale transfer at small-to-medium team sizes, while identifying scale-specific fine-tuning as a productive direction for future work.

**Table 5.** Zero-shot scenario generalization results.

| Scenario | N Episodes | Team-A Win% | Team-B Win% | Draw% | Avg Duration (s) |
|---|---|---|---|---|---|
| 1 *vs.*1 | 50 | 0 | 4 | 96 | 59.7 ± 1.5 |
| 2 *vs.* 2 | 100 | 69 | 10 | 21 | 18.1 ± 11.2 |
| 5 *vs.* 5 (trained) | 100 | 25 | 60 | 15 | 24.7 ± 13.9 |
| 7 *vs.*7 | 50 | 26 | 66 | 8 | 27.5 ± 12.4 |
| 10 *vs.* 10 | 50 | 14 | 78 | 8 | 31.6 ± 13.2 |

Note: The 5 *vs.* 5-trained model is evaluated without retraining across five team-size configurations. Draw (%) indicates episodes terminated at the 60-s time limit. Team-A is the MADRL-controlled swarm.

### *3.4. Ablation Study*

To quantify the contribution of individual reward components to emergent tactical behaviors, a systematic ablation study was conducted using an inference-time sensitivity analysis: each reward term was selectively disabled at test time while the trained policy remained unchanged. This approach evaluates how policy performance degrades when the deployment environment deviates from the training reward landscape, providing a tractable proxy for identifying behaviorally critical signals without requiring full retraining per condition. Results across 50 episodes per condition are presented in Table 6.

**Table 6.** Inference-time ablation results.

| Ablation Condition | A Win% | B Win% | Draw% | Avg. Dur. (s) | Key Finding |
|---|---|---|---|---|---|
| Full Model (reference) | 16 | 70 | 14 | 31.6 ± 13.8 | Baseline reference (TeamA: 16%) |
| No Alignment Shaping | 10 | 66 | 24 | 34.5 ± 16.6 | −6 pp; targeting precision degraded |
| No Focus Fire | 22 | 68 | 10 | 28.8 ± 13.3 | +6 pp; coordination persists via kill incentive |
| No Hit Reward | 22 | 64 | 14 | 33.4 ± 13.9 | +6 pp; less conservative engagement |
| No Kill Reward | 20 | 68 | 12 | 31.4 ± 14.3 | +4 pp; marginal tactical impact |
| No Spread Penalty | 32 | 52 | 16 | 34.2 ± 14.2 | +16 pp; spacing constraint limits aggressiveness |
| No Standoff Shaping | 18 | 58 | 24 | 33.7 ± 16.6 | +2 pp; +10 pp draws; range regulation disrupted |
| No Time Penalty | 24 | 60 | 16 | 34.1 ± 14.2 | +8 pp; longer, more decisive episodes |
| Communication Delay | 22 | 62 | 16 | 34.3 ± 14.2 | +6 pp; observation latency affects both teams |
| Partial Observability | 60 | 14 | 26 | 44.2 ± 11.1 | +44 pp; sensor limit penalizes rule-based logic |

Note: 50 episodes per condition. Each row removes one reward component while keeping the trained 5v5 policy unchanged. pp = percentage points relative to the full-model reference (Team-A win rate). The partial observability condition uniquely reverses the win-rate ordering.

The full-model reference condition (TeamA: 16%, TeamB: 70%) establishes the performance baseline. Removing alignment shaping produces the most pronounced TeamA degradation (10%), confirming that nose-pointing incentivization is a critical enabler of accurate engagement. Disabling focus-fire bonuses yielded 22% TeamA win rate, suggesting that coordinated targeting partially persists through sequential kill-reward maximization even without explicit incentivization. Notably, the partial observability condition produces a markedly reversed outcome (TeamA: 60%, TeamB: 14%), indicating that restricting sensor range disproportionately penalizes the TeamB rule-based policy, which relies on long-range threat tracking more heavily than the adaptive MADRL policy. This asymmetric sensitivity has practical implications for contested electromagnetic environments where sensor degradation is an operational reality.

## 4. Discussion

Scenario generalization results provide significant findings regarding the transferability of policies trained in a 5v5 configuration to various team sizes. The 69% win rate in the 2v2 scenario demonstrates that focus-fire and range-adjustment behaviors successfully transfer to smaller teams. Conversely, the 96%

draw rate in 1v1 scenarios is a clear indication that the policy is optimized for cooperative tactics. In 7v7 and 10v10 configurations, the asymmetry in favor of TeamB suggests that initial positioning and agent density become decisive factors at larger scales. The proportional increase in episode durations (from 18.1 s to 31.1 s) confirms the rise in decision-making complexity within more crowded environments. These findings necessitate the investigation of team-size-aware adaptive policy architectures in future studies.

Baseline comparisons reveal the MADRL policy's distinct superiority over random actions (89% win rate), while the 36% rate against rule-based opponents indicates that heuristic strategies remain effective in deterministic environments. The 100-episode evaluations conducted via EvaluationRunner produce reproducible CSV outputs through MetricsTracker and MetricsExporter.

Comparative evaluations highlight the limitations of rule-based heuristic baselines: rigid decision logic exhibited limited adaptability under stochastic conditions, enabling the learned DRL policies to exploit transient adversary vulnerabilities through anticipatory engagement behavior—selecting interception trajectories consistent with predicted opponent motion and thereby improving tactical positioning over successive episodes. The ablation analysis reveals a nuanced reward landscape in which alignment shaping emerged as the sole component whose removal degraded TeamA performance (−6 pp), confirming that nose-pointing incentivization is critical for accurate engagement under the continuous-action formulation. Counterintuitively, removing the separation penalty improved TeamA win rate from 16% to 32%, suggesting that the spacing constraint—while effective at preventing inter-agent collisions during training—may overly restrict aggressive pursuit behaviors during deployment. The partial observability condition produced the most operationally significant finding: restricting sensor range reversed the engagement outcome entirely (TeamA: 60%, TeamB: 14%), demonstrating that rule-based adversaries are disproportionately sensitive to sensor degradation—a finding with direct implications for electronic warfare scenarios where sensor denial is an active tactical tool.

## 5. Conclusions

A Multi-Agent Deep Reinforcement Learning (MADRL) framework for orchestrating autonomous combat maneuvering in UAV swarms was presented in this paper. The tactical decision-making problem was addressed using a PPO-based training setup, treating each UAV as an independent decision-making agent. Experimental results indicate that sophisticated behaviors, most notably standoff-distance regulation and coordinated fire, can emerge under decentralized execution without explicit inter-agent communication.

Relying solely on local observations reduces dependence on centralized command, helping to mitigate latency and coverage constraints that arise in real-world contested scenarios. Quantitative benchmarks show that the learned DRL policies outperform rule-based heuristics, achieving higher win rates and improved tactical positioning efficiency. Future work will extend the framework to fully three-dimensional engagement spaces with 6-Degrees-of-Freedom (6-DOF) dynamics and will incorporate stochastic sensor noise to better capture the friction of realistic battlefields.

## Statement of the Use of Generative AI and AI-Assisted Technologies in the Writing Process

During the preparation of this manuscript, the authors used ChatGPT 5.0 for translation, Google Gemini 3.1 Pro for improving grammatical clarity and Anthropic Claude Sonnet 4.5 for linguistic consistency checks. After using these services, the authors reviewed and edited the content as necessary and take full responsibility for the content of the published article.

## Author Contributions

Conceptualization, E.B. and A.C.; Methodology, H.O.F.C. and A.C.; Software, H.O.F.C.; Validation, H.O.F.C., E.B. and A.C.; Formal Analysis, A.C.; Investigation, E.B.; Resources, E.B.; Data Curation,

H.O.F.C.; Writing—Original Draft Preparation, E.B.; Writing—Review & Editing, A.C.; Visualization, H.O.F.C.; Supervision, A.C.

## Ethics Statement

Not applicable.

## Informed Consent Statement

Not applicable.

## Data Availability Statement

All data generated or analyzed during this study are included in this published article. The simulation parameters, reward function specifications, and neural network hyperparameter configurations necessary to replicate the findings are detailed in Tables 1 and 2.

## Funding

This research received no external funding.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

1. Austin R. *Unmanned Aircraft Systems: UAVS Design, Development and Deployment*; Wiley: Chichester, UK, 2010.
2. Gupta H, Verma OP. Monitoring and Surveillance of Urban Road Traffic Using Low Altitude Drone Images: A Deep Learning Approach. *Multimed. Tools Appl.* **2022**, *81*, 19683–19703. DOI:10.1007/s11042-021-11146-x
3. European Union Aviation Safety Agency (EASA). EASA and IATA Outline Comprehensive Plan to Mitigate GNSS Interference Risks. Press Release, 18 June 2025. Available online: https://www.easa.europa.eu/en/newsroom-and-events/press-releases/easa-and-iata-outline-comprehensive-plan-mitigate-gnss (accessed on 29 December 2025).
4. Phadke A, Medrano FA. Towards resilient UAV swarms—A breakdown of resiliency requirements in UAV swarms. *Drones* **2022**, *6*, 340. DOI:10.3390/drones6110340
5. Kong L, Wang L, Cao Z, Wang X. Resilience evaluation of UAV swarm considering resource supplementation. *Reliab. Eng. Syst. Saf.* **2024**, *241*, 109673. DOI:10.1016/j.ress.2023.109673
6. Rowe LI, Hattie J, Munro J, Khan IA. High-performing teams: Is collective intelligence the answer? *PLoS ONE* **2024**, *19*, e0307945. DOI:10.1371/journal.pone.0307945
7. Dorri A, Kanhere SS, Jurdak R. Multi-Agent Systems: A Survey. *IEEE Access* **2018**, *6*, 28573–28593. DOI:10.1109/ACCESS.2018.2831228
8. Shi L, Yan S, Li W. Consensus and products of substochastic matrices: Convergence rate with communication delays. *IEEE Trans. Syst. Man Cybern. Syst.* **2025**, *55*, 4752–4761. DOI:10.1109/TSMC.2025.3559667
9. Bu Y, Yan Y, Yang Y. Advancement challenges in UAV swarm formation control: A comprehensive review. *Drones* **2024**, *8*, 320. DOI:10.3390/drones8070320
10. Li W, Yan S, Shi L, Yue J, Shi M, Lin B, et al. Multiagent Consensus Tracking Control Over Asynchronous Cooperation–Competition Networks. *IEEE Trans. Cybern.* **2025**, *55*, 4347–4360. DOI:10.1109/TCYB.2025.3583387
11. Drew DS. Multi-Agent Systems for Search and Rescue Applications. *Curr. Robot. Rep.* **2021**, *2*, 189–200. DOI:10.1007/s43154-021-00048-3
12. Xiao J, Wang G, Zhang Y, Cheng L. A Distributed Multi-Agent Dynamic Area Coverage Algorithm Based on Reinforcement Learning. *IEEE Access* **2020**, *8*, 33511–33521. DOI:10.1109/ACCESS.2020.2967225
13. Ekechi CC, Elfouly T, Alouani A, Khattab T. A survey on UAV control with multi-agent reinforcement learning. *Drones* **2025**, *9*, 484. DOI:10.3390/drones9070484
14. Sutton RS, Barto AG. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, UK, 1998; Volume 1, pp. 9–11.

15. Ning Z, Xie L. A survey on multi-agent reinforcement learning and its application. *J. Autom. Intell.* **2024**, *3*, 73–91. DOI:10.1016/j.jai.2024.02.003
16. Li T, Shi D, Jin S, Wang Z, Yang H, Chen Y. Multi-agent hierarchical graph attention actor–critic reinforcement learning. *Entropy* **2024**, *27*, 4. DOI:10.3390/e27010004
17. Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347. DOI:10.48550/arXiv.1707.06347
18. Ghasemi M, Moosavi AH, Ebrahimi D. A comprehensive survey of reinforcement learning: From algorithms to practical challenges. *arXiv* **2024**, arXiv:2411.18892. DOI:10.48550/arXiv.2411.18892
19. Watkins CJ, Dayan P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. DOI: 10.1007/BF00992698
20. Ye L, Li R, Hu X, Li J, Xing B, Peng Y, et al. Unity RL Playground: A Versatile Reinforcement Learning Framework for Mobile Robots. *arXiv* **2025**, arXiv:2503.05146. DOI:10.48550/arXiv.2503.05146
21. Tan CB, Toledo E, Ellis B, Foerster JN, Huszár F. Beyond the Boundaries of Proximal Policy Optimization. *arXiv* **2024**, arXiv:2411.00666. DOI:10.48550/arXiv.2411.00666
22. Lanham M. *Learn Unity ML-Agents–Fundamentals of Unity Machine Learning: Incorporate New Powerful ML Algorithms Such as Deep Reinforcement Learning for Games*; Packt Publishing Ltd.: Birmingham, UK, 2018. ISBN 978-1-78913-813-9
23. Chen C, Song T, Mo L, Lv M, Lin D. Autonomous dogfight decision-making for air combat based on reinforcement learning with automatic opponent sampling. *Aerospace* **2025**, *12*, 265. DOI:10.3390/aerospace12030265
24. Zhu J, Kuang M, Zhou W, Shi H, Zhu J, Han X. Mastering air combat game with deep reinforcement learning. *Def. Technol.* **2024**, *34*, 295–312. DOI:10.1016/j.dt.2023.08.019