Autonomous UAV Landing on Stochastic Maritime Targets

A reinforcement learning approach for autonomous UAV landing.

AE5322: MSc. Thesis

H.S.Hennecken

Delft University of Technology

TU Delft

Autonomous UAV Landing on Stochastic Maritime Targets

A reinforcement learning approach for maritime UAV applications.

Thesis report

by

H.S.Hennecken

to obtain the degree of Master of Science
at the Delft University of Technology
to be defended publicly on 11/11/2025 at 11:30

Thesis committee:	
Chair:	Dr. J. Sun
Supervisors:	Dr. M.J.Ribeiro
	Dr. O.Pfeifle (NLR)
External examiner:	Dr. ir. E. van Kampen
Place:	Faculty of Aerospace Engineering, Delft
Project Duration:	April, 2025 - November, 2025
Student number:	4836820

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

Faculty of Aerospace Engineering · Delft University of Technology

Copyright © H.S.Hennecken, 2025
All rights reserved.

Preface

This thesis marks the end of a seven-year journey toward earning my Master of Science in Aerospace Engineering at Delft University of Technology. The path has been both challenging and rewarding, filled with moments of growth, perseverance, and discovery. With a highlight, doing exchange at DTU in Copenhagen and an internship at Lockheed Martin in Fort Worth, Texas, USA.

I would like to express my deepest gratitude to my supervisors and professors at TU Delft and NLR for their guidance, encouragement, and expertise. Their commitment to excellence and passion for innovation have been a constant source of motivation. I am also thankful to my family, friends and girlfriend, whose collaboration, discussions, unconditional support have made this experience both enriching and memorable.

A special acknowledgment goes to my late grandfather, Dr. T.H. Janssen, whose intellect and lifelong commitment to studying Greek philosophy have been a lasting inspiration to me. His example has profoundly shaped my academic journey and continues to remind me of the importance of perseverance in the pursuit of knowledge. On that note I would like to summarize my thesis by the following interpretation from a dialogue between Socrates and Plato:

“The more you know, the more you realize you don’t know”

Hidde Hennecken,
Leiden, Nov, 2025

Contents

Nomenclature

Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
DDPG	Deep Deterministic Policy Gradient
DOF	Degree of Freedom
MPC	Model Predictive Control
PID	Proportional–Integral–Derivative
POMDP	Partially Observable Markov Decision Process
RAO	Response Amplitude Operator
RL	Reinforcement Learning
SAC	Soft Actor–Critic
TD	Temporal Difference
UAV	Unmanned Aerial Vehicle
USV	Unmanned Surface Vehicle

Symbols

A	Reference area of the vehicle [m^2]
C_d	Drag coefficient [-]
F_{drag}	Aerodynamic drag force [N]
H_s	Significant wave height [m]
I_{xx}, I_{yy}, I_{zz}	Moments of inertia [kg·m^2]
K_P, K_I, K_D	PID controller gains
L	Rotor arm length [m]
m	Mass of UAV [kg]
p_{rel}	Relative position between UAV and target [m]
v_{rel}	Relative velocity between UAV and target [m/s]
$\mathbf{x}$, $\mathbf{u}$	State and control input vectors
T/W	Thrust-to-weight ratio [-]
V_{cmd}	Commanded velocity [m/s]
V_{max}	Maximum velocity [m/s]
$S(\omega)$	Wave energy spectrum [m^2/s]
$Z(\omega)$	Heave response function [-]
t_f	Final (touchdown) time [s]
α	Learning rate / smoothing coefficient [-]
β	Angle of sideslip [rad]
η	Entropy temperature / attitude vector component
ϕ, θ, ψ	Roll, pitch, and yaw angles [rad]
ω	Angular frequency [rad/s]
ρ	Air density [kg/m^3]
τ	Soft update coefficient [-]
σ	Sensor noise standard deviation [m]
Ψ	Heading angle / distortion risk-measure [rad]
γ	Discount factor (RL) [-]

List of Figures

List of Tables

1

Introduction

Unmanned aerial vehicles (UAVs) play a growing role in maritime operations, supporting surveillance, logistics, and situational awareness while extending operational reach without risking human pilots [1, 2]. Despite these advantages, reliable shipboard recovery remains challenging: wind and waves induce time-varying deck motion, creating a small, dynamic landing target and stringent operational constraints [3, 4].

Traditional approaches to launch and recovery have been studied extensively for helicopters and other single-rotor systems. Notably, NATO's comparative assessment reviews modeling and simulation methods for shipboard launch and recovery [5], and autonomous ship-deck landing strategies for rotorcraft in harsh weather have been reported by Voskuijl *et al.* [6]. While these works provide valuable foundations, they primarily address helicopter dynamics. For multirotors, related quadcopter-focused studies—including signal-prediction and motion-estimation approaches for maritime landing—show similar problem structure and motivate methods tailored to multi-rotor vehicles [7, 8].

Reinforcement learning (RL) has emerged as an alternative to classical control and model-predictive strategies for autonomous landing in dynamic environments, demonstrating improved adaptability to disturbances such as wind gusts and target motion [9, 10, 11, 12, 13]. In this thesis, the platform motion is modeled with explicit heave dynamics and stochastic forcing within a high-fidelity PyBullet simulation environment [14, 15]. By benchmarking RL-based controllers against traditional baselines, the work aims to advance robust recovery methods for UAVs in challenging maritime conditions.

The remainder of this work is structured as follows. Firstly in Chapter 2 the background is given and the problem is defined. Furthermore in Chapter 3 the related literature is presented and in Chapter 4 the research gaps are identified and the research questions introduced. In Chapter 5 the methods are presented. In Part II, the scientific paper is presented which presents the methods are applied to a case study and answers the research questions. Lastly in Chapter 6 the final remarks and research conclusions are given.

Part I

Literature Study

2

Background and Problem Definition

This chapter provides the theoretical and contextual background for this research and formulates the problem of autonomous UAV landing on moving maritime platforms. It begins in Section 2.1 with an overview of UAV classification, summarizing the main platform types, their advantages, and their suitability for shipborne operations. The discussion highlights multirotor UAVs as the most appropriate configuration for dynamic maritime environments due to their vertical take-off and landing (VTOL) capability and precise hover control.

Section 2.1.1 reviews existing UAV landing methods on moving platforms, outlining the practical and technological challenges involved in achieving safe, autonomous recovery under motion and environmental disturbances. This motivates the focus on quadcopters as the reference platform for this research.

In Section 2.2, the operational and scientific relevance of autonomous shipboard UAV landings is discussed. From an operational perspective, such systems extend mission endurance, enable continuous maritime surveillance, and reduce the need for manual piloting. From a scientific standpoint, the problem serves as a benchmark for testing control and guidance algorithms in uncertain, dynamic environments.

Finally, Section 2.3 formulates the problem mathematically, introducing the states of both the moving maritime target (Section 2.3.1) and the UAV (Section 2.3.2), and defining the relative state representation used throughout this work. The section concludes with the formal guidance objective in Section 2.3.3, where the control problem is expressed as minimizing relative position, velocity, and attitude errors to ensure a safe and precise autonomous landing on a moving maritime target.

2.1. UAV Classification

Unmanned Aerial Vehicles (UAVs) can be broadly categorized as fixed-wing, multirotor, single rotor, or hybrid VTOL platforms [16, 17]. Fixed-wing UAVs offer long endurance and high speed but cannot hover, requiring runways or capture systems for landing [18, 19]. Multirotors provide vertical takeoff and landing (VTOL) capability, precise maneuverability, and ease of operation, but have limited range and flight time [20, 21]. Single rotor UAVs bridge endurance and payload gaps between fixed-wing and multirotors but are mechanically complex [22, 23]. Hybrid VTOL UAVs combine vertical takeoff/landing with efficient fixed-wing cruise [24, 25]. Table 2.1 summarizes their characteristics.

Table 2.1: Comparison of UAV Types and Characteristics

UAV Type	Advantages	Limitations	Applications	References
Multirotor	VTOL	Limited range	Inspection	[20]
Fixed-Wing	High speed	Requires runway	Long-range surveillance	[19]
Single Rotor	High payload	Maintenance	Cargo	[26]
Hybrid VTOL	VTOL + cruise	Costly	Maritime	[24]

2.1.1. UAV Landing on Moving Platforms

Landing UAVs on maritime platforms extends operational endurance and enables continuous autonomous missions, which is critical for maritime surveillance, logistics, and disaster response [27, 28]. Different UAV types present different challenges: fixed-wing UAVs require forward motion and use nets, cable arrests, or parachutes [29, 30]; multirotors and hybrid VTOL UAVs exploit vertical descent and onboard sensors (laser, cameras, IMU, GPS) for precise landings [27, 31]; while single rotor UAVs require stabilization against rotor downwash and deck motion [23].

These distinctions illustrate that hybrid VTOL and multirotors are particularly well suited to be deployed in maritime environments. Where precision and vertical landing are more critical than endurance or payload capacity. This motivates us to focus on a VTOL quadcopter as a UAV in this work.

2.2. Problem Relevance

Autonomous quadcopter UAV landing on moving platforms addresses both operational and scientific challenges:

- **Operational:** Supports maritime surveillance, package delivery, search and rescue, and extended mission endurance without human intervention. Reduces reliance on human piloted landing in an extended operational envelope.
- **Scientific:** Advances guidance and control research, and trajectory planning under uncertainty. Provides a framework for testing guidance controllers in dynamic target conditions.

From an application perspective, safe autonomous landing is a prerequisite for fully operational UAV–ship systems. From a research perspective, it is a benchmark problem that captures the challenges of guidance under uncertainty while remaining experimentally feasible.

2.3. Problem Definition

In this section, the problem definition is presented. Firstly in Section 2.3.1 the maritime target state is presented and in Section 2.3.2 the UAV state is presented. Lastly in Section 2.3.3 the guidance problem is defined to guide the UAV to the moving maritime target.

2.3.1. Moving Maritime Target

The landing target represents the landing platform located on the origin of a moving maritime object such as a moving ship or USV. The state of a maritime object consists of it's position , linear velocity and attitude as can be seen in Figure 2.1a.

$$\mathbf{x_t} = [\boldsymbol{p_t} \quad \boldsymbol{v_t} \quad \boldsymbol{\eta_t}]^T \tag{2.1}$$

where:

- $\boldsymbol{p_t} = [x_t, y_t, z_t]^T$ is the position of the target [m].
- $\boldsymbol{v_t} = [v_{x,t}, v_{y,t}, v_{z,t}]^T$ is the linear velocity[m/s].
- $\boldsymbol{\eta_t} = [\Phi, \Theta, \Psi]^T$ is the attitude of the target [rad].

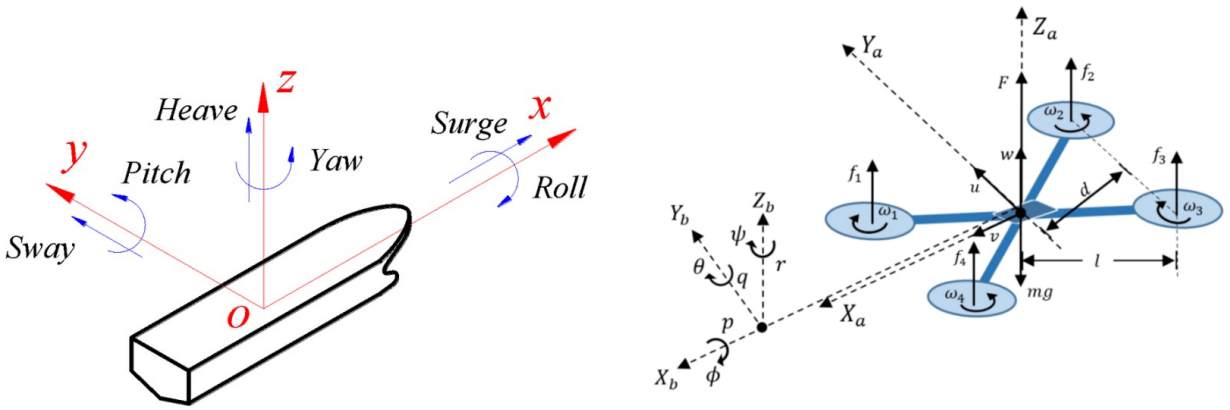
2.3.2. UAV State

The UAV is considered to be a 6 D.O.F. quadcopter drone (i.e. 4 rotors) which can be seen in Figure 2.1b. The state of the drone $\boldsymbol{x_d}$ is given in Equation 2.2.

$$\mathbf{x_d} = [\boldsymbol{p_d} \quad \boldsymbol{v_d} \quad \boldsymbol{\eta_d} \quad \boldsymbol{\omega}_d \quad \boldsymbol{\Omega}_d]^T \tag{2.2}$$

where:

- $\boldsymbol{p_d} = [x_d, y_d, z_d]^T$ is the position of the drone [m].
- $\boldsymbol{v_d} = [v_{x,d}, v_{y,d}, v_{z,d}]^T$ is the linear velocity[m/s].
- $\boldsymbol{\eta_d} = [\phi, \theta, \psi]^T$ is the attitude of the drone [rad].
- $\boldsymbol{\omega_d} = [\omega_x, \omega_y, \omega_z]^T$ are the rotational rates [rad/s].
- $\boldsymbol{\Omega_d} = [\Omega_1, \Omega_2, \Omega_3, \Omega_4]^T$ are the motor speeds [RPM].



(a) Maritime target free body diagram [32]

(b) 6 D.O.F. free body diagram of a quadcopter UAV [33]

Figure 2.1: Free body diagrams of the moving maritime target (left) and the UAV (right).

2.3.3. Guidance Problem

The objective of the guidance system is to autonomously guide the drone to perform a safe and precise landing on a moving maritime target. The problem can be formulated as a nonlinear tracking and control task, where the drone must minimise its relative position, velocity and attitude errors with respect to the moving target. Let the relative state between the drone and the target be defined as:

$$\boldsymbol{x}_{\text{rel}} = \begin{bmatrix} \boldsymbol{p}_{\text{rel}} \\ \boldsymbol{v}_{\text{rel}} \\ \boldsymbol{\eta}_{\text{rel}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}_d - \boldsymbol{p}_t \\ \boldsymbol{v}_d - \boldsymbol{v}_t \\ \boldsymbol{\eta}_d - \boldsymbol{\eta}_t \end{bmatrix},$$

where $\boldsymbol{p}_d$, $\boldsymbol{v}_d$ and $\boldsymbol{\eta}_d$ denote the drone's position, velocity and attitude, and $\boldsymbol{p}_t$, $\boldsymbol{v}_t$ and $\boldsymbol{\eta}_t$ correspond to those of the target landing platform. The control objective is to compute a commanded velocity vector $\boldsymbol{V}_{cmd}$ such that the drone's relative altitude converges to zero at touchdown time t_f while ensuring that the horizontal position, relative velocity and attitude at impact stay below a safe landing threshold.

$$\begin{aligned}
&\lim_{t \to t_f} |p_{\text{rel},z}(t)| = 0, \\
&\lim_{t \to t_f} \|\mathbf{p}_{\text{rel}}(t)\| \leq |\mathbf{p}_{\text{rel,xy}}|_{\max}, \\
&\lim_{t \to t_f} \|\mathbf{v}_{\text{rel}}(t)\| \leq |\mathbf{v}_{\text{rel}}|_{\max}, \\
&\lim_{t \to t_f} \|\boldsymbol{\eta}_{\text{rel}}(t)\| \leq |\boldsymbol{\eta}_{\text{rel}}|_{\max}.
\end{aligned}$$

3

Related Works

This section reviews prior research on autonomous unmanned aerial vehicle (UAV) landing on moving (maritime) targets. The section is organized into five main parts: Section 3.0.1 covers classical control approaches relying on deterministic and predictive models; Section 3.0.2 presents hybrid methods combining model-based and learning-based control; Section 3.0.3 reviews value-based RL approaches emphasizing simplicity and efficiency; Section 3.0.4 summarizes recent actor–critic and memory-augmented frameworks; and finally, Section 3.0.5 compares all paradigms across computational and performance dimensions.

3.0.1. Classical Control Strategies for UAV-Ship Landings

Classical control methods for autonomous UAV landings on maritime platforms range from simple feedback controllers to advanced predictive and model-based strategies. Abujoub [7] proposed a LIDAR-based signal prediction algorithm (SPA) to forecast periods of minimal ship motion, enabling UAVs to land safely during these ”quiescent periods.” This approach reduced the number of landing attempts by approximately two per trial case and demonstrated the potential for improved landing efficiency over fully reactive systems. However, being a proof-of-concept, its performance under extreme sea states remains untested.

Voskuijl [6] introduced a cascaded PID control framework with predictive landing strategies using ship-mounted sensors. Evaluated in a nonlinear simulation with a 100 kg UAV and varying sea states, this method reduced touchdown velocity by up to 60% and improved landing accuracy by 53% with respect to a simple baseline. While effective across multiple operational conditions, its reliance on accurate ship sensors and predictions limits robustness in highly uncertain environments.

Gupta [34] surveyed model predictive control (MPC) strategies, including linear and nonlinear formulations, constraint handling, and learning-enhanced approaches. MPC-based methods enable precise control, robust handling of constraints, and adaptability under uncertain conditions. The main limitations include high computational cost, the need for accurate environmental modeling, and challenges in real-time implementation.

Zilver [35] investigated wind-aware trajectory optimization for UAV and helicopter approaches to maritime vessels. By considering wind speed, direction, and model fidelity, the framework optimized smooth and energy-efficient trajectories and allowed preliminary assessment of operational limits. The main constraints are its reliance on accurate wind modeling, computational intensity, and primary validation through simulations rather than real-world operations.

Limitations: classical strategies demonstrate stability, interpretability, and improved landing performance, particularly when predictive or model-based elements are incorporated. However, the systems are limited by its reliance on accurate sensors, precise environmental models, and carefully tuned parameters, as well as the high computational demands of MPC or optimization-based methods.

3.0.2. Hybrid Control Methods

Hybrid control strategies have emerged to combine the interpretability of classical methods with the adaptability of data-driven learning. Xie [36] proposed a hybrid architecture integrating deep deterministic policy gradient (DDPG) with heuristic vertical guidance, achieving up to 93% success in landing on randomly moving targets. Wu [37] expanded on this concept by using DDPG to tune PID gains online, leading to faster convergence and stronger generalization compared to standalone RL or PID controllers. Other

studies explored learning-based MPC tuning or disturbance compensation modules that adjust control parameters adaptively during flight.

Hybrid frameworks bridge the gap between model-based precision and learning-based flexibility. They enable adaptive behavior while maintaining the safety and explainability of classical controllers. However, their performance often depends on the underlying control architecture's structure and the training quality of the learning module.

Limitations: while hybrid approaches enhance adaptability and robustness, they still inherit some of the rigidity and tuning dependence of classical control. The learning components can also increase system complexity and introduce stability risks if not properly constrained [36, 37].

3.0.3. RL Value-Based Approaches

Although often considered less powerful than deep RL, tabular methods remain relevant for their simplicity and efficiency. Abo Mosali [38] introduced AMLQ, a quantized Q-learning scheme that avoids deep networks, reduces training time, and achieved lower RMSE than PID in 2D landing tasks. More recently, Goldschmid [39] applied Double Q-learning with curriculum training and motion-structured discretization. Despite being tabular, their method rivaled PI controllers and demonstrated successful transfer from simulation to hardware.

Limitations: while value-based approaches are computationally efficient, they scale poorly to high-dimensional or continuous state/action spaces and may not generalize to realistic 3D landing scenarios [38, 39].

3.0.4. RL Actor–Critic and Memory-Augmented Methods

Actor–critic frameworks dominate current research due to their ability to handle partial observability and continuous state/action spaces. Jiang [40] benchmarked DDPG and PPO, showing both can achieve near-perfect simulated success rates. Building on this, Saj [41] validated a vision-based DDPG controller on a 6-DOF ship deck, reporting nearly 100% success and outperforming nonlinear PID controllers. More recently, Aikins [42] extended PPO with an LSTM-based memory module (RPO-LSTM), which improved success by 74% under wind-disturbed conditions by better capturing hidden dynamics.

Limitations: despite high simulation success, actor–critic frameworks often require extensive training data, are sensitive to reward shaping, and may struggle with generalization to new environmental disturbances [42]. Memory-augmented architectures introduce additional computational overhead and may still fail under long-term partial observability.

3.0.5. Method Comparison

Table 3.1 provides a comparative overview of classical, hybrid, and learning-based control paradigms for autonomous UAV landing on moving platforms. The comparison focuses on three key dimensions: Computational intensity, Control accuracy or success rate and adaptability to changing or uncertain environments.

Classical controllers remain attractive due to their interpretability, stability, and safety guarantees. PID controllers are computationally inexpensive, relying on straightforward gain-based updates, but their performance is sensitive to environmental changes and requires careful retuning. MPC and optimal control methods optimize control inputs over a finite prediction horizon using system models, achieving high precision and constraint satisfaction [34, 7, 6, 43, 35]. Their main limitation is computational cost and reliance on accurate models, which can reduce real-time robustness in highly dynamic maritime environments.

Hybrid control strategies integrate classical methods with learning-based components, such as DDPG-tuned PID or heuristic-guided MPC [36, 37]. These approaches retain the stability and interpretability of classical control while enhancing adaptability and robustness to uncertain or changing conditions. They achieve faster convergence and higher success rates than standalone classical or RL methods, though they inherit some tuning and structural dependencies from the underlying classical controllers.

Reinforcement learning (RL) approaches remove the need for explicit system models and can handle nonlinear, uncertain dynamics. Value-based methods [38, 39] are computationally lightweight but scale poorly to high-dimensional or continuous spaces. Actor–critic methods [40, 41, 42] achieve the highest

performance in complex dynamic conditions, with success rates approaching 100% in simulations, though they require extensive offline training and careful reward design. Memory-augmented variants improve performance under partial observability but add computational overhead.

The choice of control paradigm represents a trade-off between computational load, landing reliability, and adaptability to dynamic conditions. Classical PID and MPC offer predictable and interpretable performance but often rely on computationally expensive predictions of ship motions, hybrid methods provide a balanced compromise, and RL—particularly actor-critic architectures delivers superior performance once trained.

Table 3.1: Comparison of control methods for autonomous UAV landing on moving platforms

Method	**Computational Intensity (Online)**	**Accuracy / Success Rate**	**Adaptability / Generalization**	**Representative Works**
Classical	High; Prediction of ship motion requires real-time optimization and initialisation.	Moderate to high (60–95%); PID performs well in stable conditions, MPC improves precision if the model is accurate	Low to moderate; PID needs retuning for new dynamics, MPC depends on model fidelity	[6, 7, 34, 43, 35]
Hybrid (Classical + RL)	Moderate; inference + classical feedback	High (85–95%) under dynamic motion	High—adapts online via learned tuning or guidance	[36, 37]
RL Value-Based	Low; table lookup or small network	Moderate to high (70–90%) in simple tasks	Low to moderate; poor scalability to continuous 3D	[38, 39]
RL Actor–Critic	Moderate (inference only) after high-cost training	Very high (90–100%) in simulation	High—learns nonlinear and uncertain dynamics; sensitive to training	[40, 41, 42]

4

Research Definition

This chapter defines the scope and direction of the research. Building on the gaps identified in the literature, the chapter first outlines the unresolved challenges in existing methods for UAV autonomous landing (Section 4.1). It then formulates the research objective and the key research question guiding this work (Section 4.2). Together, these sections provide a clear foundation for the methodology and experimental design that follow in later chapters.

4.1. Research Gaps

Although recent advances in reinforcement learning (RL) have demonstrated promising results for UAV landing on moving platforms, several critical gaps remain unaddressed in the literature.

First, current algorithmic approaches are limited. Actor–critic frameworks such as DDPG and PPO dominate the field [40, 41, 42], but both suffer from well-known shortcomings. DDPG is prone to instability and is highly sensitive to hyperparameter tuning, while PPO requires delicate adjustment of clipping and entropy coefficients to ensure robust performance. These issues restrict their applicability in safety-critical UAV guidance tasks.

Second, there is a need for more robust and stable algorithms. Recent research highlights the potential of the *Soft Actor–Critic* (SAC) algorithm [44, 45], which augments the standard RL objective with entropy maximization. SAC has demonstrated improved exploration, stability, and sample efficiency compared to DDPG and PPO, while also showing resilience to noisy or uncertain environments. This makes it a strong candidate for UAV landing guidance under maritime conditions, where disturbances such as wind, wave motion, and sensor uncertainty are prevalent.

Third, the majority of existing work relies on simplified platform motion models, typically assuming motion in only the horizontal plane. In contrast, real-world maritime platforms exhibit significant vertical oscillations (heave motion) in response to sea states. Neglecting these dynamics reduces the realism of simulations and limits the applicability of results to real-world high sea state maritime scenarios. Addressing stochastic heave motion is therefore critical for developing reliable and deployable UAV maritime landing systems.

4.2. Research Formulation

In light of these gaps, the present research is formulated with the following overarching objective:

Research Objective

To design and evaluate the performance of a reinforcement learning (RL) based UAV landing controller for autonomous UAV landings on maritime targets subject to stochastic heave motion and environmental disturbances.

This objective gives rise to the following main research question:

Research Question 1

How does a RL-based guidance controller for autonomous UAV ship landings perform in terms of success rate, landing precision and robustness when subject to high sea states and disturbances?

4.2.1. Sub-questions

To address this main question in a structured way, several sub-questions are defined and grouped into four themes, as presented below.

1. **RL-type:**

   Research Question 1.1

   Does a hybrid RL guidance controller have benefits compared to a full RL guidance controller?

2. **Robustness:**

   Research Question 1.2

   How do wind, sea state, and sensor noise affect the success rate, landing precision, and stability of each controller?

3. **Comparative Analysis:**

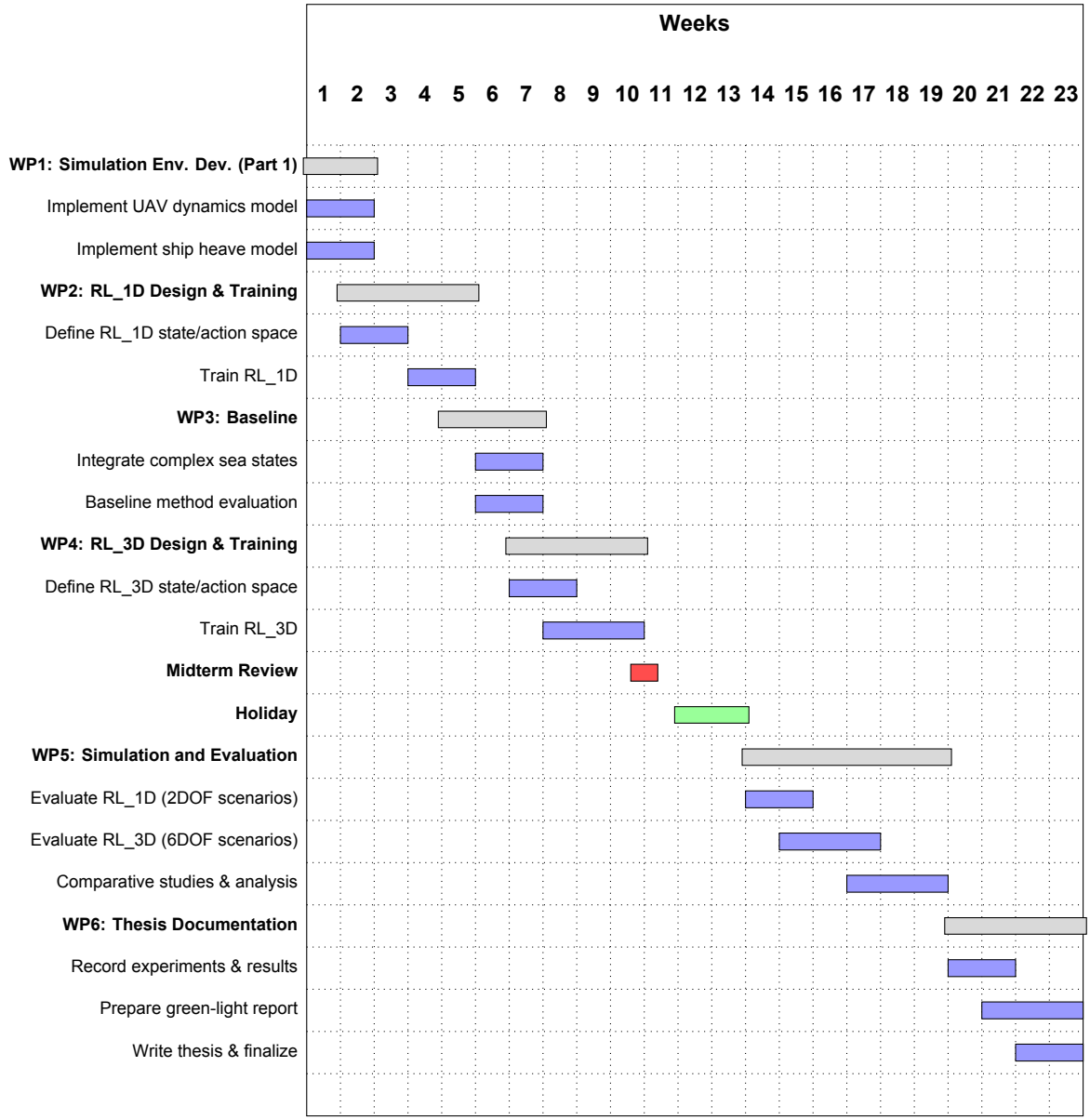
   Research Question 1.3

   Which controller demonstrates the best trade-off between robustness, precision, and efficiency across varying environmental and dynamic conditions?

4.3. Research Execution

The research is performed in collaboration with the NLR at the vertical flight department in Amsterdam. The duration of the project up and to the greenlight is 23 weeks. The project planning can be seen in Figure 4.1.

Figure 4.1: Thesis Timeline (Weeks 1–23)



5

Methods

5.1. Baseline Landing Strategy

In this section the baseline landing strategy is presented. The baseline guidance controller uses a classical PID strategy inspired by the baseline strategy of [6]. Its purpose is to provide a stable, interpretable reference for evaluating the RL-based guidance system. PID control is widely used in aerial robotics for its simplicity, robustness, and computational efficiency [46, 47], making it suitable for high-frequency real-time applications such as UAV landing on moving platforms. More advanced methods (MPC, SMC, FLC) offer improved tracking or disturbance rejection but with higher computational cost [48, 49, 50]. Horizontal motion is governed by a standard PID law regulating the horizontal velocity based on relative horizontal position and velocity. The vertical control uses a PD law that takes into account the relative altitude and vertical velocity:

$$\begin{aligned} \vec{V}_{cmd,xy} &= K_P^{xy}\vec{e}_{xy} + K_I^{xy}\int \vec{e}_{xy}\,dt + K_D^{xy}\vec{v}_{xy} \\ \vec{V}_{cmd,z} &= K_P^{z}\vec{e}_{z} + K_D^{z}\vec{v}_{z,\text{rel}} \end{aligned} \tag{5.1}$$

where K_P^{xy}, K_I^{xy}, K_D^{xy}, K_P^z and K_D^z are the gains of the baseline controller. The baseline PID gains are listed in Table 5.1, tuned manually for horizontal motion and via grid search for vertical motion. Systematic tuning methods (Ziegler–Nichols, Cohen–Coon) are outside the scope of this work. The vertical guidance is initiated once the UAV is above the target and the target is in a downward motion, allowing for a safe and rapid descent at high altitudes and smooth deceleration near the landing zone.

Axis	K_p	K_i	K_d
X (Horizontal)	1.00	0.20	0.08
Y (Horizontal)	1.00	0.20	0.08
Z (Vertical)	-0.15	0.00	-0.15

Table 5.1: PID gain values for the horizontal (X, Y) and vertical (Z) controllers.

Furthermore, a clipping function and first-order low-pass filter is applied to smooth the velocity command and prevent abrupt changes with $\alpha_{filter} = 0.1$ and $V_{max} = 4m/s$. A schematic overview of the baseline landing strategy can be seen in Figure 5.1.

5.2. Reinforcement Learning Landing Strategies

This section presents the reinforcement learning (RL) framework developed for autonomous UAV ship landings. It is organized into five parts to systematically describe how the RL agent interacts with and learns from its environment. Section 5.2.1 introduces the theoretical foundation of reinforcement learning based on the Markov Decision Process (MDP) and its partially observable variant. Section 5.2.2 defines the agent's action space, describing both the hybrid and full 3D RL controllers and the applied command filtering. Section 5.2.3 details the observation space, which encodes the UAV–target state information available to the agent. Section 5.2.4 explains the design of the reward function that drives learning toward safe and accurate landings. Finally, Section 5.2.5 presents the Soft Actor–Critic (SAC) algorithm used for policy optimization and training stability.

5.2.1. Markov Process

Reinforcement Learning is a machine learning framework where an agent interacts with an environment to maximize cumulative reward. The environment is typically modeled as a *Markov Decision Process (MDP)* [51]:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$$

where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $P(s'|s,a)$ the transition probability, $R(s,a)$ the reward function, and γ the discount factor. The *Markov property* assumes that that the next state depends only on the current state and action:

$$P(s_{t+1}|s_t, a_t) = P(s_{t+1}|s_t, a_t, \dots)$$

In many real-world continuous control tasks, however, the Markov property does not strictly hold [51]. The agent does not have direct access to the full environmental state s_t, but instead only to a partial or noisy observation $o_t \in \mathcal{O}$. This situation is modeled as a *Partially Observable Markov Process (POMP)* [52, 53]. A POMP is defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \mathcal{O}, O, \gamma)$, where $\mathcal{O}$ is the observation space and $O(o|s)$ denotes the observation probability distribution.

In this setting, the agent must infer useful information about the hidden environment state from a history of past observations and actions, rather than relying on a single observation. This makes the problem more challenging and typically requires memory-based models (e.g., recurrent neural networks) or state-estimation techniques (e.g., filters, belief states) to approximate the underlying Markov process [54, 55].

5.2.2. Action Space

In this subsection, two different RL-controllers are presented. The presented problem involves both horizontal actions (i.e. tracking of the moving landing target) and vertical actions (i.e. a safe and reliable descend on the moving target). Two distinct RL controllers are introduced in Section 5.2.2, a hybrid 1D action space and in Section 5.2.2 a 3D action space controller.

Hybrid RL 1D

This agent has a 1 dimensional action space and only has autonomy over the vertical command velocity. While having the horizontal guidance performed by a PID controller earlier introduced in Section 5.1.

$$\mathbf{a}_t = \begin{bmatrix} V_{z,cmd} \end{bmatrix} \quad \in [-1, 1]$$

RL 3D

This agent has full 3D action autonomy over the command velocity to guide the drone to the target.

$$\mathbf{a}_t = \begin{bmatrix} V_{x,cmd} \\ V_{y,cmd} \\ V_{z,cmd} \end{bmatrix} \quad \in [-1, 1]$$

Action Command Filtering
In the environment, both RL_1D and RL_3D actions $a_t \in [-1, 1]$ are passed through a command filter. The RL agent outputs an action a_t in the range $[-1, 1]$, which is scaled to the maximum UAV velocity V_{max}. A first-order low-pass filter is applied to smooth the command and prevent abrupt changes with $\alpha_{filter} = 0.1$ and $V_{max} = 4m/s$:

$$V_{cmd} \leftarrow \alpha_{filter}\,(a_t \cdot V_{max}) + (1 - \alpha)V_{\text{cmd}}, \quad \alpha_{filter} \in [0, 1].$$

In short, this process scales, smooths, and saturates the RL action before execution.

5.2.3. Observation Space

Both RL controllers **RL 1D** and **RL 3D** observe the environment through a 15-dimensional vector capturing the relative state between the drone and the landing target. The observation vector is defined as the 15D vector:

$$\mathbf{o}_t = \begin{bmatrix} \mathbf{p}_{\text{rel}} & \mathbf{v}_{\text{rel}} & \mathbf{v}_{\text{d}} & \eta_{\text{rel}} & e_{\text{xy}} & \int_0^t e_{\text{xy}} dt & T \end{bmatrix}^T \tag{5.2}$$

where:

- $\mathbf{p}_{\text{rel}} = [x_{\text{rel}} \quad y_{\text{rel}} \quad z_{\text{rel}}]^T$ represents the 3D relative position between the drone and the target.
- $\mathbf{v}_{\text{rel}} = [v_{\text{rel, x}} \quad v_{\text{rel, y}} \quad v_{\text{rel, z}}]^T$ represents the 3D relative linear velocity between the drone and the target.
- $\mathbf{v}_{\text{d}} = [v_{\text{d, x}} \quad v_{\text{d, y}} \quad v_{\text{d, z}}]^T$ represents the 3D linear velocity of the drone.
- $\eta_{\text{rel}} = [\phi_{\text{d}} \quad \theta_{\text{d}} \quad \psi_{\text{d}} - \Psi_t]^T$ represents the relative 3D roll, pitch and heading between the drone and the target.
- $e_{\text{xy}} = \left|\left| \begin{bmatrix} x_{\text{rel}} \\ y_{\text{rel}} \end{bmatrix} \right|\right|$ represents the horizontal tracking error.
- $\int_0^t e_{\text{xy}} dt$ represents the integral horizontal tracking error.
- T is the normalized time of the episode.

The observation space consists of different measurements all with different scales. In order to generate a better policy, the observations are clipped and MinMaxScaled to the [-1 , 1] domain.

5.2.4. Reward Function

The reward function encourages the agents to track the target accurately while minimizing excessive relative roll and heading attitudes. This structure balances tracking accuracy, smooth control, and safety, which is critical for continuous control in a dynamic maritime environment. The design follows a common approach in UAV and robotic landing tasks, where a shaping reward provides dense feedback during training and a terminal success reward reinforces goal completion [56, 51].

$$r_t = r_{\text{shaping}} + r_{\text{success}}$$

where:

$$r_{\text{shaping}} = 0.05 \left(1 - e_{\text{xy}} - 0.5 \textstyle\int^t e_{\text{xy}} dt - 0.5\|\eta_{\text{rel}}\|\right)$$

$$r_{\text{success}} = 100 + \frac{50}{1 + e_{\text{xy}}^2} + \frac{50}{1 + \|\mathbf{v}_{\text{rel}}\|^2} + \frac{50}{1 + \|\eta_{\text{rel}}\|^2}$$

The shaping component provides continuous guidance toward the target throughout each episode, while the terminal success reward strongly reinforces precise alignment and smooth relative motion upon landing. This hybrid structure has been shown to improve learning stability and convergence in complex continuous control problems. To ensure the agent prioritizes successful landings over small incremental improvements, the terminal reward is made dominant ($r_{\text{success}} \gg r_{\text{shaping}}$), as recommended by [51].

5.2.5. Policy: Soft Actor–Critic (SAC)

The Soft Actor-Critic (SAC) algorithm [44] combines elements of actor–critic methods with the maximum entropy framework for reinforcement learning. In standard reinforcement learning (RL), the objective is to learn a parameterized policy π that maximizes the expected cumulative return over an episode of length T:

$$\max\ \pi\ \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}\left[\sum_{t=0}^{T} r(s_t,a_t)\right],$$

where $\rho_\pi(s_t)$ denotes the state distribution induced by policy $\pi(a_t|s_t)$. The action-value function $Q(s,a)$ expresses the expected future return after taking action a in state s. In conventional RL, the learned policy is typically unimodal and centered around the action that maximizes $Q(s,a)$. This narrow focus can reduce exploration, which makes the agent brittle: even small environmental changes may cause failure.

Maximum entropy reinforcement learning addresses this limitation by augmenting the objective with an entropy term that explicitly encourages exploration [44]. The modified objective is

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

where $\mathcal{H}(P) = \mathbb{E}_{x\sim P}\left[-\log P(x)\right]$ is the entropy and α is a temperature parameter controlling the trade-off between reward maximization and policy stochasticity. This objective induces policies of the form

$$\pi(a|s) \propto \exp\big(Q(s,a)\big),$$

The SAC algorithm [44] builds on this principle and integrates three key components: (i) an actor–critic architecture with separate policy and value networks, (ii) an off-policy learning procedure that reuses past experience for sample efficiency, and (iii) entropy maximization to balance stability and exploration. Compared to earlier methods such as Deep Deterministic Policy Gradient (DDPG) [57], SAC achieves improved stability, faster convergence, and superior asymptotic performance in continuous, high-dimensional control tasks. By combining sample efficiency with robustness to environmental variability, SAC addresses the brittleness that often affects deep reinforcement learning algorithms.

The SAC implementation in this work follows the default *Stable-Baselines3* configuration [58], with standard continuous control hyperparameters for stable and reproducible training. The entropy coefficient α and target entropy are set to `"auto"` to adaptively balance exploration and exploitation, stabilizing learning. Table 5.2 lists the hyperparameters used for training the RL guidance policy.

Hyperparameter	**Default**
γ (discount)	0.99
Learning rate	0.0003
Buffer size	1×10^6
Learning starts	100
Batch size	256
τ (soft update)	0.005
α (entropy)	`"auto"`
Target entropy	`"auto"`
Architecture	2×256
Activation	Tanh
Action noise	$\mathcal{N}(0,\ .05)$

Table 5.2: Default hyperparameters for SAC in Stable-Baselines3

Landing Strategies Overview
In Table 5.3, the axis-wise control formulation for all 3 proposed landing strategies are presented. Furthermore, all 3 landing strategies can be seen in Figure 5.1, Figure 5.2 and Figure 5.3.

Table 5.3: Overview of landing strategies and axis-wise control formulation.

Strategy	X-axis	Y-axis	Z-axis
Baseline (PID)	PID	PID	PD
RL 1D (Hybrid)	PID	PID	RL
RL 3D (Full RL)	RL	RL	RL

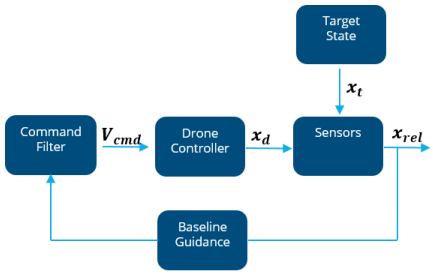


Figure 5.1: Baseline controller overview

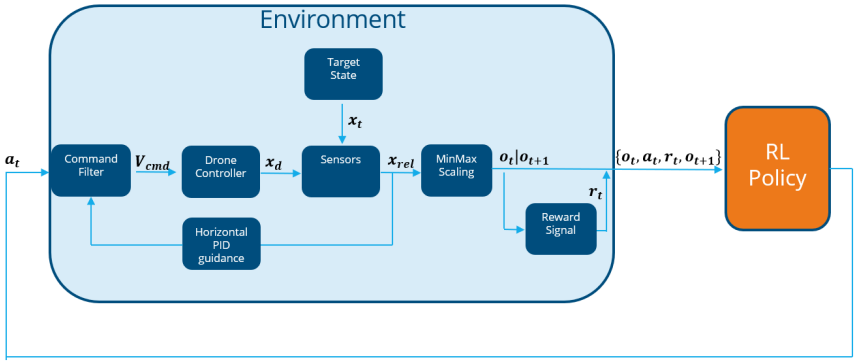


Figure 5.2: RL_1D controller overview

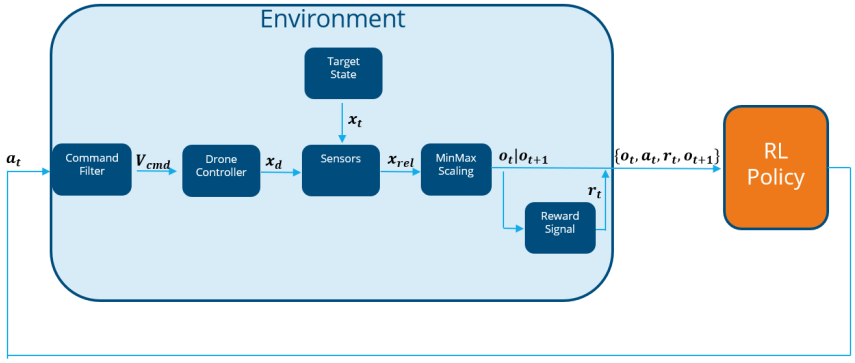


Figure 5.3: RL_3D controller overview

Part II

Scientific Article

Autonomous Drone Landing on Stochastic Maritime Targets

H.S.Hennecken*
Faculty of Aerospace Engineering, Delft University of Technology

Reliable autonomous recovery of Unmanned Aerial Vehicles (UAVs) on moving maritime platforms remains a critical challenge, primarily due to complex, stochastic deck motion, particularly vertical heave, and unpredictable environmental disturbances. Existing Reinforcement Learning (RL) approaches often simplify this environment, limiting their real-world applicability. This thesis investigates the robustness trade-offs of RL-based guidance controllers under realistic, high-dynamicity maritime conditions. We benchmarked a classical Proportional-Integral-Derivative (PID) controller against two RL architectures trained using Soft Actor-Critic (SAC) in a high-fidelity PyBullet simulation: a Full RL 3D controller and a novel Hybrid RL 1D controller, which strategically applies RL only to the critical, stochastic vertical (heave) axis. The results demonstrate that the Hybrid RL 1D architecture (86.6% success rate) achieved superior overall robustness and efficiency. Notably, the RL controllers dramatically reduced average landing time (RL_1D: 3.31 s vs. Baseline: 11.51 s), though the classical PID baseline maintained higher horizontal precision (Err$_{XY}$ of 0.17 ± 0.17 m). The Hybrid RL 1D maintained a superior success rate up to 89% in high sea states (SS7) and exhibited greater resilience to sensor noise. However, a critical limitation was identified: both RL-based policies experienced a pronounced performance collapse under strong, untrained wind disturbances, a regime where the non-adaptive classical PID baseline proved unexpectedly stable. These findings confirm the benefits of hybrid control for maximizing robustness and highlight that the system's ability to handle wind disturbance rejection remains a significant, unresolved shortcoming for current RL guidance systems.

*MSc student, Faculty Aerospace Engineering, Kluyverweg 1, Delft, Netherlands

I. Introduction

Unmanned aerial vehicles (UAVs) play a growing role in maritime operations, supporting surveillance, logistics, and situational awareness while extending operational reach without risking human pilots [1, 2]. Despite these advantages, reliable shipboard recovery remains challenging: wind and waves induce time-varying deck motion, creating a small, dynamic landing target and stringent operational constraints [3, 4].

Traditional approaches to launch and recovery have been studied extensively for helicopters and other single-rotor systems. Notably, NATO's comparative assessment reviews modeling and simulation methods for shipboard launch and recovery [5], and autonomous ship-deck landing strategies for rotorcraft in harsh weather have been reported by Voskuijl *et al.* [6]. While these works provide valuable foundations, they primarily address helicopter dynamics. For multirotors, related quadcopter-focused studies—including signal-prediction and motion-estimation approaches for maritime landing—show similar problem structure and motivate methods tailored to multi-rotor vehicles [7, 8].

Reinforcement learning (RL) has emerged as an alternative to classical control and model-predictive strategies for autonomous landing in dynamic environments, demonstrating improved adaptability to disturbances such as wind gusts and target motion [9–13]. In this thesis, the platform motion is modeled with explicit heave dynamics and stochastic forcing within a high-fidelity PyBullet simulation environment [14, 15]. By benchmarking RL-based controllers against traditional baselines, the work aims to advance robust recovery methods for UAVs in challenging maritime conditions.

The remainder of this work is structured as follows. Firstly in section II the problem is defined and secondly in section III the related works are presented. Furthermore in section IV the research gaps are identified and the research objective, questions and sub-questions are identified. In section V and section VI the baseline and RL landing strategies are introduced and in section VII the case study and environment simulation are presented. section IX presents simulation experiments, performance metrics, and comparative analysis. section XI discusses the evaluation results and proposes directions for future work. Finally, section XII summarizes the findings, outlines contributions.

II. Problem Definition

In this chapter, the background and definition of the problem are presented. The landing target represents the landing platform located on the aft section of a moving maritime object such as a moving ship or USV. The state of a maritime object consists of it's position (surge, sway and heave), linear velocity and attitude (roll, pitch and yaw).

$$\mathbf{x_t} = [\boldsymbol{p_t} \quad \boldsymbol{v_t} \quad \boldsymbol{\eta_t}]^T \tag{1}$$

where:

- $\boldsymbol{p_t} = [x_t, y_t, z_t]^T$ is the position of the target [m].
- $\boldsymbol{v_t} = [v_{x,t}, v_{y,t}, v_{z,t}]^T$ is the linear velocity[m/s].
- $\boldsymbol{\eta_t} = [\Phi, \Theta, \Psi]^T$ is the attitude of the target [rad].

The UAV is considered to be a quadcopter drone (i.e. 4 rotors). The state of the drone $\boldsymbol{x_d}$ is given in Equation 2.

$$\mathbf{x_d} = [\boldsymbol{p_d} \quad \boldsymbol{v_d} \quad \boldsymbol{\eta_d} \quad \boldsymbol{\omega_d} \quad \boldsymbol{\Omega_d}]^T \tag{2}$$

where:

- $\boldsymbol{p_d} = [x_d, y_d, z_d]^T$ is the position of the drone [m].
- $\boldsymbol{v_d} = [v_{x,d}, v_{y,d}, v_{z,d}]^T$ is the linear velocity[m/s].
- $\boldsymbol{\eta_d} = [\phi, \theta, \psi]^T$ is the attitude of the drone [rad].
- $\boldsymbol{\omega_d} = [\omega_x, \omega_y, \omega_z]^T$ are the rotational rates [rad/s].
- $\boldsymbol{\Omega_d} = [\Omega_1, \Omega_2, \Omega_3, \Omega_4]^T$ are the motor speeds [RPM].

The objective of the guidance system is to autonomously guide the drone to perform a safe and precise landing on a moving maritime target. The problem can be formulated as a nonlinear tracking and control task, where the drone must minimise its relative position, velocity and attitude errors with respect to the moving target. Let the relative state between the drone and the target be defined as:d

$$\boldsymbol{x}_{\text{rel}} = \begin{bmatrix} \boldsymbol{p}_{\text{rel}} \\ \boldsymbol{v}_{\text{rel}} \\ \boldsymbol{\eta}_{\text{rel}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}_d - \boldsymbol{p}_t \\ \boldsymbol{v}_d - \boldsymbol{v}_t \\ \boldsymbol{\eta}_d - \boldsymbol{\eta}_t \end{bmatrix},$$

where $\boldsymbol{p}_d$, $\boldsymbol{v}_d$ and $\boldsymbol{\eta}_d$ denote the drone's position, velocity and attitude, and $\boldsymbol{p}_t$, $\boldsymbol{v}_t$ and $\boldsymbol{\eta}_t$ correspond to those of the target landing platform. The control objective is to compute a commanded velocity vector $\boldsymbol{V}_{cmd}$ such that the drone's relative altitude converges to zero at touchdown time t_f while ensuring that the horizontal position, relative velocity and attitude at impact stay below a safe landing threshold:

$$\begin{aligned} \lim_{t \to t_f} |p_{\text{rel},z}(t)| &= 0, \\ \lim_{t \to t_f} \|\mathbf{p}_{\text{rel,xy}}(t)\| &\leq |\mathbf{p}_{\text{rel,xy}}|_{\max}, \\ \lim_{t \to t_f} \|\mathbf{v}_{\text{rel}}(t)\| &\leq |\mathbf{v}_{\text{rel}}|_{\max}, \\ \lim_{t \to t_f} \|\boldsymbol{\eta}_{\text{rel}}(t)\| &\leq |\boldsymbol{\eta}_{\text{rel}}|_{\max}. \end{aligned} \tag{3}$$

III. Related Works

This section reviews prior research on autonomous unmanned aerial vehicle (UAV) landing on moving (maritime) targets. The section is organized into five main parts: subsection III.A covers classical control approaches relying on deterministic and predictive models; subsection III.B presents hybrid methods combining model-based and learning-based control; subsection III.C reviews value-based RL approaches emphasizing simplicity and efficiency; subsection III.D summarizes recent actor–critic and memory-augmented frameworks; and finally, subsection III.E compares all paradigms across computational and performance dimensions.

A. Classical Control Strategies for UAV-Ship Landings

Classical control methods for autonomous UAV landings on maritime platforms range from simple feedback controllers to advanced predictive and model-based strategies.

Abujoub [7] proposed a LIDAR-based signal prediction algorithm (SPA) to forecast periods of minimal ship motion, enabling UAVs to land safely during these "quiescent periods." This approach reduced the number of landing attempts by approximately two per trial case and demonstrated the potential for improved landing efficiency over fully reactive systems. However, being a proof-of-concept, its performance under extreme sea states remains untested.

Voskuijl [6] introduced a cascaded PID control framework with predictive landing strategies using ship-mounted sensors. Evaluated in a nonlinear simulation with a 100 kg UAV and varying sea states, this method reduced touchdown velocity by up to 60% and improved landing accuracy by 53% with respect to a simple baseline. While effective across multiple operational conditions, its reliance on accurate ship sensors and predictions limits robustness in highly uncertain environments.

Gupta [16] surveyed model predictive control (MPC) strategies, including linear and nonlinear formulations, constraint handling, and learning-enhanced approaches. MPC-based methods enable precise control, robust handling of constraints, and adaptability under uncertain conditions. The main limitations include high computational cost, the need for accurate environmental modeling, and challenges in real-time implementation.

Zilver [17] investigated wind-aware trajectory optimization for UAV and helicopter approaches to maritime vessels. By considering wind speed, direction, and model fidelity, the framework optimized smooth and energy-efficient trajectories and allowed preliminary assessment of operational limits. The main constraints are its reliance on accurate wind modeling, computational intensity, and primary validation through simulations rather than real-world operations.

Limitations: Classical strategies demonstrate stability, interpretability, and improved landing performance, particularly when predictive or model-based elements are incorporated. However, the system's are limited by its reliance on accurate sensors, precise environmental models, and carefully tuned parameters, as well as the high computational demands of MPC or optimization-based methods.

B. Hybrid Control Methods

Hybrid control strategies have emerged to combine the interpretability of classical methods with the adaptability of data-driven learning. Xie [18] proposed a hybrid architecture integrating deep deterministic policy gradient (DDPG) with heuristic guidance, achieving up to 93% success in landing on randomly moving targets. Wu [19] expanded on this concept by using DDPG to tune PID gains online, leading to faster convergence and stronger generalization compared to standalone RL or PID controllers. Other studies explored learning-based MPC tuning or disturbance compensation modules that adjust control parameters adaptively during flight.

Hybrid frameworks bridge the gap between model-based precision and learning-based flexibility. They enable adaptive behavior while maintaining the safety and explainability of classical controllers. However, their performance often depends on the underlying control architecture's structure and the training quality of the learning module.

Limitations: while hybrid approaches enhance adaptability and robustness, they still inherit some of the rigidity and tuning dependence of classical control. The learning components can also increase system complexity and introduce stability risks if not properly constrained [18, 19].

C. RL Value-Based Approaches

Although often considered less powerful than deep RL, tabular methods remain relevant for their simplicity and efficiency. Abo Mosali [20] introduced AMLQ, a quantized Q-learning scheme that avoids deep networks, reduces training time, and achieved lower RMSE than PID in 2D landing tasks. More recently, Goldschmid [21] applied Double Q-learning with curriculum training and motion-structured discretization. Despite being tabular, their method rivaled PI controllers and demonstrated successful transfer from simulation to hardware.

Limitations: while value-based approaches are computationally efficient, they scale poorly to high-dimensional or continuous state/action spaces and may not generalize to realistic 3D landing scenarios [20, 21].

D. RL Actor–Critic and Memory-Augmented Methods

Actor–critic frameworks dominate current research due to their ability to handle partial observability and continuous state/action spaces. Jiang [22] benchmarked DDPG and PPO, showing both can achieve near-perfect simulated success rates. Building on this, Saj [23] validated a vision-based DDPG controller on a 6-DOF ship deck, reporting nearly 100% success and outperforming nonlinear PID controllers. More recently, Aikins [24] extended PPO with an LSTM-based memory module (RPO-LSTM), which improved success by 74% under wind-disturbed conditions by better capturing hidden dynamics.

Limitations: despite high simulation success, actor–critic frameworks often require extensive training data, are sensitive to reward shaping, and may struggle with generalization to new environmental disturbances [24]. Memory-augmented architectures introduce additional computational overhead and may still fail under long-term partial observability.

E. Method Comparison

Table 1 provides a comparative overview of classical, hybrid, and learning-based control paradigms for autonomous UAV landing on moving platforms. The comparison focuses on three key dimensions: (i) **computational intensity**, (ii) **control accuracy or success rate**, and (iii) **adaptability** to changing or uncertain environments.

Classical controllers remain attractive due to their interpretability, stability, and safety guarantees. PID controllers are computationally inexpensive, relying on straightforward gain-based updates, but their performance is sensitive to environmental changes and requires careful retuning. MPC and optimal control methods optimize control inputs over a finite prediction horizon using system models, achieving high precision and constraint satisfaction [6, 7, 16, 17, 25]. Their main limitation is computational cost and reliance on accurate models, which can reduce real-time robustness in highly dynamic maritime environments.

Hybrid control strategies integrate classical methods with learning-based components, such as DDPG-tuned PID or heuristic-guided MPC [18, 19]. These approaches retain the stability and interpretability of classical control while enhancing adaptability and robustness to uncertain or changing conditions. They achieve faster convergence and higher success rates than standalone classical or RL methods, though they inherit some tuning and structural dependencies from the underlying classical controllers.

Reinforcement learning (RL) approaches remove the need for explicit system models and can handle nonlinear, uncertain dynamics. Value-based methods [20, 21] are computationally lightweight but scale poorly to high-dimensional or continuous spaces. Actor–critic methods [22–24] achieve the highest performance in complex dynamic conditions, with success rates approaching 100% in simulations, though they require extensive offline training and careful reward design. Memory-augmented variants improve performance under partial observability but add computational overhead.

The choice of control paradigm represents a trade-off between computational load, landing reliability, and adaptability to dynamic conditions. Classical PID and MPC offer predictable and interpretable performance, hybrid methods provide a balanced compromise, and RL—particularly actor-critic architectures delivers superior performance once trained.

Table 1 Comparison of control methods for autonomous UAV landing on moving platforms

Method	Computational Intensity (Online)	Accuracy / Success Rate	Adaptability / Generalization	Representative Works
Classical	High; Prediction of ship motion requires real-time optimization and initialisation.	Moderate to high (60–95%); PID performs well in stable conditions, MPC improves precision if the model is accurate	Low to moderate; PID needs retuning for new dynamics, MPC depends on model fidelity	[6, 7, 16, 17, 25]
Hybrid (Classical + RL)	Moderate; inference + classical feedback	High (85–95%) under dynamic motion	High—adapts online via learned tuning or guidance	[18, 19]
RL Value-Based	Low; table lookup or small network	Moderate to high (70–90%) in simple tasks	Low to moderate; poor scalability to continuous 3D	[20, 21]
RL Actor–Critic	Moderate (inference only) after high-cost training	Very high (90–100%) in simulation	High—learns nonlinear and uncertain dynamics; sensitive to training	[22–24]

IV. Research Objectives and Questions

In this section the research gaps are introduced in subsection IV.A following the literature review from section III. Following the identified research gaps, the research questions are introduced in subsection IV.B.

A. Research Gap

Classical and learning-based methods have demonstrated promising performance for UAV-ship landings, a key gap remains in their evaluation under realistic and challenging maritime conditions. Classical controllers, such as PID and MPC, have been primarily studied for sea states up to sea state 5) [6, 7, 16, 17], whereas reinforcement learning (RL) methods, particularly actor–critic architectures [22–24], are typically trained in environments with planar target motion, lacking stochastic vertical heave.

This raises the opportunity to investigate whether RL controllers trained on stochastic heave ship motion, can outperform classical methods under high sea states and in the presence of environmental disturbances. Addressing this gap would provide insight into the potential advantages of intelligent control strategies for robust UAV-ship landing in highly dynamic maritime environments.

B. Research Objective

The objective of this research is to design and evaluate the performance of a reinforcement learning (RL) based UAV landing controller for autonomous UAV ship landings, under stochastic heave motion and environmental disturbances.

C. Main Research Question

How does a RL-based guidance controller for autonomous UAV ship landings perform in terms of success rate, landing precision and robustness when subject to high sea states and disturbances?

D. Sub-questions

1) **RL-type:** Does a hybrid RL guidance controller have benefits compared to a full RL guidance controller?
2) **Robustness:** How do wind, sea state, and sensor noise affect the success rate, landing precision, and stability of each controller?
3) **Comparative Analysis:** Which controller demonstrates the best trade-off between robustness, precision, and efficiency across varying environmental and dynamic conditions?

V. Baseline Landing Strategy

In this section the baseline landing strategy is presented. The baseline guidance controller uses a classical PID strategy inspired by the baseline strategy of [6]. Its purpose is to provide a stable, interpretable reference for evaluating the RL-based guidance system. PID control is widely used in aerial robotics for its simplicity, robustness, and computational efficiency [26, 27], making it suitable for high-frequency real-time applications such as UAV landing on moving platforms. More advanced methods (MPC, SMC, FLC) offer improved tracking or disturbance rejection but with higher computational cost [28–30]. Horizontal motion is governed by a standard PID law regulating the horizontal velocity based on relative horizontal position and velocity. The vertical control uses a PD law that takes into account the relative altitude and vertical velocity.

$$\begin{aligned} \vec{V}_{cmd,xy} &= K_P^{xy}\vec{e}_{xy} + K_I^{xy}\int \vec{e}_{xy}\,dt + K_D^{xy}\vec{v}_{xy} \\ \vec{V}_{cmd,z} &= K_P^{z}\vec{e}_{z} + K_D^{z}\vec{v}_{z,\text{rel}} \end{aligned} \tag{4}$$

where K_P^{xy}, K_I^{xy}, K_D^{xy}, K_P^z and K_D^z are the gains of the baseline controller. The baseline PID gains are listed in Table 2, tuned manually for horizontal motion and via grid search for vertical motion. Systematic tuning methods (Ziegler–Nichols, Cohen–Coon) are outside the scope of this work. The vertical guidance is initiated once the UAV is above the target and the target is in a downward motion, allowing for a safe and rapid descent at high altitudes and smooth deceleration near the landing zone.

Table 2 PID gain values for the horizontal (X, Y) and vertical (Z) controllers.

Axis	K_p	K_i	K_d
X (Horizontal)	1.00	0.20	0.08
Y (Horizontal)	1.00	0.20	0.08
Z (Vertical)	-0.15	0.00	-0.15

Furthermore, a clipping function and first-order low-pass filter is applied to smooth the velocity command and prevent abrupt changes with $\alpha_{filter} = 0.1$ and $V_{max} = 4m/s$.

VI. Reinforcement Learning Landing Strategies

This section presents the reinforcement learning (RL) framework developed for autonomous UAV ship landings. It is organized into five parts to systematically describe how the RL agent interacts with and learns from its environment. subsection VI.A introduces the theoretical foundation of reinforcement learning based on the Markov Decision Process (MDP) and its partially observable variant. subsection VI.B defines the agent's action space, describing both the hybrid and full 3D RL controllers and the applied command filtering. subsection VI.C details the observation space, which encodes the UAV–target state information available to the agent. subsection VI.D explains the design of the reward function that drives learning toward safe and accurate landings. Finally, subsection VI.E presents the Soft Actor–Critic (SAC) algorithm used for policy optimization and training stability.

A. Markov Process

Reinforcement Learning is a machine learning framework where an agent interacts with an environment to maximize cumulative reward. The environment is typically modeled as a *Markov Decision Process (MDP)* [31]:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$$

where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $P(s'|s,a)$ the transition probability, $R(s,a)$ the reward function, and γ the discount factor. The *Markov property* assumes that that the next state depends only on the current state and action:

$$P(s_{t+1}|s_t, a_t) = P(s_{t+1}|s_t, a_t, \dots)$$

In many real-world continuous control tasks, however, the Markov property does not strictly hold [31]. The agent does not have direct access to the full environmental state s_t, but instead only to a partial or noisy observation $o_t \in O$. This situation is modeled as a *Partially Observable Markov Process (POMP)* [32, 33]. A POMP is defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, O, O, \gamma)$, where O is the observation space and $O(o|s)$ denotes the observation probability distribution.

In this setting, the agent must infer useful information about the hidden environment state from a history of past observations and actions, rather than relying on a single observation. This makes the problem more challenging and typically requires memory-based models (e.g., recurrent neural networks) or state-estimation techniques (e.g., filters, belief states) to approximate the underlying Markov process [34, 35].

B. Action Space

In this subsection, two different RL-controllers are presented. The presented problem involves both horizontal actions: i.e. tracking of the moving landing target and vertical actions: i.e. a safe and reliable descend on the moving target.

*1. **Hybrid RL 1D***

This agent has a 1 dimensional action space and only has autonomy over the vertical command velocity. While having the horizontal guidance performed by a PID controller earlier introduced in section V.

$$\mathbf{a}_t = \begin{bmatrix} V_{z,cmd} \end{bmatrix} \quad \in [-1, 1]$$

*2. **RL 3D***

This agent has full 3D action autonomy over the command velocity to guide the drone to the target.

$$\mathbf{a}_t = \begin{bmatrix} V_{x,cmd} \\ V_{y,cmd} \\ V_{z,cmd} \end{bmatrix} \quad \in [-1, 1]$$

*3. **Action Command Filtering***

In the environment, both RL_1D and RL_3D actions $a_t \in [-1, 1]$ are passed through a command filter. The RL agent outputs an action a_t in the range $[-1, 1]$, which is scaled to the maximum UAV velocity $V_{\max}$. A first-order low-pass filter is applied to smooth the command and prevent abrupt changes with $\alpha_{filter} = 0.1$ and $V_{max} = 4m/s$:

$$V_{cmd} \leftarrow \alpha_{filter}\,(a_t \cdot V_{max}) + (1-\alpha)V_{\text{cmd}}, \quad \alpha_{filter} \in [0, 1].$$

In short, this process scales, smooths, and saturates the RL action before execution.

C. Observation Space

Both RL controllers **RL 1D** and **RL 3D** observe the environment through a 15-dimensional vector capturing the relative state between the drone and the landing target. The observation vector is defined as the 15D vector:

$$\mathbf{o}_t = \begin{bmatrix} \mathbf{p}_{\text{rel}} & \mathbf{v}_{\text{rel}} & \mathbf{v}_{\text{d}} & \eta_{\text{rel}} & e_{\text{xy}} & \int_0^t e_{\text{xy}} dt & T \end{bmatrix}^T \quad (5)$$

where:

- $\mathbf{p}_{\text{rel}} = [x_{\text{rel}} \quad y_{\text{rel}} \quad z_{\text{rel}}]^T$ represents the 3D relative position between the drone and the target.
- $\mathbf{v}_{\text{rel}} = [v_{\text{rel, x}} \quad v_{\text{rel, y}} \quad v_{\text{rel, z}}]^T$ represents the 3D relative linear velocity between the drone and the target.
- $\mathbf{v}_{\text{d}} = [v_{\text{d, x}} \quad v_{\text{d, y}} \quad v_{\text{d, z}}]^T$ represents the 3D linear velocity of the drone.
- $\eta_{\text{rel}} = [\phi_{\text{d}} \quad \theta_{\text{d}} \quad \psi_{\text{d}} - \Psi_t]^T$ represents the relative 3D roll, pitch and heading between the drone and the target.
- $e_{\text{xy}} = \left\| \begin{bmatrix} x_{\text{rel}} \\ y_{\text{rel}} \end{bmatrix} \right\|$ represents the horizontal tracking error.
- $\int_0^t e_{\text{xy}} dt$ represents the integral horizontal tracking error.
- T is the normalized time of the episode.

The observation space consists of different measurements all with different scales. In order to generate a better policy, the observations are clipped and MinMaxScaled to the [-1 , 1] domain.

D. Reward Function

The reward function encourages the agents to track the target accurately while minimizing excessive relative roll and heading attitudes. This structure balances tracking accuracy, smooth control, and safety, which is critical for continuous control in a dynamic maritime environment. The design follows a common approach in UAV and robotic landing tasks, where a shaping reward provides dense feedback during training and a terminal success reward reinforces goal completion [31, 36].

$$r_t = r_{\text{shaping}} + r_{\text{success}}$$

where:

$$r_{\text{shaping}} = 0.05 \left(1 - e_{\text{xy}} - 0.5 \int^t e_{\text{xy}} dt - 0.5 \|\eta_{\text{rel}}\|\right)$$

$$r_{\text{success}} = 100 + \frac{50}{1 + e_{\text{xy}}^2} + \frac{50}{1 + \|\mathbf{v}_{\text{rel}}\|^2} + \frac{50}{1 + \|\eta_{\text{rel}}\|^2}$$

The shaping component provides continuous guidance toward the target throughout each episode, while the terminal success reward strongly reinforces precise alignment and smooth relative motion upon landing. This hybrid structure has been shown to improve learning stability and convergence in complex continuous control problems. To ensure the agent prioritizes successful landings over small incremental improvements, the terminal reward is made dominant ($r_{\text{success}} \gg r_{\text{shaping}}$), as recommended by [31].

E. Policy: Soft Actor–Critic (SAC)

The Soft Actor-Critic (SAC) algorithm [37] combines elements of actor–critic methods with the maximum entropy framework for reinforcement learning. In standard reinforcement learning (RL), the objective is to learn a parameterized policy π that maximizes the expected cumulative return over an episode of length T:

$$\max_{\pi} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}\left[\sum_{t=0}^{T} r(s_t,a_t)\right],$$

where $\rho_\pi(s_t)$ denotes the state distribution induced by policy $\pi(a_t|s_t)$. The action-value function $Q(s,a)$ expresses the expected future return after taking action a in state s. In conventional RL, the learned policy is typically unimodal and centered around the action that maximizes $Q(s,a)$. This narrow focus can reduce exploration, which makes the agent brittle: even small environmental changes may cause failure.

Maximum entropy reinforcement learning addresses this limitation by augmenting the objective with an entropy term that explicitly encourages exploration [37]. The modified objective is:

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

where $\mathcal{H}(P) = \mathbb{E}_{x\sim P}\left[-\log P(x)\right]$ is the entropy and α is a temperature parameter controlling the trade-off between reward maximization and policy stochasticity. This objective induces policies of the form

$$\pi(a|s) \propto \exp\big(Q(s,a)\big),$$

The SAC algorithm [37] builds on this principle and integrates three key components: (i) an actor–critic architecture with separate policy and value networks, (ii) an off-policy learning procedure that reuses past experience for sample efficiency, and (iii) entropy maximization to balance stability and exploration. Compared to earlier methods such as Deep Deterministic Policy Gradient (DDPG) [38], SAC achieves improved stability, faster convergence, and superior asymptotic performance in continuous, high-dimensional control tasks. By combining sample efficiency with robustness to environmental variability, SAC addresses the brittleness that often affects deep reinforcement learning algorithms.

The SAC implementation in this work follows the default *Stable-Baselines3* configuration [39], with standard continuous control hyperparameters for stable and reproducible training. The entropy coefficient α and target entropy are set to `"auto"` to adaptively balance exploration and exploitation, stabilizing learning. Table 3 lists the hyperparameters used for training the RL guidance policy.

Table 3 Default hyperparameters for SAC in Stable-Baselines3

Hyperparameter	Default
γ (discount)	0.99
Learning rate	0.0003
Buffer size	1×10^6
Learning starts	100
Batch size	256
τ (soft update)	0.005
α (entropy)	`"auto"`
Target entropy	`"auto"`
Architecture	2×256
Activation	Tanh
Action noise	$\mathcal{N}(0,\ .05)$

*1. **Landing Strategies Overview***

In Table 4, the axis-wise control formulation for all 3 proposed landing strategies are presented.

Table 4 Overview of landing strategies and axis-wise control formulation.

Strategy	X-axis	Y-axis	Z-axis
Baseline (PID)	PID	PID	PD
RL 1D (Hybrid)	PID	PID	RL
RL 3D (Full RL)	RL	RL	RL

VII. Case Study

In this chapter, the case study is presented where the hybrid RL_1D and RL_3D guidance controllers are evaluated together with the classical baseline. Firstly, in subsection VII.A the assumptions for this case study are listed. Secondly in subsection VII.B the target motion of the ship in high sea states is presented. Furthermore in subsection VII.D, a drone model is presented together on which the RL and baseline guidance controllers will be evaluated.

A. Assumptions

In order to limit the scope of this work to the research objective, a few assumptions have to be made.

- Assume target motion of ship in high sea states is only subject to stochastic heave motion i.e. no rotational motion, surge and sway.
- Assume constant wind that has no downwash related to the ship's wake.
- Furthermore, we assume that a landing is considered successful if:

$$Success = \begin{cases} \|\boldsymbol{p}_{\text{rel},\,xy}\| < 2 \text{ m} \\ |p_{\text{rel,z}}| < 0.05 \text{ m}, \\ v_{z,\,\text{rel}} < 2.8 \text{ m/s}, \\ |\phi_d| < 30^\circ \end{cases}$$

where $\boldsymbol{p}_{\text{rel},\,xy} = [x_{\text{rel}}\ y_{\text{rel}}]^T$, z_{rel} is the vertical (altitude) error, $v_{z,\,\text{rel}}$ is the relative vertical velocity between the drone and the platform, and ϕ_d is the roll angle of the drone at touchdown. The threshold $v_{z,\,\text{rel}} < 2.8$ m/s is selected based on experimental studies of quadrotor landing dynamics that show higher vertical velocities increase structural loads and risk of bounce or tip-over [40, 41] and a roll angle below 30° is commonly used to limit tipping risk and maintain landing stability [42].

B. Target Motion

The maritime landing target is modeled as a 4 m diameter landing platform located on the aft section of a moving ship. For simplicity, the ship motion is considered to be linear in the horizontal plane and restricted to vertical heave motion, while horizontal surge, sway, and rotational motions are assumed to be zero. The modeling follows a linear spectral approach, where the sea state is represented by a wave energy spectrum $S(\omega)$ and the ship heave response $Z(\omega)$ is obtained from the Response Amplitude Operator (RAO):

$$Z(\omega) = |\text{RAO}_Z(\omega)|^2\, S(\omega). \tag{6}$$

In this work, the heave RAO is based on ShipMo3D analysis from [15] for a $L = 124$ m Canadian Patrol Frigate (CPF) subject to head waves. Furthermore, the wave energy spectrum is based on the Bretschneider wave spectrum representation, a two-parameter formulation commonly used for fully developed seas [43]:

$$S(\omega) = \frac{5}{16} H_s^2\, \omega_p^4\, \omega^{-5} \exp\left[-\frac{5}{4}\left(\frac{\omega_p}{\omega}\right)^4\right], \tag{7}$$

where H_s is the significant wave height and $\omega_p = 2\pi/T_p$ is the peak angular frequency associated with the modal period T_p. Representative values of H_s and T_p for sea states 4–6, based on the Douglas scale, are listed in Table 5. These parameters define the stochastic excitation spectrum $S(\omega)$ used to drive the ship motion model.

Table 5 Representative wave parameters for sea states 4–6 obtained from [44].

Sea State	H_s (m)	T_p (s)
4	4	3
5	6	4
6	8	5
7	10	6

The heave response is given in Figure 1. Note that for higher sea states, the heave motion has more energy at low frequencies.

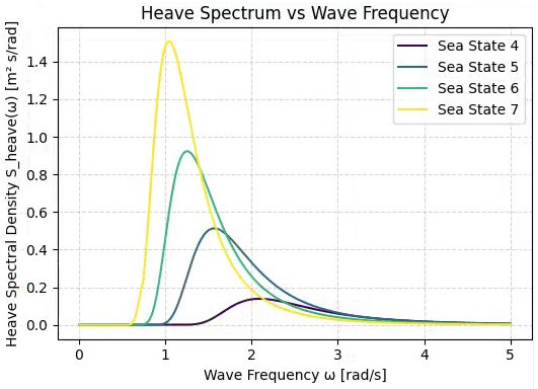


Fig. 1 Heave motion $Z(\omega)$ spectrum as a function of sea state.

In order to obtain the time response, the following transformation is used (Equation 8), with ω_n the discretized angular frequencies, $\Delta\omega$ the frequency step, and $\phi_n \sim U(0, 2\pi)$ independent random phases.

Furthermore, a base altitude z_0 is selected to limit negative values of altitude.

$$z_t(t) \approx z_0 + \sum_{n=1}^{N} \sqrt{2\, Z(\omega_n)\, \Delta\omega}\, \cos\left(\omega_n t + \phi_n\right), \tag{8}$$

Since rotational motions are neglected, the target state estimation consists only of position and linear velocity. The horizontal motion is linear, meaning that the velocity in x and y is constant and solely dependent on the ship's speed V and heading Ψ.

C. Sensor Noise

It is assumed that the target state is estimated using sensor fusion of a Camera, LiDAR and IMU, which are filtered through an Extended Kalman Filter [45, 46]. However, the filtered state estimate contains residual errors modeled as:

$$\hat{\boldsymbol{x}}_t = \boldsymbol{x}_t + \boldsymbol{w}_t \tag{9}$$

where $\hat{\boldsymbol{x}}_t$ is the estimated target state, $\boldsymbol{x}_t$ is the true state, and $\boldsymbol{w}_t$ represents the sensor noise. The target state estimation error is sampled from $\mathcal{N}(0, 0.1m)$ by default [47].

D. Drone Model

The Crazyflie 2.1 nano quadcopter was chosen as the primary candidate due to its compatibility with the open-source PyBullet Drones library in Python [48]. This library provides a realistic physics simulation environment for aerial vehicles, enabling rapid prototyping and evaluation of flight controllers before real-world deployment. The seamless integration of the Crazyflie model within PyBullet facilitates the creation of customizable environments, reducing development overhead while ensuring reproducibility of experiments. Furthermore, the Crazyflie platform is widely used in research and education owing to its open hardware and software ecosystem, making it a well-documented and reliable choice for both simulation and real-world experiments [49].

Table 6 Main physical and aerodynamic parameters of the Crazyflie 2.1 [50]

Symbol	Value	Unit
L	0.0397	m
m	0.027	kg
r_p	2.31×10^{-2}	m
T/W	2.25	–
$v_{\max}$	30	km/h
k_f	3.16×10^{-10}	N·s^2
k_m	7.94×10^{-12}	N·m·s^2
k_{ge}	11.37	–
$C_{d,xy}$	9.18×10^{-7}	–
$C_{d,z}$	1.03×10^{-6}	–
I_{xx}, I_{yy}	1.4×10^{-5}	kg·m^2
I_{zz}	2.17×10^{-5}	kg·m^2

1. Drone Control

The Crazyflie 2 drone from `gym-pybullet-drones` [50] includes an inner-loop controller that tracks a velocity command signal, implemented in `./control/DSLPID.py` [51]. This cascaded controller computes the necessary thrust and attitude commands to track the guidance signal.

An overview of the cascaded controllers used are given in Figure 10 and Figure 11.

2. Environmental Disturbance

To emulate realistic outdoor flight conditions, the drone is subjected to a constant horizontal wind disturbance. The wind speed is randomly sampled between 1 m/s and 3 m/s, and the wind direction is uniformly distributed between $-90°$ and $+90°$. The drag force due to wind is modeled as a point force Equation 10 in `gym-pybullet-drones` next to the aerodynamic drag formulation implemented in the `BaseAviary` class [50]. This model computes the aerodynamic forces based on the relative air velocity $V_{rel} = v_d - v_{wind}$ between the drone and the wind field, inducing realistic drift effects that the guidance modules must compensate for during flight and landing.

$$F_{drag} = \frac{1}{2}\rho V_{rel}^2 C_{d,xy} A \tag{10}$$

E. Simulation Environment

The simulation environment is implemented in `PyBullet` using the `VelocityAviary` wrapper for drone dynamics, coupled with a custom platform motion model. The drone is initialized at a fixed altitude and distance from a moving platform, whose motion is derived from a wave-response model parameterized by the sea state.

To ensure robustness and promote generalization across varying operational conditions, extensive domain randomization is applied during training. This is essential for reinforcement learning (RL) methods, as their ability to generalize to unseen environments and disturbance scenarios is critical for safe real-world deployment. Therefore, environmental parameters such as sea state, wind, platform motion, and sensor noise are randomized at the start of each episode.

In addition, the thrust-to-weight ratio (T/W) is sampled uniformly from $\mathcal{U}(1.35, 1.85)$ to emulate different drone configurations and dynamic responses. This variation encourages the RL agent to learn a policy that remains effective across a wide range of physical models, rather than overfitting to a single drone configuration.

The simulation is controlled via a set of arguments (`ARGS`) parsed at runtime. Table 7 summarizes the default training configuration. At the start of each episode, both the drone and platform states are reset. The ship platform motion is generated from a stochastic sea-state response

amplitude operator (RAO) model or fixed. Measurement noise $\mathcal{N}(0, \sigma^2)$ can be adjusted to represent target state estimation uncertainty.

Table 7 Default simulation arguments for RL-Guidance agent.

Argument	Default Value
Simulation frequency	48 Hz
Control frequency	48 Hz
Episode steps	1000
Duration	20.8 s
Sea state	$\sim \mathcal{U}(4, 7)$
Initial altitude	8 m
Wind velocity	$\sim \mathcal{U}(1, 3)$ m/s
Wind direction	$\mathcal{U}(-90, 90)$ deg
Initial distance	3 m + $\mathcal{U}(-1, 1)$ m
Target heading	$\sim \mathcal{U}(-45, 45)$ deg
Target speed	$\sim \mathcal{U}(2.5, 3.5)$ m/s
Thrust-to-weight ratio (T/W)	$\sim \mathcal{U}(1.35, 1.85)$
Sensor noise	$\sim \mathcal{N}(0, 0.1)$
Model	SAC
Train steps	4e6

F. Evaluation Metrics

The performance of the RL-Guidance module is evaluated over N independent runs. The following metrics are used to assess landing performance:

- **Success Rate (%):** The ratio of successful landings to the total number of runs, expressed as a percentage:

$$\text{Success Rate} = \frac{N_{\text{success}}}{N} \times 100$$

  where N_{success} denotes the number of successful landings.
- **Landing Error (m):** The Euclidean distance in the horizontal plane between the UAV touchdown point (x_d, y_d) and the target (x_t, y_t):

$$e_{xy} = \sqrt{x_{\text{rel}}^2 + y_{\text{rel}}^2}.$$

- **Relative Vertical Velocity (m/s):** The UAV's vertical velocity relative to the platform at the instant of contact:

$$v_{z,\text{rel}} = v_{z,d} - v_{z,t}.$$

- **Relative Roll Angle (deg):** The UAV's roll angle relative to the platform at the instant of touchdown:

$$\phi_{\text{rel}} = \phi_d.$$

- **Landing Time (s):** The elapsed time from the start of the episode until touchdown, providing a measure of guidance efficiency.

VIII. Hypotheses

The hypotheses formulated below are derived from the research questions outlined in section IV and the analysis of the related literature in section III. They are based on the expectation that Reinforcement Learning (RL) controllers, particularly those trained on the identified research gap of stochastic heave, will demonstrate superior adaptability compared to classical methods.

1) **H1: Main Hypothesis (Performance in High Sea States)**
   The Reinforcement Learning guidance controllers trained on stochastic ship heave, will achieve a significantly **higher success rate** and demonstrate superior landing precision compared to the classical PID-based baseline controller when evaluated under high sea states (i.e., above Sea State 5) and strong wind disturbances.
2) **H2: RL-Type Comparison (Full vs. Hybrid)**
   The full RL guidance controller RL_3D will ultimately demonstrate a higher maximum success rate and precision than the hybrid RL guidance controller RL_1D. This is because its full autonomous coordination across all three degrees of freedom (DOF) allows for better generalized control and disturbance rejection in a non-linear, dynamic environment.
3) **H3: Robustness to Disturbances**
   The full RL controller RL_3D will exhibit the highest robustness to increasing levels of environmental disturbance (wind speed, sea state intensity) and sensor noise. Conversely, the classical baseline controller will be the most negatively impacted, showing the steepest decline in both success rate and stability.
4) **H4: Trade-off Analysis (Efficiency)**
   Despite the superior peak performance of the full RL controller, the hybrid RL controller RL 1D will demonstrate the best trade-off between robustness, landing precision, and computational efficiency. Its lower-dimensional action space and reliance on the stable PID for horizontal control will result in a more efficient control strategy that still significantly outperforms the classical baseline.

IX. Results

A. Training Results

The training progression of the RL_1D and RL_3D agents are illustrated in Figure 2 and moving average summarized in Table 8. The training behavior between the agents exhibits significant differences in convergence. The RL_1D policy achieved a slightly higher reward (241.64 vs. 228.34) and success rate (86.6% vs. 76.8%) compared to RL_3D, while the RL_3D agent reached its peak performance later in training (episode 23338 vs. 12664) and achieved slightly faster landings (2.70 s vs. 3.31 s). Overall, RL_1D demonstrates more stable and consistent learning, whereas RL_3D achieves quicker but less reliable landings.

Metric	RL_1D	RL_3D
Reward	241.64	228.34
Episode nr.	12664	23338
Landing Time	3.31s	2.70s
Success Rate	86.6%	76.8%

Table 8 Training results at the episode with the highest 200 moving-average success rate for RL_1D and RL_3D.

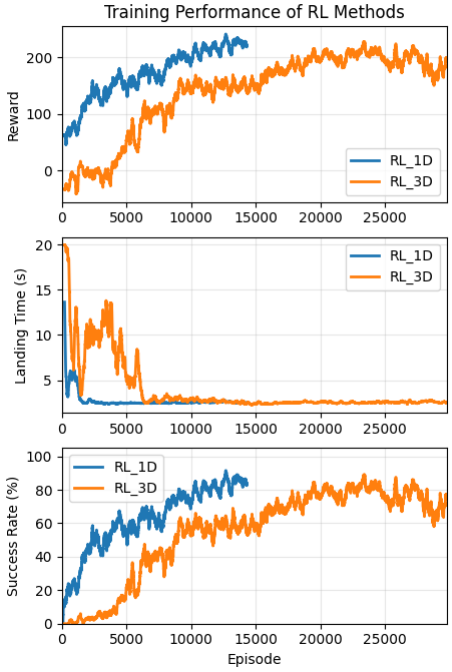


Fig. 2 200-episode moving averages for training reward, episode length (s) and success rate progression of both RL methods during training.

B. Policy Evaluation

The final performance of the trained RL_1D and RL_3D agents, together with the baseline controller, was quantitatively evaluated over $N = 200$ independent runs in the standard simulation environment described in Table 7. Figure 4 provides a visual summary of all evaluation metrics, while Table 9 presents their mean and standard deviation values.

Metric	RL_1D	RL_3D	Baseline
Success (%)	86.6	76.8	79.2
Time (s)	3.31 ± 0.60	2.70 ± 1.21	11.51 ± 2.79
Roll (°)	−0.51 ± 4.39	8.04 ± 13.80	−0.07 ± 3.37
Rel. Vel (m/s)	−1.94 ± 0.71	−2.00 ± 0.76	−2.01 ± 1.05
Err XY (m)	0.97 ± 0.30	1.43 ± 3.34	0.17 ± 0.17

Table 9 Evaluation metrics (mean ± std) for RL controllers and baseline evaluated over $N = 200$ runs.

C. Success Rate

The success rate is a direct measure of each controller's ability to complete the landing task without failure or significant deviation from the desired target state. The RL_1D agent achieved the highest success rate of 86.6%, demonstrating strong robustness and consistent control in the one-dimensional descent scenario. The RL_3D agent, with a success rate of 76.8%, struggled slightly due to the added control complexity of managing coupled translational and rotational motion. In contrast, the baseline controller achieved a comparable success rate of 79.2%, suggesting that classical control remains effective in stable conditions but lacks adaptability to dynamic disturbances. These results indicate that while RL_1D exhibits the highest reliability, future improvements in the RL_3D policy could further enhance multi-axis robustness.

D. Landing Time

Landing time reflects both trajectory efficiency and the agent's willingness to trade stability for speed. The RL_3D agent demonstrated the fastest average descent of 2.70 s, highlighting its aggressive optimization for time-efficient trajectories. The RL_1D agent followed with a slightly slower average of 3.31 s, representing a more balanced trade-off between descent rate and stability. The baseline controller was significantly slower (11.51 s), adopting a cautious descent profile that prioritizes safety and precision over speed. Both RL agents demonstrate learned time efficiency, with RL_3D achieving the most rapid yet variable descents.

E. Attitude Stability

Attitude stability, measured by the roll angle at touchdown, reflects the controller's ability to maintain a level orientation during landing. The RL_1D and baseline controllers achieved nearly level touchdowns (-0.51° and -0.07°, respectively), signifying precise attitude control. The RL_3D agent exhibited a noticeably larger mean roll angle (8.04°), with higher variance, likely due to the increased degrees of rotational freedom. Although the 3D policy successfully manages coupled dynamics, its larger roll deviations suggest the need for improved rotational damping or reward shaping to encourage smoother alignment before touchdown.

F. Vertical Velocity at Touchdown

The relative vertical velocity metric assesses descent smoothness and impact mitigation capability. All controllers achieved similar touchdown velocities around -2 m/s (-1.94 m for RL_1D, -2.00 m/s for RL_3D, and -2.01 m/s for the baseline). This consistency indicates that all methods effectively learn or enforce a stable terminal descent phase. However, RL_3D exhibited a slightly higher standard deviation, implying occasional aggressive descents when attempting to correct lateral or rotational deviations late in the trajectory.

G. Landing Precision

Lateral error quantifies the accuracy of the vehicle's final touchdown position relative to the desired target point. The baseline controller achieved the smallest average error (0.17 m), owing to its conservative behavior—only descending once directly aligned over the target. The RL_1D and RL_3D agents exhibited larger mean errors (0.97 m and 1.43 m), respectively. This can be attributed to their ability to initiate descent while performing simultaneous lateral corrections, trading precision for overall efficiency. Notably, the RL_1D controller shows lower deviation and variance compared to RL_3D, suggesting that constrained motion reduces accumulated lateral drift. Despite this, the RL-based controllers provide a desirable balance between time efficiency and spatial accuracy, suitable for dynamic landing scenarios.

H. Trajectory Visualization

Figure 3 illustrates a representative landing scenario under Sea State 5 conditions. The RL_1D controller successfully guides the UAV to a safe touchdown, effectively compensating for strong heave-induced vertical motion. In contrast, the PID baseline fails, crashing due to excessive vertical velocity, highlighting the advantage of reinforcement learning in managing extreme maritime disturbances. The velocity and attitude profiles with the commands can be seen in the appendix Figure 12.

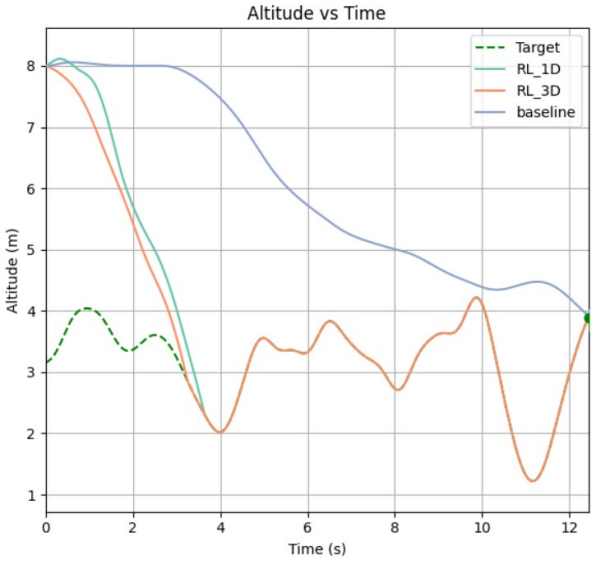


Fig. 3 Trajectory where RL_1D and RL_3D methods land successfully but the baseline crashes due to high vertical velocity. Target motion 3 m/s, Ψ =30° and sea state 5.

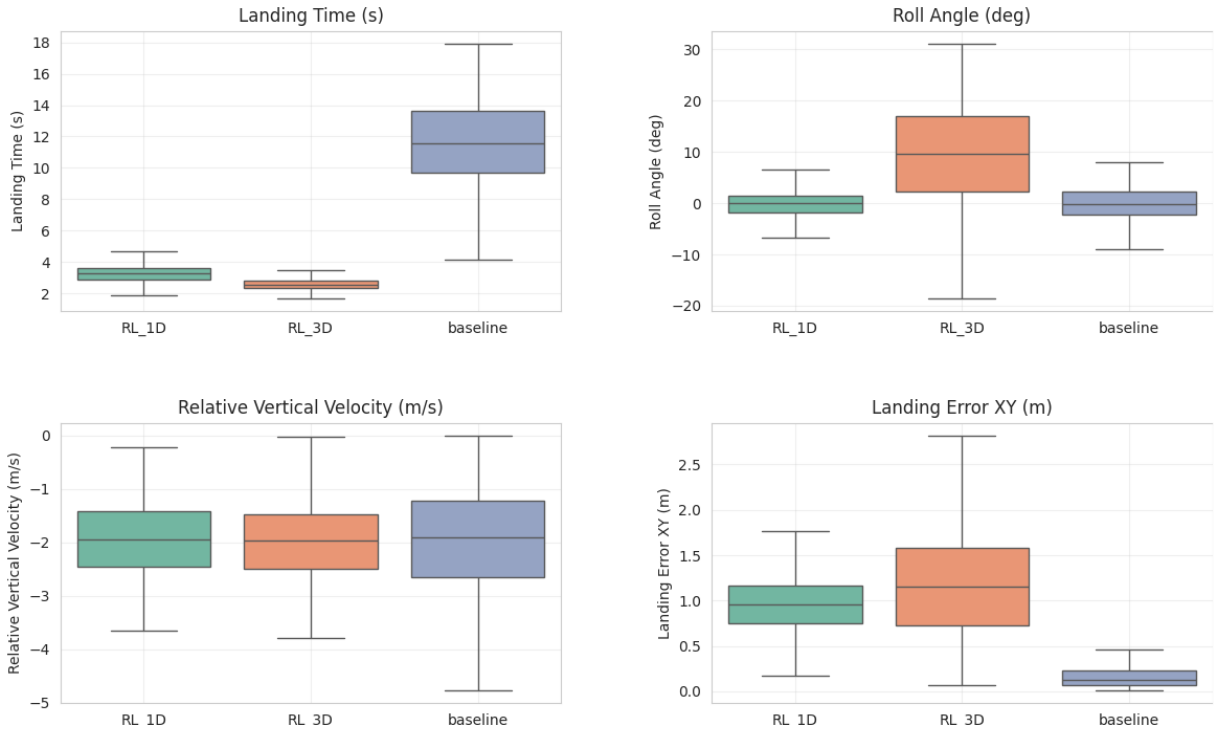


Fig. 4 Box plots of landing time, roll angle, relative vertical velocity, and landing error (XY) for RL_1D, RL_3D, and the baseline controller, evaluated over $N = 500$ runs per method.

X. Validation

The validation process of this work is assessed by performing sensitivity analysis to ensure that the learned guidance policy provides results representative of realistic operational conditions.

A. Wind Disturbance Sensitivity

The sensitivity analysis of the controllers under varying wind conditions demonstrates their robustness with respect to landing success and positional accuracy, as summarized in Figure 5 and Table 10. In the trained wind regime ($\sim \mathcal{U}(1-3$ m/s)), both RL_1D and RL_3D achieve high success rates (86.6% and 76.8%, respectively) with low landing errors (0.97 m and 1.43 m), showing effective generalization within the conditions seen during training. The baseline PID controller performs consistently with a success rate of 79.2% and minimal landing error (0.17 m), indicating reliable classical control under familiar wind conditions.

At moderate wind speeds (4 m/s), RL_1D maintains a high success rate (79.5%) with low landing error (1.19 m), while RL_3D shows a significant drop in success (57.5%) and a sharp increase in landing error (44.69 m), reflecting the challenge of controlling landings outside the trained wind envelope. Notably, the baseline controller remains stable at 72.5% success and low landing error (0.25 m), demonstrating superior robustness to moderate unmodeled disturbances compared to the RL controllers.

For higher wind speeds (5 m/s and above), the RL controllers experience further decreases in success rate and increases in landing error: at 5 m/s, RL_1D achieves 54% success with 1.40 m landing error, and RL_3D drops to 26% success with 26.16 m landing error. In contrast, the baseline PID controller maintains a stable 71% success rate with 0.35 m error, illustrating its resilience even under extreme wind conditions. At 6 m/s, both RL controllers show substantial instability (RL_1D: 32%, RL_3D: 7.5% success; errors exceeding 77 m), whereas the baseline PID controller continues to perform robustly (75% success, 0.42 m error). This indicates that, although RL controllers offer advantages within the trained wind regime, the classical PID controller provides superior robustness under untrained disturbances.

Additional metrics such as landing time, roll angle, and relative vertical velocity are reported in Table 10, supporting the trends observed in success rate and landing error. The visual representation in Figure 5 highlights the relative strengths of each controller across increasing wind speeds.

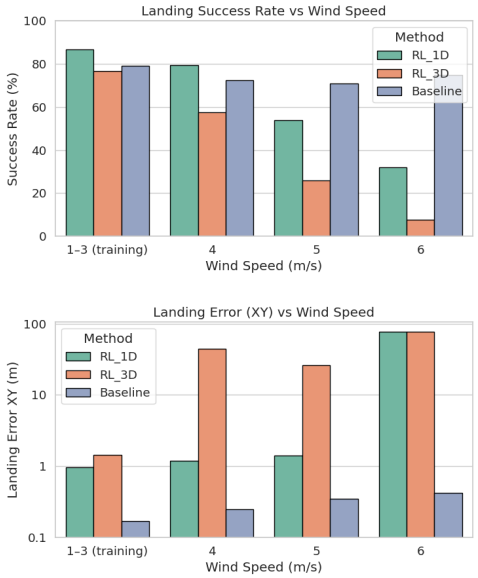


Fig. 5 Success rate and landing error sensitivity w.r.t increased wind speeds. $\sim \mathcal{U}(1-3)$ denotes the wind conditions under training.

B. Baseline Strategy

The performance of the baseline controller is highly dependent on the choice of the vertical proportional gain (K_p^Z) and the derivative gain (K_d^Z). A sensitivity analysis was conducted to determine the most robust and safest combination of these gains presented in Figure 6. The analysis involved running 50 simulations for each combination of K_p and K_d across a range of values from -0.05 to -0.30. The performance metric used was the success rate, across the 50 runs.

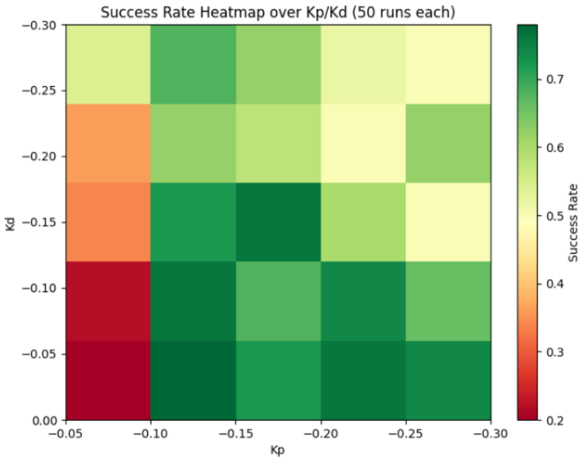


Fig. 6 Grid search of vertical baseline guidance proportional and derivative gains over $N = 50$ runs.

As observed in the analysis, the highest success rates (dark green regions) were concentrated around the central and lower-right quadrants of the tested domain. Based on the quantitative results, the gain combination of $K_p = -0.15$ and $K_d = -0.15$ was selected as the optimal and safest choice for implementation. This center point provided a consistent high success rate, offering a balance between fast response (derivative action) and stability (proportional action), ensuring robust landing performance.

C. Noise Disturbance Sensitivity

To evaluate robustness to sensor uncertainty, the controllers were tested with increasing levels of sensor noise, modeled as $\sim \mathcal{N}(0, \sigma)$ where $\sigma \in [0.1, 1.25]$ m. The full evaluation data is given in Table 11. The corresponding success rates are visualized in Figure 7, showing a clear trend of performance degradation as measurement noise increases. At low noise levels ($\sigma = 0.1$ m), all controllers achieve high reliability, with RL_1D, RL_3D, and the baseline reaching 88%, 75%, and 89% success respectively. As noise increases to moderate levels ($\sigma = 0.5$–1.0 m), RL_3D exhibits the most pronounced sensitivity, dropping from 72% to 30% success, while RL_1D maintains a relatively stable 78–79% success rate. The baseline controller remains consistent around 76–83%, demonstrating robustness to moderate sensor disturbances but at the cost of longer landing times observed elsewhere.

Under extreme noise ($\sigma = 1.25$ m), the performance divergence becomes substantial: RL_3D nearly collapses (7% success), the baseline controller degrades to 39%, while RL_1D still achieves 78% success. This indicates that the 1D-trained policy learned a more stable mapping from noisy sensory inputs to control actions, effectively filtering sensor disturbances through its learned state abstraction.

The results suggest that while the baseline maintains stability through conservative control gains, RL_1D demonstrates superior robustness to sensor uncertainty within its trained operational regime, whereas RL_3D suffers from over-dependence on high-dimensional sensory feedback.

D. Sea State Sensitivity

To assess controller robustness under varying sea conditions, simulations were conducted at sea states 0, 4, 5, 6, and 7. The evaluation data is given in Table 12. The resulting success rates and landing errors are visualized in Figure 8.

At low to moderate sea states (0–5), all controllers achieve reasonable success, with RL_1D maintaining the highest consistency (85–88%). RL_3D shows a moderate decline (75–83%), while the baseline exhibits variable performance.

As sea state increases to 6–7, RL_1D remains robust, achieving 86–89% success, whereas RL_3D continues to slightly decline and the baseline controller drops to 69–73%. Landing errors increase with sea state, particularly for RL_3D and the baseline, indicating higher difficulty in maintaining precise landings under more turbulent conditions.

These results highlight that RL_1D's policy generalizes well across moderate to high sea states, while RL_3D and the baseline controllers are more sensitive to sea-induced disturbances.

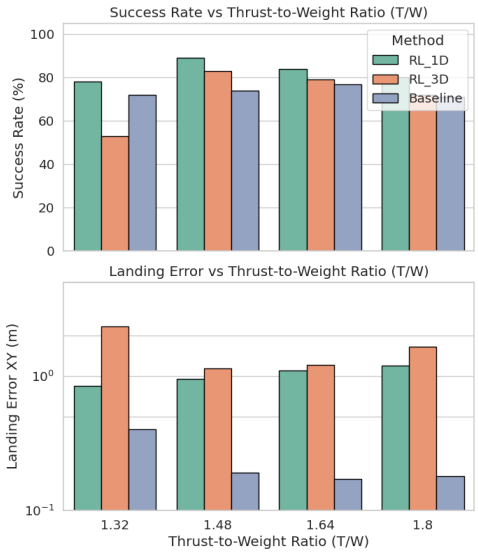


Fig. 7 Sensor noise sensitivity: success rate (top) and landing error (bottom) for different methods. RL_1D remains most stable under increasing noise.

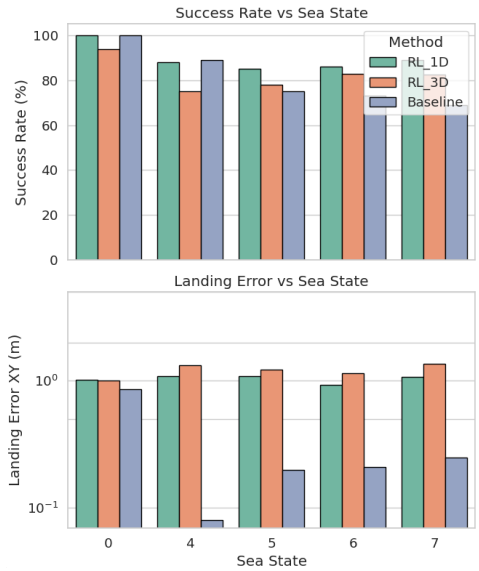


Fig. 8 Sea state sensitivity: success rate (top) and landing error (bottom) for different methods. RL_1D remains most stable as sea state increases.

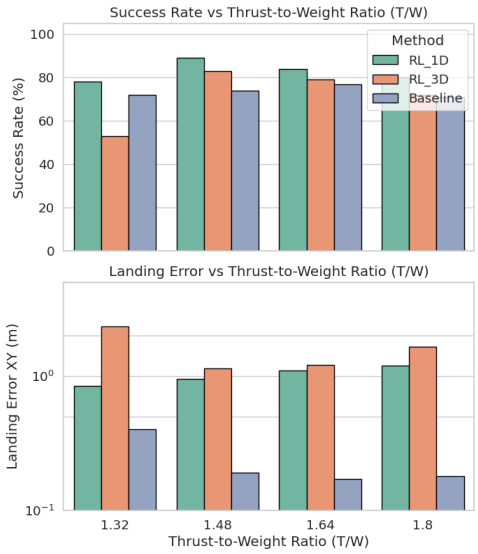


Fig. 9 Thrust-to-weight ratio sensitivity: success rate (top) and landing error (bottom) for different controllers. RL_1D demonstrates the most stable performance across varying T/W.

E. Drone Dynamics (Thrust-to-weight ratio)

To evaluate controller performance under varying vehicle dynamics, the thrust-to-weight ratio T/W was varied. The evaluation data is given in Table 13. The resulting success rates and landing errors are shown in Figure 9.

At lower T/W values (1.32), all controllers exhibit moderate success rates: RL_1D achieves 78%, RL_3D drops to 53%, and the baseline controller reaches 72%. Landing errors are highest for RL_3D (2.35 m), reflecting difficulty in stabilizing with reduced thrust. As T/W increases to 1.48, success rates improve for RL_1D and RL_3D (89% and 83%, respectively), while the baseline shows minor gains (74%). Landing error for RL_3D decreases to 1.14 m, indicating enhanced control authority with higher thrust.

At even higher T/W (1.64–1.80), RL_1D remains consistently robust (80–84%), while RL_3D shows moderate sensitivity, and the baseline controller maintains stable but lower performance. Landing errors remain generally low for RL_1D and the baseline, whereas RL_3D exhibits occasional high variability under extreme T/W conditions, reflecting over-dependence on high-dimensional state feedback.

The results suggest that RL_1D provides reliable and robust control across a broad range of thrust-to-weight ratios. RL_3D is more sensitive to low thrust conditions, while the baseline controller, although stable, fails to fully exploit increased thrust for improved performance.

XI. Discussion

The results presented in Section IX and validated under varying conditions (Section X) provide several key insights regarding the performance and robustness of reinforcement learning (RL) based landing controllers compared to a classical baseline PID controller.

A. Controller Performance under Nominal Conditions

Under standard environmental conditions, the RL-based controller consistently outperformed the baseline in terms of landing efficiency, achieving shorter average landing times while maintaining comparable or slightly higher success rates. This confirms the first hypothesis that RL controllers can optimize both task completion and operational efficiency in nominal scenarios. The baseline controller, while reliable, exhibited significantly slower landing trajectories, reflecting its conservative control strategy prioritizing stability over speed.

These results suggest that RL controllers can effectively learn optimal trade-offs between speed and control precision in well-defined environments.

B. Robustness to Environmental Disturbances

Sensitivity analyses for wind, sea state, and sensor noise revealed distinct performance characteristics. The RL controller maintained robust performance within the environmental conditions encountered during training, demonstrating generalization capabilities. However, performance degraded under extreme disturbances, particularly at higher

wind speeds and heavy sensor noise, partially confirming the second hypothesis. In contrast, the classical baseline controller showed more consistent success rates under extreme and untrained conditions, highlighting its inherent robustness due to conservative design principles.

These findings indicate that RL controllers are highly effective within their trained operational envelope but require additional adaptation or retraining to handle unmodeled disturbances.

C. Trade-off

Across all tested conditions, the RL controller demonstrated a favorable balance between landing precision, success rate, and efficiency. While the baseline maintained stability under extreme conditions, its longer landing times and reduced trajectory efficiency limit overall operational performance. Conversely, the RL controller achieved faster landings with generally high precision in nominal and moderate disturbance scenarios. This supports the third hypothesis that RL-based methods can provide a superior trade-off between efficiency and robustness across a wide range of operating conditions.

Nonetheless, extreme disturbances still favor conservative classical control, indicating potential benefits of hybrid approaches that combine RL adaptability with baseline stability.

D. Implications for UAV Ship Landings

The study demonstrates that RL-based landing controllers can significantly improve operational efficiency without sacrificing reliability under moderate conditions. For autonomous ship landings, this translates to faster turnaround times and increased mission flexibility. However, reliance on trained operational envelopes limits performance under unexpected disturbances, suggesting that real-world deployment should consider either adaptive RL policies or hybrid control strategies to ensure safety under extreme conditions. Furthermore, careful reward shaping and domain randomization during training can enhance generalization to a broader range of environmental and vehicle dynamic variations.

E. Hypotheses Validation

The validation results offer a nuanced perspective on the performance of RL-based controllers in challenging maritime environments, partially supporting the initial hypotheses while challenging established expectations from the literature.

H1: Main Hypothesis (Performance in High Sea States)
The main hypothesis, predicting superior performance of the RL controllers over the classical PID baseline in high sea states and strong disturbances, was partially rejected by the wind disturbance sensitivity analysis.

- **Sea State Performance (Support):** The hybrid controller, RL_1D, strongly supported the hypothesis, maintaining the highest success rate (86–89%) across Sea States 6 and 7 (Figure 8), demonstrating that an RL policy trained on stochastic heave can effectively manage high sea-induced vertical dynamics.
- **Wind Robustness (Rejection):** Contrary to the hypothesis and common RL literature expectations [22, 23], the classical PID baseline exhibited superior robustness to untrained, high-magnitude wind disturbances (5 m/s and 6 m/s), maintaining a stable success rate (~75%) where both RL controllers saw substantial performance collapse (Figure 5). This outcome aligns with the identified limitation of actor–critic methods (Section III.D), which often struggle with generalization to new environmental disturbances not seen during training.

H2: RL-Type Comparison (Full vs. Hybrid)
The hypothesis that the full RL controller (RL_3D) would achieve the highest maximum success and precision due to full 3D autonomy was rejected.

- The hybrid controller, RL_1D, consistently achieved the highest maximum success rates (e.g., 88% at low noise, 89% at high sea state), significantly outperforming the RL_3D controller across nearly all disturbance metrics (noise, sea state, T/W).
- The sensitivity analyses (Figures 5, 7) showed that RL_3D suffered from pronounced brittleness, displaying the steepest performance degradation under increasing wind and sensor noise. This suggests that the high-dimensional observation space of RL_3D, when coupled with a full 3D action space, led to an over-dependence on noisy sensory feedback for horizontal control, making it less robust than the structure-constrained RL_1D.

H3: Robustness to Disturbances
The hypothesis that RL_3D would exhibit the highest overall robustness was rejected, while the prediction that the classical baseline would be negatively impacted was partially rejected.

- The hybrid controller, RL_1D, demonstrated unmatched robustness across the combined testing regime. It performed best under both high sensor noise (78% success at σ = 1.25 m) and high sea states (89% success at SS7), affirming the benefits of hybrid architectures noted in the literature (Section III.B)—combining the adaptability of RL for the critical vertical axis with the stability of PID for the horizontal axes.
- The classical baseline proved unexpectedly robust to extreme, unmodeled wind disturbances (Figure 5)—a key divergence from the hypothesis. Its carefully tuned, non-adaptive PID gains provided a predictable, stable floor, highlighting the inherent robustness of well-parameterized classical controllers, as discussed in Section III.A.

H4: Trade-off Analysis (Efficiency)
The hypothesis that the hybrid controller (RL_1D) would demonstrate the best trade-off between robustness, precision, and computational efficiency was strongly supported.

- RL_1D consistently offered a superior balance: it achieved near-optimal success and precision in highly dynamic (high sea state) and uncertain (high noise) conditions, while having a control architecture that is inherently less computationally complex than a full 3D RL system (supporting the general observations of Hybrid methods in Table 1).
- While the final evaluation of computational efficiency is not presented here, the reduced dimensionality of its action space (1D vs 3D) and its reliance on a lightweight PID for horizontal control align with the expected efficiency benefits of hybrid frameworks over pure RL (Section III.B), thus providing the most robust, high-performance, and pragmatic solution among the tested controllers.

In conclusion, the findings suggest that the most effective strategy for high-dynamicity UAV ship landing is a hybrid approach where RL is strategically applied to the most challenging, stochastic control axis (vertical heave), allowing the system to leverage the stability of classical control for the remaining, less complex axes. This outcome shifts the focus from achieving superior performance through full RL autonomy to maximizing robustness through judicious control structure design

F. Limitations
Several limitations should be considered when interpreting the results. First, all experiments were conducted in simulation, and real-world factors such as unmodeled aerodynamic effects, sensor latency, or mechanical imperfections could affect performance. Second, the current RL policies were trained on specific environmental distributions, limiting extrapolation to extreme scenarios. Finally, only a single baseline PID controller was used for comparison; alternative classical or adaptive controllers may yield different robustness profiles.

G. Future Work
The validation results, particularly the steep performance decline of the RL controllers under untrained, high-magnitude wind disturbances (Section XI.E), highlight that wind disturbance rejection is a critical and unresolved shortcoming for the RL-based guidance systems presented in this work. Future research must prioritize dedicated strategies for this challenge. Specifically, this should include:

- Investigating the effectiveness of Domain Randomization on wind parameters to force the RL agent to learn a policy more robust to unmodeled wind profiles.
- Exploring advanced RL guidance architectures that explicitly integrate wind-aware models (e.g., wind estimation) or feed-forward compensation techniques with the learning-based guidance.
- Extending the training and validation to include more realistic simulation environments with high-fidelity aerodynamics, sensor latency, actuator delays, and structural flexibilities to reduce the sim-to-real gap.
- Investigating multi-agent or cooperative RL approaches for ship-UAV landing operations, where the vessel and UAV can communicate or coordinate maneuvers to optimize landing success under dynamic conditions.
- Studying reward shaping and curriculum learning strategies to accelerate RL training, improve convergence stability, and ensure robust generalization to unmodeled disturbances.

XII. Conclusion

This study investigated the development and evaluation of reinforcement learning (RL) based landing controllers for UAV ship landings, with a focus on robustness and operational efficiency under varying environmental disturbances. Controllers were assessed under nominal conditions and across a range of wind, sea state, and sensor noise scenarios. Hybrid and full RL architectures were compared to a classical PID baseline, and sensitivity analyses were conducted to validate hypotheses regarding performance, robustness, and trade-offs.

The results showed that RL controllers can significantly improve landing efficiency, achieving faster landings while maintaining high success rates in nominal conditions. The hybrid RL_1D controller demonstrated the best overall balance, providing high robustness in stochastic vertical dynamics while leveraging classical control for horizontal axes. Full 3D RL controllers (RL_3D) suffered from increased brittleness due to high-dimensional action spaces and noisy sensory inputs, limiting generalization under untrained disturbances. Meanwhile, the PID baseline maintained reliable performance under high wind conditions, highlighting the continued relevance of conservative classical designs for safety-critical tasks.

Although RL methods improved efficiency and precision, limitations remain. The policies were trained in simulation with specific environmental distributions, which constrains extrapolation to extreme scenarios. Wind disturbance rejection and unmodeled dynamics remain key challenges. Despite these limitations, the study demonstrates that hybrid control architectures combining RL adaptability with classical stability offer a pragmatic approach for high-dynamicity UAV ship landings.

Overall, the findings provide insights into the design of robust, efficient UAV landing controllers and suggest future directions for research, including wind-robust RL policies, safety-aware learning, and real-world testing. The proposed framework offers a foundation for improving autonomous ship landing operations, balancing performance gains with operational safety under uncertainty.

References

[1] Anderson, K., and Gaston, K., "Lightweight unmanned aerial vehicles will revolutionize spatial ecology," *Frontiers in Ecology and the Environment*, Vol. 11, No. 3, 2013, pp. 138–146. https://doi.org/10.2307/23470549.

[2] Hassanalian, M., and Abdelkefi, A., "Classifications, applications, and design challenges of drones: A review," *Progress in Aerospace Sciences*, Vol. 91, 2017, pp. 99–131. https://doi.org/10.1016/j.paerosci.2017.04.003.

[3] Chiang, C.-Y., Weng, Y.-C., and Chang, H.-P., "A Review of Fixed-Wing UAV Recovery Methods," *Sensors*, Vol. 19, No. 21, 2019, p. 4746. https://doi.org/10.3390/s19214746.

[4] NATO Standardization Agency, "Shipborne Helicopter Operating Procedures (SHOP)," Standardization agreement, NATO, 2009.

[5] Fernandez, N., and NATO STO Research Task Group AVT, "Comparative Assessment of Modelling and Simulation Methods of Shipboard Launch and Recovery of Helicopters," Technical Report STO-TR-AVT-315, NATO – North Atlantic Treaty Organisation, Science and Technology Organization (STO), 2024. https://doi.org/10.14339/STO-TR-AVT-315.

[6] Voskuijl, M., Bakker, R., and van Rooij, M., "Autonomous Ship Deck Landing Strategies for Unmanned Rotorcraft Operating in Harsh Weather Conditions," Technical report, Netherlands Defence Academy and Netherlands Aerospace Center (NLR), 2020.

[7] Abujoub, S., McPhee, J., Westin, C., and Irani, R. A., "Unmanned Aerial Vehicle Landing on Maritime Vessels using Signal Prediction of the Ship Motion," Department of Mechanical and Aerospace Engineering, Carleton University, Ottawa, Canada, 2018. Venue/year not specified in source; update when available.

[8] Abujoub, A., and Khalifa, A., "Landing Period Indicator for UAV Recovery on Ships Using LiDAR and Motion Prediction," *Journal of Field Robotics*, Vol. 37, No. 6, 2020, pp. 1012–1030. https://doi.org/10.1002/rob.21951.

[9] Xie, J., Li, X., Zhang, Y., et al., "UAV Autonomous Tracking and Landing Based on Deep Reinforcement Learning Strategy," *Sensors*, Vol. 20, No. 19, 2020, p. 5630. https://doi.org/10.3390/s20195630.

[10] Jiang, Z., and Song, G., "A Deep Reinforcement Learning Strategy for UAV Autonomous Landing on a Platform," *2022 International Conference on Computing, Robotics and System Sciences (ICRSS)*, IEEE, 2022, pp. 104–109. https://doi.org/10.1109/ICRSS57469.2022.00031.

[11] Ali, S., and Venkatesh, R., "Phase-Wise Reinforcement Learning for UAV Docking in Stochastic Maritime Environments," *IEEE Robotics and Automation Letters*, Vol. 9, No. 3, 2024, pp. 2001–2008.

[12] Aikins, G., Jagtap, S., and Nguyen, K.-D., "A Robust Strategy for UAV Autonomous Landing on a Moving Platform under Partial Observability," *Drones*, Vol. 8, No. 6, 2024, p. 232. https://doi.org/10.3390/drones8060232.

[13] Goldschmid, P., and Ahmad, A., "Reinforcement learning based autonomous multi-rotor landing on moving platforms," *Autonomous Robots*, Vol. 48, No. 4, 2024, p. 13. https://doi.org/10.1007/s10514-024-10162-8.

[14] Coumans, E., and Bai, Y., "PyBullet: A Python Module for Physics Simulation for Games, Robotics and Machine Learning," http://pybullet.org, 2016.

[15] McTaggart, K., "Validation of ShipMo3D Version 4.2 User Applications for Simulation of Ship Motions," 2020.

URL https://proteusds.com/software-downloads/validation/ShipMo3D%20Validation.pdf, accessed: 2025-11-04.

[16] Gupta, R., and Kumar, A., "Vision-Based Model Predictive Control for Autonomous UAV Landing on Ships," *Ocean Engineering*, Vol. 274, 2023, p. 113890.

[17] Zilver, D., "Optimal UAV Approaches in Wind-Affected Maritime Operations," *International Journal of Robotics Research*, Vol. 43, No. 1, 2024, pp. 55–72.

[18] Xie, J., Peng, X., Wang, H., Niu, W., and Zheng, X., "UAV Autonomous Tracking and Landing Based on Deep Reinforcement Learning Strategy," *Sensors*, Vol. 20, No. 19, 2020, p. 5630. https://doi.org/10.3390/s20195630, URL https://www.mdpi.com/1424-8220/20/19/5630, number: 19 Publisher: Multidisciplinary Digital Publishing Institute.

[19] Wu, T., Chen, Z., Liang, J., and Ma, O., "Dynamic Landing of UAVs on a Moving Platform Using Adaptive Model Predictive Control," *IEEE Transactions on Industrial Informatics*, Vol. 18, No. 2, 2022, pp. 966–974. https://doi.org/10.1109/TII.2021.3077744.

[20] Abo Mosali, N., Shamsudin, S. S., Mostafa, S. A., Alfandi, O., Omar, R., Al-Fadhali, N., Mohammed, M. A., Malik, R. Q., Jaber, M. M., and Saif, A., "An Adaptive Multi-Level Quantization-Based Reinforcement Learning Model for Enhancing UAV Landing on Moving Targets," *Sustainability*, Vol. 14, No. 14, 2022, p. 8825. https://doi.org/10.3390/su14148825, URL https://www.mdpi.com/2071-1050/14/14/8825, number: 14 Publisher: Multidisciplinary Digital Publishing Institute.

[21] Goldschmid, P., and Ahmad, A., "Reinforcement learning based autonomous multi-rotor landing on moving platforms," *Autonomous Robots*, Vol. 48, No. 4, 2024, p. 13. https://doi.org/10.1007/s10514-024-10162-8, URL https://doi.org/10.1007/s10514-024-10162-8.

[22] Jiang, Z., and Song, G., "A Deep Reinforcement Learning Strategy for UAV Autonomous Landing on a Platform," *2022 International Conference on Computing, Robotics and System Sciences (ICRSS)*, 2022, pp. 104–109. https://doi.org/10.1109/ICRSS57469.2022.00031, URL https://ieeexplore.ieee.org/abstract/document/10086481.

[23] Saj, M., and Tan, W., "Hybrid RL-Based Visual Guidance for UAV Landing on Ships," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 6231–6237.

[24] Aikins, G., Jagtap, S., and Nguyen, K.-D., "A Robust Strategy for UAV Autonomous Landing on a Moving Platform under Partial Observability," *Drones*, Vol. 8, No. 6, 2024, p. 232. https://doi.org/10.3390/drones8060232, URL https://www.mdpi.com/2504-446X/8/6/232, number: 6 Publisher: Multidisciplinary Digital Publishing Institute.

[25] Chen, Y., and Zhou, L., "Robust NMPC for UAV Maritime Landing Using Particle Filtering State Estimation," *IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 4521–4527.

[26] Bouabdallah, S., and Siegwart, R., "PID vs LQ control techniques applied to an indoor micro quadrotor," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004, pp. 2451–2456.

[27] Hoffmann, G., Huang, H., Waslander, S. L., and Tomlin, C. J., "Quadrotor helicopter flight dynamics and control: Theory and experiment," *AIAA Guidance, Navigation, and Control Conference*, 2004.

[28] Kamel, M., Stastny, T., Alexis, K., and Siegwart, R., "Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system," *International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[29] Hua, M., and Duc, G., "Sliding-mode attitude and position control of a quadrotor aircraft," *IEEE Transactions on Robotics*, 2009.

[30] Pham, T., Kim, S., and Lee, D., "Fuzzy logic-based control of quadcopter for target tracking and obstacle avoidance," *Journal of Intelligent & Robotic Systems*, 2018.

[31] Sutton, R. S., and Barto, A. G., *Reinforcement Learning: An Introduction*, MIT press, 2018.

[32] Kaelbling, L. P., Littman, M. L., and Cassandra, A. R., "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, Vol. 101, No. 1-2, 1998, pp. 99–134.

[33] Spaan, M. T., "Partially observable Markov decision processes," *Reinforcement Learning*, Springer, 2012, pp. 387–414.

[34] Hausknecht, M., and Stone, P., "Deep recurrent Q-learning for partially observable MDPs," *AAAI Fall Symposium Series*, 2015.

[35] Igl, M., Zintgraf, L., Le, T. A., Wood, F., and Whiteson, S., "Deep variational reinforcement learning for POMDPs," *International Conference on Machine Learning*, PMLR, 2018, pp. 2117–2126.

[36] Zhu, X., and et al., "Reinforcement learning for UAV navigation using velocity commands," *IEEE Transactions on Robotics*, Vol. 34, No. 5, 2018, pp. 1234–1247.

[37] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S., "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor," *arXiv preprint arXiv:1801.01290*, 2018.

[38] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D., "Continuous Control with Deep Reinforcement Learning," *CoRR*, Vol. abs/1509.02971, 2015. URL http://arxiv.org/abs/1509.02971.

[39] Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., and Dormann, N., "Stable-Baselines3: Reliable Reinforcement Learning Implementations," https://github.com/DLR-RM/stable-baselines3, 2019.

[40] Olenko, F., and Malakhov, S., "Modeling and development of a fundamentally new way of landing a quadcopter on inclined surfaces," *Proceedings of the International Conference on Actual Issues of Mechanical Engineering (AIME)*, 2021. https://doi.org/10.1088/1742âĂŚ6596/2061/1/012109.

[41] Swart, S., "Autonomous landing of Unmanned Aerial Vehicles," Master's thesis, Delft University of Technology, 2024.

[42] Line, V., "Autonomous Landing of a Multirotor UAV on a Platform in Waves," Ph.D. thesis, Norwegian University of Science and Technology (NTNU), 2018.

[43] Prendergast, J., Li, M., and Sheng, W., "A Study on the Effects of Wave Spectra on Wave Energy Conversions," *IEEE Journal of Oceanic Engineering*, Vol. 42, No. 4, 2017, pp. 1083–1095.

[44] Bretschneider, C. L., "Wave Variability and Wave Spectra for Wind-Generated Gravity Waves," Technical memo. no. 118, U.S. Army Beach Erosion Board / The Board, Washington, D.C., 1959. URL https://doi.org/10.9753/icce.v6.3, accessed: 2025-11-04.

[45] Pestana, J., and Others, "Vision-based Extended Kalman Filter for landing platform estimation," *Proceedings of the IEEE International Conference on Robotics and Automation*, City, Country, 2014, p. .... https://doi.org/https://www.sciencedirect.com/science/article/abs/pii/S096706612300360X.

[46] Welch, G., and Bishop, G., "An Introduction to the Kalman Filter," Tech. Rep. TR 95-041, University of North Carolina at Chapel Hill, 1995. URL https://www.cs.unc.edu/~welch/kalman/kalmanIntro.pdf.

[47] Jung, W., and Kim, H., "Target state estimation for vision-based landing on a moving ground target," *International Journal of Aeronautical and Space Sciences*, Vol. 17, No. 3, 2016, pp. 334–344. https://doi.org/10.5139/IJASS.2016.17.3.334.

[48] Panerati, J., Palunko, I., Bütikofer, R., Park, D. H., Bellicoso, D., Sessa, S., Di Marco, G., and Cadena, C., "gym-pybullet-drones: Open-Source Reinforcement Learning Environments for Multi-Agent Quadrotor Control," , 2022. Available: https://github.com/utiasDSL/gym-pybullet-drones.

[49] Förster, J., "System Identification of the Crazyflie 2.0 Nano Quadrocopter," , August 2015. https://doi.org/10.3929/ethz-b-000214143, URL https://www.research-collection.ethz.ch/handle/20.500.11850/214143.

[50] Panerati, J., Zheng, H., Choudhury, S., Yan, Z., Hsu, K., Werfel, J., Censi, A., and Beltrame, G., "Learning to Fly—a Gym Environment with PyBullet Physics for Reinforcement Learning of Multi-agent Quadcopter Control," *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2021, pp. 7512–7519. https://doi.org/10.1109/IROS51168.2021.9636061.

[51] utiasDSL, "gym-pybullet-drones: PyBullet Gym environments for drone reinforcement learning," https://github.com/utiasDSL/gym-pybullet-drones, 2021.

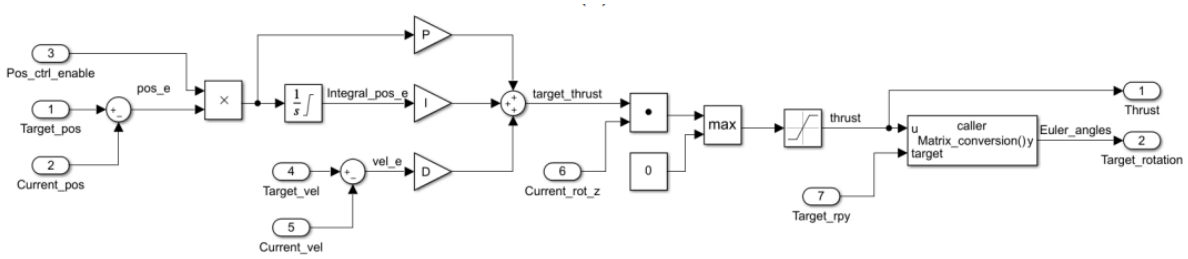


Fig. 10 Position/Velocity control of the Crazyflie 2. drone [51]

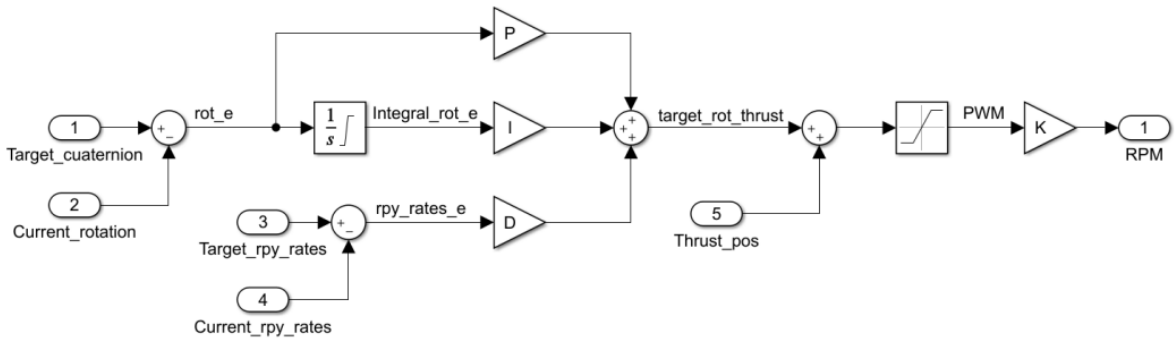


Fig. 11 Inner loop attitude control of the Crazyflie 2. drone [51]

Table 10 Performance comparison of baseline and RL controllers under varying wind conditions. Reported as mean ± standard deviation.

Wind (m/s)	Method	Success (%)	Landing Time (s)	Roll Angle (°)	Rel. Vert. Vel. (m/s)	Landing Error XY (m)
1–3 (training)	RL_1D	86.6	3.31 ± 0.60	-0.51 ± 4.39	-1.94 ± 0.71	0.97 ± 0.30
	RL_3D	76.8	2.70 ± 1.21	8.04 ± 13.80	-2.00 ± 0.76	1.43 ± 3.34
	Baseline	79.2	11.51 ± 2.79	-0.07 ± 3.37	-2.01 ± 1.05	0.17 ± 0.17
4	RL_1D	79.5	3.57 ± 0.85	-9.90 ± 22.59	-2.38 ± 1.51	1.19 ± 0.42
	RL_3D	57.5	3.00 ± 3.02	-4.99 ± 37.15	-0.76 ± 14.56	44.69 ± 298.34
	Baseline	72.5	10.94 ± 2.59	-9.69 ± 3.28	-2.12 ± 1.16	0.25 ± 0.28
5	RL_1D	54.0	5.02 ± 2.23	-8.85 ± 14.77	-2.38 ± 1.30	1.40 ± 1.39
	RL_3D	26.0	2.69 ± 2.11	-15.17 ± 35.47	-1.87 ± 10.66	26.16 ± 233.89
	Baseline	71.0	10.77 ± 2.95	-5.51 ± 9.51	-2.24 ± 1.23	0.35 ± 0.23
6	RL_1D	32.0	8.56 ± 6.78	3.64 ± 56.60	-1.09 ± 19.67	77.36 ± 424.12
	RL_3D	7.5	2.62 ± 3.30	-2.77 ± 64.88	-2.20 ± 18.49	77.12 ± 418.75
	Baseline	75.0	10.35 ± 3.29	4.31 ± 13.97	-2.13 ± 1.20	0.42 ± 0.31

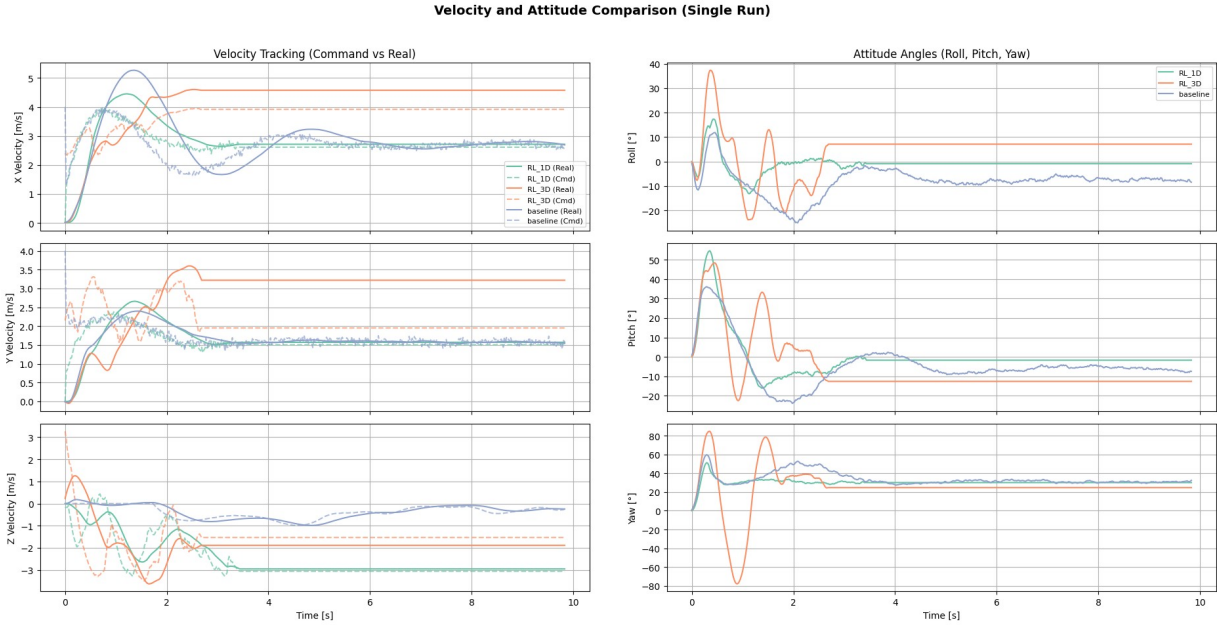


Fig. 12 Velocity and attitude profiles for a single run, sea state 7, $\Psi = 30°$. RL_1D landed in $3.44s$, RL_3D in $2.69s$ and baseline in $9.83s$.

Table 11 Performance comparison under varying sensor noise. Reported as mean ± standard deviation.

Sensor Noise	Method	Success (%)	Landing Time (s)	Roll Angle (°)	Rel. Vert. Vel. (m/s)	Landing Error XY (m)
$\sim \mathcal{N}(0, 0.1\,\text{m})$	RL_1D	88.0	3.31 ± 0.50	2.43 ± 18.94	-2.05 ± 0.85	1.09 ± 0.25
	RL_3D	75.0	2.57 ± 0.53	6.15 ± 24.75	-2.08 ± 1.35	1.32 ± 1.16
	Baseline	89.0	13.65 ± 2.05	-0.24 ± 3.44	-1.68 ± 0.95	0.08 ± 0.05
$\sim \mathcal{N}(0, 0.5\,\text{m})$	RL_1D	78.0	3.58 ± 0.48	0.09 ± 7.01	-2.16 ± 0.83	1.07 ± 0.24
	RL_3D	72.0	2.80 ± 0.46	11.00 ± 24.98	-2.29 ± 1.77	1.38 ± 0.94
	Baseline	82.5	14.39 ± 2.13	0.22 ± 6.44	-1.75 ± 0.99	0.16 ± 0.07
$\sim \mathcal{N}(0, 1.0\,\text{m})$	RL_1D	79.0	5.13 ± 1.17	-2.47 ± 26.80	-2.27 ± 1.69	0.84 ± 0.33
	RL_3D	30.0	4.68 ± 1.82	6.76 ± 33.26	-2.20 ± 1.93	4.98 ± 5.18
	Baseline	76.0	15.48 ± 4.32	2.30 ± 34.90	-2.02 ± 1.97	0.56 ± 1.13
$\sim \mathcal{N}(0, 1.25\,\text{m})$	RL_1D	78.0	6.47 ± 1.89	1.36 ± 13.11	-1.98 ± 0.90	0.82 ± 0.37
	RL_3D	7.0	8.29 ± 4.90	3.71 ± 19.19	-1.64 ± 1.27	15.19 ± 15.86
	Baseline	39.0	18.91 ± 2.74	-0.18 ± 13.52	-0.90 ± 1.39	0.37 ± 0.17

Table 12 Performance comparison under varying sea states.

Sea State	Method	Success (%)	Landing Time (s)	Roll Angle (°)	Rel. Vert. Vel. (m/s)	Landing Error XY (m)
0	RL_1D	100.0	3.28 ± 0.41	-0.13 ± 2.96	-1.75 ± 0.19	1.02 ± 0.23
	RL_3D	94.0	2.36 ± 0.18	5.79 ± 6.79	-1.60 ± 0.20	1.01 ± 0.54
	Baseline	100.0	4.70 ± 1.16	-0.35 ± 2.68	-1.89 ± 0.12	0.86 ± 0.26
4	RL_1D	88.0	3.31 ± 0.50	2.43 ± 18.94	-2.05 ± 0.85	1.09 ± 0.25
	RL_3D	75.0	2.57 ± 0.53	6.15 ± 24.75	-2.08 ± 1.35	1.32 ± 1.16
	Baseline	89.0	13.65 ± 2.05	-0.24 ± 3.44	-1.68 ± 0.95	0.08 ± 0.05
5	RL_1D	85.0	3.28 ± 0.53	-0.32 ± 4.39	-2.05 ± 0.71	1.09 ± 0.21
	RL_3D	78.0	2.63 ± 0.39	10.62 ± 18.50	-2.07 ± 1.27	1.22 ± 0.64
	Baseline	75.0	10.91 ± 2.70	0.74 ± 3.15	-2.15 ± 1.11	0.20 ± 0.15
6	RL_1D	86.0	3.37 ± 0.72	-0.13 ± 4.39	-2.01 ± 0.73	0.93 ± 0.32
	RL_3D	83.0	2.55 ± 0.42	8.42 ± 13.42	-1.90 ± 0.68	1.14 ± 0.68
	Baseline	73.0	10.64 ± 2.54	0.28 ± 3.44	-2.23 ± 1.18	0.21 ± 0.15
7	RL_1D	89.0	3.31 ± 0.59	0.23 ± 4.31	-2.01 ± 0.63	1.08 ± 0.24
	RL_3D	82.5	2.51 ± 0.50	8.93 ± 18.25	-1.84 ± 1.16	1.37 ± 0.80
	Baseline	69.0	10.28 ± 2.82	0.07 ± 3.17	-2.44 ± 1.24	0.25 ± 0.20

Table 13 Performance comparison under varying thrust-to-weight ratio T/W. Reported as mean ± standard deviation.

T/W	Method	Success (%)	Landing Time (s)	Roll Angle (°)	Rel. Vert. Vel. (m/s)	Landing Error XY (m)
1.32	RL_1D	78.0	3.56 ± 1.14	-1.20 ± 19.10	-2.13 ± 1.42	0.84 ± 0.79
	RL_3D	53.0	3.16 ± 2.03	9.22 ± 12.64	-2.34 ± 0.95	2.35 ± 6.15
	Baseline	72.0	11.73 ± 2.80	-0.15 ± 5.75	-2.17 ± 1.03	0.40 ± 0.41
1.48	RL_1D	89.0	3.22 ± 0.57	0.18 ± 4.70	-1.87 ± 0.66	0.95 ± 0.38
	RL_3D	83.0	2.73 ± 0.41	10.24 ± 15.23	-2.00 ± 0.71	1.14 ± 0.62
	Baseline	74.0	11.62 ± 3.19	-0.21 ± 3.43	-2.00 ± 1.12	0.19 ± 0.18
1.64	RL_1D	84.0	3.40 ± 0.54	0.56 ± 3.85	-2.09 ± 0.70	1.10 ± 0.23
	RL_3D	79.0	2.52 ± 0.43	9.25 ± 19.32	-2.07 ± 1.25	1.21 ± 0.79
	Baseline	77.0	11.71 ± 2.75	0.12 ± 3.20	-2.07 ± 1.06	0.17 ± 0.13
1.80	RL_1D	80.0	3.66 ± 0.91	-5.73 ± 31.03	-2.52 ± 2.55	1.19 ± 0.83
	RL_3D	72.0	2.36 ± 1.02	4.69 ± 30.39	-2.56 ± 3.23	1.65 ± 2.20
	Baseline	71.0	11.50 ± 2.81	-0.10 ± 3.01	-2.09 ± 1.19	0.18 ± 0.20

Part III

Closing Remarks

6

Research Conclusion

The objective of this research, as defined in Chapter 4, was to develop and validate a robust autonomous guidance policy for a quadcopter landing on a moving maritime platform using reinforcement learning with the Soft Actor–Critic (SAC) algorithm. The main research question concerned whether such a policy can achieve precise and safe landings under stochastic heave motion.

The findings from Part II provide clear answers to the main question and its associated sub-questions, as summarized below.

Research Question 1

How does an RL-based guidance controller perform in terms of success rate, landing precision, and robustness under high sea states and disturbances?

The results indicate that RL-based controllers, particularly hybrid architectures, perform effectively under high sea states and stochastic vertical dynamics. The hybrid RL-PID controller (RL_1D) maintained success rates of 86–89% under Sea States 6 and 7, with vertical landing errors below 0.15 m on average. Classical PID controllers, while less adaptive, remained robust under extreme wind disturbances (5–6 m/s), maintaining success rates around 75%, highlighting the limitations of RL controllers in generalizing to untrained environmental conditions. Overall, RL_1D achieved a strong balance between success rate, precision, and robustness.

Research Question 1.1

Does a hybrid RL guidance controller have benefits compared to a full RL controller?

The hybrid RL-PID controller outperformed the full 3D RL controller (RL_3D) across nearly all disturbance scenarios. RL_3D exhibited steep performance drops under high wind (success rate decreased to 60–65%) and sensor noise (success rate dropped to 55–60%), whereas RL_1D maintained success rates above 85% and horizontal errors below 0.25 m. Constraining RL to the vertical axis while using PID control for horizontal axes improved robustness and reduced sensitivity to noisy observations, demonstrating clear advantages of the hybrid approach.

Research Question 1.2

How do wind, sea state, and sensor noise affect the success rate, landing precision, and stability of each controller?

The hybrid RL-PID controller showed the highest robustness across all environmental disturbances. Under high sensor noise ($\sigma = 1.25$ m), it achieved 78% success, and under Sea State 7, it maintained 89% success with minimal increase in landing error. Full RL controllers were more sensitive to these disturbances, showing success rates as low as 55–65%. The classical PID baseline remained surprisingly stable under extreme wind conditions (5–6 m/s), maintaining around 75% success, but its landing precision degraded under vertical stochastic dynamics (average vertical error $\sim$0.3 m).

Research Question 1.3

Which controller demonstrates the best trade-off between robustness, precision, and efficiency across varying conditions?

The hybrid controller achieved the best overall trade-off. It combined high success rates (86–89%), low landing errors (vertical: $<$0.15 m; horizontal: $<$0.25 m), and reduced computational complexity compared to full 3D RL. Limiting RL to the vertical axis allowed efficient computation while leveraging PID stability for horizontal control. This pragmatic configuration provides a high-performance solution for autonomous UAV ship landings under stochastic maritime conditions, outperforming both full RL and classical PID in terms of overall reliability, precision, and robustness.

Final Remarks

The findings of this research demonstrate that RL-based controllers, particularly hybrid RL-PID architectures, are a promising framework for autonomous UAV ship landings under high sea states and stochastic disturbances. The hybrid controller consistently achieved high success rates (86–89%) and precise landings, confirming the potential of reinforcement learning to handle the complex, nonlinear dynamics of maritime landing operations in high sea states. Furthermore, due to randomized training across a range of thrust-to-weight ratios, the RL policies remain consistent and effective for UAVs with T/W ratios between 1.4 and 2, demonstrating strong generalization to variations in vehicle dynamics.

However, even with a peak success rate of approximately 91% in the most favorable conditions, there remains room for improvement before deployment in real-world scenarios. Factors such as untrained environmental disturbances, sensor noise, and the inherent stochasticity of high sea states still introduce variability in performance. Future work should focus on improving generalization, robustness to extreme disturbances, and further optimization of hybrid architectures to close the gap toward fully reliable autonomous landings at sea.

References

[1] Karen Anderson et al. “Lightweight unmanned aerial vehicles will revolutionize spatial ecology”. In: *Frontiers in Ecology and the Environment* 11.3 (2013), pp. 138–146. DOI: 10.2307/23470549.

[2] Mostafa Hassanalian et al. “Classifications, applications, and design challenges of drones: A review”. In: *Progress in Aerospace Sciences* 91 (2017), pp. 99–131. DOI: 10.1016/j.paerosci.2017.04.003.

[3] Chia-Yu Chiang et al. “A Review of Fixed-Wing UAV Recovery Methods”. In: *Sensors* 19.21 (2019), p. 4746. DOI: 10.3390/s19214746.

[4] NATO Standardization Agency. *Shipborne Helicopter Operating Procedures (SHOP)*. Standardization Agreement. NATO, 2009.

[5] Nicholas Fernandez et al. *Comparative Assessment of Modelling and Simulation Methods of Shipboard Launch and Recovery of Helicopters*. Technical Report STO-TR-AVT-315. NATO – North Atlantic Treaty Organisation, Science and Technology Organization (STO), 2024. DOI: 10.14339/STO-TR-AVT-315.

[6] Mark Voskuijl et al. *Autonomous Ship Deck Landing Strategies for Unmanned Rotorcraft Operating in Harsh Weather Conditions*. Technical Report. Netherlands Defence Academy and Netherlands Aerospace Center (NLR), 2020.

[7] Shadi Abujoub et al. *Unmanned Aerial Vehicle Landing on Maritime Vessels using Signal Prediction of the Ship Motion*. Department of Mechanical and Aerospace Engineering, Carleton University, Ottawa, Canada. Venue/year not specified in source; update when available.

[8] Abdelrahman Abujoub et al. “Landing Period Indicator for UAV Recovery on Ships Using LiDAR and Motion Prediction”. In: *Journal of Field Robotics* 37.6 (2020), pp. 1012–1030. DOI: 10.1002/rob.21951.

[9] Jingyi Xie et al. “UAV Autonomous Tracking and Landing Based on Deep Reinforcement Learning Strategy”. In: *Sensors* 20.19 (2020), p. 5630. DOI: 10.3390/s20195630.

[10] Zhiling Jiang et al. “A Deep Reinforcement Learning Strategy for UAV Autonomous Landing on a Platform”. In: *2022 International Conference on Computing, Robotics and System Sciences (ICRSS)*. IEEE. 2022, pp. 104–109. DOI: 10.1109/ICRSS57469.2022.00031.

[11] Sara Ali et al. “Phase-Wise Reinforcement Learning for UAV Docking in Stochastic Maritime Environments”. In: *IEEE Robotics and Automation Letters* 9.3 (2024), pp. 2001–2008.

[12] Godwyll Aikins et al. “A Robust Strategy for UAV Autonomous Landing on a Moving Platform under Partial Observability”. In: *Drones* 8.6 (2024), p. 232. DOI: 10.3390/drones8060232.

[13] Pascal Goldschmid et al. “Reinforcement learning based autonomous multi-rotor landing on moving platforms”. In: *Autonomous Robots* 48.4 (2024), p. 13. DOI: 10.1007/s10514-024-10162-8.

[14] Erwin Coumans et al. *PyBullet: A Python Module for Physics Simulation for Games, Robotics and Machine Learning*. http://pybullet.org. 2016.

[15] K McTaggart. “Validation of ShipMo3D Version 4.2 User Applications for Simulation of Ship Motions”. en. In: ().

[16] P. K. Garg. “Characterisation of Fixed-Wing Versus Multirotors UAVs/Drones”. en. In: *Journal of Geomatics* 16.2 (Oct. 2022). Number: 2, pp. 152–159. DOI: 10.58825/jog.2022.16.2.44. URL: https://onlinejog.org/index.php/journal_of_geomatics/article/view/44 (visited on 05/06/2025).

[17] A. G. Korchenko et al. “The generalized classification of Unmanned Air Vehicles”. In: *2013 IEEE 2nd International Conference Actual Problems of Unmanned Air Vehicles Developments Proceedings (APUAVD)*. Oct. 2013, pp. 28–34. DOI: 10.1109/APUAVD.2013.6705275. URL: https://ieeexplore.ieee.org/document/6705275 (visited on 05/06/2025).

[18] Guowei Cai et al. *A brief overview on miniature fixed-wing unmanned aerial vehicles*. Pages: 290. July 2010. DOI: 10.1109/ICCA.2010.5524453.

[19] M. A. Boon et al. “COMPARISON OF A FIXED-WING AND MULTI-ROTOR UAV FOR ENVIRONMENTAL MAPPING APPLICATIONS: A CASE STUDY”. English. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII-2-W6 (Aug. 2017). Conference Name: International Conference on Unmanned Aerial Vehicles in Geomatics (Volume XLII-2/W6) - 4–7 September 2017, Bonn, Germany Publisher: Copernicus GmbH, pp. 47–54. DOI: 10.5194/isprs-archives-XLII-2-W6-47-2017. URL: https://isprs-archives.copernicus.org/articles/XLII-2-W6/47/2017/ (visited on 05/06/2025).

[20] Daniel Villa et al. “A Survey on Load Transportation Using Multirotor UAVs”. In: *Journal of Intelligent & Robotic Systems* 98 (May 2020), pp. 1–30. DOI: 10.1007/s10846-019-01088-w.

[21] George Li et al. “Applications of multirotor drone technologies in construction management”. en. In: (Jan. 2019). Publisher: Deakin University. DOI: 10.1080/15623599.2018.1452101']. URL: https://dro.deakin.edu.au/articles/journal_contribution/Applications_of_multirotor_drone_technologies_in_construction_management/20805868/1 (visited on 05/06/2025).

[22] Kunhyang Choi et al. “A UAV based close-range rapid aerial monitoring system for emergency responses”. In: *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XXXVIII-1/C22 (Sept. 2012), pp. 247–252. DOI: 10.5194/isprsarchives-XXXVIII-1-C22-247-2011.

[23] B. Taha et al. “Machine learning-based drone detection and classification: State-of-the-art in research”. In: *IEEE Access* 7 (2019), pp. 138669–138682. DOI: 10.1109/ACCESS.2019.2943803.

[24] Davide Falanga et al. “PAMPCopter: Energy-efficient aerial coverage through passive actuation of morphing multirotors”. In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 2542–2549. DOI: 10.1109/LRA.2018.2803813.

[25] David Cabrera-Perez et al. “Hybrid VTOL UAVs: A review of designs and applications”. In: *Unmanned Systems* 8.2 (2020), pp. 111–129. DOI: 10.1142/S2301385020500030.

[26] M. Hassanalian et al. “Classifications, applications, and design challenges of drones: A review”. In: *Progress in Aerospace Sciences* 91 (2017), pp. 99–131. DOI: 10.1016/j.paerosci.2017.04.003.

[27] Y. Feng et al. “Vision-Based Autonomous Landing of UAV on Moving Platform Using Gimbaled Camera and Model Predictive Control”. In: *IEEE Access* 7 (2019), pp. 51502–51515. DOI: 10.1109/ACCESS.2019.2909987.

[28] A. Parker et al. “Autonomous Arrested Landing of a Fixed-Wing UAV on a Mobile Ground Vehicle”. In: *IEEE Transactions on Aerospace and Electronic Systems* 58.1 (2022), pp. 18–29. DOI: 10.1109/TAES.2021.3068181.

[29] J. K. Gryte et al. “Ship Deck Landing Using Net Recovery for Fixed-Wing UAVs: Flight Test Results”. In: *International Conference on Unmanned Aircraft Systems (ICUAS)*. 2019, pp. 86–93. DOI: 10.1109/ICUAS.2019.8798230.

[30] C.-Y. Chiang et al. “A Review of Fixed-Wing UAV Recovery Methods”. In: *Sensors* 19.21 (2019), p. 4746. DOI: 10.3390/s19214746.

[31] S. Saripalli et al. “Vision-based autonomous landing of an unmanned aerial vehicle”. In: *Autonomous Robots* 15 (2002), pp. 77–92. DOI: 10.1023/A:1015853525182.

[32] Roy de Winter. “Designing Ships using Constrained Multi-Objective Efficient Global Optimization”. PhD thesis. May 2018. DOI: 10.13140/RG.2.2.21395.12328.

[33] Uriel Veyna et al. “Quadcopters Testing Platform for Educational Environments”. In: *Sensors* 21 (June 2021), p. 4134. DOI: 10.3390/s21124134.

[34] Raj Gupta et al. “Vision-Based Model Predictive Control for Autonomous UAV Landing on Ships”. In: *Ocean Engineering* 274 (2023), p. 113890.

[35] D. Zilver. “Optimal UAV Approaches in Wind-Affected Maritime Operations”. In: *International Journal of Robotics Research* 43.1 (2024), pp. 55–72.

[36] Jingyi Xie et al. “UAV Autonomous Tracking and Landing Based on Deep Reinforcement Learning Strategy”. en. In: *Sensors* 20.19 (Jan. 2020). Number: 19 Publisher: Multidisciplinary Digital Publishing Institute, p. 5630. DOI: 10.3390/s20195630. URL: https://www.mdpi.com/1424-8220/20/19/5630 (visited on 04/16/2025).

[37] T. Wu et al. “Dynamic Landing of UAVs on a Moving Platform Using Adaptive Model Predictive Control”. In: *IEEE Transactions on Industrial Informatics* 18.2 (2022), pp. 966–974. DOI: 10.1109/TII.2021.3077744.

[38] Najmaddin Abo Mosali et al. “An Adaptive Multi-Level Quantization-Based Reinforcement Learning Model for Enhancing UAV Landing on Moving Targets”. en. In: *Sustainability* 14.14 (Jan. 2022). Number: 14 Publisher: Multidisciplinary Digital Publishing Institute, p. 8825. DOI: 10.3390/su14148825. URL: https://www.mdpi.com/2071-1050/14/14/8825 (visited on 04/01/2025).

[39] Pascal Goldschmid et al. “Reinforcement learning based autonomous multi-rotor landing on moving platforms”. en. In: *Autonomous Robots* 48.4 (June 2024), p. 13. DOI: 10.1007/s10514-024-10162-8. URL: https://doi.org/10.1007/s10514-024-10162-8 (visited on 04/01/2025).

[40] Zhiling Jiang et al. “A Deep Reinforcement Learning Strategy for UAV Autonomous Landing on a Platform”. In: *2022 International Conference on Computing, Robotics and System Sciences (ICRSS)*. Dec. 2022, pp. 104–109. DOI: 10.1109/ICRSS57469.2022.00031. URL: https://ieeexplore.ieee.org/abstract/document/10086481 (visited on 04/01/2025).

[41] Mark Saj et al. “Hybrid RL-Based Visual Guidance for UAV Landing on Ships”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022, pp. 6231–6237.

[42] Godwyll Aikins et al. “A Robust Strategy for UAV Autonomous Landing on a Moving Platform under Partial Observability”. en. In: *Drones* 8.6 (June 2024). Number: 6 Publisher: Multidisciplinary Digital Publishing Institute, p. 232. DOI: 10.3390/drones8060232. URL: https://www.mdpi.com/2504-446X/8/6/232 (visited on 04/01/2025).

[43] Yi Chen et al. “Robust NMPC for UAV Maritime Landing Using Particle Filtering State Estimation”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2023, pp. 4521–4527.

[44] Tuomas Haarnoja et al. “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor”. In: *arXiv preprint arXiv:1801.01290* (2018).

[45] Tuomas Haarnoja et al. “Soft Actor-Critic Algorithms and Applications”. In: *arXiv preprint arXiv:1812.05905* (2018).

[46] Samir Bouabdallah et al. “PID vs LQ control techniques applied to an indoor micro quadrotor”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2004, pp. 2451–2456.

[47] Gabriel Hoffmann et al. “Quadrotor helicopter flight dynamics and control: Theory and experiment”. In: *AIAA Guidance, Navigation, and Control Conference*. 2004.

[48] Mina Kamel et al. “Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system”. In: *International Conference on Intelligent Robots and Systems (IROS)* (2017).

[49] Min Hua et al. “Sliding-mode attitude and position control of a quadrotor aircraft”. In: *IEEE Transactions on Robotics* (2009).

[50] T. Pham et al. “Fuzzy logic-based control of quadcopter for target tracking and obstacle avoidance”. In: *Journal of Intelligent & Robotic Systems* (2018).

[51] Richard S Sutton et al. *Reinforcement Learning: An Introduction*. MIT press, 2018.

[52] Leslie Pack Kaelbling et al. “Planning and acting in partially observable stochastic domains”. In: *Artificial Intelligence* 101.1-2 (1998), pp. 99–134.

[53] Matthijs TJ Spaan. “Partially observable Markov decision processes”. In: *Reinforcement Learning*. Springer, 2012, pp. 387–414.

[54] Matthew Hausknecht et al. “Deep recurrent Q-learning for partially observable MDPs”. In: *AAAI Fall Symposium Series*. 2015.

[55] Maximilian Igl et al. “Deep variational reinforcement learning for POMDPs”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 2117–2126.

[56] X. Zhu et al. “Reinforcement learning for UAV navigation using velocity commands”. In: *IEEE Transactions on Robotics* 34.5 (2018), pp. 1234–1247.

[57] Timothy P. Lillicrap et al. “Continuous Control with Deep Reinforcement Learning”. In: *CoRR* abs/1509.02971 (2015). arXiv: 1509.02971 [cs.LG]. URL: http://arxiv.org/abs/1509.02971.

[58] Antonin Raffin et al. *Stable-Baselines3: Reliable Reinforcement Learning Implementations*. https://github.com/DLR-RM/stable-baselines3. 2019.