POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Joint Aircraft Design and Tactical Learning via Reinforcement Learning for Air Combat

Tesi di Laurea Magistrale in
Computer Science and Engineering - Ingegneria Informatica

Author: **Marco Grazi**

Student ID: 231849
Advisor: Prof. Andrea Bonarini
Co-advisors: Fabio Valerio Ferrari
Academic Year: 2025-26

Abstract

The design of effective close-range air combat tactics depends on a tight coupling between aircraft dynamic capabilities, control architectures, and decision-making strategies. This thesis presents an integrated simulation and reinforcement learning framework for the joint analysis and development of aircraft concepts and tactical control policies in subsonic air-to-air combat.

A custom six-degree-of-freedom aircraft simulator is developed, combining parametric mass, inertia, propulsion, and aerodynamic models with a hierarchical control stack that separates high-frequency stabilization from low-frequency tactical decision making. On top of this simulator, a geometry-aware reinforcement learning environment is constructed, featuring invariant observation representations, simplified engagement rules based on weapon and vulnerability zones, and a modular reward structure shaping flight stability, pursuit geometry, closure control, and weapon usage. Soft Actor–Critic is adopted as the learning algorithm and embedded within a curriculum learning and self-play framework.

Experimental results show that the framework reliably produces tactically coherent behaviors, including controlled intercepts, speed management across engagement regimes, and disciplined firing decisions. In one-versus-one self-play, learned policies converge toward interaction patterns closely resembling real-world Basic Fighter Maneuvers. Comparative self-play across aircraft variants indicates that dynamic differences primarily affect stability margins and failure modes rather than inducing fundamentally distinct tactics. Extension to two-versus-two scenarios shows that non-trivial multi-agent interaction patterns can emerge from shared objectives, while also highlighting the limitations of symmetric, non-cooperative training setups.

Overall, this work establishes a flexible engineering tool for studying the interplay between aircraft design choices and reinforcement-learning-driven tactical behavior, supporting future investigations into asymmetric manned–unmanned combat and coordinated multi-agent air combat.

Keywords: reinforcement learning, air combat simulation, aircraft dynamics, self-play, multi-agent systems

Abstract - Italiano

La progettazione di tattiche efficaci per il combattimento aereo a corto raggio richiede una stretta integrazione tra le capacità dinamiche del velivolo, l'architettura di controllo e i meccanismi decisionali. Questa tesi presenta un framework integrato di simulazione e reinforcement learning per l'analisi congiunta e lo sviluppo di concetti aeronautici e politiche tattiche di controllo nel combattimento aria–aria subsonico.

Viene sviluppato un simulatore aeronautico a sei gradi di libertà, basato su modelli parametrici di massa, inerzia, propulsione e aerodinamica, integrato con un'architettura di controllo gerarchica che separa la stabilizzazione ad alta frequenza dalle decisioni tattiche a bassa frequenza. Su questa base è costruito un ambiente di reinforcement learning orientato alla geometria dell'ingaggio, con rappresentazioni osservative invarianti, regole di ingaggio semplificate basate su zone offensive e di vulnerabilità, e una funzione di ricompensa modulare che guida stabilità di volo, geometria di inseguimento, controllo della chiusura e impiego dell'armamento. L'algoritmo Soft Actor–Critic è adottato all'interno di una strategia di curriculum learning e self-play.

I risultati sperimentali mostrano che il framework produce comportamenti tatticamente coerenti, includendo intercetti controllati, gestione della velocità nei diversi regimi di ingaggio e decisioni di fuoco disciplinate. Nel self-play uno-contro-uno, le politiche apprese convergono verso schemi di interazione riconducibili alle Basic Fighter Maneuvers. Il confronto tra varianti di velivolo indica che le differenze dinamiche influenzano soprattutto i margini di stabilità e le modalità di fallimento. L'estensione a scenari due-contro-due mostra che strutture di interazione multi-agente non banali possono emergere da obiettivi condivisi.

Nel complesso, questo lavoro introduce uno strumento ingegneristico flessibile per lo studio dell'interazione tra progettazione del velivolo e comportamenti tattici guidati dal reinforcement learning.

Parole chiave: reinforcement learning, simulazione del combattimento aereo, dinamica del velivolo, self-play, sistemi multi-agente

Contents

1 | Introduction

As modern warfare shifts increasingly toward the use of autonomous systems powered, at many levels, by machine learning algorithms, air combat is expected to follow a similar trajectory. The emphasis placed on the development and integration of artificial intelligence within defense systems is clearly visible in recent government studies, such as the 2025 briefing for the European Parliament titled *Defense and Artificial Intelligence* [4]. This document explicitly highlights the growing role of autonomous platforms, including their potential application to close-range air combat and related operational domains.

Another well-established trend is the focus on low observability and stealth as defining characteristics of new aircraft designs. As discussed in Rebecca Grant's *The Radar Game: Understanding Stealth and Aircraft Survivability* [7], reducing radar and infrared signatures substantially increases survivability and expands the range of feasible mission profiles. Current and near-future airframes do not achieve true invisibility, but they significantly reduce detection ranges against modern sensing systems, altering the geometry and timing of potential engagements.

The convergence of these two technological directions—greater autonomy and enhanced stealth—has contributed to the emergence of the Collaborative Combat Aircraft (CCA) concept, as outlined in recent U.S. Air Force studies submitted to the U.S. Congress [5]. A CCA is envisioned as an advanced UCAV (Unmanned Combat Aerial Vehicle) featuring modern low-observable design principles and AI-driven combat capabilities, able to coordinate with similar autonomous platforms under the supervision of a manned sixth-generation fighter acting as a team leader.

In this context, the present work, developed in collaboration with Leonardo S.p.A., aims to develop a tool for exploring, evolving, and comparing combinations of combat tactics—represented by reinforcement learning decision policies—and parametric aircraft designs, within scenarios where stealthy UCAVs detect each other at short range and are therefore compelled to engage in close-range maneuvering combat. Such a tool could support early-stage concept development for future CCAs and is designed to accommodate engagements involving multiple aircraft per team and more than two opposing teams.

To achieve this objective, a custom 6-degree-of-freedom aircraft dynamics simulator was developed, with emphasis on computational efficiency and the ability to incorporate parametric definitions of different airframe geometries. On top of this simulator, a reinforcement learning environment was created to implement the studied combat scenario, including the definition of observation and action spaces and the construction of a suitable reward function through multiple iterations of reward shaping. Furthermore, a multi-phase training pipeline was designed, consisting of an initial curriculum-learning stage against a simplified opponent—repeated for each aircraft variant—followed by a competitive self-play phase in which the aircraft-agent combinations train against one another to refine their capabilities and ultimately identify the most effective configuration.

The remainder of this thesis follows this logical structure. It begins with an overview of the relevant background concepts, then presents the development of the aircraft dynamics simulator and parametric airframe models. This is followed by a description of the reinforcement learning environment, algorithms, and training pipeline, together with the resulting agent behaviors and performance comparisons. Finally, the contributions, limitations, and possible future evolutions of the developed tool are discussed.

2 | Background and Related Work

2.1. Air Combat Fundamentals

As explained in the introduction, the scenario modeled in this work is a close-range air-to-air engagement, commonly referred to as a dogfight. From a reinforcement learning perspective, this type of scenario is more interesting because it requires frequent and precise maneuvering decisions, whereas at longer ranges engagements tend to be dominated by standardized missile exchanges. In those cases, weapon performance and sensor range become the central factors, turning the encounter into a "range game", where the aircraft equipped with the longer-range missiles simply needs to remain outside the adversary's effective envelope. At close range, instead, sensor and missile limitations place stronger constraints on feasible engagement geometries, and maneuvering skill becomes decisive.

2.1.1. Sensors, Missile Types, and Short-Range Dynamics

Engagement range strongly influences the type of sensors, missiles, and countermeasures involved. At short distances, heat-seeking infrared (IR) missiles become predominant due to their high agility and ability to track thermal signatures. Such missiles typically operate within a defined field of view, may lose effectiveness when the target's engine heat is partially shielded, and can be deceived through countermeasures such as flares. For the purposes of this work, it is sufficient to recall that short-range engagements are primarily conducted with IR-guided missiles and that their use is tightly coupled to aircraft positioning and attitude.

2.1.2. Dogfighting Geometry and Positional Concepts

On the positional and geometric side of dogfighting maneuvers, several concepts are central to the modeling used in later sections of this thesis. The most relevant are the Line of Sight (LOS), the track angle, and the adverse angle. The LOS vector is defined as the vector pointing from the current aircraft to the adversary's position. The track angle is the angle between the aircraft's forward attitude vector and the LOS, while the adverse

angle is the analogous angle computed from the adversary's forward vector to the LOS. From a tactical perspective, optimal geometry corresponds to minimizing the track angle and maximizing the adverse angle, effectively placing the aircraft directly behind the opponent and pointing straight at it. This is also the most favorable position for the seeker head of an infrared missile, which has an unobstructed view of the engine exhaust.

2.1.3. Closure Rate and Overshoot Considerations

Another important quantity for close-range engagements is the closure rate, defined as the relative speed at which the two aircraft approach or separate. Closure rate depends on the speed and heading of both aircraft. The optimal value is not monotonic. When far from the adversary, a positive closure rate is desirable, allowing the attacker to enter sensor and weapon range. Once this is achieved, however, excessive closure becomes dangerous: if the pursuing aircraft overshoots, it risks flying past the target and placing itself directly in front of the adversary—an extremely vulnerable position. Managing closure rate is therefore essential for maintaining positional advantage.

2.1.4. Missile Engagement Feasibility

Although this work does not model missiles as fully simulated entities, it is useful to outline the major factors influencing a missile's ability to intercept a target. These include the initial orientation at launch, the missile's time to intercept (dependent on its acceleration and maximum speed), the seeker's tracking quality or "tone", and the relative maneuverability of the missile and aircraft. In principle, one can estimate the feasible set of positions reachable by both missile and aircraft and determine whether their reachable sets overlap. If they fully overlap, a hit is highly likely. However, this forms a coupled nonlinear dynamic system with interdependent variables such as flight time and maneuver trajectory, which is difficult to solve analytically and expensive to simulate numerically.

Given that simulating missiles as independent agents with full guidance logic would impose a significant computational burden on the environment, this work adopts a simplified probabilistic model based primarily on the missile's initial orientation and seeker tone. This abstraction preserves the essential tactical implications of missile employment while remaining computationally efficient for RL training.

2.2. Reinforcement Learning

Reinforcement Learning (RL) provides a framework for autonomous agents to learn decision-making strategies through interaction with an environment, guided only by sparse evaluative feedback [17]. In contrast to supervised learning, where labeled examples define the desired behavior, RL focuses on estimating long-term consequences of actions and on discovering policies that maximize cumulative reward. This makes RL a natural fit for sequential control problems such as air combat maneuvering, where agents must continuously select actions based on dynamic state information and where optimal behavior emerges only through sustained interaction and exploration.

2.2.1. Policy Gradient Methods

Policy gradient methods directly optimize the parameters of a stochastic policy by estimating the gradient of expected return with respect to those parameters [17]. Rather than deriving a policy from a value function, these methods assume a parametric form $\pi_\theta(a|s)$ and adjust θ in the direction that improves performance. This approach is particularly well suited for continuous action spaces, such as aircraft control inputs, where value-based methods struggle due to the need for action discretization.

In its simplest form, the policy gradient theorem shows that:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(a|s)\, Q^{\pi_\theta}(s,a) \right],$$

meaning that actions that yield higher returns increase their probability under the policy. Because this estimator can suffer from high variance, practical implementations typically incorporate baselines, variance reduction techniques, or critic networks to stabilize learning. Nonetheless, the fundamental idea remains the same: the policy is adjusted incrementally toward behaviors that lead to higher long-term rewards.

2.2.2. Actor–Critic Architectures

Actor–critic architectures combine the strengths of policy gradient methods with value function approximation [17]. In this framework, the *actor* represents the policy and proposes actions according to a parametric distribution, while the *critic* evaluates these actions by estimating a state-value or action-value function. The critic provides a low-variance training signal—typically in the form of an advantage estimate—that guides the actor's policy updates.

This division of roles enables more stable and sample-efficient learning compared to pure policy gradient methods. For continuous control tasks, actor–critic methods are particularly advantageous because they allow smooth policy updates and leverage differentiable function approximators, such as neural networks, without requiring discretization of the action space. Many of the most effective modern RL algorithms, including SAC, PPO, and TD3, follow this actor–critic paradigm.

2.2.3. Soft Actor–Critic Algorithm

Soft Actor–Critic (SAC) is an off-policy actor–critic algorithm based on the maximum entropy reinforcement learning framework. In addition to maximizing expected cumulative reward, SAC also maximizes the entropy of the policy, which encourages exploration and leads to smoother, more robust behaviors. This is especially valuable in environments like air combat, where the agent must continuously adapt its maneuvers and avoid converging prematurely to suboptimal deterministic strategies.

SAC uses a stochastic policy represented by a neural network, two separate Q-function approximators to mitigate overestimation bias, and a temperature parameter that balances exploration and exploitation. The algorithm optimizes the objective:

$$J(\pi) = \sum_t \mathbb{E}_{(s_t,a_t)\sim\pi} \left[r(s_t, a_t) + \alpha \, \mathcal{H}(\pi(\cdot|s_t)) \right],$$

where α controls the weight of the entropy term. The inclusion of entropy regularization has been shown to improve training stability and convergence in complex continuous-control tasks, making SAC a strong candidate for maneuvering and tactical decision-making problems. Its off-policy nature also allows efficient reuse of collected data, which is critical for computationally intensive simulations such as the 6DOF aircraft model used in this work.

2.3. Curriculum Learning, Self-Play and Multi-Agent Reinforcement Learning

In this thesis, agent training is divided into two major phases. The first phase relies on the concept of Curriculum Learning, in which the agent is progressively exposed to increasingly difficult tasks in order to build up the capabilities required for the final close-range air combat scenario. The second phase adopts self-play reinforcement learning strategies, enabling competitive co-evolution of agent policies and providing a robust

evaluation framework for comparing different aircraft–policy combinations. The multi-agent evolution of the aerial combat task is explored with 2 vs 2 encounter scenarios and has similar objectives to the self-play training phase, just extended to look for emergence of cooperative tactics.

2.3.1. Curriculum Learning

Curriculum Learning (CL) is a training paradigm in which learning tasks are structured in a meaningful progression of increasing difficulty, rather than sampled uniformly from the full task distribution. This idea, originally formalized by Bengio et al. [1], draws inspiration from human and animal learning, where simpler skills are mastered before more complex ones are introduced. In reinforcement learning contexts, CL has been shown to accelerate convergence, reduce exploration difficulties, and improve generalization across task variations [12].

The core principle is to design a sequence of intermediate tasks that form a ladder toward the final desired behavior. These tasks may differ in the variety and difficulty of initial states, the skill level of opponents, environmental complexity, or even the size of the effective state and action spaces. By doing so, the agent can quickly acquire basic competencies that would be difficult or impossible to learn if trained directly on the full-complexity scenario.

In this work, the curriculum consists of a progression in the maneuvering capabilities of the dummy adversary and an increasing variety of initial states in terms of position, orientation, distance from the adversary, and speed. This proved crucial for enabling stable learning of the skills needed for the task, from basic flying behavior to pursuit dynamics and management of missile tone and engagement geometry.

2.3.2. Self-Play

Self-play is a reinforcement learning paradigm in which agents learn by competing against versions of themselves or against a dynamically maintained population of policies. This approach was first applied in adversarial games such as chess and Go [16], and later in high-dimensional continuous environments such as Dota 2 [2]. Self-play is particularly effective in domains where the optimal behavior depends strongly on the strategy of the opponent.

The central advantage of self-play in RL is that by training an agent against older versions of itself—or against a co-evolving population—it naturally generates an implicit curricu-

lum. As the agent improves, the difficulty of its opponents increases as well, encouraging the emergence of increasingly complex strategies. Self-play also mitigates overfitting, since a single static adversary cannot be exploited repeatedly, and older policy snapshots may reappear as opponents with non-zero probability. Maintaining a history of past policies is therefore important to prevent the agent from forgetting how to counter simpler strategies while adapting to more advanced ones.

In the context of close-range air combat, self-play enables the co-evolution of tactics between competing agents and supports the comparative evaluation of different aircraft configurations. The details of the self-play implementation used in this work, including the tournament structure and its integration with TrueSkill-based ranking, will be discussed later in this chapter.

2.3.3. Multi-Agent Reinforcement Learning

Multi-Agent Reinforcement Learning (MARL) extends standard RL to environments involving multiple learning agents that interact within the same state space. In such environments, the presence of multiple adaptive policies makes the learning dynamics non-stationary, as the reward distribution and transition dynamics experienced by any given agent evolve as other agents update their policies [8]. This creates challenges for stability, convergence, and exploration not present in single-agent RL.

In adversarial MARL settings such as air combat, agents must simultaneously reason about maneuvering constraints, engagement geometry, and opponent intent. Self-play provides one mechanism to stabilize MARL by generating a diverse population of behaviors against which each agent can train. In this thesis, multi-agent concepts are incorporated into the implementation, following a CTDE (Centralized Training, Decentralized Execution) adversarial approach, similar to the one used in self-play. Observations and results will be discussed in later chapters.

2.3.4. Matchmaking and Skill Estimation (TrueSkill)

In competitive and adversarial reinforcement learning settings, it is often necessary to estimate the relative skill of agents based on the outcomes of repeated interactions. Simple ranking methods, such as win-rate statistics or Elo ratings, are limited in this context, as they assume deterministic outcomes, require large numbers of matches for stability, and do not explicitly model uncertainty in skill estimates. These limitations become particularly problematic in stochastic environments, where match outcomes may vary significantly due to random initial conditions or probabilistic elements of the simulation.

TrueSkill is a Bayesian skill rating system originally developed by Microsoft Research to address these shortcomings [10]. In TrueSkill, each agent's skill is modeled as a probability distribution rather than a single scalar value, typically represented as a Gaussian distribution with a mean corresponding to the estimated skill level and a variance representing the uncertainty of that estimate. Match outcomes are treated as noisy observations of the underlying skill variables, and Bayesian inference is used to update both the estimated skill and its associated uncertainty after each game.

One of the key advantages of TrueSkill over Elo is its natural extension to multi-player and team-based scenarios, allowing individual contributions to be inferred even when outcomes result from collective interactions [10]. This property makes it particularly well suited for reinforcement learning environments involving multiple agents or evolving populations, where direct pairwise comparisons may be insufficient to capture relative performance.

In the context of this thesis, TrueSkill provides a principled framework for ranking competing aircraft–policy combinations during self-play training, supporting both robust evaluation and adaptive opponent selection. The specific way in which TrueSkill is integrated into the training pipeline is described in later chapters.

2.4. Related Work

The development of this thesis was inspired by several works spanning both reinforcement learning applied to continuous control problems and research focused on air combat modeling, aircraft dynamics, and engagement rules. This section reviews the most relevant contributions that influenced the design choices adopted in this work. Additional sources consulted for more specific implementation decisions are referenced in their respective chapters.

2.4.1. RL in Aircraft Control

When investigating the applicability of reinforcement learning to aircraft control, particular attention was given to algorithms capable of handling continuous state and action spaces, as well as highly nonlinear dynamics. In this context, the comparative study presented in *A Comparison of PPO, TD3 and SAC Reinforcement Algorithms for Quadruped Walking Gait Generation* [11] proved useful despite its different application domain. While focused on legged locomotion, the work analyzes policy gradient methods under challenging continuous-control conditions characterized by stability constraints, delayed rewards, and complex system dynamics. These characteristics closely resemble those encountered

in aggressive aircraft maneuvering tasks, making the comparison between PPO, TD3, and SAC informative for algorithm selection in this thesis.

A more directly related contribution is *A Deep Reinforcement Learning-Based Intelligent Maneuvering Strategy for the High-Speed UAV Pursuit-Evasion Game* [19], which applies a TD3-based method to a pursuit–evasion scenario involving high-speed unmanned aerial vehicles. This work addresses several challenges central to the present thesis, including continuous control of aircraft dynamics, adversarial interaction between agents, and the design of reward functions capable of guiding complex maneuvering behavior. Although SAC was ultimately selected as the learning algorithm in this work, the paper provided a valuable reference for understanding the trade-offs involved in training agents for air combat–like scenarios, particularly with respect to stability, convergence speed, and reward shaping.

2.4.2. RL in Aerial Combat

Regarding the definition of the reinforcement learning environment—including observation and action spaces, reward formulation, and engagement rules—the most influential reference was *Hierarchical Reinforcement Learning for Air Combat at DARPA's AlphaDogfight Trials* [14]. This work explores a self-play dogfighting scenario and employs the same SAC algorithm used in this thesis. It provides a clear definition of key geometric quantities such as track angle and adverse angle, along with a thorough explanation of air combat terminology and engagement logic. Particularly relevant was the discussion of reward shaping strategies, several elements of which inspired the reward design adopted in this work.

However, the AlphaDogfight Trials focus primarily on the exploration of different policy archetypes—such as aggressive, conservative, and control-zone behaviors—and consider engagements based exclusively on the aircraft's internal gun. As a result, the rules of engagement and tactical constraints differ in several respects from those considered here, where missile employment and probabilistic hit modeling play a central role.

Another closely related contribution is *A Hierarchical Deep Reinforcement Learning Framework for 6-DOF UCAV Air-to-Air Combat* [3], which investigates air combat scenarios using a hierarchical control architecture. In this work, a high-level policy determines maneuvering objectives, while a lower-level controller executes these objectives through direct control of aerodynamic surfaces. A distinction is made between the decision-making frequency of the two layers, with the higher-level policy operating at a lower frequency suitable for long-term tactical planning and the lower-level controller running at a higher fre-

quency to ensure fast and stable control response. This separation of time scales strongly influenced the control architecture adopted in this thesis, although an additional intermediate abstraction layer was introduced to improve learning stability in the considered scenario.

2.4.3. 6DOF Simulation Frameworks

With respect to aircraft dynamics modeling, initial investigations considered existing simulation libraries designed for reinforcement learning applications. In particular, the *PyFlyt – UAV Simulation Environments for Reinforcement Learning Research* framework [18] was evaluated as a potential foundation. While PyFlyt offers an accessible and RL-oriented simulation environment, it was found to lack the flexibility required to define fully parametric aircraft models and relies on simplified aerodynamic representations. Additionally, its computational performance raised concerns when extended to multi-agent scenarios.

These limitations motivated the development of a custom six-degree-of-freedom aircraft dynamics simulator. For this purpose, the primary reference was the master's thesis *Guidance and Control for a Fixed-Wing UAV* by Farì [6], which provides a detailed derivation of the equations of motion, aerodynamic force modeling, and control strategies for fixed-wing aircraft. This work served as a foundation for the implementation of a computationally efficient and customizable simulation model, later extended to support reinforcement learning and multi-agent training.

3 | Aircraft Dynamics and Simulation Framework

3.1. 6DOF Equations of Motion

The design of a reasonably realistic simulation for a parametric aircraft model was based on the master's thesis cited above [6], which provides a clear and systematic description of the equations of motion, reference frames, and mathematical transformations required for this task. This section therefore focuses on presenting an overview of the main elements of the dynamic modeling, with particular emphasis on the definition of reference frames, which will be essential for understanding the observation space described in later chapters.

3.1.1. Reference Frames and Conventions

The description of the simulated world begins with the *World* reference frame. This inertial frame defines the origin of the three spatial dimensions and is expressed by the coordinate vector

$$\mathbf{p} = (x, y, z).$$

In the implemented environment, the spatial domain is bounded such that the x and y coordinates lie within the interval $[0, \text{max\_size}]$, while the z coordinate lies within $[0, \text{max\_altitude}]$.

From the World frame, the *Vehicle* reference frame is defined through a pure translation. Its origin O' is located at the aircraft position, given by the vector

$$\mathbf{p}_v = (x', y', z') = (p_x, p_y, p_z),$$

expressed in the World frame. Throughout the simulation, the axes of the World and Vehicle frames remain parallel, as no rotation is applied between these two reference frames.

The third reference frame, referred to as the *Body* frame, is obtained by rotating the Vehicle frame according to the aircraft attitude, defined by the Euler angles (ψ, θ, ϕ), corresponding respectively to yaw (rotation about the z-axis), pitch (rotation about the y-axis), and roll (rotation about the x-axis). This frame represents the point of view of an ideal pilot seated in the aircraft. Its axes are denoted as

$$(x'', y'', z''),$$

and allow a natural definition of forward, lateral, and vertical directions relative to the aircraft.

A standard aircraft sign convention is adopted, consistent with the reference work [6]: the positive directions are defined as forward, rightward, and downward. Using the right-hand rule, this corresponds to the index finger pointing forward and the thumb pointing downward, with the middle finger indicating the positive lateral direction.

An additional reference frame used in later parts of this work is the *Wind* frame. This frame can be interpreted as a rotation of the Body frame about the (x'', z'') and (x'', y'') planes by the angles α and β, respectively. These angles are known as the angle of attack (AoA) and sideslip angle (SS), and are defined as the angles between the aircraft's forward axis and the relative wind velocity vector. The term *relative wind* refers to the airflow perceived in the Body frame, which results from the combination of the ambient wind velocity and the wind induced by the aircraft's own motion.

When required, transformations between reference frames are performed using rotation matrices, including the Vehicle-to-Body and Wind-to-Body transformations. The transformation from the Vehicle frame to the Body frame is defined through the composition of three successive rotations parameterized by the Euler angles yaw ψ, pitch θ, and roll ϕ. Following the yaw–pitch–roll convention, the corresponding rotation matrix is expressed as

$$\mathbf{R}_v^b = \mathbf{R}_x(\phi)\,\mathbf{R}_y(\theta)\,\mathbf{R}_z(\psi),$$

where the individual rotation matrices are defined as

$$\mathbf{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}, \quad \mathbf{R}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}, \quad \mathbf{R}_z(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

An additional transformation is defined between the Body frame and the Wind frame, parameterized by the angle of attack α and the sideslip angle β. The rotation from the

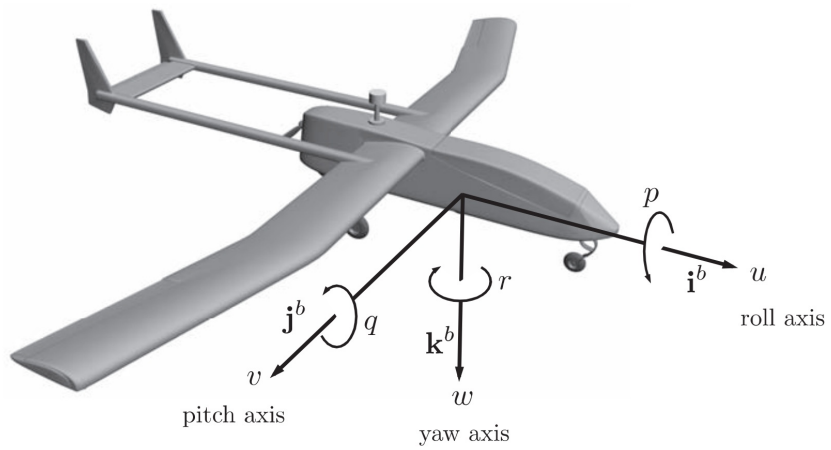

Figure 3.1: Definition of *body* frame axis and rotational positive conventions

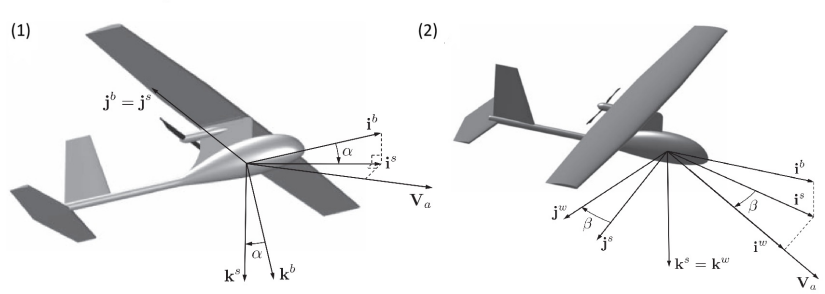

Figure 3.2: Definition of (1): Angle of Attack, (2): Sideslip

Wind frame to the Body frame is given by

$$\mathbf{R}_w^b = \mathbf{R}_y(\alpha)\,\mathbf{R}_z(\beta),$$

with

$$\mathbf{R}_y(\alpha) = \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{bmatrix}, \quad \mathbf{R}_z(\beta) = \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

These rotation matrices are used throughout the simulation to transform vectors between reference frames, including velocity, force, and moment vectors, following the conventions described in [6].

The standard right-hand convention is also applied to rotational directions, with the thumb aligned with the axis of rotation and the fingers indicating the positive direction of rotation. A schematic representation of the reference frames and angles introduced in this section is shown in Figure 3.1 and figure 3.2.

3.1.2. Translational Dynamics

The translational dynamics of the aircraft describe the evolution of its linear motion in space and are expressed through the time integration of acceleration, velocity, and position. In the implemented model, linear acceleration is computed in the *Body* frame, integrated to obtain the velocity vector, and then rotated into the *Vehicle* frame before being integrated to update the aircraft position.

Formally, the translational equations of motion are given by

$$\dot{\mathbf{v}}_b = \frac{1}{m}\mathbf{F}_b - \boldsymbol{\omega}_b \times \mathbf{v}_b,$$

$$\dot{\mathbf{p}} = \mathbf{R}_b^v \, \mathbf{v}_b,$$

where $\mathbf{v}_b$ is the velocity vector expressed in the Body frame, $\mathbf{p}$ is the position vector expressed in the Vehicle (inertial) frame, m is the aircraft mass, $\boldsymbol{\omega}_b$ is the angular velocity vector in the Body frame, and $\mathbf{R}_b^v$ is the rotation matrix from Body to Vehicle frame.

The total force vector $\mathbf{F}_b$ is computed in the Body frame as the sum of all forces acting on the aircraft:

$$\mathbf{F}_b = \mathbf{F}_{\text{aero}} + \mathbf{F}_{\text{thrust}} + \mathbf{F}_{\text{weight}},$$

with each term rotated into the Body frame if originally computed in another reference frame.

The aerodynamic force $\mathbf{F}_{\text{aero}}$ is decomposed into lift, drag, and lateral components, and further expressed as the combined contribution of the airframe and control surfaces, including elevators, rudders, ailerons, and the aerodynamic brake. This decomposition enables the definition of parametric aerodynamic models for each component, which is a key aspect of the aircraft model flexibility discussed in later sections.

3.1.3. Rotational Dynamics

The rotational dynamics govern the evolution of the aircraft attitude and angular motion. Analogously to the translational case, angular acceleration is integrated to obtain angular velocity, which is then integrated to update the attitude. In this formulation, linear quantities are replaced by their rotational counterparts: mass is replaced by the inertia tensor, forces by moments, and linear velocity by angular velocity.

The rotational equations of motion expressed in the Body frame are

$$\mathbf{I}\,\dot{\boldsymbol{\omega}}_b = \mathbf{M}_b - \boldsymbol{\omega}_b \times (\mathbf{I}\boldsymbol{\omega}_b),$$

where $\boldsymbol{\omega}_b$ is the angular velocity vector, $\mathbf{I}$ is the inertia matrix expressed in the Body frame, and $\mathbf{M}_b$ is the total moment vector acting on the aircraft.

In this work, the inertia matrix is simplified by retaining only its diagonal terms, assuming negligible products of inertia. The moment vector $\mathbf{M}_b$ is computed from the aerodynamic forces acting at their respective centers of force, using the vector cross product between the lever arm and the applied force. The weight force produces no moment, while thrust is assumed to act through the center of mass and therefore does not generate a moment. This represents a modeling simplification adopted to reduce complexity and computational cost.

The centers of force associated with each aerodynamic component are configurable parameters of the aircraft model, allowing different airframe designs to be represented consistently within the same simulation framework.

In summary, the complete six-degree-of-freedom rigid-body dynamics model implemented in this work can be written as

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{R}_b^v\,\mathbf{v}_b \\ \dot{\mathbf{v}}_b = \frac{1}{m}\mathbf{F}_b - \boldsymbol{\omega}_b \times \mathbf{v}_b \\ \dot{\boldsymbol{\omega}}_b = \mathbf{I}^{-1}\left(\mathbf{M}_b - \boldsymbol{\omega}_b \times (\mathbf{I}\boldsymbol{\omega}_b)\right) \end{cases}$$

The attitude is updated through numerical integration of the angular velocity, using the Euler angle representation defined in the previous section. Together, these equations describe the coupled translational and rotational motion of the aircraft under the action of aerodynamic, propulsive, and gravitational forces.

3.1.4. State Variables and Implemented System

In order to define the aircraft dynamics in a compact and systematic way, the six-degree-of-freedom model is expressed in terms of a state vector composed of translational, rotational, and attitude variables. This state representation is also used as the basis for defining the observation space of the reinforcement learning environment in later chapters.

The inertial position of the aircraft is expressed in the *Vehicle* (inertial) frame using a

North–East–Down (NED) convention:

$$\mathbf{p} = \begin{bmatrix} p_n & p_e & p_d \end{bmatrix}^T,$$

where p_n, p_e, and p_d denote the inertial north, east, and down positions, respectively.

The linear velocity is expressed in the *Body* frame as

$$\mathbf{v}_b = \begin{bmatrix} u & v & w \end{bmatrix}^T,$$

where u, v, and w represent the forward, lateral, and vertical velocity components.

The aircraft attitude is represented using Euler angles

$$\boldsymbol{\eta} = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T,$$

corresponding to roll, pitch, and yaw, defined with respect to the *Vehicle* frame following the conventions introduced earlier.

Finally, the angular velocity is expressed in the Body frame as

$$\boldsymbol{\omega}_b = \begin{bmatrix} p & q & r \end{bmatrix}^T,$$

where p, q, and r denote the roll, pitch, and yaw rates.

Collecting all terms, the complete aircraft state vector is defined as

$$\mathbf{x} = \begin{bmatrix} p_n & p_e & p_d & u & v & w & \phi & \theta & \psi & p \\ q & r & & & & & & & & \end{bmatrix}^T.$$

The time evolution of the inertial position is obtained by rotating the Body-frame velocity into the Vehicle frame:

$$\begin{bmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{p}_d \end{bmatrix} = \mathbf{R}_b^v \begin{bmatrix} u \\ v \\ w \end{bmatrix},$$

where $\mathbf{R}_b^v$ is the rotation matrix defined by the current Euler angles.

The translational dynamics in the Body frame are given by

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{bmatrix} + \frac{1}{m} \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix},$$

where f_x, f_y, and f_z are the total external forces acting on the aircraft expressed in the Body frame.

The kinematic relationship between Euler angle rates and angular velocities is expressed as

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}.$$

Finally, the rotational dynamics are governed by

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \mathbf{I}^{-1} \left(\begin{bmatrix} L \\ M \\ N \end{bmatrix} - \begin{bmatrix} (I_z - I_y)qr \\ (I_x - I_z)pr \\ (I_y - I_x)pq \end{bmatrix} \right),$$

where L, M, and N are the aerodynamic moments expressed in the Body frame and $\mathbf{I} = \mathrm{diag}(I_x, I_y, I_z)$ is the simplified diagonal inertia matrix.

This formulation defines the complete nonlinear state-space model used in the simulator and provides a direct mapping between the physical aircraft dynamics and the state variables exposed to the reinforcement learning agent.

3.2. Aircraft Model Variants Definition

In order to fully exploit the flexibility offered by the custom parametric aircraft model described in the previous chapter, a complete specification of aerodynamic, inertial, and propulsive properties was required. These include aerodynamic force models as functions of angle of attack and sideslip, mass and inertia matrices, and thrust characteristics. To demonstrate both the freedom of configuration of the proposed simulation framework and the methodology required to obtain such parameters, this work includes the design of multiple aircraft variants and the use of Computational Fluid Dynamics (CFD) simulations to inform their aerodynamic models.

Two baseline aircraft designs were developed and subsequently extended into additional

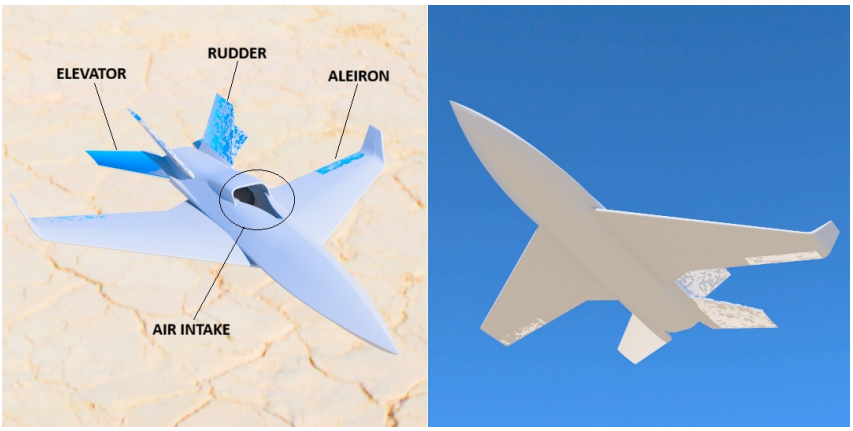

Figure 3.3: Render of 1st airframe variant $UCAV_0$

variants, allowing the reinforcement learning agents to be trained and evaluated on platforms with different dynamic characteristics.

3.2.1. Airframe and Dynamic Properties Design

The design of the two initial aircraft variants was inspired by publicly available concepts related to Collaborative Combat Aircraft (CCA) programs, which emphasize autonomy, agility, and operation alongside manned sixth-generation fighters [5]. While no specific platform was replicated, the overall design philosophy follows common trends observed in these concepts.

The first aircraft variant adopts a relatively classical configuration, characterized by a clear distinction between fuselage, wings, and control surfaces. Conventional elevators, ailerons, and rudders are used, and the airframe geometry is optimized for straightforward control authority and stability. A rendering of this configuration is shown in Figure 3.3.

The second aircraft variant explores a more unconventional layout, featuring delta wings combined with canards, which serve a role analogous to elevators but are positioned ahead of the main wing. In this design, the fuselage is also shaped to contribute to lift generation, resulting in a blended wing–body configuration. This layout, illustrated in Figure 3.4, is expected to exhibit different aerodynamic and control characteristics, particularly at high angles of attack.

To further increase the diversity of dynamic behavior and explore how reinforcement learning agents adapt to different performance envelopes, the two aircraft variants were also differentiated by propulsion configuration. The first design employs a single-engine setup, while the second uses a twin-engine configuration. This choice significantly affects the thrust-to-weight ratio and overall mass distribution, leading to different maneuvering

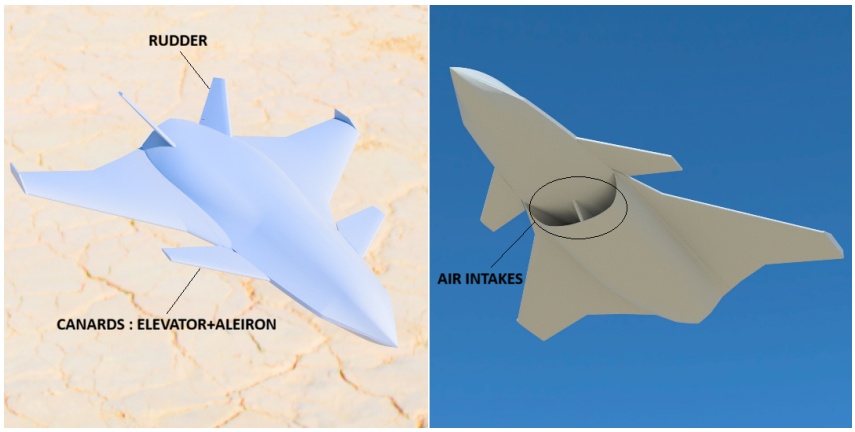

Figure 3.4: Render of 2nd airframe variant $UCAV_1$

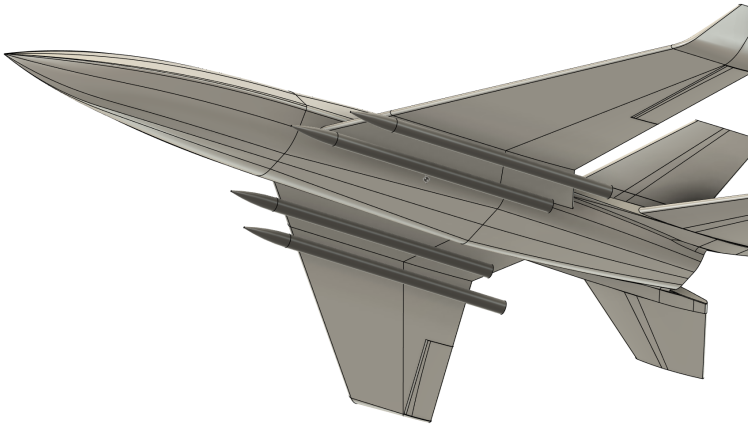

Figure 3.5: Scale comparison between *UCAV-0* and *Meteor Missile*

capabilities and energy management strategies.

Thrust levels were estimated by referencing real-world engines of comparable size and intended application. In particular, the EJ200 turbofan engine, developed for the Eurofighter Typhoon, was used as a benchmark for thrust magnitude and scaling considerations [15]. While the modeled aircraft are smaller and unmanned, the EJ200 provides a realistic upper bound for high-performance military propulsion systems.

The overall size of the aircraft was chosen to be on the order of 10 meters for both length and wingspan. This decision was guided by practical considerations related to payload capacity, specifically the ability to carry multiple missiles. As a reference, the dimensions of the MBDA Meteor missile were considered when defining internal and external payload volumes [9]. A schematic comparison between the aircraft geometry and the reference missile size is shown in Figure 3.5.

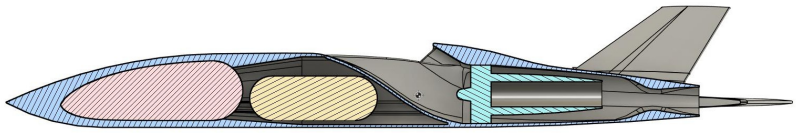

Figure 3.6: Section of *UCAV-0* airframe showing electronics, fueltank and engine volumes. Center Of Mass is also shown

After completing the geometric design, appropriate materials were assigned to each component, and reference volumes were defined for fuel tanks, avionics, and propulsion systems as shown in Figure 3.6.

Using these specifications, the total mass, center of mass location, and rotational inertia matrix were extracted directly from the CAD models. These quantities constitute a fundamental part of the aircraft dynamic model and directly influence both translational and rotational behavior during simulation.

3.2.2. CFD Simulation Setup

Computational Fluid Dynamics (CFD) simulations were conducted to characterize the aerodynamic behavior of the designed aircraft variants and to obtain quantitative estimates of lift and drag over a range of angles of attack and sideslip angles. The simulations were performed using the default steady-state settings provided by the selected CFD tool, as the objective was not to capture transient or unsteady flow phenomena, but rather to extract static aerodynamic coefficients suitable for use in a real-time simulation environment.

All simulations were carried out under fixed environmental conditions, with constant air density and airspeed. In particular, lift and drag forces were measured for a set of discrete angles of attack and sideslip angles at a fixed velocity. These measurements were later used to derive interpolated models of the coefficient of lift (C_L) and coefficient of drag (C_D), expressed as functions of the aerodynamic angles.

The aerodynamic forces were related to their corresponding coefficients through the stan-

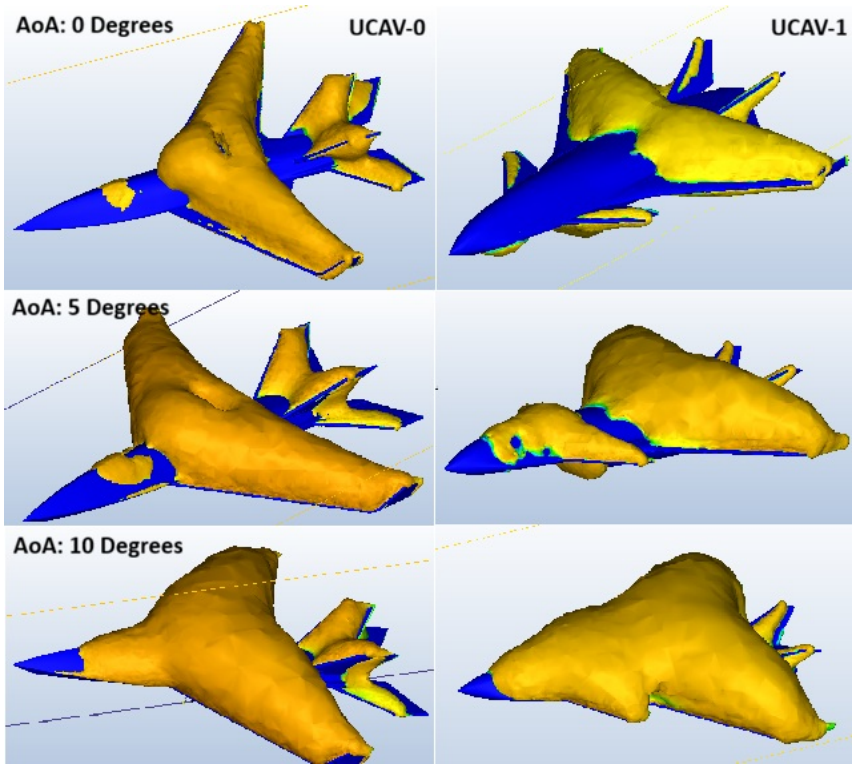

Figure 3.7: Simulation images for both designs at 0, 5 and 10 Degrees of AoA

dard aerodynamic formulation

$$F = \frac{1}{2}\,\rho\, S\, v^2\, C,$$

where F denotes the aerodynamic force (lift or drag), ρ is the air density, S is the reference surface area, v is the airspeed, and C is the corresponding aerodynamic coefficient. Inverting this relation allows the computation of the coefficients from the CFD-derived forces:

$$C = \frac{2F}{\rho\, S\, v^2}.$$

In this work, a constant air density of $\rho = 1.239$ kg/m^3 was assumed, corresponding to standard atmospheric conditions at low altitude. The reference airspeed used in the simulations was $v = 300$ km/h, which corresponds to approximately 83.3 m/s. By keeping both density and airspeed constant, variations in the measured forces could be directly attributed to changes in angle of attack and sideslip.

Figure 3.7 shows representative CFD visualizations for both aircraft designs at different angles of attack.

Regions of lower pressure are visible on the upper surfaces of the airframes, indicating suction effects that generate lift. The distribution and intensity of these regions vary with

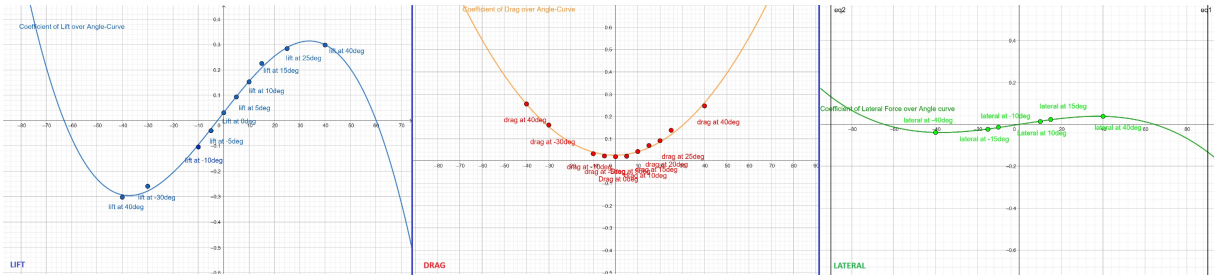

Figure 3.8: CL, CD and CY polynomial curve fitting from simulation data points

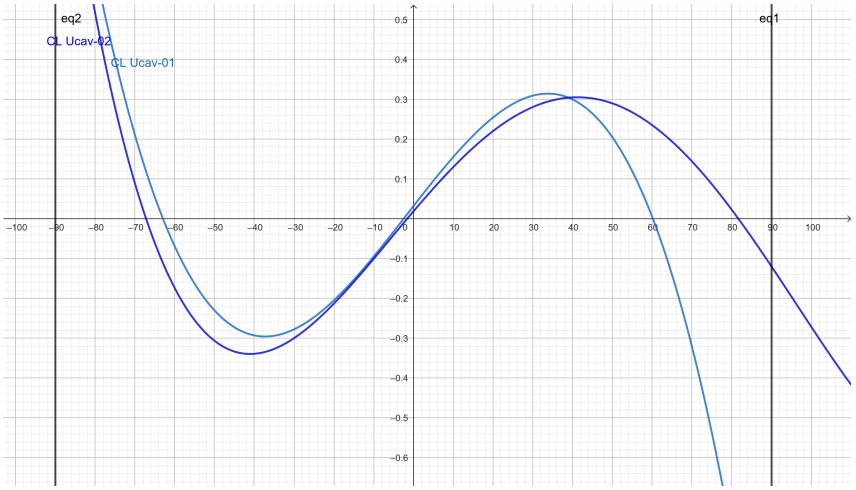

Figure 3.9: Comparison between CL for *UCAV-0* (*Ucav-01* in the image) and *UCAV-1* (*Ucav-02* in the image)

angle of attack and differ between the two designs, reflecting their distinct aerodynamic characteristics.

The discrete lift and drag coefficient samples obtained from the CFD simulations were then used to fit interpolated polynomial regression curves. Figure 3.8 illustrates the resulting interpolation for both C_L and C_D, showing good agreement between the fitted curves and the original data points.

The coefficients of these interpolating functions are directly integrated into the parametric aircraft configuration and constitute the core of the aerodynamic force model used in the simulator.

A comparison between the aerodynamic characteristics of the two aircraft variants is presented in Figure 3.9, where the differences in lift and drag curves highlight the impact of the chosen geometric configurations on overall performance and maneuverability.

Several simplifying assumptions were adopted in this approach and are important to explicitly acknowledge. First, stall dynamics were not modeled explicitly. While the

interpolated lift curve exhibits a plateau and eventual inversion at angles of attack of approximately 40°, in reality aerodynamic stall typically manifests in a more critical and asymmetric manner at significantly lower angles, often around 20°. In such conditions, flow separation may occur unevenly across the wings, leading to unpredictable loss of lift and potentially uncontrollable roll moments. Modeling these phenomena accurately would require unsteady aerodynamic simulations and significantly more complex control logic. For this reason, maximum allowable angles of attack and minimum airspeeds were introduced as termination conditions in the environment, as discussed in later chapters.

A second simplification concerns control surface effectiveness. In the implemented model, aerodynamic control surfaces are assumed to maintain full authority at all times. In real flight conditions, however, control surfaces may experience partial or total loss of effectiveness due to flow separation or wake interference from the airframe. These effects were neglected in favor of a more tractable and computationally efficient model.

Finally, the flight regime considered in this work is restricted to subsonic conditions. A maximum allowable airspeed of 343 m/s, corresponding approximately to the speed of sound at standard atmospheric conditions, was imposed. Beyond this threshold, aerodynamic behavior changes significantly due to compressibility effects and shock wave formation, requiring the adoption of fundamentally different aerodynamic models. As such, supersonic flight dynamics were considered outside the scope of this work.

3.3. Chapter Synthesis

This chapter presented the modeling approach adopted for the aircraft dynamics and aerodynamic properties used throughout this work. A six-degree-of-freedom rigid-body formulation was introduced, together with a clear definition of reference frames, state variables, and the numerical integration scheme employed to simulate aircraft motion in three-dimensional space.

Two parametric aircraft variants were designed and characterized in terms of geometry, mass properties, propulsion, and aerodynamic behavior. Static CFD simulations were used to inform lift and drag coefficient models as functions of angle of attack and sideslip, providing a physically grounded yet computationally efficient representation of aerodynamic forces. The resulting coefficients, together with simplified assumptions on control surface effectiveness and subsonic flight conditions, define the operating envelope of the simulator.

Several modeling simplifications were deliberately introduced, including steady-state aero-

dynamics, the absence of explicit stall dynamics, and the restriction to subsonic regimes. These choices were motivated by the need to balance physical realism with computational tractability, especially in the context of large-scale reinforcement learning training and self-play experiments. Importantly, the adopted abstractions preserve the key couplings between aircraft geometry, dynamic response, and control authority that are central to close-range air combat maneuvers.

The resulting aircraft model provides a flexible and extensible foundation upon which reinforcement learning agents can be trained and evaluated. By exposing a consistent state representation and parametrically defined dynamics, the simulator enables the study of how different aircraft designs interact with learned control and decision-making policies. The next chapter builds on this foundation by introducing the reinforcement learning environment, control architecture, and interaction mechanisms through which agents operate within the simulated air combat scenario.

4 | Control Architecture

A description of the structure and design choices underlying the control stack used in this work is provided in this chapter. The discussion proceeds through each layer connecting the policy output to its effects in the physical simulation environment. Particular attention is given to the separation between low-frequency decision-making signals and high-frequency control loops, as well as to the role of a geometric action translation layer, which was found to significantly improve both training stability and final policy performance.

4.1. Overview of the Control Stack

The overall structure of the control stack is illustrated in Figure 4.1.

The reinforcement learning policy receives observations from the environment, which will be described in detail in later chapters, and produces high-level tactical commands that define the desired behavior of the controlled aircraft.

Specifically, the policy outputs a polar velocity vector expressed in the *Body* frame, parameterized by the tuple

$$(\mathit{UpAngle},\ \mathit{SideAngle},\ \mathit{Speed}),$$

using the same sign conventions introduced for the six-degree-of-freedom dynamic model. In addition to the velocity command, the policy outputs a binary *Fire* command, which is used to trigger missile launch decisions.

These outputs are passed to an intermediate *Action Translation* layer which performs a geometric transformation of the policy commands into a set of control-relevant target quantities:

$$(\alpha,\ \beta,\ \phi,\ V,\ \mathit{Fire}),$$

corresponding respectively to angle of attack, sideslip angle, roll angle, airspeed, and fire command.

At this stage, a change in control frequency is introduced. While the policy and action

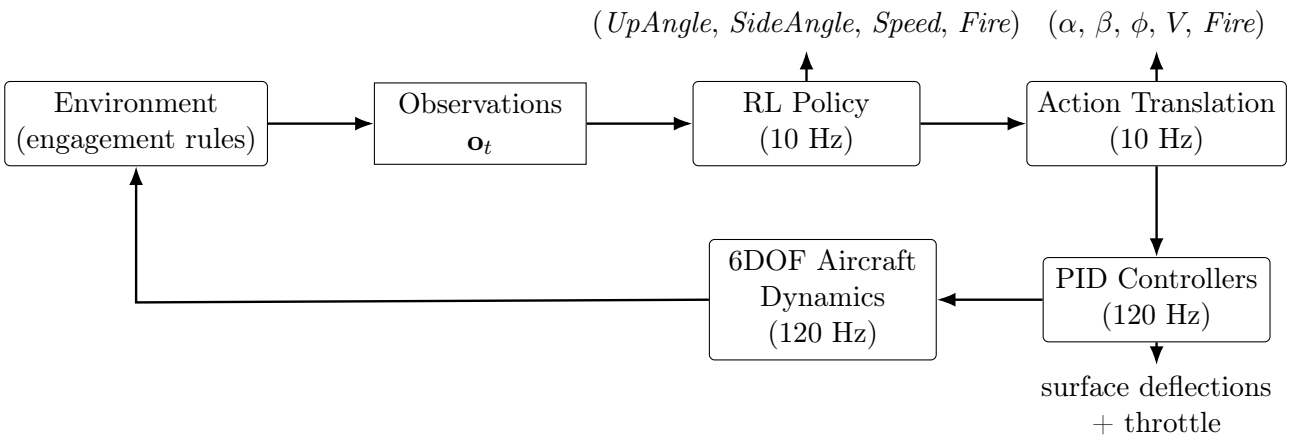

Figure 4.1: Overview of the hierarchical control stack. A low-frequency policy (10 Hz) outputs high-level geometric commands, which are translated into control-relevant targets and tracked by high-frequency PID controllers (120 Hz) interacting with the 6DOF aircraft dynamics model.

translation operate at a frequency of 10 Hz, the resulting target commands are tracked by a set of classical PID controllers running at 120 Hz, matching the update rate of the physical simulation. Each PID controller acts independently on the corresponding control surfaces, producing deflections that are then fed into the aircraft dynamics model, ultimately generating forces and moments that affect the simulated environment.

This hierarchical control structure was inspired by the approach adopted in the DARPA AlphaDogfight Trials [14]. However, the present work departs significantly from that formulation by replacing the learned low-level control policy with a classical PID-based control layer, resulting in a clearer separation between tactical decision-making and high-frequency control.

4.2. High-Level Action Space Definition

As introduced above, the policy outputs a polar velocity vector expressed in the *Body* frame, together with a *Fire* command. This choice emerged from several empirical iterations involving alternative action representations and was found to provide the most stable and effective learning behavior.

A key advantage of this formulation lies in the consistency between the policy output space and the geometric representation of the environment observations. The relative position and velocity of other aircraft are expressed as polar vectors, and quantities such as track angle and adverse angle can be interpreted as rotations of the forward axis in

the *Body* frame. As a result, the policy is required to learn a mapping within the same vectorial domain, effectively selecting a target direction and speed that align with the observed engagement geometry.

This induces a quasi-identity relationship between observations and actions, significantly reducing the complexity of the function to be learned by the policy. By operating directly in a geometrically meaningful space, the learning problem is simplified, leading to improved training stability and enhanced final performance. This concept is illustrated schematically in Figure 4.2.

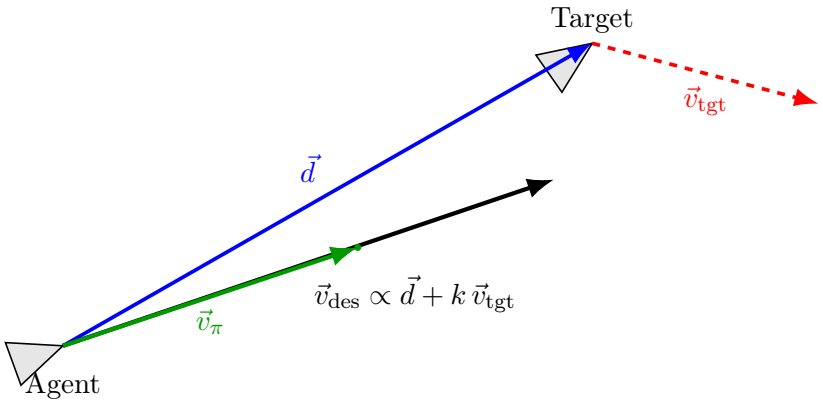

Figure 4.2: Geometric consistency between observations and actions. The policy outputs a desired velocity direction $\vec{v}_{\pi}$ (green) in the same vectorial domain used to represent the relative geometry, expressed by the relative position vector $\vec{d}$ (blue) and the target velocity $\vec{v}_{\text{tgt}}$ (red). The resulting mapping closely resembles an identity transformation in the geometric space.

4.2.1. Geometric Interpretation of Agent Commands

To enable the aforementioned simplification, an *Action Translation* layer was introduced between the policy output and the low-level control system. This layer serves two distinct but complementary roles.

The first role is to translate the polar velocity command produced by the policy into control-relevant target quantities that can be directly tracked by the PID controllers. Given a normalized desired velocity direction expressed in the *Body* frame as

$$\mathbf{v}_d = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T,$$

the translation is performed as

$$\alpha = \arccos(v_x), \quad \beta = 0, \quad \phi = \arctan 2(v_y, v_z),$$

where α denotes the target angle of attack, β the sideslip angle, and ϕ the roll angle. This formulation preserves the geometric intent of the policy command while expressing it in terms of quantities that naturally induce banked turn maneuvers. Such maneuvers would otherwise require the policy to implicitly learn complex couplings between roll, pitch, and yaw dynamics.

There are specific situations in which the general translation described above is intentionally bypassed. When the requested target vector corresponds to a negative *UpAngle* and a *SideAngle* below a 5° threshold, or when only small lateral corrections are required, the layer maps the *UpAngle* and *SideAngle* directly to angle of attack and sideslip targets. This enables fine adjustments using elevators and rudders alone and prevents unnecessary aircraft inversion when the agent commands a downward pointing maneuver, thereby reducing the risk of dynamic instability.

The second role of the *Action Translation* layer addresses the fact that the PID controllers used for low-level control operate independently and do not explicitly coordinate their actions. When aggressive combined maneuvers are commanded, this independence can lead to transient instabilities. To mitigate this effect, the translation layer incorporates a maneuver progression mechanism that softens angle-of-attack commands during the early phase of a maneuver. Formally, when the maneuver progress parameter $\gamma \in [0, 1]$ satisfies $\gamma < 0.6$ and the requested angle of attack exceeds 10°, the target angle of attack is limited according to

$$\alpha \leftarrow \text{clip}\left(1.025\,\alpha_{\text{current}},\ -\alpha,\ \alpha\right).$$

This heuristic was tuned empirically and proved effective in reducing controller transients while preserving the responsiveness required for close-range combat maneuvers.

4.3. Low-Level Control via PID Controllers

As discussed in the previous sections, low-level control and actuation of the aircraft were delegated to a set of classical PID controllers. This design choice was motivated both by the initial difficulty of training a policy to directly control aerodynamic surfaces and by the inherent simplicity and robustness of PID-based control. In addition, this approach enables a clear separation between tactical decision-making and fast, stabilizing control loops, which is further discussed in the next section.

The low-level control layer consists of four independent PID controllers, responsible respectively for tracking targets in angle of attack, sideslip, roll, and airspeed. These controllers act directly on the elevators, rudders, ailerons, and engine throttle. Each controller receives a target value generated by the action translation layer and outputs a corresponding actuation command applied to the aircraft model.

The PID gains were tuned empirically by analyzing the response of the physical model to isolated step commands of increasing amplitude. For each controlled quantity, three representative step magnitudes were evaluated in order to balance fast and accurate response for small target variations with stability under larger and more aggressive commands. An example of the step response used during the tuning of the angle-of-attack controller is shown in Figure 4.3.

This tuning procedure was carried out independently for each aircraft variant. The resulting PID parameters were incorporated into the parametric definition of each vehicle, ensuring that differences in performance between variants arise from their aerodynamic and inertial properties rather than from a shared or biased control configuration. This design choice also preserves the extensibility of the framework, allowing further aircraft models to be introduced with their own tailored control parameters.

Across all variants, the final tuning emphasized proportional and derivative gains, while the integral term played a comparatively minor role. This reflects the fact that the primary objective of the low-level controllers is to provide responsive and stable tracking of rapidly changing targets, rather than long-term elimination of steady-state error.

4.4. Control Frequencies and Stability Considerations

A key concept, highlighted in several prior works and further validated by the experiments conducted in this thesis, is the decoupling of command frequency from control frequency. In the proposed architecture, this is implemented by running policy inference at a frequency of 10 Hz, while the PID controllers and physical simulation operate at 120 Hz.

This separation is fundamental because it isolates two tasks that are difficult to learn and manage simultaneously: the high-level mission objective, such as pursuit and engagement, and the low-level aircraft stabilization and maneuver execution. When both tasks are handled directly by a single reinforcement learning policy, the agent receives immediate feedback related to control errors, while the tactical consequences of its actions manifest over longer time horizons. This mismatch complicates temporal credit assignment and

forces the choice of the discount factor to become a compromise between control precision and tactical awareness.

By separating decision-making from control, this issue is largely mitigated. The reinforcement learning policy operates at a time scale appropriate for tactical reasoning, while the PID controllers handle fast dynamics and ensure stable execution of the requested maneuvers.

The specific choice of control frequencies was informed by empirical analysis of the PID step responses for the most critical control channels. These tests indicated that approximately 20 to 40 control iterations were required for a medium-amplitude command to be reliably achieved while maintaining stability. Given computational constraints, a direct 10 Hz to 400 Hz separation was deemed impractical. Instead, both frequencies were adjusted to obtain a suitable compromise, resulting in the final configuration of 10 Hz for policy inference and 120 Hz for control and physics simulation.

Under this configuration, each policy action is held constant for 12 physics steps. While this does not fully span the complete convergence time of a maneuver, it proved sufficient to avoid control instabilities even when the policy exhibits abrupt or exploratory behavior, while maintaining acceptable computational performance.

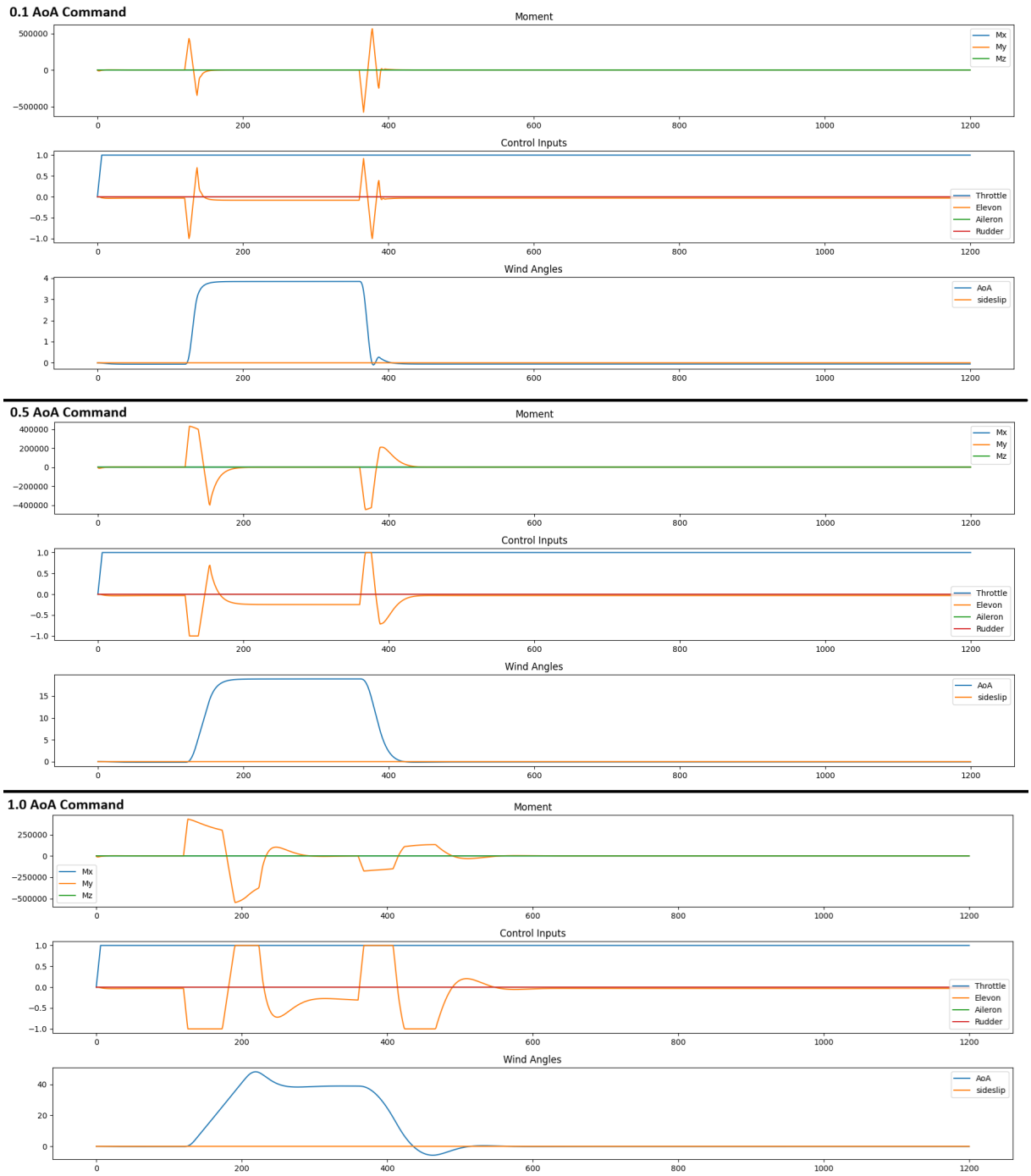

Figure 4.3: Angle-of-attack (AoA) controller tuning via isolated step commands of increasing amplitude. The three panels correspond to command magnitudes of 0.1, 0.5, and 1.0 (normalized). For each case, the resulting moment components, control inputs (throttle and surface deflections), and aerodynamic angles are shown over time. The comparison highlights the trade-off between fast tracking for small commands and stability/saturation effects under larger, more aggressive AoA requests.

5 | Reinforcement Learning Environment Design

In this chapter, the design and structure of the reinforcement learning environment are discussed in detail. The focus is placed on the conceptual and architectural decisions underlying the environment implementation, rather than on low-level code details.

The software developed for this thesis is organized into three main components. The physical modeling aspects were addressed in Chapter 3 and are implemented in the `Physics.py` module. The present chapter focuses on the reinforcement learning environment, implemented in the `Environment.py` module. Finally, the training procedures, algorithm configuration, and hyperparameter tuning are discussed in subsequent chapters and are implemented in the `TrainingScript.py` module.

Each of these modules corresponds to a dedicated class, and interaction between components is handled through object instantiation and well-defined interfaces. In particular, the environment class instantiates multiple *Aircraft* objects, each derived from the *FixedWingAircraft* class, and is itself instantiated within the training script to interface with the reinforcement learning algorithm.

Given this high-level implementation overview, the remainder of this chapter focuses on the design of the reinforcement learning environment, starting from the definition of the observation space.

5.1. Observation Space

The observation space was designed to provide the agent with sufficient information to perform effective decision-making in a close-range air combat scenario, while keeping the dimensionality and structure of the observations as compact and invariant as possible.

To this end, the observation vector is divided into two main components. The first component describes the agent's own state, capturing the most relevant information about

the controlled aircraft. The second component provides information about other aircraft present in the environment, including both friendly and adversarial agents.

5.1.1. Agent Self-Observation

The agent's self-observation encodes the fundamental state variables of the aircraft dynamic model, enriched with additional information relevant to the combat task. Specifically, the following quantities are included:

- Altitude, expressed using a z-down convention and normalized by a reference altitude.
- Linear acceleration in the *Body* frame, normalized by a safety-scale factor.
- Linear velocity in the *Body* frame, normalized by the maximum allowed speed in the scenario.
- Aircraft orientation, represented using sine and cosine encoding of Euler angles to avoid discontinuities.
- Angular velocity in the *Body* frame, normalized by a constant scaling factor.
- Angle of attack and sideslip angle, each encoded using sine and cosine representations.
- Distance from the centroid of the environment bases, projected on the horizontal plane and normalized by the environment size.
- Missile tone indicators, representing both the attack tone acquired on an adversary and the defensive tone received from a potential pursuer.

This representation provides the agent with a complete description of its own kinematic and dynamic state, while also supplying minimal positional information required to remain within the combat envelope and avoid collisions with the environment boundaries.

Importantly, spatial information is expressed relative to the aircraft's *Body* frame whenever possible, rather than using absolute world-frame quantities. This design choice contributes to observation invariance, as identical engagement geometries produce similar observations regardless of their absolute position within the environment.

5.1.2. Observations of Other Aircraft

The second part of the observation space provides information about all other aircraft present in the environment. For each other agent, the following quantities are included:

- Relative position expressed in the ego aircraft *Body* frame using a polar encoding.
- Relative velocity expressed in the ego aircraft *Body* frame using a polar encoding.
- Closure rate, defined as the projection of relative velocity along the line of sight and normalized.
- Track angle and adverse angle, each encoded using sine and cosine representations.
- A binary flag indicating whether the other aircraft is alive.
- A binary friend-or-foe indicator.

This structure enables the agent to reason about the geometry and dynamics of the engagement with each aircraft independently, while remaining agnostic to the absolute position of the encounter within the environment. Both friendly and adversarial agents are included, allowing the same observation structure to support future extensions to multi-agent and team-based scenarios.

The guiding principle behind the iterative design of the observation space was to maximize invariance with respect to absolute reference frames and to encode information in a geometrically meaningful way. By expressing positions, velocities, and angles relative to the agent's own *Body* frame, equivalent engagement configurations produce nearly identical observations regardless of their location in the world.

This invariance significantly reduces the effective complexity of the observation space, making it easier for the reinforcement learning policy to generalize across scenarios. Compared to observation structures based on absolute reference frames, this approach proved to improve training stability and convergence speed in the experiments conducted during this work.

5.2. Action Space

The action space used in this work has already been introduced in the Control Architecture chapter. For completeness, a brief summary is provided here, focusing on how the policy output is exposed at the environment level and how it is scaled to interface with the control stack.

At each decision step, the policy outputs a high-level command consisting of a target velocity vector expressed in polar form together with a firing command:

$$(UpAngle,\ SideAngle,\ Speed,\ Fire).$$

As discussed previously, this representation was chosen to maintain consistency between the observation and action spaces, allowing the policy to operate within a coherent geometric domain and effectively learn quasi-identity mappings between perceived engagement geometry and desired motion.

Each continuous action component is normalized in the interval $[-1, 1]$ and subsequently rescaled before being passed to the next stage of the control stack. For the angular components (*UpAngle* and *SideAngle*), the normalization factor corresponds to $\pm 30°$. This value was selected by considering the worst-case angle-of-attack request that can emerge from the Action Translation layer and comparing it against the maximum allowable angle of attack enforced by the physical model. This choice preserves the agent's ability to request extreme maneuvers—potentially leading to instability or crashes if sustained—while still respecting the aerodynamic constraints of the simulated aircraft.

The *Speed* command is scaled by the maximum allowable airspeed, set to 343 m/s, corresponding to the speed of sound under standard atmospheric conditions. This implicitly restricts the action space to the subsonic regime, consistent with the aerodynamic modeling assumptions discussed in Chapter 3.

The *Fire* command is treated differently. Although it is ultimately interpreted as a binary decision, it is modeled as a continuous output of the policy. A firing event is triggered when this value exceeds a predefined threshold. This design choice allows the internal evolution of the firing intent to be observed during training: empirically, the agent tends to increase the fire command smoothly as positional advantage improves and missile tone accumulates.

5.3. Engagement Rules and Scenario Configuration

The reinforcement learning task is defined through a set of engagement rules embedded in the environment design. At a high level, the objective of each agent is to approach an adversarial aircraft, maneuver into a favorable position behind it, maintain that positional advantage long enough to acquire missile tone, and eventually shoot the target down, while remaining within the combat envelope defined by the environment constraints.

Weapon effectiveness and vulnerability are modeled through two geometric regions associ-

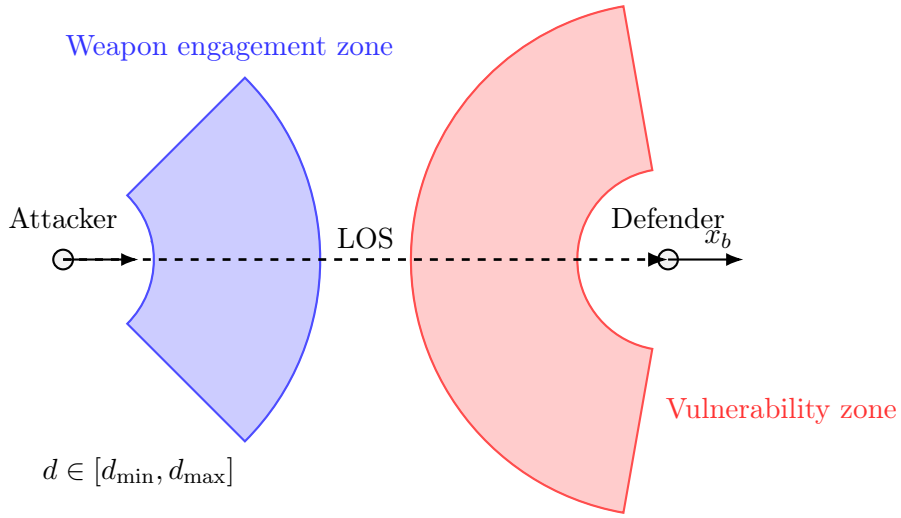

Figure 5.1: Clean 2D schematic of the engagement geometry used in the environment. The attacker's forward *weapon engagement zone* and the defender's rear *vulnerability zone* are modeled as annular sectors defined by an opening angle and by minimum/maximum effective range. The line-of-sight (LOS) between aircraft is used to derive track/adverse angles and distance-dependent engagement conditions for tone accumulation and probabilistic hit evaluation.

ated with each aircraft: a *weapon engagement zone* and a *vulnerability zone*. Both regions are represented as conical volumes aligned with the aircraft longitudinal axis, defined by an opening angle and a minimum and maximum effective range.

The weapon engagement zone is oriented forward and represents the region in which missile guidance and tracking are feasible. Across the aircraft variants considered in this work, its opening angle ranges approximately between 90° and 120°. The vulnerability zone, oriented backward, represents the region in which the aircraft is most exposed to an opponent's weapons. Its opening angle ranges between 160° and 200°, allowing for lateral engagements to be partially effective even when the geometry is not perfectly aligned. Both zones share a minimum effective range of approximately 500 m and a maximum range of 5000 m. All these parameters are defined at the aircraft model level, reflecting differences in sensor coverage and weapon characteristics between variants. Figure 5.1 shows the forward weapon engagement zone and rear vulnerability zone that define the rules of engagement and the geometric quantities used for reward shaping.

Missile tone accumulation is modeled as a simple discrete-time process. When an adversarial aircraft lies within the weapon engagement zone, a tone variable is incremented at each time step. The increment value is configurable; for the experiments conducted in this work, it was set to 0.01. This choice enforces the requirement that a favorable

engagement geometry must be maintained for a non-negligible duration before a shot can be taken. Taking a shot also resets the counter so that multiple successive shots cannot be taken at each step after missile tone acquisition.

Once the accumulated tone exceeds a predefined threshold, the firing command becomes effective. If, at the same time, the policy output for the *Fire* action exceeds its activation threshold, a probabilistic hit evaluation is performed. Rather than explicitly simulating missile dynamics, a Bernoulli trial is used to determine whether the shot results in a hit.

Let θ_{track} denote the track angle between the attacker's forward axis and the line of sight to the target, and let Θ_{att} be the opening angle of the attacker's weapon engagement cone. An angular alignment factor is defined as

$$A = 0.2 + 0.8 \left(\frac{\frac{\Theta_{\text{att}}}{2} - \pi\theta_{\text{track}}}{\frac{\Theta_{\text{att}}}{2}} \right),$$

where A increases as the attacker's pointing direction aligns more closely with the center of the engagement cone. The hit probability is then computed as

$$P_{\text{hit}} = A \cdot T,$$

where T denotes the current missile tone value. A uniform random sample $u \sim \mathcal{U}(0, 1)$ is drawn, and a hit is registered if $u < P_{\text{hit}}$.

If a hit occurs, the target aircraft is marked as destroyed, excluded from the step execution queue, and left in its final position in the environment. The episode ends when all live aircraft belong to the same team.

In addition to combat outcomes, several termination conditions are enforced to define the combat envelope and maintain physical realism. These include limits on maximum angle of attack, maximum acceleration, minimum altitude, and boundary violations of the simulated environment. All termination thresholds are configurable.

5.4. Reward Function Design

The reward function plays a central role in shaping the behavior learned by the reinforcement learning agents. In the context of close-range air combat, the reward must simultaneously encourage stable flight, effective pursuit maneuvers, disciplined weapon usage, and successful engagement outcomes, while discouraging unsafe or physically unrealistic behaviors.

The overall structure of the reward function adopted in this work draws strong inspiration from the reward design presented in the DARPA AlphaDogfight Trials [14], particularly with respect to the decomposition of the task into flight stability, pursuit geometry, and sparse combat events. However, several components were adapted and extended to better suit missile-based engagements and continuous control with explicit speed management.

To address the requirements of the task, the reward function is composed of multiple terms, grouped into three main categories:

- flight envelope and stability shaping rewards,
- pursuit and engagement geometry shaping rewards,
- sparse event-based rewards and penalties.

Each reward component is computed at every simulation step and combined linearly using configurable weighting factors. This modular structure allows individual contributions to be monitored, tuned, and analyzed independently during training.

5.4.1. Flight Envelope and Stability Shaping

A first group of reward terms is dedicated to encouraging stable and physically plausible flight behavior. These terms act as continuous penalties that increase smoothly as the aircraft approaches unsafe or undesirable regions of the flight envelope.

Penalties are applied to excessive angle of attack α, sideslip angle β, low airspeed V, and deviations from a preferred altitude band. Each quantity is normalized with respect to a terminal threshold and shaped using a smooth sigmoid-based penalty function.

Taking the angle of attack as a representative example, the normalized quantity is defined as

$$\alpha_{\text{norm}} = \frac{|\alpha|}{\alpha_{\text{term}}},$$

where α_{term} denotes the terminal (non-recoverable) angle-of-attack threshold. A critical midpoint is defined as

$$\alpha_{\text{mid}} = \frac{\alpha_{\text{crit}}}{\alpha_{\text{term}}},$$

with α_{crit} representing the onset of strongly undesirable behavior. The corresponding penalty term is then computed as

$$R_\alpha = -w_\alpha \left(\frac{1}{1 + \exp(-k_\alpha \left(\alpha_{\text{norm}} - \alpha_{\text{mid}}\right))} \right),$$

where w_α is a weighting coefficient and k_α controls the steepness of the transition. An analogous formulation is applied to sideslip angle, low-speed penalties, and altitude deviations, with each term using its own critical and terminal thresholds.

This formulation ensures that small deviations from nominal flight conditions incur only mild penalties, while approaching unsafe regions of the flight envelope results in rapidly increasing negative reward, without introducing discontinuities in the reward signal.

In addition, a command smoothing penalty discourages abrupt changes in steering commands between consecutive time steps. Let $u^{(t)} = (u_{\text{up}}^{(t)}, u_{\text{side}}^{(t)})$ denote the vertical and lateral steering commands at time step t. The average command variation is defined as

$$\Delta u = \frac{1}{2}\left(\left|u_{\text{up}}^{(t)} - u_{\text{up}}^{(t-1)}\right| + \left|u_{\text{side}}^{(t)} - u_{\text{side}}^{(t-1)}\right|\right).$$

A critical variation threshold Δu_{crit} is defined, beyond which command changes are considered excessively aggressive. The corresponding smoothing penalty is computed as

$$R_{\text{smooth}} = -w_{\text{smooth}}\left(\frac{1}{1+\exp(-k_\Delta\,(\Delta u - \Delta u_{\text{crit}}))}\right),$$

where w_{smooth} is the smoothing weight and k_Δ controls the sharpness of the penalty onset.

This term promotes smoother control inputs and reduces high-frequency oscillations that could destabilize the low-level controllers, while still allowing rapid maneuvering when required by the tactical situation.

5.4.2. Pursuit and Engagement Geometry Shaping

A second group of reward terms focuses on the tactical geometry of air combat engagements. These rewards are computed relative to a single reference opponent, selected as the closest alive adversarial aircraft.

The primary pursuit shaping term is based on the angular advantage between the attacker and the target. Let θ_{track} denote the track angle and θ_{adv} the adverse angle. An angular advantage measure is defined as

$$\Delta\theta = \theta_{\text{adv}} - \theta_{\text{track}}.$$

Rather than using a linear or sigmoidal mapping, this angular advantage is transformed

using a bounded tangential shaping function

$$R_{\text{pursuit}} = w_A \frac{\tan\left(\frac{\pi}{\tau}\Delta\theta\right)}{\tan\left(\frac{\pi}{\tau}\right)},$$

where τ controls the steepness of the mapping and w_A is a weighting factor.

This shaping choice was selected after empirical comparison with linear and sigmoid-based alternatives. The tangential formulation proved more effective, as it provides an increasingly strong gradient as the agent approaches optimal engagement geometry, while remaining bounded. This behavior was observed to be particularly beneficial during exploration and late-stage policy refinement.

In addition to angular alignment, a second pursuit-related reward term explicitly couples relative distance, closure rate, and pointing accuracy into a single shaping mechanism. Unlike many approaches that treat these components independently, the formulation adopted in this work combines them multiplicatively and additively within a single reward expression. As a consequence, achieving a high reward requires the agent to simultaneously address all three aspects of pursuit behavior.

Let d denote the distance to the target, $\dot{d}$ the closure rate, and $d^\star$ the center of an optimal engagement zone defined by the opponent's vulnerability cone. Three engagement regimes are distinguished:

- when $d > d^\star$, positive closure and accurate pointing are encouraged to rapidly intercept the target;
- when $d \approx d^\star$, closure is discouraged and relative velocity matching is favored to maintain a stable position behind the target;
- when $d < d^\star$, negative closure is encouraged to avoid overshooting and loss of positional advantage.

This behavior is implemented through distance-dependent weighting functions applied to the closure term, yielding a combined shaping reward of the form

$$R_{\text{closure}} = w_C \left(\lambda_1(d)\,\dot{d}(1-\theta_{\text{track}}) + \lambda_2(d)\,(-\dot{d}) + \lambda_3(d)\,(1-|\dot{d}|)\right),$$

where $\lambda_i(d)$ are smooth, distance-dependent weighting functions.

Crucially, no single component of this expression can yield a high reward in isolation. For example, aggressive closure without adequate angular alignment, or correct pointing without appropriate distance and speed management, results in a reduced or null contribution.

This coupling effectively prevents degenerate behaviors in which the agent attempts to exploit only one aspect of the reward function while neglecting the others.

Through extensive empirical iteration, this combined formulation proved particularly effective in allowing speed control behavior to emerge alongside heading control. In practice, it significantly reduced overshooting errors and mitigated the tendency of angular control to dominate speed regulation, resulting in more realistic and tactically effective pursuit behavior.

5.4.3. Weapon Usage and Sparse Event Rewards

In addition to continuous shaping rewards, several sparse reward components are included to model weapon usage discipline and combat outcomes. These terms are designed to encourage correct coordination between engagement geometry, missile lock quality, and firing decisions, while preventing degenerate behaviors such as indiscriminate trigger activation.

Trigger Discipline. A trigger discipline penalty is applied to discourage premature or poorly timed firing attempts. Let $u_{\text{fire}} \in [-1, 1]$ denote the continuous firing command produced by the policy, and let τ_{fire} be the activation threshold above which a firing event is considered. When the agent attempts to fire without sufficient missile tone, a penalty proportional to the deviation from the threshold is applied:

$$R_{\text{trigger}} = -w_{\text{trig}}\, \mathbb{I}(T_{\text{att}} = 1)\, |\text{clip}(u_{\text{fire}}, -1, \tau_{\text{fire}}) - \tau_{\text{fire}}|\,,$$

where $T_{\text{att}} \in [0, 1]$ denotes the attack missile tone, w_{trig} is a penalty weight, and $\mathbb{I}(\cdot)$ is the indicator function.

This formulation penalizes trigger activation when firing conditions are not fully satisfied, while allowing smooth modulation of firing intent as the engagement geometry improves. As a result, the agent is encouraged to align firing decisions with sustained positional advantage rather than exploiting the stochastic nature of hit resolution.

Missile Tone-Based Shaping. Additional sparse shaping rewards are associated with missile tone accumulation, which serves as a proxy for lock quality and engagement confidence.

An attack shaping reward is applied when attack tone is accumulated under favorable tracking conditions:

$$R_{\text{attack}} = w_{\text{att}}\, T_{\text{att}}\, \theta_{\text{track}},$$

where w_{att} is a weighting coefficient and θ_{track} denotes the normalized track angle. This term encourages the agent to maintain correct pointing while building missile lock.

Conversely, a defensive shaping penalty is applied when the agent is exposed to adversarial threats:

$$R_{\text{defence}} = -w_{\text{def}}\, T_{\text{def}}\, \theta_{\text{adv}},$$

where T_{def} denotes the defensive missile tone and θ_{adv} the adverse angle. This term discourages prolonged exposure to enemy weapon engagement zones and promotes evasive behavior when under threat.

Kill Reward. A sparse kill bonus is awarded when an adversarial aircraft is successfully destroyed. This reward is scaled by the attack missile tone at the time of the kill:

$$R_{\text{kill}} = w_{\text{kill}}\, T_{\text{att}},$$

reinforcing the importance of achieving and maintaining a strong lock prior to firing.

Scaling the kill reward by tone discourages opportunistic firing and aligns terminal success with correct engagement geometry and timing.

5.4.4. Termination Penalties and Final Reward Aggregation

Unsafe or physically implausible behaviors result in immediate termination of the episode for the affected agent, with an associated penalty. Termination conditions include collisions, excessive angles, low airspeed (stall), altitude violations, and exit from the combat envelope. When any termination condition is triggered, the agent is marked as destroyed and a terminal penalty is applied.

Final Reward Formulation. At each simulation step, the reward is computed as a weighted combination of flight-envelope shaping, pursuit shaping, and sparse event-based terms. The overall reward is defined as

$$R = w_F\, R_{\text{flight}} + w_P\, R_{\text{pursuit}} + R_{\text{sparse}}, \tag{5.1}$$

where w_F and w_P regulate the relative contribution of stability-oriented shaping and tactical shaping.

The flight component aggregates individual penalty terms:

$$R_{\text{flight}} = R_\alpha + R_\beta + R_V + R_h + R_{\Delta u}, \tag{5.2}$$

corresponding respectively to angle-of-attack penalty, sideslip penalty, low-speed penalty, altitude penalty, and command smoothing penalty.

The pursuit component combines angle-advantage shaping and distance-aware closure shaping:

$$R_{\text{pursuit}} = w_A \, \tilde{R}_{\text{pursuit}} + w_C \, \tilde{R}_{\text{closure}}, \tag{5.3}$$

where $\tilde{R}_{\text{pursuit}}$ denotes the tangentially-shaped angular advantage term and $\tilde{R}_{\text{closure}}$ the coupled distance–closure–alignment term.

Finally, the sparse reward term collects event-driven rewards and penalties:

$$R_{\text{sparse}} = R_{\text{trigger}} + R_{\text{attack}} + R_{\text{defence}} + R_{\text{kill}} + R_{\text{term}}, \tag{5.4}$$

where R_{term} is applied only upon termination.

Reward Configuration. Table 5.1 summarizes the final set of weights and parameters adopted for the experiments reported in this thesis. All values are configurable through the environment configuration file.

As a final implementation note, the software developed for this thesis supports automated evaluation of multiple reward configurations through repeated training trials. In particular, reward parameters can be defined as sets of candidate values, enabling systematic sweeps (grid searches) where each trial is trained and evaluated under a distinct reward configuration. This functionality proved useful during reward shaping and was used to converge toward the final configuration reported in Table 5.1.

5.5. Multi-Agent Environment Capabilities

To conclude this chapter, it is important to emphasize that while the reward function and decision-making logic are defined at the level of individual agents, the environment itself is designed to support fully multi-agent and multi-team scenarios.

The environment implementation maintains a global state that tracks multiple teams, each composed of an arbitrary number of aircraft. Both agent-controlled and scripted ("dummy") aircraft can coexist within the same scenario, allowing flexible combinations

of learning agents and predefined opponents. This design enables curriculum learning against progressively more capable adversaries, as well as competitive self-play among learned policies.

In addition to per-agent reward computation, the environment handles team-level logic, including match termination conditions, victory determination, and episode-level outcome tracking. Randomized spawning procedures are supported, allowing initial positions, orientations, and velocities to be sampled according to configurable distributions. This functionality plays a central role in curriculum design and robustness evaluation, as discussed in later chapters.

From an analysis and evaluation standpoint, the environment also provides multiple visualization and logging tools. These include lightweight two-dimensional rendering for large-scale rollout inspection and final video generation, as well as three-dimensional trajectory reconstruction for post-hoc analysis of engagement geometry, maneuvering behavior, and emergent tactics.

Many of the logical mechanisms enabled by this multi-agent design—such as opponent selection, policy freezing, ranking, and evaluation scheduling—are exercised primarily during training and benchmarking. Accordingly, they will be discussed in greater detail in the training and results chapters, where their impact on learning dynamics and performance becomes evident.

Table 5.1: Reward function configuration used in the final experiments.

Parameter	Value	Description
w_F (GFW)	0.2	Global weight of flight-envelope shaping (stability / safety).
w_P (PW)	0.8	Global weight of pursuit/tactical shaping.
w_C (CW)	0.8	Closure shaping weight inside R_{pursuit}.
w_A (AW)	0.2	Angle-advantage shaping weight inside R_{pursuit}.
AoA_W	0.0	Angle-of-attack shaping weight (disabled in final configuration).
Sideslip_W	0.0	Sideslip shaping weight (disabled in final configuration).
Speed_W	0.4	Low-speed penalty weight.
Altitude_W	0.3	Altitude penalty weight.
Smoothing_W	0.3	Command smoothing penalty weight.
Critical_AoA	15°	Onset threshold for AoA shaping.
Terminal_AoA	40°	Termination threshold for AoA violation.
Critical_Sideslip	15°	Onset threshold for sideslip shaping.
Terminal_Sideslip	40°	Termination threshold for sideslip violation.
Critical_Speed	150 m/s	Onset threshold for low-speed shaping.
Terminal_Speed	50 m/s	Termination threshold for stall/low-speed.
Critical_Altitude	2000 m	Onset threshold for altitude shaping (relative to mid-altitude band).
Critical_Delta	0.1	Command-change threshold for smoothing penalty.
optimal_zone	1000 m	Width of the distance band around $d^\star$ used in closure shaping.
tan_parameter	3	Steepness parameter for tangential angular shaping.
att_tone_bonus	0.5	Reward coefficient tied to attack tone accumulation.
def_tone_bonus	10	Penalty coefficient tied to defensive tone exposure.
trigger_penalty	0.2	Penalty coefficient for poor trigger discipline.
kill_bonus	250	Sparse reward for a successful kill.
terminal_penalty	1000	Penalty applied on termination due to envelope violation/collision.
killed_penalty	500	Penalty associated with being killed (if modeled separately).

6 | Soft Actor–Critic Algorithm

6.1. Algorithm Selection and Implementation Framework

The reinforcement learning algorithm selected for this work is Soft Actor–Critic (SAC), an off-policy actor–critic method designed for continuous action spaces and stochastic policies. SAC was chosen for its stability properties, sample efficiency, and its explicit handling of the exploration–exploitation trade-off through entropy regularization.

In addition to these theoretical advantages, the choice of SAC was also informed by comparative studies in continuous-control domains. In particular, the work [11], which compares PPO, TD3, and SAC on a quadruped locomotion task, reports superior stability and convergence behavior for SAC when dealing with highly coupled dynamics and continuous actuation. Although the application context differs from aerial combat, the underlying control challenges share important similarities, such as nonlinear dynamics, delayed reward attribution, and the need for smooth yet expressive control policies.

The objective optimized by SAC augments the standard discounted return with an entropy term:

$$J(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \left(r(s_t, a_t) + \alpha\, \mathcal{H}(\pi(\cdot|s_t))\right)\right],$$

where γ is the discount factor, α is the entropy temperature, and $\mathcal{H}(\cdot)$ denotes the policy entropy. This formulation encourages sustained exploration while allowing the policy to gradually converge toward deterministic behavior as training progresses.

The algorithm was implemented using RLlib, which provides a scalable and modular implementation of SAC with native support for off-policy replay, parallel environment execution, and multi-agent training. In the following sections, the focus is placed on the design choices and empirical tuning performed on top of this baseline implementation.

6.1.1. Discount Factor

The discount factor γ controls the relative importance of future rewards in the value function and therefore plays a central role in temporal credit assignment. In the close-range air combat scenario considered in this work, many tactical decisions have delayed consequences: maneuvering actions may only translate into a favorable engagement geometry, missile tone accumulation, or a successful kill after several seconds of flight.

From a practical standpoint, this means that sparse but decisive rewards—such as aircraft destruction or episode termination—must be able to propagate backward through a potentially long sequence of intermediate states and actions. This propagation can be controlled either by adjusting the magnitude of the sparse rewards themselves or by increasing the discount factor. In this work, the discount factor was treated as the primary mechanism to establish an appropriate temporal horizon for the task, while reward magnitudes were refined afterward to balance the contribution of sparse and dense components.

Empirical testing showed that lower values of γ biased learning toward short-term stabilization behavior, such as maintaining safe flight conditions or minimizing immediate penalties, at the expense of long-term tactical objectives. In these cases, the influence of sparse rewards decayed too rapidly, preventing meaningful credit assignment to the actions that ultimately led to successful engagements.

Conversely, values of γ approaching unity increased the variance of value estimates and slowed convergence, particularly during early exploration phases, as distant rewards dominated the learning signal. A discount factor of

$$\gamma = 0.99$$

was found to provide a suitable compromise. At this value, sparse rewards such as successful kills or terminal penalties were able to propagate over the expected engagement time horizon, while still allowing dense shaping rewards to guide short-term maneuvering behavior.

Once this temporal horizon was fixed through the choice of γ, the relative influence of sparse and continuous reward components was further refined by tuning their respective magnitudes. In particular, attack and defense tone rewards were kept intentionally small to prevent them from overwhelming flight stability and pursuit shaping terms at this discount factor. This two-step procedure—first selecting an appropriate discount factor based on task time scale, then balancing reward magnitudes accordingly—proved effective in stabilizing training and enabling the emergence of coherent tactical behavior.

6.1.2. Training Batch Size

As an off-policy method, Soft Actor–Critic relies on stochastic gradient updates computed from samples drawn from a replay buffer. The training batch size therefore plays a central role in determining gradient variance, convergence stability, and the balance between data efficiency and computational cost.

In the RLlib implementation used in this work, the training batch is constructed by aggregating trajectory fragments collected in parallel by multiple environment runners. Episodes are not necessarily completed before sampling occurs; instead, fixed-length fragments from different agents and episodes are concatenated into a single training batch. Once the accumulated number of samples reaches the specified training batch size, a policy update is performed. As a result, the batch size influences not only the number of samples used per update, but also the temporal diversity and mixing of experience drawn from different phases of multiple episodes.

From a practical standpoint, smaller batch sizes resulted in highly noisy gradient updates and unstable learning dynamics, particularly during early exploration phases when experience is strongly stochastic. In contrast, larger batch sizes produced smoother updates by averaging over a broader set of states, actions, and engagement conditions, but at the cost of increased computational load and diminishing marginal returns.

A systematic empirical evaluation was conducted by testing batch sizes of 200, 400, 500, 700, 1000, and 1200 samples per learner update. The most significant improvements in training stability and convergence behavior were observed when increasing the batch size from 200 up to approximately 700 samples. Beyond this point, further increases continued to yield performance gains, but with progressively smaller and less consistent improvements.

Based on these observations, a batch size of

$$N_{\text{batch}} = 1200$$

was selected as a reasonable compromise. This value provided stable gradient estimates and robust learning behavior while remaining computationally feasible given the available parallelism and training resources.

6.2. Neural Network Architecture

The policy and value networks were implemented as multilayer perceptrons. Rather than adopting deep or highly complex architectures, the design of these networks was informed by insights from the deep reinforcement learning literature on scaling neural network size in RL.

The study "Training Larger Networks for Deep Reinforcement Learning" analyzes the effect of increased network capacity on reinforcement learning performance and stability [13]. Contrary to the trends in supervised learning, where very deep and wide networks have driven significant gains, the authors observe that naively increasing network depth or width in RL can lead to degraded performance or unstable training. Specifically, the paper shows that simply deepening networks with a fixed unit count often fails to improve performance on continuous-control tasks, and may even harm convergence due to complex loss landscapes and sensitivity to training choices.

Motivated by these findings, several network configurations were empirically evaluated in this work. Both shallow and deeper architectures were tested, with hidden layer sizes of 128, 256, 512, and 1024 units, and depths ranging from one to three hidden layers. Consistent with the conclusions of [13], merely increasing depth did not consistently improve performance and often introduced training instability. For example, networks with two or three hidden layers typically exhibited lower performance both in terms of behavior and in terms of training stability after the same time of training.

Regarding network width, while larger networks can in principle provide increased representational capacity, the marginal gains diminished as the number of units increased. Empirically, a single hidden layer of 256 units with ReLU activation provided the best trade-off between capacity, stability, and computational efficiency.

Formally, both the actor and critic networks implement a function of the form

$$f(s) = W_2\,\sigma(W_1 s + b_1) + b_2,$$

where $\sigma(\cdot)$ denotes the ReLU activation function. The relatively simple architecture was found to offer sufficient representational power for the nonlinear control and decision-making tasks encountered in close-range air combat, while remaining robust across repeated training runs.

Table 6.1: Final Soft Actor–Critic configuration used in training.

Parameter	Value	Description
Hidden layers	[256]	Single hidden-layer MLP for actor and critic.
Activation	ReLU	Nonlinear activation function.
Actor learning rate	3×10^{-5}	Learning rate for policy network updates.
Critic learning rate	3×10^{-4}	Learning rate for value network updates.
Entropy learning rate	3×10^{-4}	Learning rate for entropy temperature updates.
Initial α	1.0	Initial entropy temperature.
τ	0.005	Target network soft-update coefficient.
Replay buffer capacity	500 000	Maximum number of stored transitions.
Batch size	1200	Samples per learner update.
Discount factor γ	0.99	Reward discount factor.
Batch mode	Truncate episodes	Fixed-length rollout fragments.

6.3. Final Algorithm Configuration

The final Soft Actor–Critic configuration used for the experiments reported in this thesis is summarized in Table 6.1. This configuration was selected after an initial phase of hyperparameter exploration and was found to provide stable learning behavior and consistent performance on early versions of the aerial combat task.

Once this baseline configuration was established, it was intentionally kept fixed throughout the remainder of the work. This decision was made to isolate the effects of reward shaping, curriculum learning, and self-play dynamics from algorithmic and architectural variations. By freezing the learning algorithm configuration, subsequent improvements in performance could be attributed with greater confidence to changes in environment design and training strategy rather than to low-level optimization artifacts.

As a result, the focus of the following chapters shifts from algorithmic tuning to the design and evaluation of training curricula, competitive self-play mechanisms, and the resulting emergent combat behaviors.

7 | Curriculum Learning

7.1. Motivation and Design Principles

The curriculum learning strategy adopted in this work is designed to progressively scale the agent's ability to perform the close-range air combat task. As discussed in previous chapters, this task can be decomposed into at least three tightly coupled sub-tasks, which are also reflected in the structure of the reward function:

- maneuvering and engagement geometry management,
- closure and relative speed control,
- firing discipline based on missile tone and situational awareness.

Rather than introducing these sub-tasks sequentially, the approach adopted here exposes the agent to all of them from the beginning of training. The curriculum progression is instead realized by gradually increasing the difficulty of the environment through changes in initial conditions and adversary maneuvering capabilities. This design choice was motivated by two main considerations.

First, keeping the reward structure fixed across curriculum stages avoids the need for repeated reward redesign and ensures that all sub-tasks are consistently reinforced throughout training. Second, introducing all sub-tasks from the start encourages the emergence of balanced policies, in which maneuvering, speed control, and firing decisions co-evolve, rather than being learned in isolation. Alternative approaches that introduce sub-tasks sequentially risk creating imbalances in proficiency, potentially leading to degenerate behaviors that exploit only a subset of the reward components.

At the same time, exposing an untrained policy to a complex task with multiple reward signals carries the risk of overwhelming the learning process with noisy gradients. For this reason, the initial curriculum stage was iteratively simplified until a configuration was identified that enabled stable early learning. As a consequence, the analysis of this phase focuses primarily on the first curriculum stage, while subsequent stages are discussed in terms of qualitative behavioral changes and observed improvements.

7.2. Curriculum Structure

The curriculum is structured as a sequence of training stages characterized by increasing complexity in adversary behavior and initial condition variability. Across all stages, the controlled agent faces a scripted adversary (referred to as a *dummy* aircraft), whose maneuvering logic is progressively enriched.

The guiding principle of this progression is to start with predictable and fixed adversary behavior, allowing the agent to learn how to respond and adapt its maneuvering and tactics to well-defined conditions. As training progresses, the adversary becomes increasingly dynamic, requiring the policy to generalize previously acquired skills and react to changes in the opponent's behavior in real time.

7.2.1. Stage 1: Deterministic Linear Adversary

The first curriculum stage represents the simplest training scenario. The dummy adversary follows a deterministic linear trajectory, maintaining a fixed heading and speed throughout the episode. No random maneuvering or speed changes are introduced at this stage.

Initial conditions are constrained to reduce variability: the two aircraft are spawned at a fixed separation distance of approximately 3000 meters, with limited sets of relative orientations and speeds. This configuration emphasizes basic pursuit geometry, alignment with the target, and initial exposure to missile tone accumulation and firing logic, while minimizing uncertainty in the adversary's behavior.

The expectation at this stage is that the agent learns to align with the target and manage speed in order to reach a favorable engagement position. The linear motion of the adversary is particularly well suited for training speed management across the different regimes discussed in the reward design chapter. Once the agent reaches weapon range, the engagement geometry remains relatively stable, shifting the emphasis of the reward signal toward trigger discipline and firing decisions.

Although several configuration parameters are defined, the dominant factor at this stage is the dummy behavior type, which is set to `line`. Other parameters related to turning, speed changes, or randomization are not actively used by this adversary model.

7.2.2. Stage 2: Randomized Maneuver Selection

In the second curriculum stage, the dummy adversary behavior is upgraded to a randomized maneuvering model. At regular intervals, the dummy selects between maintaining straight-line flight and executing a coordinated turn. The duration of each maneuver, the turn radius, and the turn direction are sampled from predefined sets.

In addition to adversary behavior, this stage introduces randomized spawning. While the aircraft are still initially positioned facing each other, their relative yaw orientation is sampled within a wider angular range, significantly increasing variability in engagement geometry.

This randomized spawning was introduced after the first self-play training runs revealed a systematic disadvantage for agents that had never encountered certain initial orientations during curriculum training. In particular, policies trained only on fixed initial alignments performed poorly when required to act as adversaries starting from unfamiliar configurations. Introducing randomized spawning in this stage effectively removed this bias and improved generalization.

Compared to Stage 1, this configuration introduces uncertainty both in the future evolution of the adversary's trajectory and in the initial engagement setup, requiring the agent to generalize beyond deterministic pursuit strategies.

7.2.3. Stage 3: Dynamic Speed and Aggressive Maneuvering

The third stage further increases difficulty by allowing the dummy adversary to dynamically change its speed in addition to its heading. Maneuver change intervals are shortened and drawn from a broader set of values, while turn radii span a wider range, including more aggressive maneuvers.

The spawning distance is increased, extending the time horizon required to achieve engagement. As a result, the agent must learn to manage closure rate more carefully and avoid overshooting during pursuit, particularly when the adversary executes speed changes while maneuvering.

This stage places greater emphasis on coordinated control of heading and speed, reinforcing the coupled pursuit behavior encouraged by the reward design.

7.2.4. Stage 4: Full Curriculum Configuration

The final curriculum stage represents the most challenging scripted-adversary scenario. Adversary maneuvering remains fully randomized, with frequent changes in turn radius, direction, and speed. Spawn distance is increased further, and the range of initial orientations and speeds is maintained at its widest.

Missile engagement dynamics, safety constraints, and termination conditions are unchanged from previous stages, ensuring consistency in the task definition. The increased difficulty arises purely from the adversary's ability to generate complex, unpredictable trajectories over extended engagement durations.

At this stage, the environment closely approximates the conditions required for transition to self-play training, with the adversary being sufficiently dynamic to induce reactive and adaptive behavior in the trained policy, as discussed in subsequent chapters.

7.3. Training Results

This section presents and discusses the behaviors learned by the agent at each stage of the curriculum. As motivated in the previous chapter, particular emphasis is placed on the first curriculum stage, where foundational behaviors emerge, while subsequent stages are analyzed primarily in terms of behavioral adaptation and refinement.

7.3.1. Evaluation Methodology

Training performance is evaluated through periodic rollouts using frozen policy parameters. At each checkpoint, approximately every 1000 training iterations over a total of 20 000 iterations, a set of 30 evaluation episodes is generated. Each training iteration corresponds to a policy update performed on a mixed batch of 1200 environment steps, collected in parallel as described in Chapter 6.

The evaluation episodes presented in the following sections are representative samples distilled from this set. Across the evaluated episodes, some behaviors were observed consistently, while others occurred less frequently. The analysis therefore focuses on the most prevalent behavioral patterns for each sub-task, while also highlighting notable outliers. Even in the more challenging curriculum stages, successful behaviors were observed with high frequency, with reduced prevalence primarily attributable to longer episodes in which target elimination was not achieved within the evaluation horizon.

7.3.2. Step 1: Results

To illustrate the behaviors learned during the first curriculum stage, three representative evaluation episodes are presented. Each episode highlights proficiency in one or more of the core sub-tasks: engagement geometry management, closure and speed control, and firing discipline.

Episode 1: Geometry Reversal and Controlled Intercept Figure 7.1 shows an episode in which the agent starts from a disadvantageous orientation and rapidly inverts its heading to engage the adversary head-on. During this initial maneuver, the agent closely matches the adversary's speed and avoids excessive closure, passing the target at a controlled lateral distance. This behavior indicates successful assimilation of the distance-gated closure reward component.

After the initial pass, the agent executes a coordinated banked turn, briefly increasing speed before settling into velocity matching with the adversary. This maneuver brings the agent into an ideal firing position, from which multiple missile shots are executed. The episode terminates on the last shot, with firing occurring at approximately 0.8 missile tone, deliberately exceeding the minimum tone threshold of 0.5.

While the primary inversion maneuver is performed via an aggressive banked turn, upward looping maneuvers were also observed in other episodes, albeit less frequently. Throughout the engagement, the adversary steadily descends from an initial altitude of approximately 5000 m to 3800 m. The agent mirrors this descent while maintaining a vertical separation of roughly 100–200 m, eventually ending slightly below the adversary at around 3600 m.

Detailed telemetry for this episode is shown in Figure 7.2. Control commands remain largely stable, with small-amplitude high-frequency oscillations consistent with fine attitude regulation. Toward the end of the episode, a clear increase in the trigger command is visible, culminating in missile launch following a gradual buildup throughout the engagement.

The corresponding reward evolution is shown in Figure 7.3. Although the composite nature of the reward complicates direct interpretation, several trends are evident. During the head-on closing phase, indicated between the second and third vertical markers, the closure component transitions smoothly from positive to negative as engagement geometry evolves. Following the successful reversal and pursuit from behind, the closure signal returns to the positive domain. The attack reward component exhibits a sawtooth pattern beginning near the fourth marker, reflecting missile tone accumulation dynamics.

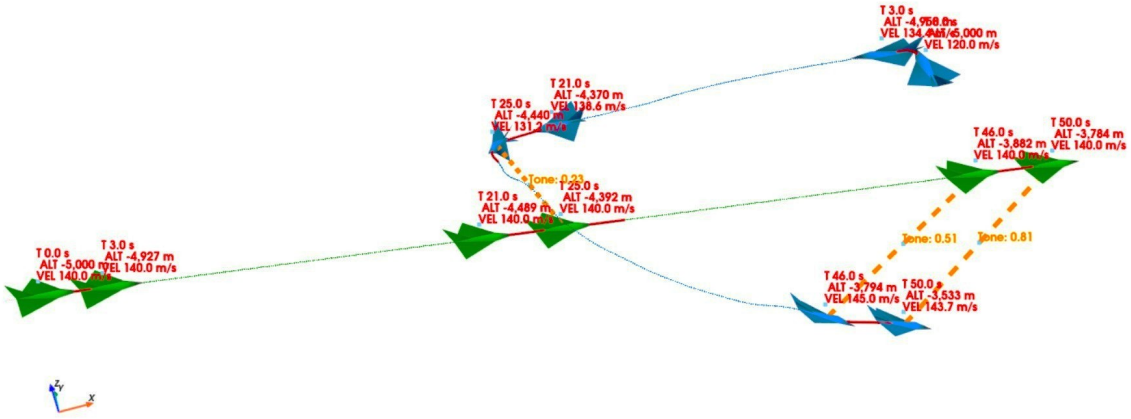

Figure 7.1: Curriculum Step 1, Sample Episode 1: Render

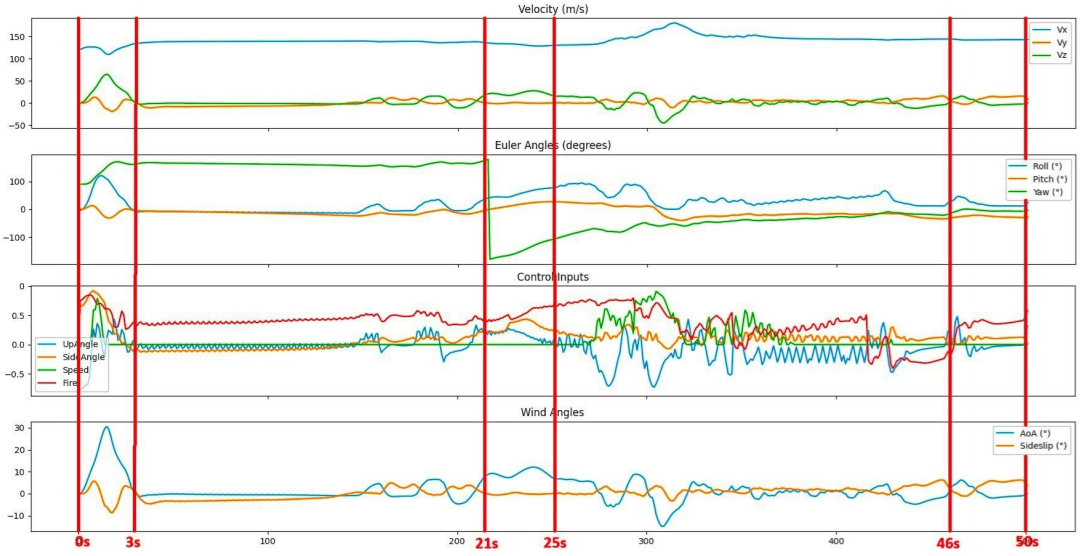

Figure 7.2: Curriculum Step 1, Sample Episode 1: Telemetry Plot

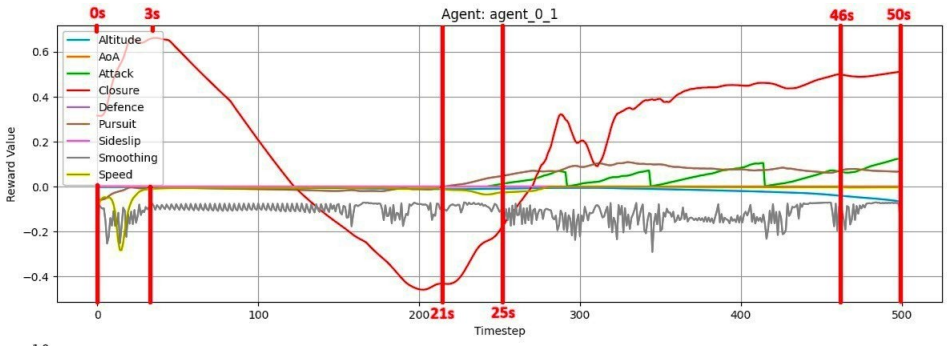

Figure 7.3: Curriculum Step 1, Sample Episode 1: Reward Plot

Episode 2: Aggressive Speed Management A second representative episode is shown in Figure 7.4, highlighting speed management across multiple engagement regimes. The adversary flies at relatively low speed, while the agent starts from a perpendicular initial orientation. The agent immediately aligns its heading toward the target and accelerates aggressively, reaching approximately 290 m/s to rapidly enter the vulnerability cone.

Upon entering the engagement zone, the agent sharply reduces speed. Although exact velocity matching is not achieved, the agent stabilizes at approximately 135 m/s while the adversary maintains 120 m/s, sustaining this regime until missile tone reaches 0.8 and a successful shot is executed.

Telemetry and reward plots for this episode are shown in Figures 7.5 and 7.6. Heading remains stable throughout the episode, while speed adjustments clearly reflect the intended closure shaping behavior. The closure reward decays as relative speed decreases, demonstrating the intended coupling between speed regulation and engagement geometry. For clarity, the plotted closure signal reflects the combined relationship between closure and alignment, rather than the fully distance-gated formulation described earlier.

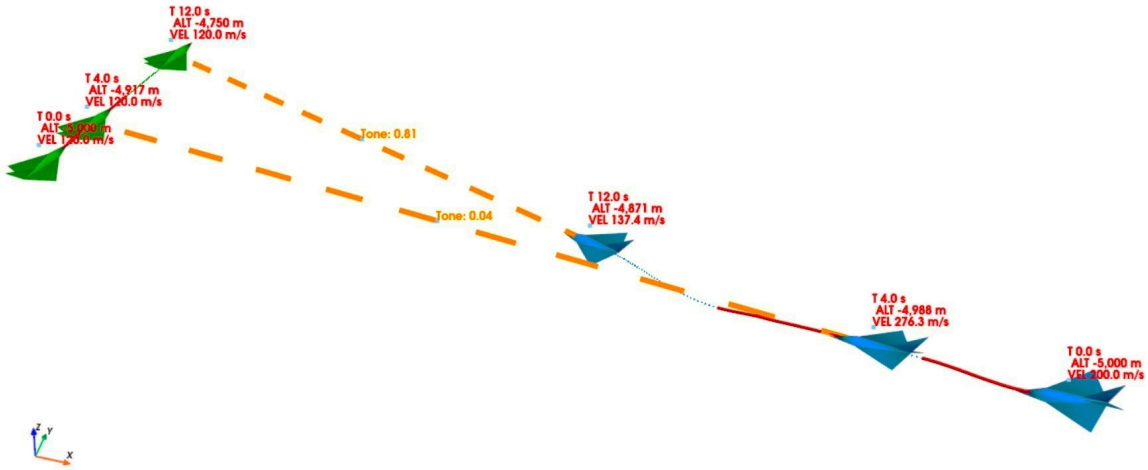

Figure 7.4: Curriculum Step 1, Sample Episode 2: Render

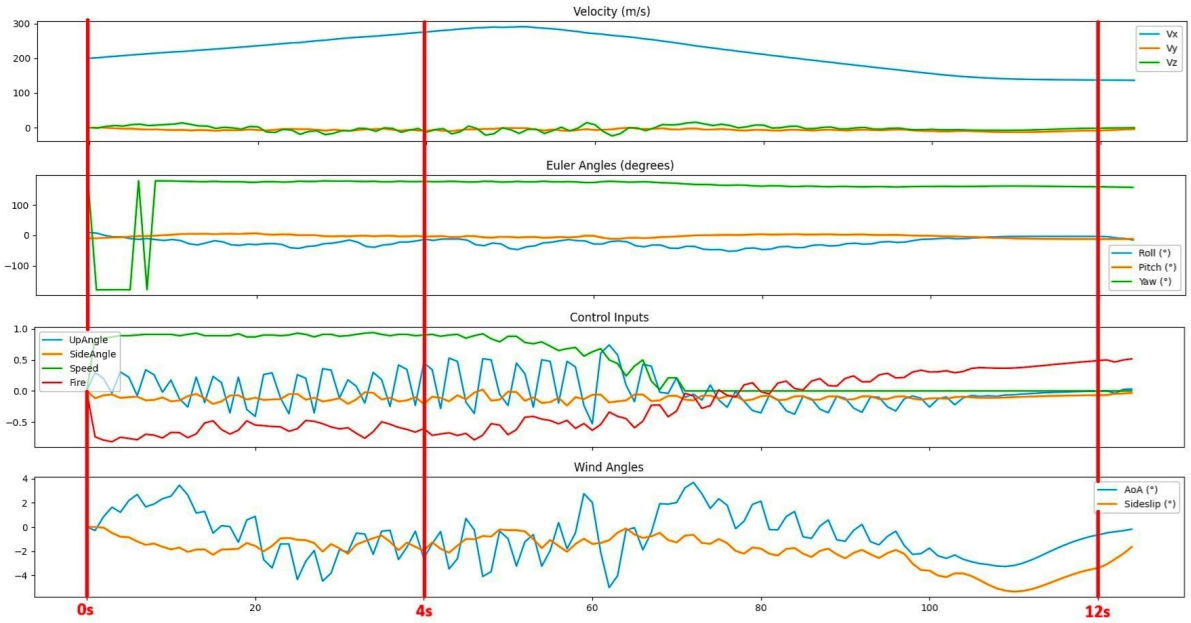

Figure 7.5: Curriculum Step 0.9, Sample Episode 2: Telemetry Plot

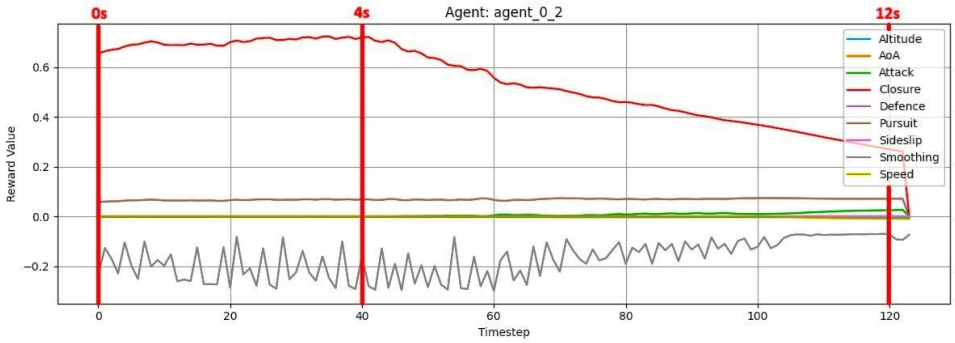

Figure 7.6: Curriculum Step 1, Sample Episode 2: Reward Plot

Episode 3: High-Speed Pursuit The third episode, shown in Figure 7.7, demonstrates pursuit at the limits of the modeled flight envelope. The agent executes a rapid banked turn at high angles of attack to reverse direction and pursue an adversary flying away at high speed. During this pursuit, the agent reaches the maximum allowed speed of 343 m/s and maintains maneuverability at this limit.

In similar scenarios, the agent often chooses to sacrifice perfect alignment to avoid risky high-speed maneuvers, compensating instead through increased missile tone accumulation. In this instance, however, the agent prioritizes alignment and fires immediately upon reaching the tone threshold. Telemetry and reward evolution for this episode are shown in Figures 7.8 and 7.9.

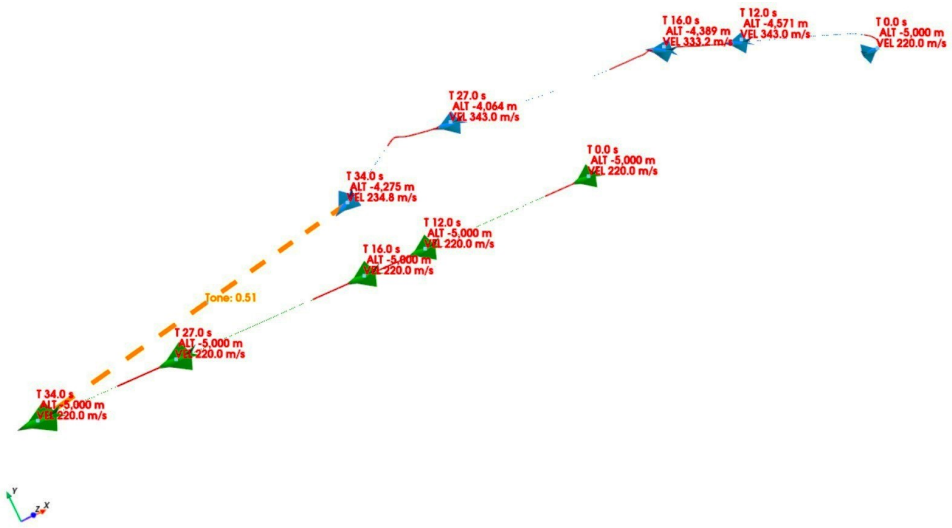

Figure 7.7: Curriculum Step 1, Sample Episode 3: Render

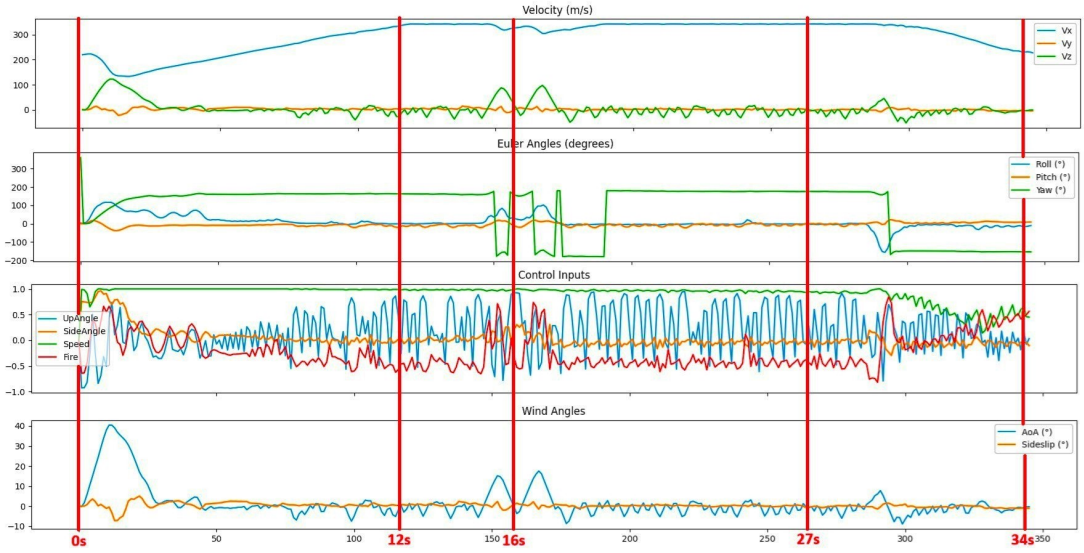

Figure 7.8: Curriculum Step 1, Sample Episode 3: Telemetry Plot

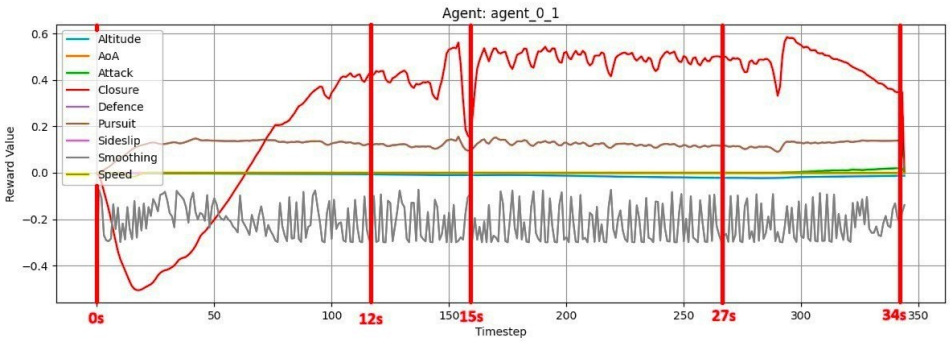

Figure 7.9: Curriculum Step 1, Sample Episode 3: Reward Plot

Training Metrics Figure 7.10 summarizes key training metrics for the first curriculum stage, including mean episodic reward, mean kills per episode, critic loss, and entropy temperature. During the first 5000 training iterations, performance remains low and critic loss is high, indicating limited understanding of the environment dynamics and reward structure. This phase also coincides with elevated entropy values, resulting in highly stochastic behavior that is particularly challenging in a control-sensitive task such as aircraft maneuvering.

As critic loss decreases, both mean reward and kill rate increase sharply before converging to a plateau. A mean kill rate of approximately 0.8 indicates that the agent successfully eliminates the adversary in eight out of ten training episodes, confirming effective learning of the core engagement behaviors during this stage.

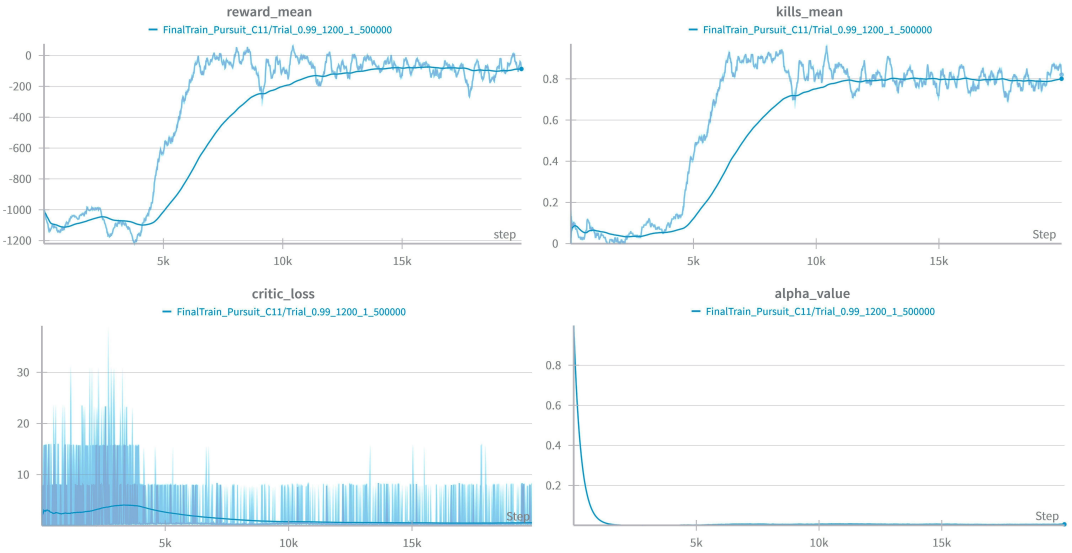

Figure 7.10: Curriculum Step 1, Critical Training Metrics

7.3.3. Step 2: Results

As discussed in the previous section, the second curriculum stage introduces a first form of randomized adversary maneuvering, requiring the agent to react to changes in the opponent's trajectory rather than pursuing a fixed direction. Evaluation episodes show that the fundamental behaviors learned in Step 1 remain embedded in the policy, but are adapted to cope with a maneuvering target.

The representative episode presented for this stage highlights the emergence of a more reactive pursuit strategy, in which the agent follows the adversary's motion instead of committing to a predefined intercept path. Two recurrent traits can be observed.

First, the agent exhibits a tendency to maintain relatively low airspeed when the target is maneuvering, often remaining slower than the adversary. Second, inspection of the reward evolution reveals that once a favorable geometric position is achieved, the closure-related reward component may decrease as the closure rate becomes negative. These two effects are closely linked. The former suggests that, in the presence of turning adversaries, the agent prefers to acquire positional advantage by "cutting the corner" rather than increasing speed and executing riskier maneuvers. The latter indicates that, in some situations, the policy accepts a temporary reduction in dense shaping rewards in order to preserve positional advantage and ultimately obtain the sparse kill reward.

As before, Figure 7.11, Figure 7.12, and Figure 7.13 provide a detailed view of the episode. It is worth noting how the *UpAngle* and *SideAngle* control signals operate at significantly lower amplitudes compared to Step 1, even during turning maneuvers. This indicates a refinement of control smoothness and a reduced reliance on aggressive steering inputs.

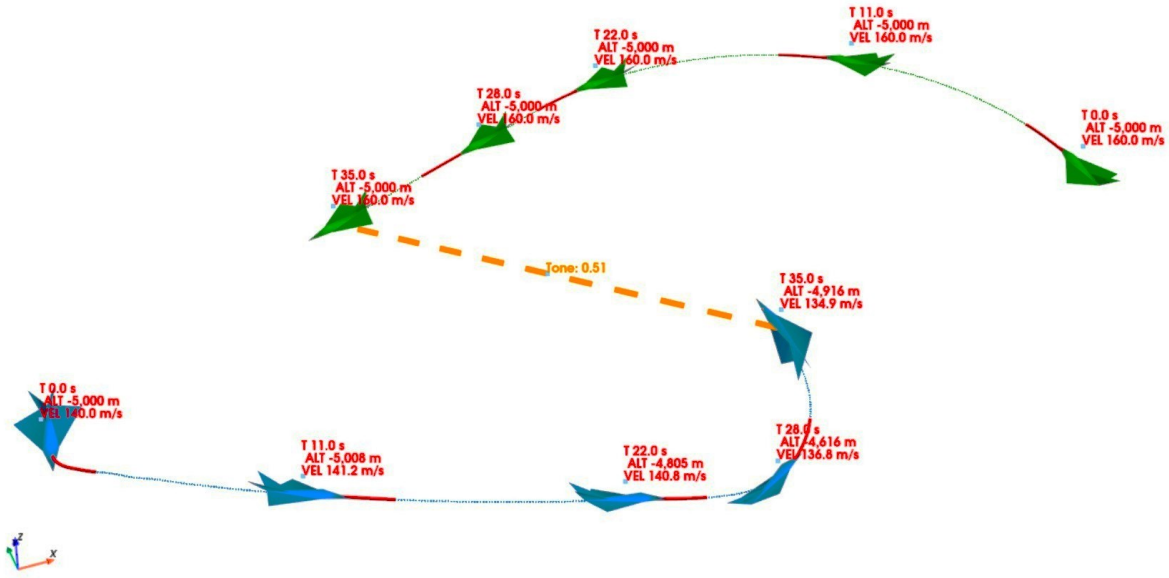

Figure 7.11: Curriculum Step 2, Sample Episode 1: Render

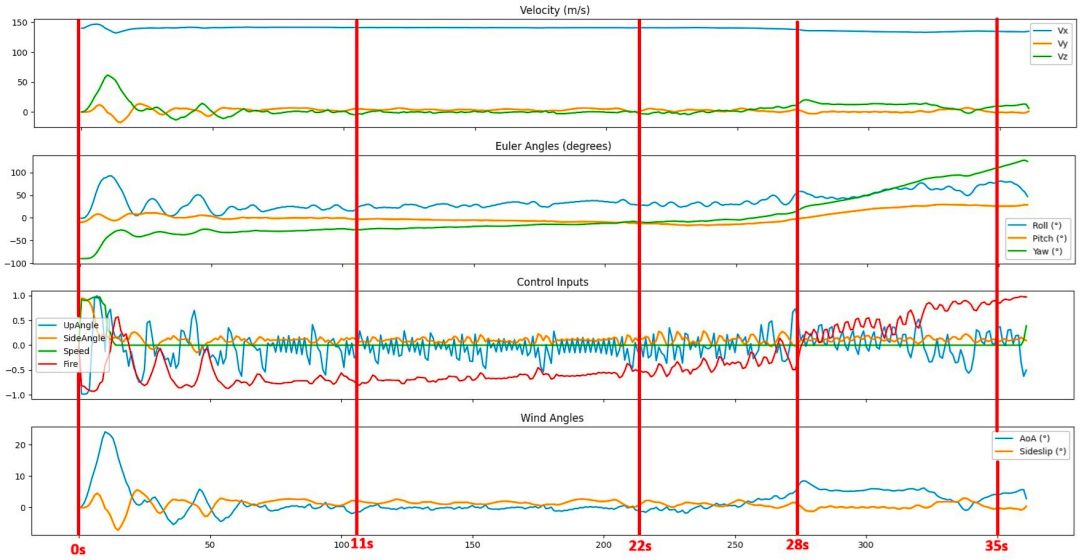

Figure 7.12: Curriculum Step 2, Sample Episode 1: Telemetry Plot

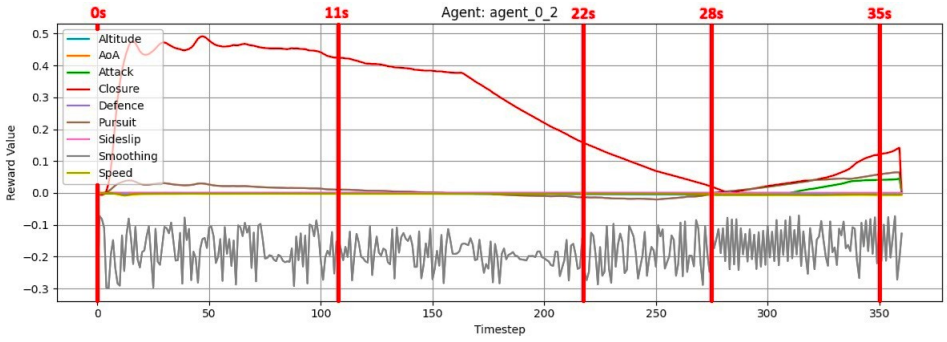

Figure 7.13: Curriculum Step 2, Sample Episode 1: Reward Plot

As a final observation for this training stage, Figure 7.14 reports the same set of critical training metrics shown for Step 1. Unlike the previous stage, performance improves sharply from the very beginning of training, the critic loss remains close to the value reached at the end of Step 1, and the entropy temperature α, restored from the previous checkpoint, initially increases before gradually decreasing. This behavior reflects an adaptive modulation of exploration driven by the increased task complexity. All these effects stem from initializing training from a policy already shaped by the previous curriculum stage.

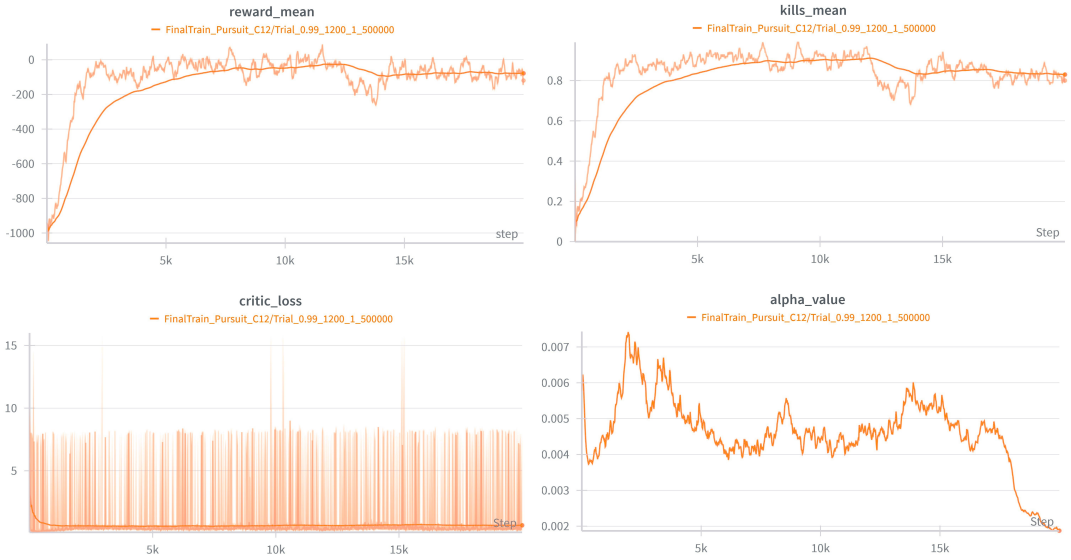

Figure 7.14: Curriculum Step 2, Critical Training Metrics

7.3.4. Step 3: Results

The third curriculum stage does not introduce fundamentally new behavioral patterns, but significantly increases the dynamism of the adversary. This results in longer episodes and more complex engagements, in which predictive maneuvering and sustained pursuit become increasingly important.

The representative episode presented for this stage illustrates how the agent refines the skills acquired previously. In particular, the agent demonstrates improved anticipation of the adversary's trajectory and smoother transitions between pursuit phases. Figures 7.15 and 7.16 highlight a characteristic situation observed multiple times during evaluation, in which the agent accumulates maximum missile tone but delays firing.

This behavior is not easily explained deterministically. One plausible interpretation is that, in the presence of highly dynamic adversary behavior, the policy adopts a conservative firing strategy, accumulating tone beyond the minimum threshold to reduce the risk of a missed shot. The occasional exaggeration of this mechanism, resulting in delayed firing despite optimal conditions, appears to be a transient artifact of policy adaptation at this stage.

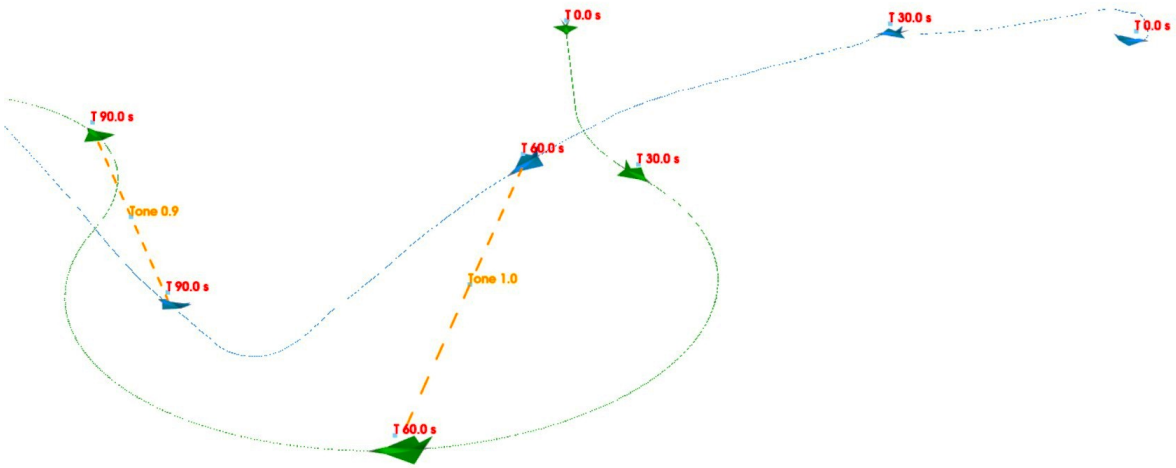

Figure 7.15: Curriculum Step 3, Sample Episode 1, Part 1: Render

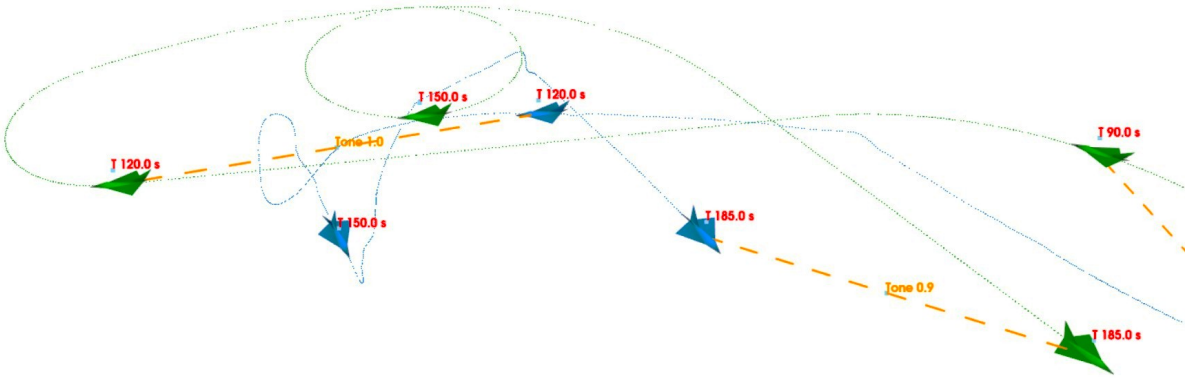

Figure 7.16: Curriculum Step 3, Sample Episode 1, Part 2: Render

7.3.5. Step 4: Results

The fourth curriculum stage represents the most challenging scripted-adversary configuration and closely approaches the conditions later encountered during self-play training. The qualitative nature of the engagements observed at this stage is largely consistent with those of Step 3, with increased variability arising from more complex initial conditions and adversary maneuvering patterns.

As a result, a full episode walkthrough is not required. Instead, two notable trends emerge. First, the delayed-trigger behavior observed in Step 3 is significantly mitigated, with the agent more consistently firing when favorable conditions are achieved. Second, overshoot-

ing errors become more frequent, occasionally placing the agent in disadvantageous or dangerous positions. In such cases, the agent either gets shot down or executes evasive maneuvers to recover geometric advantage.

Figure 7.17 shows the evolution of training metrics for this stage. Compared to Step 2, performance improves more gradually, reflecting the larger increase in task difficulty between Steps 3 and 4. The mean episode length decreases steadily as training progresses, indicating that the agent learns to resolve engagements more efficiently and with fewer prolonged pursuit phases.

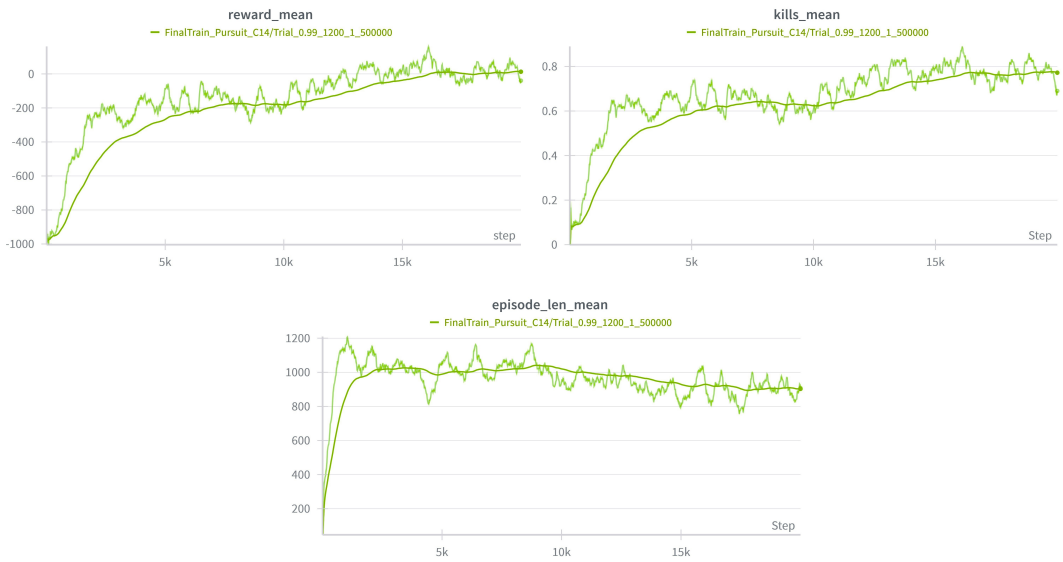

Figure 7.17: Curriculum Step 4, Critical Training Metrics

7.4. Chapter Summary

This chapter presented the curriculum learning strategy adopted in this work and analyzed the progressive emergence of air combat behaviors across successive training stages. Rather than decomposing the task into isolated sub-skills, the curriculum exposed the agent to the full structure of the engagement problem from the outset, while gradually increasing environmental complexity through adversary maneuvering and initial condition variability.

The results show that fundamental behaviors such as pursuit geometry management, closure control, and firing discipline emerge early and are refined as the curriculum progresses. Subsequent stages do not overwrite previously learned skills, but instead adapt and extend them to cope with increasingly dynamic and unpredictable adversaries. The

reuse of policy checkpoints between stages proved effective in enabling rapid performance recovery and stable learning, even under significant increases in task difficulty.

Qualitative analysis of representative evaluation episodes highlighted the agent's ability to balance aggressive maneuvering with stability constraints, to manage relative speed across different engagement regimes, and to coordinate firing decisions with missile tone and positional advantage. Observed failure modes, such as overshooting or delayed trigger activation, provided valuable insight into the evolving policy structure and informed the interpretation of later training phases.

Overall, the curriculum learning framework established in this chapter produces a robust and adaptable policy that serves as a suitable initialization for the self-play training described in the following chapter, where competitive co-evolution further refines tactical performance.

8 | Self-Play and Competitive Training

8.1. Aircraft Variant Generation

To enable meaningful self-play and comparative evaluation, a population of aircraft variants was defined by parametrically modifying a small set of physically and tactically relevant aircraft characteristics. Each variant shares the same control architecture and policy structure, but differs in mass properties, propulsion limits, aerodynamic reference dimensions, and engagement geometry.

This approach allows differences in performance to be attributed to well-defined aircraft characteristics rather than to changes in learning setup or reward design. The resulting population consists of two base models and four derived variants, each designed to emphasize a specific design trade-off such as agility, acceleration, or survivability.

8.1.1. Aircraft Variant Parameters

Tables 8.1–8.3 summarize the defining parameters of each aircraft variant. Variants are presented in pairs to highlight their relationships.

Table 8.1: Base aircraft variants: Model 0 and Model 1

Parameter	**Model 0**	**Model 1**
Mass m [kg]	5000	6000
Max thrust $T_{\max}$ [N]	60 000	120 000
Max speed $V_{\max}$ [m/s]	343	343
Max acceleration $a_{\max}$ [g]	20	20
Attack cone $\mathcal{C}_A$ [deg, m]	[120, 1000–5000]	[120, 1000–5000]
Defence cone $\mathcal{C}_D$ [deg, m]	[200, 1000–5000]	[200, 1000–5000]
Reference surface S [m^2]	88.6	118
Inertia (I_{xx}, I_{yy}, I_{zz}) [kg m^2]	$(6.8, 35.5, 41.5) \times 10^3$	$(19, 19, 22) \times 10^3$

Table 8.2: Variants derived from Model 0

Parameter	Model 2	Model 4
Mass m [kg]	4000	5000
Max thrust $T_{\max}$ [N]	60 000	80 000
Max speed $V_{\max}$ [m/s]	343	343
Max acceleration $a_{\max}$ [g]	20	20
Attack cone $\mathcal{C}_A$ [deg, m]	[120, 1000–5000]	[120, 1000–5000]
Defence cone $\mathcal{C}_D$ [deg, m]	[200, 1000–5000]	[140, 1000–5000]
Reference surface S [m^2]	75	88.6
Inertia (I_{xx}, I_{yy}, I_{zz}) [kg m^2]	$(5.8, 30, 35) \times 10^3$	$(6.8, 35.5, 41.5) \times 10^3$

Table 8.3: Variants derived from Model 1

Parameter	Model 3	Model 5
Mass m [kg]	5000	6000
Max thrust $T_{\max}$ [N]	120 000	140 000
Max speed $V_{\max}$ [m/s]	343	343
Max acceleration $a_{\max}$ [g]	20	20
Attack cone $\mathcal{C}_A$ [deg, m]	[120, 1000–5000]	[120, 1000–5000]
Defence cone $\mathcal{C}_D$ [deg, m]	[200, 1000–5000]	[140, 1000–5000]
Reference surface S [m^2]	100	118
Inertia (I_{xx}, I_{yy}, I_{zz}) [kg m^2]	$(15, 15, 17) \times 10^3$	$(19, 19, 22) \times 10^3$

8.1.2. Design Differences Between Variants

Models 0 and 1 represent two distinct base designs. Model 0 is heavier in pitch and yaw inertia and lower in thrust, favoring smoother, energy-conservative maneuvering. Model 1 exhibits higher thrust-to-weight ratio and lower inertia, enabling more aggressive acceleration and turn initiation.

Models 2 and 3 reduce mass and inertia relative to their respective base models, increasing agility and responsiveness. Models 4 and 5 instead focus on improved survivability and energy dominance by increasing thrust and reducing the defence cone aperture, thereby lowering rear-aspect vulnerability.

This structured population enables controlled self-play experiments in which performance differences can be directly linked to aircraft design choices rather than to policy architecture or training procedure.

For each aircraft variant, the low-level PID gains and actuator rate limits were recalibrated to account for differences in mass, inertia, and aerodynamic response, while the same high-level policy network trained for the corresponding base design was retained. As a result, derived variants start self-play with a slight initial disadvantage, as the policy was not explicitly trained for their exact dynamics. However, the magnitude of the parametric variations was intentionally kept limited, and empirical evaluation showed that these differences did not significantly hinder the agent's ability to engage effectively in combat. This design choice allows performance differences observed during self-play to be primarily attributed to aircraft characteristics rather than to mismatched control or policy architectures.

For each aircraft variant, the low-level PID gains and actuator rate limits were recalibrated to account for differences in mass, inertia, and aerodynamic response, while the same high-level policy network trained for the corresponding base design was retained. As a result, derived variants start self-play with a slight initial disadvantage, as the policy was not explicitly trained for their exact dynamics. However, the magnitude of the parametric variations was intentionally kept limited, and empirical evaluation showed that these differences did not significantly hinder the agent's ability to engage effectively in combat. This design choice allows performance differences observed during self-play to be primarily attributed to aircraft characteristics rather than to mismatched control or policy architectures.

8.2. TrueSkill Evaluation Method

To track the relative performance of each aircraft–policy pair throughout self-play, a statistical skill evaluation method was required. While the Elo rating system was initially considered, it was ultimately discarded in favor of the TrueSkill framework, as described in [10].

Unlike Elo, which assigns a single scalar rating to each competitor, TrueSkill models player skill as a probability distribution. Specifically, each agent i is associated with a Gaussian distribution

$$s_i \sim \mathcal{N}(\mu_i, \sigma_i^2),$$

where μ_i represents the estimated mean skill level and σ_i quantifies the uncertainty associated with that estimate. At initialization, all agents are assigned identical prior distributions with high uncertainty, reflecting the absence of performance information.

8.2.1. TrueSkill Update Mechanism

After each match, the outcome is used to update the posterior skill distributions of the participating agents. If agent i defeats agent j, the update increases μ_i and decreases μ_j, while also reducing the corresponding uncertainty terms σ_i and σ_j. The magnitude of the update depends on both the match outcome and the prior uncertainty: early results produce larger changes, while later updates become progressively smaller as confidence in the estimates increases.

A key advantage of TrueSkill is its ability to handle:

- repeated matches between the same competitors,
- uneven match schedules,
- multi-agent and team-based encounters,
- estimation of individual contribution within teams.

Although this work focuses on one-versus-one engagements, the latter capabilities were considered valuable for future extensions of the system toward coordinated multi-aircraft combat.

8.2.2. Interpretation of Skill Distributions

The final outcome of the evaluation process is a set of Gaussian skill distributions, one for each aircraft–policy pair. These distributions are typically visualized as overlapping curves on a common axis.

In this representation:

- the mean μ indicates the most likely skill level of the agent;
- the standard deviation σ represents uncertainty in that estimate;
- limited overlap between two distributions indicates strong statistical confidence in a performance difference;
- significant overlap suggests that the relative ordering is uncertain.

Importantly, TrueSkill does not aim to predict the outcome of a single match, but rather to estimate long-term relative strength. An agent with slightly lower μ but significantly smaller σ may be considered more reliably evaluated than an agent with higher but uncertain skill.

For ranking purposes, a conservative skill estimate is often used,

$$s_i^{\text{cons}} = \mu_i - k\sigma_i,$$

with $k \in [2, 3]$, providing a lower confidence bound on the agent's true ability. This metric is particularly useful when comparing agents with different numbers of matches played.

8.2.3. Sample Size Considerations

The accuracy of TrueSkill estimates depends on both the number of observed matches and the structure of those matches. As a Bayesian rating system, TrueSkill progressively reduces posterior uncertainty as additional outcomes are incorporated, with the variance parameter σ serving as an explicit measure of confidence in the estimated skill.

In one-versus-one settings, experience from prior applications indicates that a number greater or equal to 12 matches is typically sufficient to achieve a stable ordering, provided that opponents are reasonably balanced and that match outcomes are informative. In this work, the number of evaluation matches after each round of training was set to 15 for one-versus-one engagements. This proved sufficient to significantly reduce posterior uncertainty and to yield consistent relative rankings across successive evaluation phases.

TrueSkill naturally extends to team-based scenarios, such as two-versus-two engagements, which are explored in the following chapter. In such settings, convergence of individual skill estimates required more than 20 matches according to TrueSkill documentation, and this was indeed the reference used in this work for two-versus-two engagements

8.3. Champion–Challenger Training Loop

The self-play experiments conducted in this work, both in one-versus-one and two-versus-two settings, are organized around a champion–challenger training architecture. The objective of this loop is to drive competitive co-evolution among aircraft–policy pairs while ultimately identifying the most effective combination.

Training is performed over a total of 150,000 iterations, subdivided into rounds of 5,000 training iterations each. During each round, two aircraft–policy entities are selected and trained against one another in self-play, with both agents actively updating their policies in response to the current adversary.

At the end of each round, a fixed number of evaluation episodes is executed using frozen policy parameters. The outcomes of these evaluation matches are used to update the

TrueSkill ratings of the competing entities. The aircraft–policy pair with the higher posterior skill estimate is retained as the *champion* for the subsequent round.

The opposing entity, referred to as the *challenger*, is sampled from a pool of previously trained aircraft–policy pairs. Sampling is performed according to a probability distribution biased by TrueSkill ratings, such that higher-skill entities are more likely to be selected, while lower-skill or older variants retain a non-zero probability of participation. This mechanism ensures that the champion is regularly exposed to competitive opponents, fostering continuous refinement of tactics, while also preventing catastrophic forgetting by occasionally revisiting simpler or outdated strategies.

Through successive rounds of competition, evaluation, and replacement, this process induces a selective pressure favoring increasingly effective aircraft–policy pairs. The final champion emerging from this loop represents the best-performing configuration identified by the self-play framework, which constitutes the primary objective of the training methodology developed in this work.

8.4. Results

The outcome of the self-play training process is summarized in Figure 8.1, which reports the final TrueSkill posterior distributions for the aircraft–policy pairs retained at the end of training. Each curve represents a Gaussian belief over the latent skill of a given variant, inferred from the sequence of evaluation matches conducted during the champion–challenger loop. In the legend, variants are identified by aircraft model index and checkpoint identifier (e.g., `P0`, `P1`).

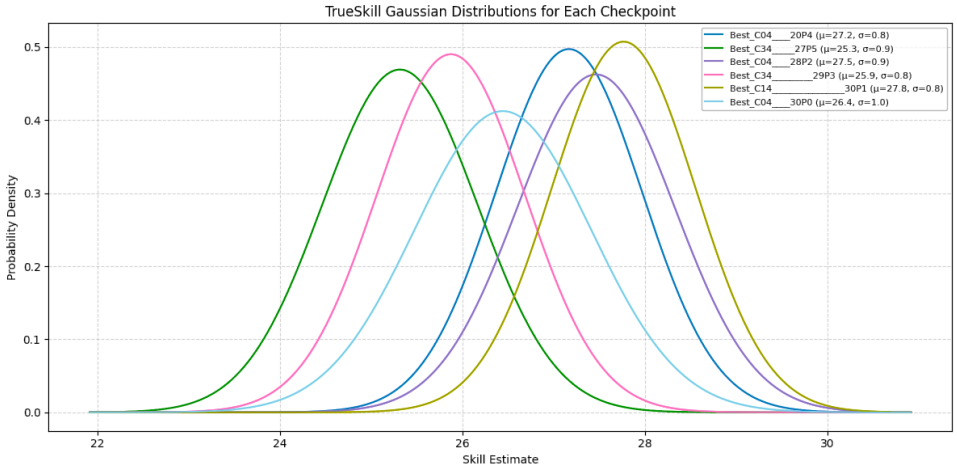

Figure 8.1: Final TrueSkill posterior distributions for the retained aircraft–policy pairs after self-play training.

As expected in a stochastic and highly dynamic domain such as close-range air combat, the resulting TrueSkill posterior distributions exhibit substantial overlap. This overlap reflects both the inherent variability of engagement outcomes and the fact that all retained aircraft–policy variants reach a broadly comparable level of tactical competence after extensive self-play training. For this reason, the TrueSkill estimates are not interpreted as defining a strict total ordering among variants.

Nevertheless, meaningful relative performance trends can still be identified. In particular, comparisons between variants located at the extremes of the skill range, as well as contrasts between policies derived from the same aircraft model at different stages of training, provide useful insight into the effects of continued self-play refinement.

Qualitative inspection of the evaluation episodes supports the overall picture suggested by Figure 8.1. The aircraft–policy combinations with the highest mean skill values are consistently observed to exhibit more stable maneuvering behavior and greater robustness across engagements. At the same time, their empirical win–loss statistics remain similar, which explains the significant overlap of their posterior skill distributions. In contrast, where the overlap between distributions is reduced—most notably between the best and worst performing combinations—the differences in observed outcomes become more pronounced.

It is important to note that a large fraction of engagements do not end with the destruction of one aircraft by missile fire. Instead, many episodes terminate due to stalls induced by extreme angle-of-attack maneuvers or through impacts with the boundaries of the simulation envelope. These outcomes are nonetheless informative with respect to skill, as they reflect the agent's ability to balance aggressive maneuvering with flight envelope constraints.

In this regard, the two lowest-performing aircraft–policy combinations correspond to extreme variants of the already highly responsive base Model 1. When controlled effectively, this model family yields some of the best overall performance; however, in these particular variants the increased dynamic responsiveness leads to frequent instability and stall. In other cases, likely as a reaction to this instability, the policy adopts overly conservative maneuvering strategies, resulting in easy engagement opportunities for the opponent and a higher probability of being shot down.

Interestingly, stalls and envelope violations do not appear to be attributable to a single aircraft or policy in isolation. Instead, certain match-ups consistently produce less stable engagements than others, suggesting that these failure modes are often triggered by the interaction between the behaviors of both aircraft. This observation reinforces the inter-

pretation of skill as a relative, interaction-dependent property rather than an absolute measure of policy quality.

From the perspective of tactical evolution, further insight is provided by the evolution of the `kill_mean` metric over the course of self-play training (see Figure 8.2).

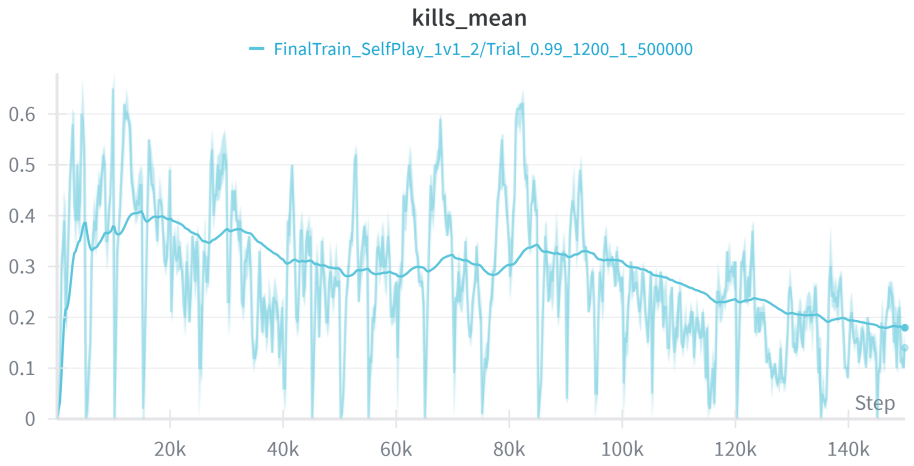

Figure 8.2: Kills-episodes proportion metric evolution during Self-Play training run

The proportion of engagements ending in a missile kill decreases from approximately 0.5 in the early stages of self-play to around 0.3 in later stages. This trend is consistent with an evolutionary pressure toward more extreme and aggressive maneuvering, where engagements are increasingly resolved through envelope violations rather than clean firing opportunities.

Overall, the evaluation suggests that, while clear-cut dominance between variants is rare, distinct behavioral regimes and failure modes emerge consistently across match-ups. For this reason, the remainder of this section focuses on three complementary analyses. First, common behavioral trends and failure modes observed across variants are identified and discussed. Second, the tactical evolution of the best-performing aircraft–policy pair is examined in detail. Finally, the limitations of the current training and modeling framework are discussed, together with directions for future improvements.

8.4.1. Common Behavioral Trends and Failure Modes

Across all evaluated match-ups and aircraft–policy combinations, the observed engagements consistently converge toward a small number of equilibrium interaction patterns. In the absence of large performance asymmetries or decisive execution errors, these equi-

libria tend to trap the engagement in prolonged maneuvering phases, from which resolution occurs only through mistakes, extreme maneuvers, or termination due to envelope violations.

Remarkably, the most frequently observed equilibria correspond closely to well-known real-world Basic Fighter Maneuvers (BFM). Before discussing these behaviors in detail, it is important to note that the engagement rules adopted in this work occupy an intermediate regime between classical gun-only dogfighting and missile-based beyond-visual-range combat. While the modeled weapon dynamics are missile-inspired, the requirement to maintain sustained geometric advantage resembles gun engagements. As a consequence, certain BFMs that originate from gun-only doctrine emerge clearly, while others appear less frequently or are associated with higher risk.

The most prevalent convergent behavior corresponds to the classical *two-circle fight*, illustrated in Figure 8.3.

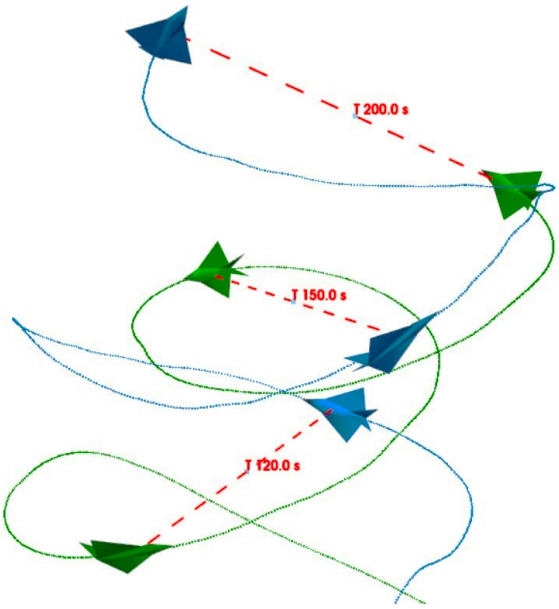

Figure 8.3: Example of two-circle fight behavior in evaluation episode

In this configuration, both aircraft turn in opposite directions around a common center, emphasizing sustained turn rate and lateral separation.

When one aircraft begins to gain a positional advantage, several responses are theoretically available:

- increasing angle of attack to reduce turn radius,

- introducing a vertical component to exchange altitude for speed or vice versa,
- breaking the turn and attempting a reversal.

Each option involves trade-offs. Steeper turning increases drag and risks energy depletion if thrust is insufficient. Vertical maneuvering sacrifices altitude or kinetic energy and often leads to large excursions in the flight envelope. Turn reversals, while viable in gun-only engagements, are particularly risky in the present setting, as they temporarily expose the aircraft to the opponent's forward firing zone.

These considerations are clearly reflected in the evaluation episodes. Two-circle encounters most often evolve through increased angle-of-attack or vertical maneuvering, while attempted reversals frequently result in the rapid destruction of the aircraft. Vertical maneuvers in particular are a common precursor to another frequent termination mode: collision with the upper or lower bounds of the simulation envelope.

A second dominant equilibrium corresponds to the *one-circle fight*, shown in Figure 8.4. In this configuration, the two aircraft repeatedly meet in head-on passes, separate through coordinated turns in the same direction, and re-enter another head-on merge.

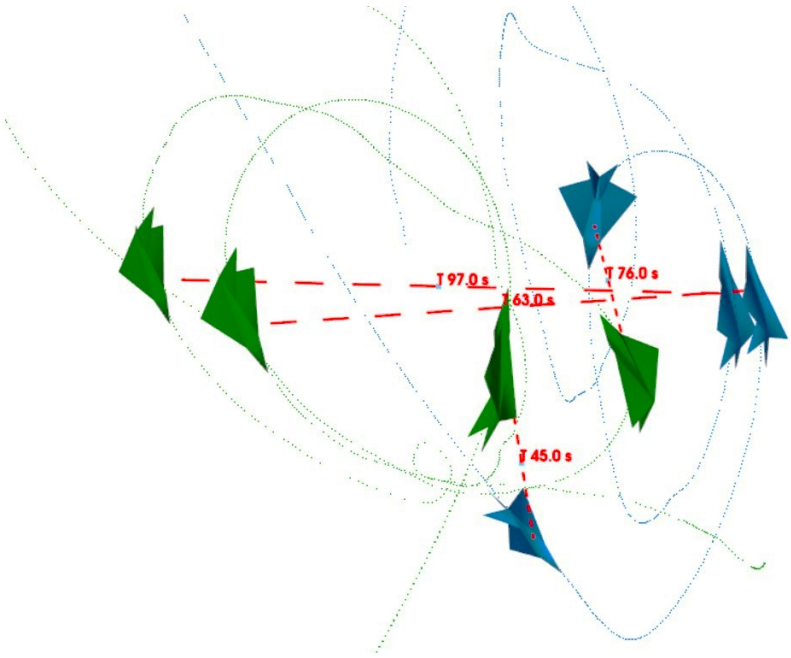

Figure 8.4: Example of one-circle fight behavior in evaluation episode

As with two-circle engagements, breaking this equilibrium requires committing to aggressive radius-minimizing maneuvers, vertical displacement, or reversals, each of which introduces risk. In the evaluated episodes, one-circle engagements frequently persist for

extended durations, gradually driving both aircraft toward more extreme maneuvering conditions.

Both one- and two-circle equilibria induce an evolutionary pressure toward increasingly aggressive control actions. A common failure mode resulting from this pressure is termination due to stall.

Notably, despite the frequent convergence toward low-speed engagements, stalls are rarely triggered by insufficient airspeed. Instead, they are predominantly caused by excessive angle of attack during highly aggressive turning maneuvers.

Finally, a third BFM-like behavior was observed, albeit far less frequently: the *scissors*. In the example shown in Figure 8.5, the engagement takes the form of a flat scissors, where both aircraft cross in a head-on pass and one performs a rapid turn inversion. If the opponent fails to execute a timely counter-reversal, the maneuvering aircraft gains immediate access to the adversary's rear hemisphere. When both aircraft repeatedly execute such reversals, the interaction evolves into a double scissors.

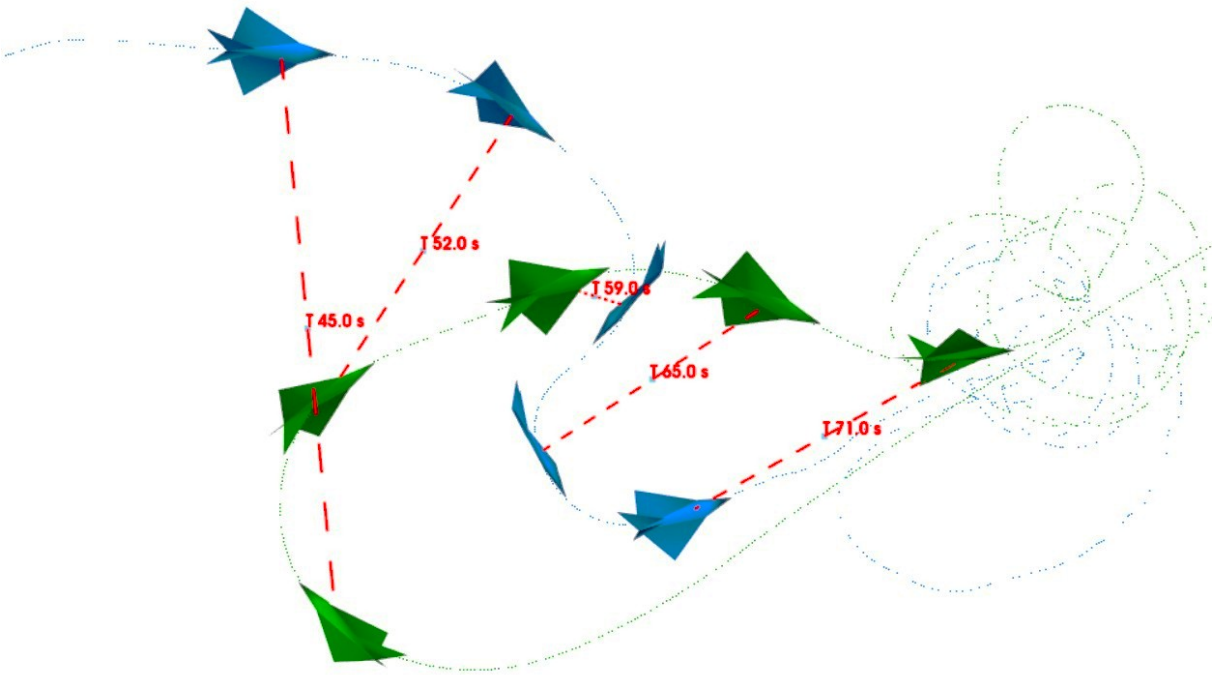

Figure 8.5: Example of flat scissors behavior in evaluation episode

The spontaneous convergence toward classical BFM patterns is a significant result of this work. It indicates that, despite the absence of any explicit encoding of air combat doctrine,

the combination of observation design, control architecture, and reward structure leads to equilibria that closely mirror real-world fighter maneuvering strategies.

At the same time, the ubiquity of these behaviors across aircraft variants suggests that the performance differences between models are not sufficiently large to fundamentally alter high-level tactical decisions. Instead, they manifest primarily through stability margins and susceptibility to failure modes, rather than through qualitatively distinct engagement strategies.

As a final note on common behavior convergence, it is worth mentioning an emergent defensive maneuver induced by the chosen engagement rules. By maintaining a sufficiently tight and continuous turn, the defending aircraft prevents the adversary from remaining within the vulnerability zone long enough to accumulate the required missile tone. Although the attacker repeatedly achieves momentary geometric advantage, firing opportunities are denied due to the temporal nature of tone accumulation. This behavior highlights how the engagement logic promotes sustained defensive maneuvering rather than instantaneous evasive actions.

8.4.2. Tactical Evolution of the Best Aircraft–Policy Pair

To illustrate how tactical competence evolves throughout the self-play training process, this section presents representative encounters sampled from early, intermediate, and late stages of training. All examples correspond to match-ups between the best-performing aircraft–policy combination at the end of training (aircraft model 1) and a fixed adversary (aircraft model 0), allowing changes in behavior to be attributed primarily to policy refinement rather than opponent variability.

Figure 8.6 shows an early-stage engagement. Although the policy already exhibits a basic understanding of offensive and defensive maneuvering, its reactions are slow and hesitant. Control inputs are conservative, and evasive actions do not yet organize into recognizable one-circle or two-circle fight patterns. Instead, maneuvering appears reactive rather than anticipatory, with large temporal gaps between opponent actions and corresponding responses.

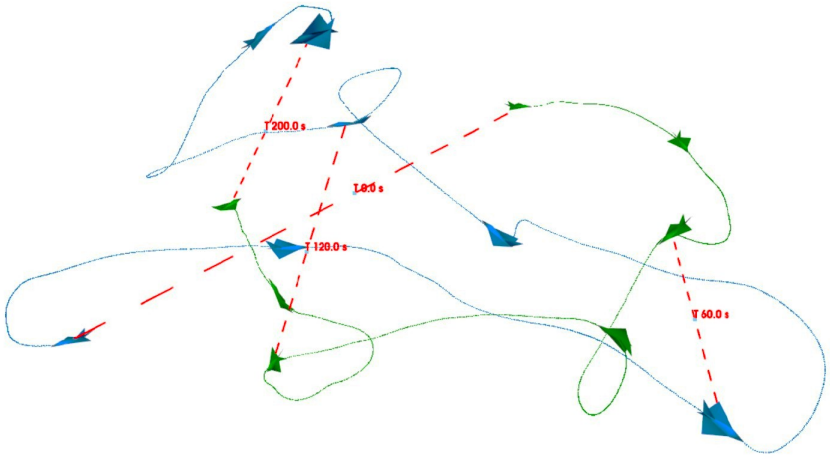

Figure 8.6: Example of early training behavior in evaluation episode

A markedly different interaction emerges in the intermediate-stage example shown in Figure 8.7. Here, the engagement begins with an explicit attempt to force a head-on merge, followed by the formation of a two-circle fight. Both aircraft maneuver continuously and decisively to maintain angular separation and deny tail positioning. However, as the engagement unfolds, this equilibrium degenerates: one aircraft breaks the circular engagement, while the opponent fails to exploit the resulting opportunity and instead transitions into a tight defensive circling maneuver. This behavior mirrors the defensive equilibrium discussed in the previous section, where positional denial is favored over aggressive pursuit.

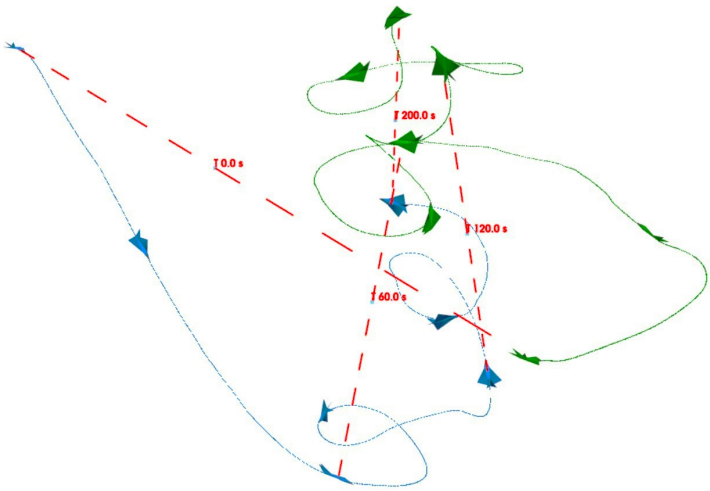

Figure 8.7: Example of mid-training suboptimal behavior in evaluation episode

The late-stage encounter presented in Figure 8.8 demonstrates a further refinement of tactical timing and responsiveness. Both policies consistently employ canonical BFM patterns, with timely attempts to break stalemates and equally prompt counteractions. Unlike earlier stages, these attempts are neither delayed nor overly conservative, resulting in prolonged, stable engagements in which neither aircraft can decisively capitalize on momentary advantages. The encounter thus converges to a dynamic equilibrium characterized by sustained maneuvering rather than abrupt resolution.

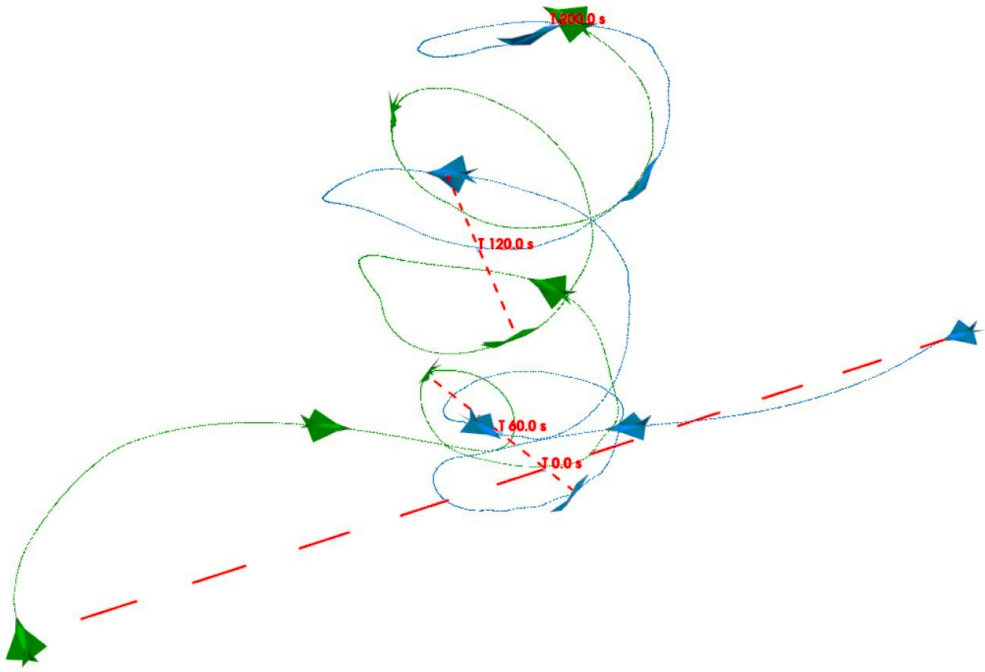

Figure 8.8: Example of converged, late training behavior in evaluation episode

This pattern of convergence was observed across aircraft variants and match-ups. It reinforces the conclusions drawn in the previous section: the environment and reward formulation encourage the emergence of realistic air-combat equilibria, while relatively small differences in aircraft performance are insufficient to decisively break these equilibria without substantial tactical asymmetry.

8.4.3. Conclusions, Limitations, and Future Improvements

The self-play experiments presented in this chapter demonstrate that a carefully defined close-range air-combat task, constrained to subsonic flight and a bounded operational envelope, can lead to the spontaneous emergence of behaviors closely resembling real-world Basic Fighter Maneuvers. Despite the absence of explicit maneuver templates or

scripted tactics, the learned policies converge toward well-known engagement equilibria, validating both the environment design and the training methodology adopted in this work.

At the same time, these results highlight an important limitation. Because the aircraft variants considered exhibit only a moderate difference in performance, tactical disparities are often mitigated by compensatory control strategies learned by the policies. As a consequence, engagements frequently settle into prolonged equilibria rather than being decisively resolved through superior kinematic capability alone.

Future extensions of this work could explore more asymmetric scenarios. In particular, contrasting an unmanned aircraft optimized for extreme maneuverability against an opponent constrained by human physiological limits—especially with respect to sustained acceleration—could yield qualitatively different equilibria. Additionally, expanding the modeled flight envelope to include supersonic regimes and higher operational ceilings, potentially incorporating engine performance degradation or airflow limitations, may fundamentally alter the nature of the converged tactics.

Such extensions would not only improve realism but also provide further insight into how aircraft performance envelopes and engagement rules shape the space of viable air-combat strategies.

9 | Multi-Agent Competitive Training

9.1. Multi-Agent 2 vs 2 Self-Play Setup

The final phase of the experimental campaign extends the self-play framework from one-versus-one engagements to a two-versus-two air combat scenario. Compared to the self-play configuration described in the previous chapter, the environment and learning setup remain largely unchanged, with two main differences.

First, the number of active aircraft per team at episode start is increased from one to two. Second, the number of aircraft variants considered is reduced from six to three, in order to limit combinatorial complexity and to focus the analysis on qualitative behavioral trends rather than exhaustive ranking. All other implementation and configuration aspects—including training round length, TrueSkill evaluation, matchmaking strategy, and the overall tournament co-evolution loop—are kept identical to the one-versus-one setting.

Importantly, no explicit cooperation mechanisms are introduced. Agents do not share observations, internal states, or communication channels, and the reward function does not include any term that explicitly incentivizes coordination, role assignment, or spatial separation between teammates. Each aircraft is controlled by an independent policy instance trained via self-play against the opposing team.

This configuration enables the investigation of whether coordinated or cooperative behaviors can emerge purely from shared objectives and interaction dynamics, without the introduction of explicit multi-agent cooperation incentives.

9.2. Observed Emergent Behaviors

From a quantitative perspective, the three aircraft variants considered in this phase (model 0, model 1, and the extreme variant derived from model 1) do not exhibit a

clear skill hierarchy. The corresponding TrueSkill posterior distributions show extensive overlap, indicating comparable overall performance across variants.

Qualitative inspection of evaluation episodes is consistent with this observation. Across match-ups, the different variants exhibit largely similar tactical behaviors, with no persistent or systematic advantages emerging for a specific aircraft model. At the same time, the two-versus-two self-play setting is characterized by a high degree of variability and noise. Individual engagements often diverge significantly depending on initial conditions, aircraft pairing, and early maneuvering decisions.

Despite this variability, a small set of recurring behavioral patterns emerges consistently across a large portion of the evaluated encounters. The most prominent of these behaviors, which becomes clearly visible after the midpoint of training and persists through later stages, is a tendency for aircraft belonging to the same team to converge spatially and form loose line-astern configurations, effectively flying in queues.

In many episodes, both agents of a team end up committing to the same opponent, resulting in close proximity and partial alignment of their flight paths. This pairing behavior does not appear to be rigid or pre-planned. Rather, it emerges dynamically during the engagement and may dissolve or reform as the situation evolves.

Explicit division of roles—such as one aircraft maintaining a purely defensive or support position while the other attacks—is not consistently observed. Nevertheless, the frequent occurrence of temporary leader–wingman-like configurations, together with the observable evolution of these behaviors from early to late training iterations, suggests a non-random structure in the learned policies. Representative examples of these dynamics can be observed in evaluation episodes from early and late stages of training.

9.3. Interpretation: Emergent Wingman-Like Dynamics

From a qualitative standpoint, the observed queuing behavior is compatible with simplified forms of real-world two-versus-two Basic Fighter Maneuvers. In operational air combat, fully coordinated bracket or sandwich maneuvers require explicit communication, training, and role assignment. However, temporary line-astern alignment and target convergence are commonly observed in opportunistic or reactive engagements, particularly when both aircraft commit to the same adversary.

In this sense, the emergent behavior observed in the simulations can be interpreted as a

rudimentary form of wingman dynamics. One aircraft often acts as the primary engager, while the second follows a similar trajectory and remains in close support, occasionally providing implicit coverage of the teammate's rear hemisphere. This occurs despite the absence of any enforced leader–wingman hierarchy or coordination mechanism in the learning setup.

At the same time, a purely mechanistic explanation remains plausible. Given the shared reward structure and symmetric observations, both agents are naturally driven toward the same high-reward regions of the state space. Random initial motion can bring teammates into proximity early in the engagement, and subsequent pursuit and maneuvering toward a common target further reinforces spatial alignment. In the absence of penalties for proximity or incentives for role diversity, policy gradients may therefore favor convergence rather than dispersion.

Figure 9.1 shows a moment in evaluation episode that most clearly exemplify the leader–wingman formation for both teams, with the two pairs engaged into a two-circle fight situation.

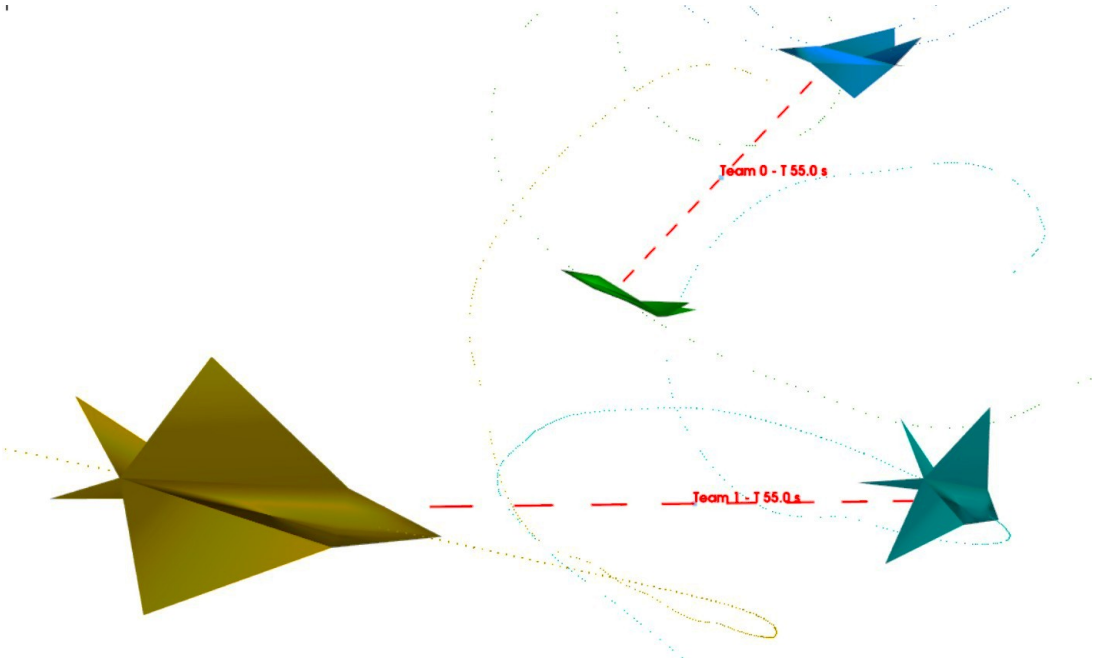

Figure 9.1: Queue formation in two-circle fight maneuver in evaluation episodes

As a result, while the observed behaviors are compatible with basic cooperative patterns seen in real air combat, they cannot be conclusively attributed to deliberate coordination or emergent teamwork. Instead, they are best interpreted as the outcome of shared objectives and interaction dynamics in a symmetric multi-agent learning environment.

9.4. Limitations and Ambiguity of Multi-Agent Results

The two-versus-two self-play results are subject to several important limitations. First, the stochastic nature of the environment and the increased complexity of multi-agent interactions lead to high variance in engagement outcomes, making it difficult to extract stable quantitative performance rankings between aircraft variants or policies.

Second, while spatial convergence and queuing behaviors are frequently observed, more structured cooperative tactics—such as sustained role specialization, explicit support maneuvers, or coordinated target switching—do not consistently emerge. This suggests that the current reward formulation and observation model are insufficient to reliably induce higher-level teamwork.

Finally, the absence of explicit coordination mechanisms makes it inherently difficult to distinguish intentional cooperative behavior from incidental alignment driven by shared incentives. For these reasons, the results of this chapter are best interpreted as exploratory and qualitative, rather than as definitive evidence of learned cooperation.

9.5. Discussion and Future Directions

Despite the above limitations, the multi-agent experiments provide valuable insight into how structured air combat behaviors can emerge from relatively simple learning rules. The tendency of agents to form loose queues and converge on common targets demonstrates that shared objectives alone can produce non-trivial interaction patterns, even in the absence of explicit coordination mechanisms. In this sense, the experiments also serve as a validation of the flexibility and expressive power of the simulation and training framework developed in this work.

Future extensions could build on this foundation by introducing explicit cooperation incentives, such as rewards for spatial coverage, role differentiation, or mutual support. Alternative approaches may include asymmetric policy training, explicit leader–wingman role assignment, or limited communication channels between agents.

10 | Conclusions and Future Work

10.1. Conclusions

This thesis presented the design, implementation, and evaluation of a unified simulation and learning framework for close-range air combat, with the explicit goal of enabling joint exploration of aircraft characteristics, control architectures, and reinforcement-learning-driven tactical policies. Rather than focusing solely on the performance of a single trained agent, the work was conceived as an engineering tool for studying how aircraft dynamics, control abstractions, reward formulations, and learning dynamics interact to shape emergent air combat behavior.

At the foundation of the framework lies a custom six-degree-of-freedom aircraft dynamics simulator with parametric aerodynamic and propulsion modeling. By deriving multiple aircraft variants from a shared modeling pipeline and embedding them in a common environment, the framework allows controlled comparison between platforms while preserving physical consistency. This capability is essential for disentangling the effects of vehicle design choices from those of policy architecture or training procedure, and positions the simulator as a reusable component for aircraft-centric learning experiments.

A central contribution of this work is the hierarchical control architecture that decouples tactical decision-making from low-level stabilization. By introducing a geometric action translation layer between the reinforcement learning policy and classical PID controllers, the framework enables policies to operate in a physically meaningful action space aligned with engagement geometry. This separation significantly improves training stability and interpretability, while preserving responsiveness and robustness across different aircraft variants. More broadly, it demonstrates how classical control and learning-based decision-making can be integrated into a coherent stack suitable for complex, safety-critical maneuvering tasks.

The reinforcement learning environment was designed to encode air combat geometry ex-

plicitly through invariant, body-frame-relative observations and a reward structure that couples pursuit geometry, closure management, and firing discipline. This design was critical in enabling the emergence of tactically meaningful behaviors without explicit encoding of air combat doctrine. Through curriculum learning, agents reliably acquired foundational skills such as controlled intercepts, overshoot avoidance, velocity matching, and disciplined missile employment. Subsequent self-play training further refined these skills through competitive co-evolution, exposing agents to increasingly adaptive adversaries.

One-versus-one self-play experiments demonstrated that the framework consistently induces engagement patterns closely resembling classical Basic Fighter Maneuvers, including one-circle and two-circle fights, scissors, and sustained defensive circling. Importantly, these behaviors emerged spontaneously from the interaction between aircraft dynamics, reward shaping, and learning, rather than from scripted tactics. Comparative evaluation across multiple aircraft variants showed that moderate differences in platform characteristics tend to manifest through stability margins and susceptibility to failure modes, rather than through qualitatively distinct tactical strategies. This highlights the role of the framework as a tool for analyzing relative performance envelopes and interaction-driven outcomes, rather than producing absolute rankings.

The extension to two-versus-two self-play further demonstrated the scalability of the framework to multi-agent scenarios. Even in the absence of explicit coordination mechanisms, agents exhibited non-trivial interaction patterns, such as temporary wingman-like spatial configurations and joint commitment to common targets. While these behaviors cannot be conclusively interpreted as deliberate cooperation, their emergence illustrates how shared objectives and interaction dynamics alone can produce structured multi-agent behavior. This result reinforces the suitability of the framework for exploratory studies in multi-agent air combat, while also exposing clear limitations of symmetric, coordination-free training setups.

Taken together, the results of this thesis demonstrate that reinforcement learning can be effectively embedded within a carefully engineered simulation and control framework to produce interpretable, tactically relevant air combat behaviors. Beyond individual trained policies, the primary outcome of this work is a flexible experimental platform for joint aircraft design and tactical policy exploration, capable of supporting controlled studies across vehicle configurations, engagement rules, and learning paradigms.

10.2. Future Work

The framework developed in this thesis is intended as an extensible tool for joint aircraft–policy design and tactical learning experiments. Building on the current results, three development routes appear both realistic and high-impact.

Increased physical realism and expanded flight envelope. A first direction is to extend the simulator beyond the current steady-state, subsonic operating envelope. Priority improvements include explicit stall and post-stall modeling (instead of handling these regimes only through termination constraints), more realistic control-surface effectiveness degradation at high angles, and a richer propulsion model (e.g., thrust limits and performance variation with speed/altitude). Extending the valid regime toward higher altitudes and transonic/supersonic conditions would enable the study of energy-management tactics that are currently outside scope.

Stronger asymmetries between contenders. A second direction is to explore deliberately asymmetric match-ups where aircraft constraints are fundamentally different. A particularly relevant case is contrasting unmanned platforms that can sustain extreme maneuvering with opponents constrained by human physiological limits (e.g., sustained g tolerance and recovery constraints). Introducing these asymmetries at the aircraft-model and/or rules level would allow the framework to investigate how doctrine and optimal tactics shift when one side can exploit maneuvers that are infeasible for a human pilot.

Multi-agent cooperation mechanisms. Finally, the two-versus-two experiments suggest that shared objectives alone can induce non-trivial interaction patterns, but not reliable teamwork. Future work should therefore introduce explicit coordination mechanisms, such as team-level reward terms for spatial coverage and mutual support, role differentiation (leader–wingman), or limited communication channels. This would enable the framework to move from incidental alignment toward reproducible cooperative tactics in team combat scenarios.

Bibliography

[1] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. URL https://doi.org/10.1145/1553374.1553380.

[2] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. de Oliveira Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang. Dota 2 with large scale deep reinforcement learning. *CoRR*, abs/1912.06680, 2019. URL http://arxiv.org/abs/1912.06680.

[3] J. Chai, W. Chen, Y. Zhu, Z.-X. Yao, and D. Zhao. A hierarchical deep reinforcement learning framework for 6-dof ucav air-to-air combat. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 53(9):5417–5429, 2023. doi: 10.1109/TSMC.2023.3270444.

[4] S. Clapp. Defense and artificial intelligence. Technical report, European Parliament, Panel for the Future of Science and Technology (STOA), 2025. URL https://www.europarl.europa.eu/RegData/etudes/BRIE/2025/769580/EPRS_BRI%282025%29769580_EN.pdf. Briefing document.

[5] J. DiMascio. U.s. air force collaborative combat aircraft (cca). Technical Report IF12740, Congressional Research Service, Washington, D.C., November 28 2025. URL https://www.congress.gov/crs-product/IF12740.

[6] S. Farì. Guidance and control for a fixed-wing UAV. Master's thesis, Politecnico di Milano, 2017. URL https://hdl.handle.net/10589/137455. M.S. thesis.

[7] R. Grant. The radar game, 2010.

[8] P. Hernández-Leal, B. Kartal, and M. E. Taylor. A survey and critique of multiagent

deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6): 750–797, 2019. doi: 10.1007/s10458-019-09421-1.

[9] MBDA. Meteor beyond visual range air-to-air missile. URL `https://www.mbda-systems.com/product/meteor/`.

[10] T. Minka, R. Cleven, and Y. Zaykov. Trueskill 2: An improved bayesian skill rating system. Technical Report MSR-TR-2018-8, Microsoft, March 2018. URL `https://www.microsoft.com/en-us/research/publication/trueskill-2-improved-bayesian-skill-rating-system/`.

[11] J. W. Mock and S. S. Muknahallipatna. A comparison of ppo, td3 and sac reinforcement algorithms for quadruped walking gait generation. *Journal of Intelligent Learning Systems and Applications*, 15(1):31–44, 2023. URL `https://www.scirp.org/journal/paperinformation?paperid=123401`.

[12] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020. URL `http://jmlr.org/papers/v21/20-212.html`.

[13] K. Ota, D. K. Jha, and A. Kanezaki. Training larger networks for deep reinforcement learning, 2021. URL `https://arxiv.org/abs/2102.07920`.

[14] A. P. Pope, J. S. Ide, D. Mićović, H. Diaz, J. C. Twedt, K. Alcedo, T. T. Walker, D. Rosenbluth, L. Ritholtz, and D. Javorsek. Hierarchical reinforcement learning for air combat at darpa's alphadogfight trials. *IEEE Transactions on Artificial Intelligence*, 4(6):1371–1385, 2023. doi: 10.1109/TAI.2022.3222143.

[15] Rolls-Royce. Ej200 turbofan engine, 2023. URL `https://www.rolls-royce.com/products-and-services/defence/aerospace/combat-jets/ej200.aspx`.

[16] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. P. Lillicrap, K. Simonyan, and D. Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *CoRR*, abs/1712.01815, 2017. URL `http://arxiv.org/abs/1712.01815`.

[17] R. Sutton and A. Barto. *Reinforcement Learning, second edition: An Introduction*. Adaptive Computation and Machine Learning series. MIT Press, 2018. ISBN 9780262352703. URL `https://books.google.it/books?id=uWV0DwAAQBAJ`.

[18] J. J. Tai, J. Wong, M. Innocente, N. Horri, J. Brusey, and S. K. Phang. Pyflyt –

uav simulation environments for reinforcement learning research, 2023. URL https://arxiv.org/abs/2304.01305.

[19] T. Yan, C. Liu, M. Gao, Z. Jiang, and T. Li. A deep reinforcement learning-based intelligent maneuvering strategy for the high-speed uav pursuit-evasion game. *Drones*, 8(7), 2024. ISSN 2504-446X. doi: 10.3390/drones8070309. URL https://www.mdpi.com/2504-446X/8/7/309.

List of Figures

List of Tables

Acknowledgements

Giunto al termine di questo percorso universitario e di tesi, desidero dedicare questo spazio alle persone che hanno accompagnato e guidato i miei passi lungo il cammino. Un sincero ringraziamento va al Professor Andrea Bonarini, che accettando di supervisionare questo lavoro ne ha reso possibile la realizzazione e, con i suoi consigli e la sua gentile professionalità, ne ha sostenuto lo sviluppo fino al completamento. Ringrazio inoltre Leonardo S.p.A. per aver promosso il programma di tesi in azienda "Deep Dive", che ha dato avvio a questo progetto e lo ha supportato mettendo a disposizione le competenze e le risorse di una grande realtà industriale. In modo ancora più significativo, questa esperienza mi ha permesso di incontrare persone straordinarie, sia umanamente sia professionalmente. Un grazie particolare a Fabio e Roberto, che hanno saputo rispondere con competenza — e con la necessaria pazienza — alle numerose domande che hanno accompagnato lo sviluppo della tesi. Un pensiero va infine a Michele, Davide, Matteo, Nicola, Lorenzo e a tutti i colleghi e amici con cui ho condiviso con piacere i mesi di lavoro e trasferta a Genova, rendendo questa esperienza non solo formativa, ma anche profondamente arricchente dal punto di vista umano.