Autonomous Drone Combat: A Mutli-Agent Reinforcement Learning Approach

Hansen Julien
University of Liège
julien.hansen@student.uliege.be

Louette Arthur
University of Liège
Arthur.Louette@uliege.be

Leroy Pascal
University of Liège
pleroy@uliege.be

Ernst Damien
University of Liège
dernst@uliege.be

Abstract

Drones have become essential tool in various industries, from agriculture to surveillance and are now increasingly deployed on battlefields for detection, recognition, identification, and combat. While most systems remain controlled by human, the shift toward autonomy is intensifying, driven by breakthroughs in artificial intelligence, notably in reinforcement learning and scalable simulation techniques. This paper presents two contributions. A multi agent reinforcement learning environment for drone combat built on IsaacLab. An in-depth comparison between decentralized learning and self-play scheme in competitive settings. Our work confirmed the benefits of self-play methods for autonomous drone combat.

1 Introduction

Drones have become an important asset in various applications [1, 2]. In recent years, their numbers have drastically increased in combat zones, as they offer a low-cost solution to detect and identify strategic targets such as armed vehicles or critical infrastructures like power plants [3]. Additionally, they can easily be converted into lethal, low-cost weapons, representing a threat for these targets [4]. The conflict in Ukraine highlights the growing deployment of weaponized versions of commercial drones [5–7]. Their small size and speed impose constraints and complexity on designing countermeasures that require minimal resources and time. Multiple ways of dealing with these threats are currently considered [8, 9], a distinction in these countermeasures can be made between physical and electronic systems. Physical defenses include measures such as air defense systems, lasers, net guns, and interceptor drones [10]. Various combinations were explored, such as, integrating anti-aircraft guns with radar and laser systems [11], or deploying drones to pursue and neutralize other drones [12]. However, these approaches face key challenges: drones are highly agile, making them difficult to target, and the cost of defense systems, particularly missiles, often exceeds the value of the drones they aim to destroy [13]. In contrast, electronic systems exploit tactics such as radio jamming, eavesdropping, and information injection [14]. In this work, we study the control of an interceptor drone to intercept and neutralize adversarial drones actively. Interceptor drones offer multiple advantages. Their speed and rapid movements make them a highly adaptable countermeasure capable of operating on various terrains. They are cost-efficient, can be deployed from commercially available drones [7, 15], and do not rely on high-cost, military-grade weapons. Finally, the majority of their components can be fabricated through additive manufacturing techniques such as 3D printing [16], enabling fast assembly.

A well-established framework of artificial intelligence for control is reinforcement learning (RL), where agents learn to make sequences of decisions by interacting with an environment. It has shown

great capabilities in robotics applications, like the ability to control first-person view (FPV) drones better than the best pilots [17], or the ability to learn human actions like walking or running [18]. These examples consider a single agent training in the environment and, when there are several, reinforcement learning extends to multi-agent reinforcement learning (MARL). MARL achieves human-level performance under challenging tasks such as competing with some of the best human players in StarCraft II [19] and beating Chess Grandmasters [20]. It also demonstrates the capacity to solve complex drone control tasks such as flying in a formation pattern [21] or exploring an unknown environment through drone swarms [22]. There is no doubt that MARL could reshape the battlefield landscape by enabling autonomous systems to coordinate, adapt, and make complex decisions collectively. This could allow swarms of drones to respond to threats dynamically, and carry out missions with reduced human intervention. However, there is still a gap before deploying autonomous agents in a real battlefield [23]. These previous examples typically make strong hypotheses both on the drones and the environments that should be relaxed to be closer to real combat situation.

This paper analyzes and explores the inherent challenges of MARL, such as the continual co-adaptation of multiple agents as they learn from interactions with one another, known as the moving target problem [24, 25]. This non-stationarity induced by learning agents can lead to cyclic dynamics [26], whereby a learning agent adapts to the changing policies of other agents, which in turn adapt their policies to the learning agent's policy, and so on, creating possible infinite cycles in their strategies. Unlike single-agent settings where an optimal policy is defined with respect to a stationary environment, in MARL, an agent's policy is inherently linked to the policies of all other agents [27, 28]. Thus, defining and achieving optimal behavior in multi-agent environments requires different criteria from the single-agent optimality concepts. This interdependence between agents requires additional theoretical frameworks, largely drawn from game theory. Solution concepts in multi-agent settings often revolve around achieving some form of equilibrium, such as the widely studied Nash equilibrium [29]. Other concepts like correlated equilibria or evolutionarily stable strategies also offer valuable frameworks for analyzing multi-agent learning outcomes [30, 31]. Furthermore, the dynamic and often uncertain nature of multi-agent interactions underscores the importance of developing robust policies, i.e., policies that can maintain performance despite variations in opponents' strategies [32, 33].

To address the presented challenges, self-play methods are central in MARL by confronting an agent with its own weaknesses. Traditionally, this means an agent competes directly against copies of its current policy: in doing so, it discovers and exploits weaknesses in its play. There are several examples of games where human-level performance has been achieved with this particular scheme, such as Stratego, Go and chess [20, 34, 35]. Self-play also shown benefits for autonomous driving [36]. In our approach, an asymmetric self-play scheme is applied only the defender which periodically faces off against frozen, past checkpoints of the attacker during training.

Nowadays, high-fidelity physics simulators have a significant impact on RL research for environment creation and agent training. Training in a simulator is often divided into two steps: data collection and policy updates, where a policy refers to the agent's decision-making function that maps observations to actions. Nowadays policies are neural networks trained on GPUs, a high-fidelity simulator that also runs on GPU can significantly improve the speed of training. For this reason, the IsaacLab framework [37] and the IsaacSim simulator [38] were selected to design a new adversarial environment involving two agents. This new environment models a combat scenario where one drone attempts to reach a target, while the other tries to prevent it by colliding with it. Crazyflie drones [39] were selected for this task. Their open-source nature simplifies the acquisition of necessary data, such as their mass or thrust-to-weight ratio.

This paper is organized as follows. In Section 2, related works on autonomous drone control are reviewed. Section 3 covers the theoretical background. In Section 4, our approach to the problem is presented. Section 5 discusses the results, and finally, Section 6 concludes the paper and outlines potential future work.

2 Related Work

Game theory provides a framework for modeling strategic interactions among rational agents and for formally classifying different categories of games. In cooperative games, all agents optimize a shared reward function and pursue a common objective. In contrast, competitive or zero-sum games

are characterized by strictly opposing interests, where the gain of one agent is exactly the loss of another. More generally, general-sum games represent settings in which the outcomes of one player are not necessarily related to those of others. Important contributions in Game theory, such as Nash's equilibrium [29], have laid the groundwork for analyzing the strategic outcomes and stability of policies learned by agents in competitive and cooperative environments. Other solution concepts exist, such as maxmin / minmax strategies [40], iterated elimination of dominated strategies [41], and correlated equilibria [42], that have further enriched our understanding of multi-agent dynamics. Game theory and RL have shaped the current field of MARL [26].

Generally, MARL methods fall between a fully centralized [43] or fully decentralized paradigms [44]. In the centralized paradigm, a joint policy is learned with global state information. Decentralized methods, on the other hand, train each agent independently based on local observations and individual rewards. While decentralized learning can handle general-sum games, it often suffers from instability due to the non-stationarity of the environment, even in simple settings [45]. An emerging compromise is the framework of centralized training and decentralized execution (CTDE) [25], in which, during training, agents exploit global information about the environment to mitigate the non-stationary challenges caused by simultaneous learning, but at execution, each agent selects actions based only on its own local observations. Recent works showed that CTDE methods consistently achieve state-of-the-art results on standard cooperative multi-agent reinforcement learning benchmarks [46]. There are two prominent classes of CTDE methods: actor-critic and value decomposition approaches. In actor-critic approaches, each agent learns a decentralized policy using a centralized critic that has access to the joint observation history during training. This centralized critic provides better value estimates than a local critic, thereby improving policy learning [47]. At execution time, the critic is no longer required; each agent acts independently based solely on its local observation history. In contrast, in value decomposition methods a centralized learner trains decentralized value functions by factorizing the joint action-value function into individual agent-specific value functions. These individual function enables decentralized execution [48].

CTDE strategies have proven highly effective in a range of cooperative drone applications requiring tight inter-agent coordination. In urban surveillance, for example, such frameworks enable drones swarms to collaboratively navigate and efficiently monitor target areas. Huang et al. [49] introduce a Multi-Agent Critic-Actor learning scheme, where a centralized critic maximize the discounted global rewards considering both safety and energy efficiency and an actor per drone to find decentralized policies to avoid collisions. In the field of mobile edge computing, MARL models trained with centralized knowledge have been applied to optimize drone trajectories and resource allocation. Park et al. [50] propose a method allowing drones to act as mobile base stations, to improve service quality in mobile access networks. Precision agriculture also benefits from these strategies, Sahu et al. [51] study a multi-agent approach to field coverage, where drones dynamically adapt to environmental changes to ensure complete monitoring while minimizing redundancy.

While competitive MARL scenarios for drone control have received less attention than cooperative settings, they present unique challenges, as agents must learn to counter adversaries with opposing objectives. To deal with these challenges, recent research combined hierarchical decompositions and self-play under a CTDE paradigm. Many approaches relies on hierarchical framework separating decision-making into macro- and micro-levels. Chai et al. [52] propose a two-tier architecture for air-to-air combat, where an outer strategic planner selects high-level combat objectives and an inner maneuver controller translates those objectives into precise flight commands. This framework relies on self-play during training. Building on this concept, Selmonaj et al. [53] design a hierarchical multi-agent system in which a high-level "commander" issues macro commands to subordinate agents. These agents, in turn, rely on self-play to refine group tactics and adapt to evolving strategies. More recently, Pang et al. [54] introduce a three-tier Leader-Follower Multi-Agent Proximal Policy Optimization scheme for drones combat. In their design, the top tier performs battlefield assessment, the middle tier determines optimal engagement angles, and the bottom tier issues flight commands.

Although most autonomous drone studies adhere to the CTDE paradigm, recent success of Schroeder de Witt et al. [55] on the StarCraft Multi-Agent Challenge, demonstrated that decentralized agents trained with independent proximal policy optimization (IPPO), relying only on local observation during training, can match or exceed the performance of agents trained with centralized critics, which have access to global information, and, surprisingly, with little hyperparameter tuning. Beyond StarCraft, recent work by Batra et al. [56] shows that fully decentralized RL can produce interesting swarm behaviors. They demonstrate advanced flocking behaviors, perform aggressive maneuvers

in tight formations while avoiding collisions with each other, break and re-establish formations to avoid collisions with moving obstacles, and efficiently coordinate in pursuit-evasion tasks. Our work follows this approach by investigating autonomous drone combat based on independent learning and self-play methods.

3 Theoretical Background

The stochastic game (SG), also called a Markov game [44], is at the foundation of MARL. In an SG, a set of agents interact with the environment by observing its state, choosing actions, and receiving rewards over a time sequence.

Definition: A stochastic game is defined by the tuple $(n, \mathcal{S}, \{\mathcal{U}_i\}_{i\in\mathcal{A}}, \mathcal{P}, \{R_i\}_{i\in\mathcal{A}}, \gamma, p, T)$, where $\mathcal{A} = \{1, \dots, n\}$ denotes the set of $n \geq 1$ agents, $\mathcal{S}$ denotes the state space, and $\mathcal{U}_i$ denotes the action space of agent i. Let $\mathcal{U} := \mathcal{U}_1 \times \cdots \times \mathcal{U}_n$, then $\mathcal{P} : \mathcal{S} \times \mathcal{U} \to \Delta(\mathcal{S})$ denotes the transition probability from any state $s \in \mathcal{S}$ to any state $s' \in \mathcal{S}$ for any joint action $\mathbf{u} \in \mathcal{U}$. The reward function $R_i : \mathcal{S} \times \mathcal{U} \times \mathcal{S} \to \mathbb{R}$ determines the immediate reward received by agent i. Finally, $\gamma \in [0, 1)$ is the discount factor, $p \in \Delta(\mathcal{S})$ is the initial state distribution, and $T \in \mathbb{N} \cup \{\infty\}$ is the time horizon.

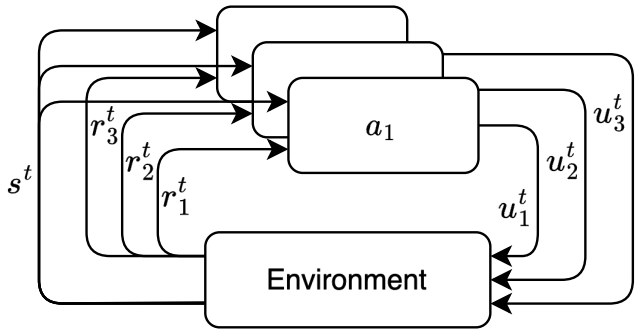

Figure 1: Interaction of three agents with the environment in a stochastic game [57]. Agents $i \in \{1, \dots, 3\}$ have access to the state s^t to select actions u_i^t. As a consequence, they each receive a reward r_i^t and the environment transitions into a new state s^{t+1}.

The SG formalism is the standard for a fully observable environment, which is the case of this work. Within this framework, agent behavior is typically analyzed through game-theory principles. In particular, many solution concepts rely on the notion of best response, where a strategy is optimal for an agent given the fixed strategies of all others, in terms of maximizing its expected cumulative reward. This notion is inherently static [58, 59], as it assumes the strategies of opponents are fixed. A Nash equilibrium is a stable point in which each agent's strategy is a best response to the strategies of the others. The existence of such a solution in any general-sum, non-repeated, normal-form game was first proven in the seminal work of Nash [29]. To define a Nash equilibrium, we first define the value function V_i of agent i, and the joint policy $\pi = (\pi_i, \pi_{-i})$ where $-i$ represents all agents but agent i.

$$V_i^{\pi_i, \pi_{-i}}(s) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t R_i(s^t, \mathbf{u}^t) \,\middle|\, s_0 \sim p(.),\, \mathbf{u}^t \sim \pi(\cdot \mid s^t),\, s^{t+1} \sim \mathcal{P}(\cdot \mid s^t, \mathbf{u}^t)\right]. \tag{1}$$

A Nash equilibrium is a joint policy $\boldsymbol{\pi}^* = (\pi_i^*, \pi_{-i}^*)$ satisfying:

$$V_i^{\pi_i^*, \pi_{-i}^*}(s) \geq V_i^{\pi_i, \pi_{-i}^*}(s), \quad \forall \pi_i \in \Pi \quad \forall s \in \mathcal{S} \quad \forall i \in \mathcal{A} \tag{2}$$

Intuitively, the Nash equilibrium represents a joint policy where no agent can improve its expected cumulative reward by changing their own policy while others keep theirs fixed [29]. In a Nash equilibrium, each agent's strategy is a best response to the joint strategy of all other agents [60]. Even though Nash equilibria are guaranteed to exist, verifying their presence is intractable in practice for high-dimensional environments [61]. Classical reinforcement learning algorithms, like policy gradient methods, lack theoretical guarantees for convergence to Nash equilibria in multi-agent settings, and may instead converge to suboptimal or unstable strategies [62]. Especially in general-sum or partially

observable environments, where agents have limited access to the state of the environment. This has motivated the self-play paradigms in recent research [63].

Once training is complete, the performance of agents in a multi-agent setting can be assessed through several metrics. A direct and widely adopted approach, especially in single-agent settings, is to evaluate the expected cumulative reward, as formalized in the value function $V_i^{\pi_i,\pi_{-i}}(s)$, defined in Equation 1. This measures the expected return for agent i, assuming it follows policy π_i while the other agents follow π_{-i}. Reward-based evaluations are straightforward but often fail to capture strategic nuance in competitive settings. Indeed, high returns may simply reflect exploitation of weak opponents rather than genuinely robust play. Consequently, game-theory metrics are frequently required in adversarial domains. One such metric is the Elo system originally developed for chess, which assigns each player a score within a population [64]. Let R_A and R_B be the Elo ratings of agents A and B, respectively. The estimated probability of winning for agent A against agent B, and vice versa, can be estimated as follows:

$$E_A = \frac{1}{1 + 10^{(R_B - R_A)/400}}, \quad E_B = \frac{1}{1 + 10^{(R_A - R_B)/400}} \tag{3}$$

Note that $E_A + E_B = 1$. The constant 400 determines the steepness of the logistic function, for instance, if agent A has a rating 400 points higher than agent B, A is expected to win with a probability ten times greater than that of B. After each episode, the Elo ratings are updated according to the formula:

$$R_i' = R_i + K(S_i - E_i) \tag{4}$$

where R_i is the current Elo rating of agent i, R_i' the updated rating, E_i the expected score, $S_i \in \{0, 0.5, 1\}$ the actual result (loss, draw, or win), and K a constant controlling the update magnitude.

Another commonly used performance metric is the win rate, which measures the proportion of matches an agent wins against others in the population. This metric offers valuable insights into an agent's relative strengths and weaknesses, particularly when evaluated against specific subgroups within the population. Unlike Elo ratings, which provide a general measure of performance, the win rate can highlight whether an agent performs especially well or poorly against certain subsets of opponents.

4 Experimental Setup

In this section, the components of the SG of our experiments are defined in addition to the training procedure.

State Space $\mathcal{S}$

The state space $\mathcal{S}$ consists of all possible states of the environment. In our drone combat scenario, the state space $\mathcal{S}$ is defined as:

$$\mathcal{S} = \{\mathbf{p_d^w}, \mathbf{q_d^w}, \mathbf{p_a^w}, \mathbf{q_a^w}, \mathbf{p_t^w}, \mathbf{v_d^w}, \boldsymbol{\omega_d^w}, \mathbf{v_a^w}, \boldsymbol{\omega_a^w}, \mathbf{g_d^b}, \mathbf{g_a^b}\} \in \mathbb{R}^{35} \tag{5}$$

In which $\mathbf{p_d^w}$, $\mathbf{p_a^w}$, $\mathbf{p_t^w}$ represent respectively the position vectors in $\mathbb{R}^3$ of the defender drone, the attacker and the target in the world frame. Vectors $\mathbf{q_d^w}$, $\mathbf{q_a^w}$ represent the quaternions in $\mathbb{R}^4$ encoding the orientation of the defender and attacker drones with respect to the world frame.The vectors $\mathbf{v_d^w}$, $\mathbf{v_a^w} \in \mathbb{R}^3$ and $\boldsymbol{\omega_d^w}$, $\boldsymbol{\omega_a^w} \in \mathbb{R}^3$ both represent the linear velocity and angular velocity vectors of each drone in their base frame. Finally, $\mathbf{g_a^b}$, $\mathbf{g_d^b} \in \mathbb{R}^3$ represents the projected gravity vector in the base frame of each drone, which provide orientation of each drone.

Joint Action Space

A joint action $\mathbf{u}$ is:

$$\mathbf{u} = (\mathbf{u_1}, \mathbf{u_2}) \in \mathcal{U}_1 \times \mathcal{U}_2 \tag{6}$$

The action space for each agent consists of a vector $\mathbf{u_i} = (u_{i,0}, u_{i,1}, u_{i,2}, u_{i,3})$, where:

- $u_{i,0}$ represents the thrust, normalized between -1 and 1.
- $u_{i,1}$, $u_{i,2}$, and $u_{i,3}$ represent the moments applied around the roll, pitch, and yaw axes, also normalized between -1 and 1.

The thrust is scaled according to the drone's weight and a predefined thrust-to-weight ratio. The ratio for the Crazyflie model is approximately 1.9 [39]. Each agent controls a drone through applied forces and torques see Figure 2.

$$\text{thrust}_i = \text{thrust\_to\_weight} \times \text{robot\_weight} \times (\frac{u_{i,0} + 1}{2}) \tag{7}$$

This shift ensures that the thrust ranges from zero to the maximum achievable thrust. The moments are adjusted based on a moment scaling factor, moment_scale:

$$\text{moment}_i = \text{moment\_scale} \times (u_{i,1}, u_{i,2}, u_{i,3}) \tag{8}$$

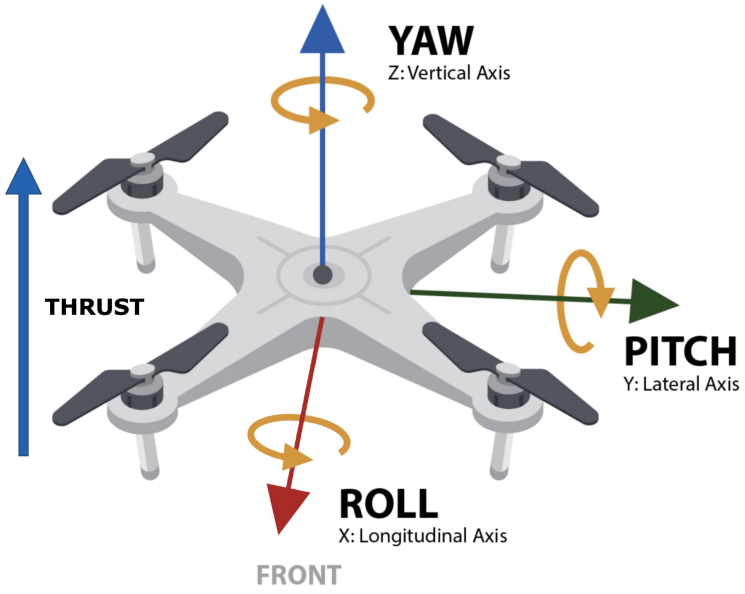

Figure 2: Representation of the action spaces

Reward Functions R_i

Each agent i receives a reward according to a tailored reward function R_i, implying that our environment is a general-sum game. The reward r_1 received by the defender drone is:

$$r_1 = r_{\text{distance}} - r_{\text{lin\_vel}} - r_{\text{ang\_vel}} + r_{\text{win}} - r_{\text{pen}} - r_{\text{alt\_pen}} \tag{9}$$

The terms of the reward function include a distance-based component encouraging the defender to get closer to the attacker, this term is computed as follow:

$$r_{\text{distance}} = 1 - \tanh(\frac{||p_d^w, p_a^w||}{c_1}) \tag{10}$$

where $||\,.||$ is the euclidean distance $\in \mathbb{R}^3$ between the attacker and the target, and where c_1 is a hyperparameter depending on the size of the environment.

$r_{\text{lin\_vel}}$ and $r_{\text{ang\_vel}}$ represent linear and angular velocity penalties, their values are quite small ranging from -0.01 to -1. These terms motivate the agent to learn a more human-like flight. Without these terms, drones start to develop very turbulent flight. r_{win} represents a large positive reward (+300) for successful interception of the attack drone. However, if the attacker reaches the target point, a penalty r_{pen} (-300) is assigned to the defender. Finally, an altitude penalty $r_{\text{alt\_pen}}$ (-10) is added to simulate the crash of a drone. When it happens, the other drone gains a positive reward as it can be considered a win.

Similarly, the reward r_2 received by the attacker drone according to the reward function R_2 is calculated as:

$$r_2 = r_{\text{distance}} - r_{\text{lin\_vel}} - r_{\text{ang\_vel}} + r_{\text{win}} - r_{\text{penalty}} - r_{\text{alt\_pen}} \tag{11}$$

For the attacker drone, the distance component encourages proximity to the goal. This term is computed as :

$$r_{\text{distance}} = 1 - \tanh\left(\frac{||p_d^w, p_a^w||}{c_2}\right) \tag{12}$$

where c_2 is another hyperparameter depending on the distance to the target.

5 Methods

In this paper, we compare two approaches: Independent proximal policy optimization (IPPO) and an modified version of IPPO, Self-play based IPPO (S-IPPO).

IPPO

IPPO was implemented via the SKRL library [65] following the algorithm details presented by Schroeder et al. [66]. IPPO is an extension of PPO to multi-agent settings. PPO is an algorithm derived from Trust Region Policy Optimization (TRPO) [67], which is a class of policy-gradient methods that restricts the update of a policy to within the trust region of the behavior policy by enforcing a KL divergence constraint on the policy update at each iteration. Initially, TRPO optimizes the following:

$$\max_{\theta} \quad \mathbb{E}_{s_t,u_t}\left[\frac{\pi_\theta(u_t \mid s_t)}{\pi_{\theta_{\text{old}}}(u_t \mid s_t)}\hat{A}(s_t,u_t)\right], \qquad \text{subject to } \mathbb{E}_{s_t,u_t}\left[\text{KL}\left(\pi_{\theta_{\text{old}}}, \pi_\theta\right)\right] \leq \delta \tag{13}$$

where θ_{old} are the parameters of the policy before the update and $\hat{A}(s_t, u_t)$ is an approximation of the advantage function. This formulation is computationally expensive due to the computation of multiple Hessian-vector products when approximating the KL constraint. To solve this, PPO[68] approximates the trust region constraints by policy ratio clippings, i.e. the policy loss becomes:

$$\mathcal{L}(\theta) = \mathbb{E}_{s_t,u_t}\left[\min\left(\frac{\pi_\theta(u_t \mid s_t)}{\pi_{\theta_{\text{old}}}(u_t \mid s_t)}\hat{A}(s_t,u_t), \text{clip}\left(\frac{\pi_\theta(u_t \mid s_t)}{\pi_{\theta_{\text{old}}}(u_t \mid s_t)}, 1-\epsilon, 1+\epsilon\right)\hat{A}(s_t,u_t)\right)\right] \tag{14}$$

In IPPO every agents learn a decentralized policies π^i with individual policy clipping, where each agent's independent policy updates are clipped based on the objective defined in Equation 17. A variant of the advantage function, where each agent i learns a local observation based critic $V_\phi(o_t^i)$ parameterised by ϕ is considered, using *Generalized Advantage Estimation* [69]. The network parameters θ, ϕ are not shared across critics and actors in this implementation. An entropy regularization term is also added to the final policy loss [70]. For each agent i, the advantage estimation is computed as:

$$\hat{A}_t^i = \sum_{l=0}^{\infty}(\gamma\lambda)^l\delta_{t+l}^i, \quad \text{where} \quad \delta_t^i = r_t^i + \gamma V_{\phi_{i,\text{old}}}(s_{t+1}^i) - V_{\phi_{i,\text{old}}}(s_t^i) \tag{15}$$

where δ_t^i is the temporal difference at time step t. The team reward $r_t(s_t, i_t)$ approximates $r_t(o_t^i, u_t^i)$. Following that, the final policy loss for each agent i becomes:

$$\mathcal{L}^i(\theta) = \mathbb{E}_{o_t^i, u_t^i}\left[\min\left(\frac{\pi_\theta(u_t \mid o_t^i)}{\pi_{\theta_{\text{old}}}(u_t \mid o_t^i)}\hat{A}_t^i, \text{clip}\left(\frac{\pi_\theta(u_t \mid o_t^i)}{\pi_{\theta_{\text{old}}}(u_t \mid o_t^i)}, 1-\epsilon, 1+\epsilon\right)\hat{A}_t^i\right)\right] \tag{16}$$

Value Clipping is applied to restrict the update of the critic function as proposed by [69]:

$$\mathcal{L}^i(\theta) = \mathbb{E}_{o^i}\left[\min\left\{(V_\phi(o_t^i) - \hat{V_t^i})^2, (V_{\phi_{old}}(o_t^i)) + \text{clip}(V_\phi(o_t^i) - V_{\phi_{old}}(o_t^i), -\epsilon, +\epsilon) - \hat{V_t^i})^2\right\}\right] \tag{17}$$

where ϕ_{old} are parameters before the update and $\hat{V}_t^i = A_t^i + V_\phi(o_t^i)$. This restriction of the value function to within the trust region helps avoiding overfitting to the most recent batch of data. For each agent the overall learning loss becomes:

$$\mathcal{L}(\phi, \theta) = \sum_{i=1}^{n} \mathcal{L}^i(\theta) + \lambda_{critic}\mathcal{L}^i(\phi) + \lambda_{entropy}\mathcal{H}(\pi^i) \tag{18}$$

where $\mathcal{H}(\pi^i)$ is the entropy of policy π^i and λ_{critic} and $\lambda_{entropy}$ are hyperparameter set to 1.0 and 0.001 respectively.

Learning Architecture: We use orthogonal initialization with a gain of $\sqrt{2}$ to initialize the parameters of both the policy and value networks, a strategy known to perform well with ELU activations. The input of each network consists of a flatten state vector, which is passed through two fully connected layers with 256 and 128 hidden units respectively, each followed by ELU activations [71]. The policy network outputs the mean of a Gaussian distribution over actions, with a state-independent log standard deviation parameter. Advantage normalization is applied once before training by subtracting the mean and dividing by the standard deviation over the full rollout buffer. A KL-adaptive learning rate scheduler with a threshold of 0.008 is used. The PPO-specific hyperparameters include a clipping ratio of 0.2 and discount factor λ= 0.99.

S-IPPO

A new Self-play variant of IPPO was also implemented, this new approach aims to improve the defender robustness by modifying the training opponent selection process. The core modification involves maintaining a fixed population of M distinct and frozen attacker policies.

Training is divided in chunks. During training, the defender policy $\pi_{\theta_{\text{def}}}$ primarily collects experience by interacting with the current version of the opponent policy. However, with some probability p, at the end of a chunk, the defender instead interacts with a past version of the attacker policy sampled randomly from previously saved opponent checkpoints. The defender's policy $\pi_{\theta_{\text{def}}}$ is then updated following the standard IPPO algorithm detailed previously (Equations 15-18), exploiting the experience gathered from these interactions. The idea behind this training scheme is to help the defender develop more robust and generalized behaviors.

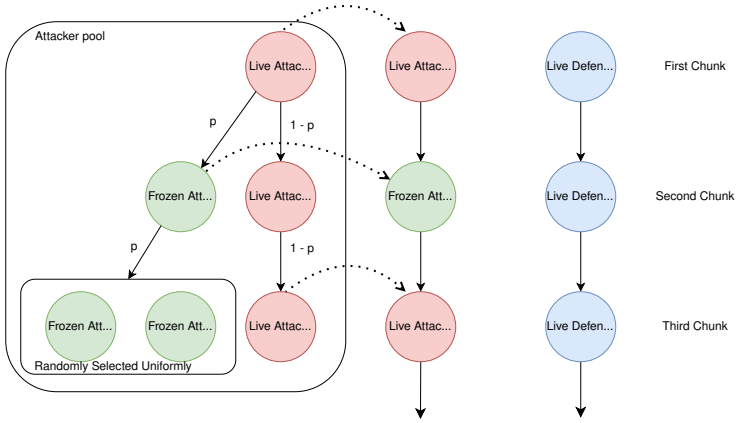

Figure 3: Overview of S-IPPO.

To test both methods we trained agents during 1 billion timesteps with 1000 environments running in parallel, checkpoints for both the attackers and the defenders were taken every chunks, the probability p of switching from the live version of the attacker to a past frozen version of it is 0,2.

6 Results

As stated in Section 5, during training, a checkpoint of each agent is saved every chunk, each chunk is composed of 10 million timesteps, resulting in a population of 100 distinct versions of attackers and defenders. To evaluate their respective Elo scores, each attacker–defender pair was matched 5 times, yielding a total of 50,000 simulated combats. The matches are randomly ordered to ensure statistical fairness and to avoid bias introduced by the ordering of evaluations. To make sure the elo score converge each pair play 100 episodes against each other. This methodology was performed eight times and the results were averaged across these runs.

Figure 4 and 5 presents the Elo score evolution of both teams trained with IPPO. The attackers achieves higher Elo ratings than the defender. This discrepancy may be partially attributed to the unfairness of the tasks: the attacker's target remains fixed, allowing it to better optimize its strategy, while the defender must adapt to the evolving and increasingly competent behavior of the attacker. The defender's Elo slighty decrease at first then remains relatively constant over time, suggesting that earlier defender might learn more general strategies than later one. The constant elo observed after suggests that more trained defender may forget early attacker and are only strong at each chunk against a subset of attacker. However elo scores obtained from S-IPPO are slightly higher, regarding the defender. The final defender saved achieves a mean elo score of $\tilde{1}450$ compared to $\tilde{1}350$ for IPPO, suggesting that more trained defender don't forget early attacker.

Figure 4: IPPO Elo Score

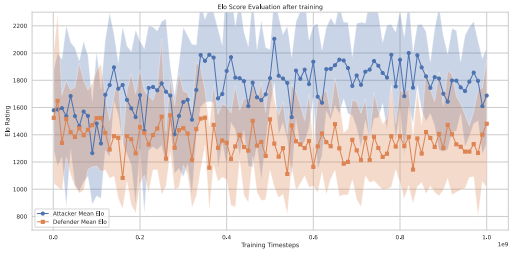

Figure 5: S-IPPO Elo Score

Our suggestion is further reinforced by the win rate analysis presented in Figures 6 and 7, which display the average win rates across all runs between ten defender and every attackers, these heatmaps were clipped between 0,25 and 0,45 to better visualize the average trend present in each methods. Under the IPPO framework, each defender exhibits marginally higher win rates against attacker

policies generated around the same stage of training. This trend indicates a form of temporal overfitting, where defenders adapt primarily to the strategies of their current attackers, but fail to generalize effectively to opponents trained significantly earlier or later. Notably, a marked decline in defender performance becomes evident beyond approximately 600 million timesteps, suggesting a degradation in the ability to counter earlier attack strategies over time.

In contrast, the S-IPPO approach demonstrates higher win rates against earlier attacker checkpoints. This outcome highlights the benefits of training defenders against a diverse set of opponent policies drawn from different stages of training, thereby improving robustness and mitigating the forgetting effect.

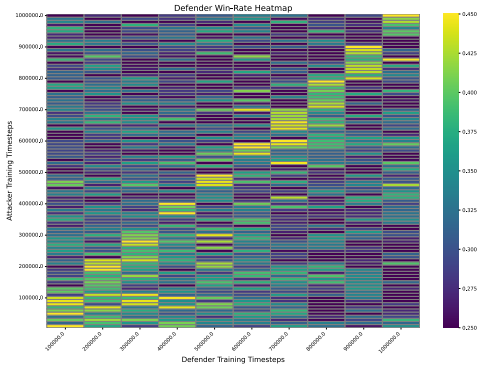

Figure 6: Win rate of IPPO.

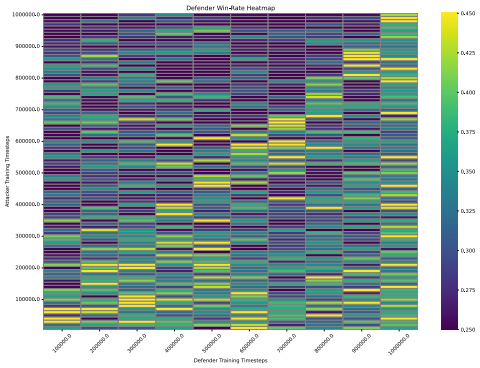

Figure 7: Win rate of S-IPPO.

Regarding the variance in performance, S-IPPO displays a higher variance in win rates compared to IPPO. This observation suggests that the outcome of training under S-IPPO is more sensitive to the rate and diversity of opponent exposure during training, potentially due to the broader behavioral spectrum introduced by replaying past attacker checkpoints. This could also come from inherent stochasticity of RL. Two runs can output really different results leading to an high variance.

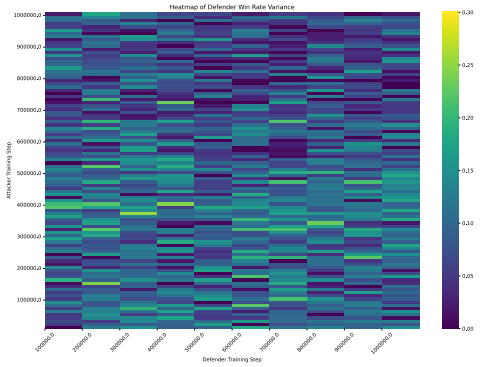

Figure 8: Variance heatmap for IPPO defenders against attacker checkpoints.

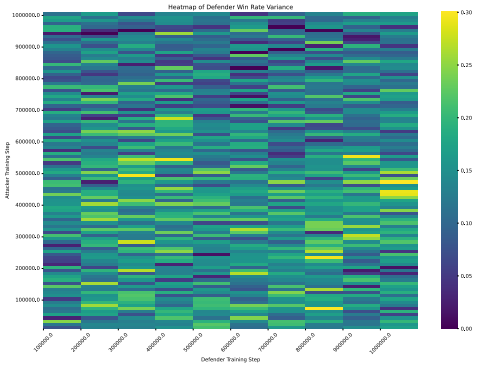

Figure 9: Variance heatmap for S-IPPO defenders against attacker checkpoints.

7 Conclusion and future works

In this paper, we develop and evaluate multi-agent reinforcement learning approaches for autonomous drone combat. We introduce a competitive MARL environment built on IsaacLab, featuring a combat scenario where an attacker drone attempts to reach a target while a defender drone aims to intercept it. Agents control Crazyflie drones through thrust and torques. Our primary investigation compare a fully decentralized learning paradigm, IPPO, against an asymmetric self-play scheme, S-IPPO, in which the defender trains against a population of attacker policies. This approach specifically addresses the moving target problem inherent in competitive multi-agent settings. We evaluate the performance

of these approaches at the end of training using the Elo rating system, and we analyze the win rates of several matchups during training to support the results provided by the Elo scores. Our results highlight the benefits of training a defender against past version of the attacker, demonstrating that S-IPPO improves the development of more robust defenders.

This work constitutes a first investigation of two-team drone competitive tasks using fully decentralized methods, and we propose several future research directions. First, we suggest performing the same experiments in more complex environments with more than two agents and under partially observable settings. Given their impressive results in cooperative tasks, a comparison between CTDE methods and fully decentralized approaches may yield interesting insights. Another research direction consists in comparing a symmetric self-play scheme to S-IPPO. We also recommend conducting behavioral and policy analyses to better understand why some teams achieve higher Elo scores and how strategy diversity emerges within a single training population.

References

1 Pajares, G. (2017). "Unmanned aerial systems for civil applications: A review". *Drones*, *1*(1), 2. https://doi.org/10.3390/drones1010002

2 Fan, B., Li, Y., Zhang, R., & Fu, Q. (2020). "Review on the technological development and application of UAV systems". *Chinese Journal of Electronics*, *29*(2), 199–207. https://doi.org/10.1049/cje.2019.12.006

3 DSIAC. (2018). "Unmanned aerial systems for intelligence, surveillance, reconnaissance" [Defense Systems Information Analysis Center (DSIAC)].

4 Chamola, V., Kotesh, P., Agarwal, A., Naren, Gupta, N., & Guizani, M. (2021). "A comprehensive review of unmanned aerial vehicle attacks and neutralization techniques". *Ad Hoc Networks*, *111*, 102324. https://doi.org/10.1016/j.adhoc.2020.102324

5 Kallenborn, Z. (2022). "Will the drone war come home? ukraine and the weaponization of commercial drones" [Modern War Institute at West Point].

6 Kallenborn, Z. (2023). "Ukraine is the first "hackers' war"". *IEEE Spectrum*. https://spectrum.ieee.org/ukraine-hackers-war

7 Al-Garni, A. D. (2022). "Drones in the ukrainian war: Will they be an effective weapon in future wars?" [Unpublished manuscript].

8 Sharaf, M., Abdelaziz, A., & Al-Shaer, E. (2020). "Protect your sky: A survey of counter unmanned aerial vehicle systems". *IEEE Communications Surveys Tutorials*, *22*(4), 2837–2884. https://doi.org/10.1109/COMST.2020.3016902

9 Tyurin, V., Martyniuk, O., Mirnenko, V., Open'ko, P., & Korenivska, I. (2019). "General approach to counter unmanned aerial vehicles". *2019 IEEE 5th International Conference Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD)*. https://doi.org/10.1109/APUAVD47061.2019.8943859

10 Oliver Parken, T. R. (2024). "Chinese soldiers train to fend off fpv drones".

11 Kunertova, D. (2023). "The war in ukraine shows the game-changing effect of drones depends on the game". *Bulletin of the Atomic Scientists*.

12 Karadeniz, G., Ozcan, A., Bayram, M., & Ince, G. (2024). "Drone wars 3d: A game-based simulation platform for testing aerial defence strategies against drone swarms". *Journal of Aeronautics and Space Technologies*.

13 Silva, D. L. D., Machado, R., Coutinho, O. L., & Antreich, F. (2023). "A soft-kill reinforcement learning counter unmanned aerial system (c-uas) with accelerated training". *IEEE Access*.

14 He, D., Chan, S., & Guizani, M. (2017). "Communication security of unmanned aerial vehicles". *IEEE Wireless Communications*.

15 Modovolo. (2025). "The ukrainian drone order of battle: Lessons for commercial drone operators".

16 Muralidharan, N., Pratheep, V., Shanmugam, A., Hariram, A., Dinesh, P., & Visnu, B. (2021). "Structural analysis of mini drone developed using 3d printing technique". *Heliyon*, *7*(12), e08519. https://doi.org/10.1016/j.heliyon.2021.e08519

17 Song, Y., Steinweg, M., Kaufmann, E., & Scaramuzza, D. (2021). "Autonomous drone racing with deep reinforcement learning".

18 Song, S., Kidziński, Ł., Peng, X. B., Ong, C., Hicks, J., Levine, S., Atkeson, C. G., & Delp, S. L. (2021). "Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation". *Journal of NeuroEngineering and Rehabilitation*, *18*(1).

19 Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Sasha, A., Vezhnevets, M., Yeo, A., Makhzani, H., Kuttler, J., Agapiou, J., Schrittwieser, J., Quan, S., Gaffney, S., Petersen, K., Simonyan, T., Schaul, H., Silver, D., & Lillicrap, T. (2017). "StarCraft II: A new challenge for reinforcement learning".

20 Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., & Hassabis, D. (2017). "Mastering chess and shogi by self-play with a general reinforcement learning algorithm".

21 Xie, Y., Yu, C., Zang, H., Gao, F., Tang, W., Huang, J., Chen, J., Xu, B., Wu, Y., & Wang, Y. (2024). "Multi-UAV formation control with static and dynamic obstacle avoidance via reinforcement learning".

22 Liao, G., Wang, J., Yang, D., & Yang, J. (2024). "Multi-UAV escape target search: A multi-agent reinforcement learning method". *Sensors*, *24*(21), 6859. https://doi.org/10.3390/s24216859

23 et al, A. L. (2025). "Existing gaps in reinforcement learning for drone warfare" (Technical Report) (Accessed: 2025-06-03). University of Liège. https://orbi.uliege.be/bitstream/2268/331713/1/Existing_Gaps_In_Reinforcement_Learning_For_Drone_Warfare.pdf

24 Tan, M. (1993). "Multi-agent reinforcement learning: Independent vs. cooperative agents". *Proceedings of the tenth international conference on machine learning*, 330–337.

25 Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Abbeel, P., & Mordatch, I. (2017). "Multi-agent actor-critic for mixed cooperative-competitive environments". *Advances in neural information processing systems (NIPS)*, *30*, 6379–6390. https://proceedings.neurips.cc/paper/2017/hash/68a9750337a418a86fe06c1991a1d64c-Abstract.html

26 Albrecht, S. V., Christianos, F., & Schäfer, L. (2024). "Multi-agent reinforcement learning: Foundations and modern approaches". MIT Press. https://www.marl-book.com/download/marl-book.pdf

27 Zhang, K., Yang, Z., & Başar, T. (2021). "Multi-agent reinforcement learning: A selective overview of theories and algorithms". In K. G. Vamvoudakis, F. L. Lewis, D. V. Dimarogonas, & Y. Lu (Eds.), *Handbook of reinforcement learning and control* (pp. 321–384). Springer. https://doi.org/10.1007/978-3-030-60990-0_12

28 Buşoniu, L., Babuşka, R., & Schutter, B. D. (2008). "A comprehensive survey of multi-agent reinforcement learning". *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *38*(2), 156–172. https://doi.org/10.1109/TSMCC.2007.913919

29 Nash, J. F., Jr. (1950). "Equilibrium points in n-person games". *Proceedings of the National Academy of Sciences of the United States of America*, *36*(1), 48–49. https://doi.org/10.1073/pnas.36.1.48

30 Shoham, Y., & Leyton-Brown, K. (2008). "Multiagent systems: Algorithmic, game-theoretic, and logical foundations". Cambridge University Press. https://www.cambridge.org/core/books/multiagent-systems/B11B69E0CB9032D6EC0A254F59922360

31 Nowak, M. A. (2006). "Evolutionary dynamics: Exploring the equations of life". Harvard University Press. https://www.hup.harvard.edu/books/9780674023383

32 Pinto, L., Davidson, J., Sukthankar, R., & Gupta, A. (2017, August). "Robust adversarial reinforcement learning". In D. Precup & Y. W. Teh (Eds.), *Proceedings of the 34th international conference on machine learning* (pp. 2817–2826, Vol. 70). PMLR. https://proceedings.mlr.press/v70/pinto17a.html

33 Zhang, K., Sun, T., Tao, Y., Genc, S., Mallya, S., & Başar, T. (2020). "Robust multi-agent reinforcement learning with model uncertainty". *Advances in Neural Information Processing Systems*, *33*, 1–12. https://proceedings.neurips.cc/paper/2020/hash/774412967f19ea61d448977ad9749078-Abstract.html

34 Perolat, J., de Vylder, B., Hennes, D., Tarassov, E., Strub, F., de Boer, V., Muller, P., Connor, J. T., Burch, N., Anthony, T., McAleer, S., Elie, R., Cen, S. H., Wang, Z., Gruslys, A., Malysheva, A., Khan, M., Ozair, S., Timbers, F., ... Tuyls, K. (2022). "Mastering the game of stratego with model-free multiagent reinforcement learning" [arXiv preprint arXiv:2206.15378]. https://arxiv.org/abs/2206.15378

35 Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., & Hassabis, D. (2016). "Mastering the game of go with deep neural networks and tree search". *Nature*, *529*, 484–489. https://doi.org/10.1038/nature16961

36 Cornelisse, D., & Vinitsky, E. (2024). "Human-compatible driving partners through data-regularized self-play reinforcement learning" [arXiv preprint arXiv:2403.19648]. https://arxiv.org/abs/2403.19648

37 Mittal, M., Yu, C., Yu, Q., Liu, J., Rudin, N., Hoeller, D., Yuan, J. L., Singh, R., Guo, Y., Mazhar, H., Mandlekar, A., Babich, B., State, G., Hutter, M., & Garg, A. (2023). "Orbit: A unified simulation framework for interactive robot learning environments" [arXiv preprint arXiv:2301.04195]. https://arxiv.org/abs/2301.04195

38 NVIDIA. (2023). "Isaac sim: Robotics simulation and synthetic data generation" [Accessed: 2025-05-13].

39 Bitcraze AB. (2023). "Crazyflie 2.1 open source quadcopter drone" [Accessed: 2025-05-13].

40 von Neumann, J. (1928). "Zur theorie der gesellschaftsspiele". *Mathematische Annalen*, *100*(1), 295–320. https://doi.org/10.1007/BF01448847

41 Fudenberg, D., & Tirole, J. (1991). "Game theory". MIT Press.

42 Aumann, R. J. (1974). "Subjectivity and correlation in randomized strategies". *Journal of Mathematical Economics*, *1*(1), 67–96. https://doi.org/10.1016/0304-4068(74)90037-8

43 Claus, C., & Boutilier, C. (1998). "The dynamics of reinforcement learning in cooperative multiagent systems". *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, 746–752. https://aaai.org/papers/00746-AAAI98-106-the-dynamics-of-reinforcement-learning-in-cooperative-multiagent-systems/

44 Littman, M. L. (1994). "Markov games as a framework for multi-agent reinforcement learning". *Proceedings of the Eleventh International Conference on Machine Learning*, 157–163. https://doi.org/10.1016/B978-1-55860-335-6.50027-1

45 Foerster, J. N., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., & Mordatch, I. (2017). "Learning with opponent-learning awareness". *arXiv preprint arXiv:1709.04326*. https://arxiv.org/abs/1709.04326

46 Foerster, J. N., Farquhar, G., Nardelli, N., Torr, P. H., & Whiteson, S. (2018). "Counterfactual multi-agent policy gradients". *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, *31*, 2859–2867. https://arxiv.org/abs/1705.08926

47 Lyu, X., Baisero, A., Xiao, Y., Daley, B., & Amato, C. (2023). "On centralized critics in multi-agent reinforcement learning". *Journal of Artificial Intelligence Research*, *77*, 295–354. https://doi.org/10.1613/jair.1.14386

48 Rashid, T., Samvelyan, M., Foerster, J. N., Zambaldi, V., Tuyls, K., Degris, T., Hessel, M., Silver, D., Vinyals, O., & Lanctot, M. (2018). "Qmix: Monotonic value function factorization for deep multi-agent reinforcement learning". *Proceedings of the 35th International Conference on Machine Learning (ICML)*, *80*, 4295–4304. https://arxiv.org/abs/1803.11485

49 Huang, S., Zhang, H., & Huang, Z. (2022). "Multi-uav collision avoidance using multi-agent reinforcement learning with counterfactual credit assignment". *arXiv preprint arXiv:2204.08594*. https://arxiv.org/abs/2204.08594

50 Park, C., Park, S., Jung, S., Cordeiro, C., & Kim, J. (2022). "Cooperative multi-agent deep reinforcement learning for reliable and energy-efficient mobile access via multi-uav control". *arXiv preprint arXiv:2210.00945*. https://arxiv.org/abs/2210.00945

51 Sahu, R., Sahu, P., Sahu, S., & Sahu, S. (2023). "An analysis of the robustness of uav agriculture field coverage using multi-agent reinforcement learning". *Innovations in Systems and Software Engineering*, *19*(1), 1–12. https://doi.org/10.1007/s41870-023-01264-0

52 Chai, J., Chen, W., Zhu, Y., Yao, Z.-x., & Zhao, D. (2022). "A hierarchical deep reinforcement learning framework for 6-dof ucav air-to-air combat". *arXiv preprint arXiv:2212.03830*. https://arxiv.org/abs/2212.03830

53 Selmonaj, A., Zhang, Y., Zhang, X., Yang, Z., Zhang, Y., Li, Y., & Zhang, X. (2023). "Hierarchical multi-agent reinforcement learning for air combat maneuvering". *arXiv preprint arXiv:2309.11247*. https://arxiv.org/abs/2309.11247

54 Pang, J., He, J., Abdelaal, N. M., Mohamed, A., Lin, C., Zhang, Z., & Hao, X. (2025). "A hierarchical reinforcement learning framework for multi-uav combat using leader-follower strategy". *arXiv preprint arXiv:2501.13132*. https://arxiv.org/abs/2501.13132

55 Schroeder de Witt, C., Gupta, T., Makoviichuk, D., Makoviychuk, V., Torr, P. H. S., Sun, M., & Whiteson, S. (2020). "Is independent learning all you need in the starcraft multi-agent challenge?" *arXiv preprint arXiv:2011.09533*. https://doi.org/10.48550/arXiv.2011.09533

56 Batra, S., Huang, Z., Petrenko, A., Kumar, T., Molchanov, A., & Sukhatme, G. S. (2021). "Decentralized control of quadrotor swarms with end-to-end deep reinforcement learning". *5th Conference on Robot Learning, CoRL 2021, 8-11 November 2021, London, England, UK*. https://arxiv.org/abs/2109.07735

57 Pascal, L. (2024, July). "Contributions to multi-agent reinforcement learning" [PhD thesis]. University of Liege.

58 Weinberg, M., & Rosenschein, J. S. (2004). "Best-response multiagent learning in non-stationary environments". *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, 506–513.

59 Hernandez-Leal, P., Kartal, B., & Taylor, M. E. (2017). "A survey of learning in multiagent environments: Dealing with non-stationarity". *Journal of Artificial Intelligence Research*, *65*, 1–50.

60 Osborne, M. J., & Rubinstein, A. (1994). "A course in game theory". MIT Press.

61 Daskalakis, C., Goldberg, P. W., & Papadimitriou, C. H. (2009). "The complexity of computing a nash equilibrium". *Journal of the ACM*, *56*(3), Art. 14, 57 pgs. https://doi.org/10.1145/1536414.1536420

62 Mazumdar, E., Ratliff, L. J., Jordan, M. I., & Sastry, S. S. (2019). "Policy-gradient algorithms have no guarantees of convergence in linear quadratic games". *arXiv preprint arXiv:1907.03712*. https://arxiv.org/abs/1907.03712

63 Xu, Z., Yu, C., Liang, Y., Wu, Y., & Wang, Y. (2025). "Learning global nash equilibrium in team competitive games with generalized fictitious cross-play". *Journal of Machine Learning Research*, *26*(44), 1–30. https://jmlr.org/papers/v26/24-1503.html

64 Elo, A. E. (1978). "The rating of chessplayers, past and present". Arco Publishing. https://archive.org/details/ratingofchesspla00unse

65 Serrano-Muñoz, A., Chrysostomou, D., Bøgh, S., & Arana-Arexolaleiba, N. (2023). "Skrl: Modular and flexible library for reinforcement learning". *Journal of Machine Learning Research*, *24*(254), 1–9. http://jmlr.org/papers/v24/23-0112.html

66 Schroeder de Witt, C., Gupta, T., Makoviichuk, D., Makoviychuk, V., Torr, P. H., Sun, M., & Whiteson, S. (2020). "Is independent learning all you need in the starcraft multi-agent challenge?" *arXiv preprint arXiv:2011.09533*. https://arxiv.org/abs/2011.09533

67 Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., & Moritz, P. (2015). "Trust region policy optimization". *Proceedings of the 32nd International Conference on Machine Learning*, 1889–1897. https://proceedings.mlr.press/v37/schulman15.html

68 Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). "Proximal policy optimization algorithms". *arXiv preprint arXiv:1707.06347*. https://arxiv.org/abs/1707.06347

69 Schulman, J., Moritz, P., Levine, S., Jordan, M. I., & Abbeel, P. (2016). "High-dimensional continuous control using generalized advantage estimation". *Proceedings of the 4th International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1506.02438

70 Mnih, V., Puigdomènech Badia, A., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). "Asynchronous methods for deep reinforcement learning". *Proceedings of the 33rd International Conference on Machine Learning*, 1928–1937. https://proceedings.mlr.press/v48/mniha16.html

71 Clevert, D.-A., Unterthiner, T., & Hochreiter, S. (2016). "Fast and accurate deep network learning by exponential linear units (elus)". *Proceedings of the 4th International Conference on Learning Representations (ICLR)*.