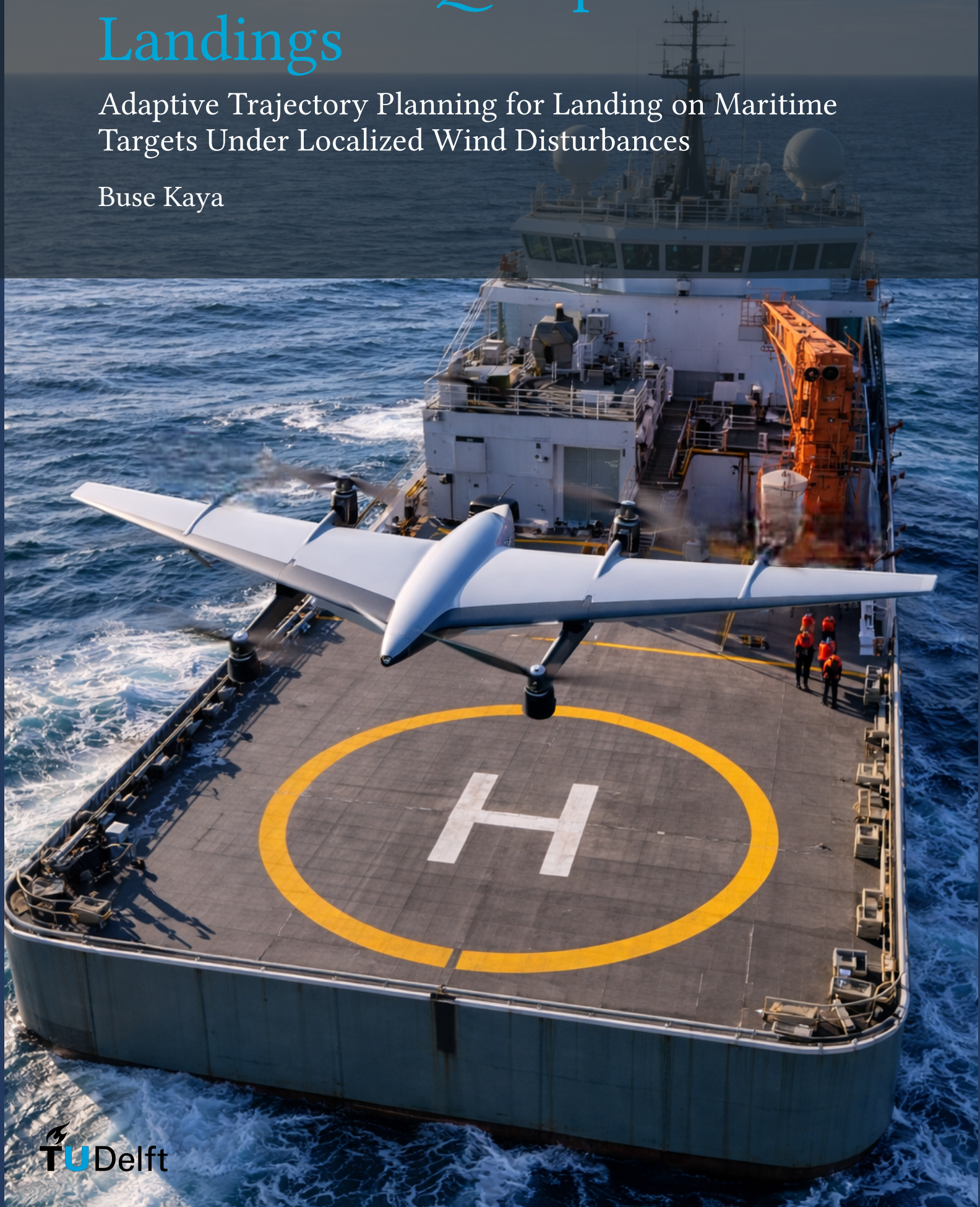


Reinforcement Learning Control For Quadplane Landings

Adaptive Trajectory Planning for Landing on Maritime
Targets Under Localized Wind Disturbances

Buse Kaya



Reinforcement Learning Control For Quadplane Landings

Adaptive Trajectory Planning for Landing on Maritime
Targets Under Localized Wind Disturbances

Thesis report

by

Buse Kaya

to obtain the degree of Master of Science
at the Delft University of Technology
to be defended publicly on August 26, 2026 at 13:00

Supervisors:	Dr.ir. Erik-Jan Van Kampen Dr.ir. Ole Pfeifle Dr.ir. Tijmen Pollack
Chair:	Dr.ir. Coen de Visser
External examiner:	Dr.ir. Erwin Mooij
Place:	Faculty of Aerospace Engineering, Delft
Project Duration:	January, 2026 - August, 2026
Cover Page:	Designed with the assistance of ChatGPT (OpenAI) and Canva.

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



Copyright © Buse Kaya, 2026
All rights reserved.

PREFACE

This thesis marks the completion of a five-year journey through the Faculty of Aerospace Engineering at TU Delft. My journey in Delft has been extraordinary. It has been a time of learning, challenging myself, and continuously pushing my boundaries. I leave this chapter with a strong belief that a good engineer, equipped with solid problem-solving skills and the willingness to learn, can take on and achieve anything they set their mind to. This thesis has certainly proven that to me.

I was fortunate to have the greatest supervisory team, for which I am deeply grateful. Erik-Jan van Kampen, thank you for always welcoming me with an open door and for sharing your knowledge and support throughout this journey. Ole Pfeifle and Tijmen Pollack, thank you for proposing the framework for this problem and for your dedication, field expertise, and guidance throughout the project. Your support and insights have been invaluable to me.

My biggest thanks go to my parents. You have always given me the opportunities and support I needed to reach for the stars, and you have believed in me every step of the way. Thank you for always putting my education first and for giving me the freedom to pursue my dreams. Your unconditional love has always been, and will always be, my greatest source of happiness and strength.

To my brother, thank you for being the spark in my life. Thank you for reminding me, especially at times when I get too caught up in the problems I am trying to solve, that those problems are not nearly as important as the memories we make together.

And finally, to my dearest boyfriend, Jordi: thank you for being there through it all. You picked me back up when I was exhausted, celebrated my wins, and stood by me through my losses. I could not have asked for a better person to share this journey with. I love you.

*Buse Kaya
Delft, August 2026*

CONTENTS

Preface	i
List of Figures	ii
List of Tables	vi
List of Symbols and Abbreviations	viii
1 Introduction	2
2 Literature Review	3
2.1 Maritime Landing Problem	3
2.2 Classical Control Strategies for UAV Landing	8
2.3 RL Based Control Strategies for UAV Landing	9
2.4 Curriculum Learning Approaches	19
2.5 Lift & Cruise Quadplane Modeling	21
2.6 Estimating UAV States	25
2.7 Ship Dynamics	28
2.8 Wind and Wake Modeling	31
3 Research Framework	37
3.1 Research Questions	37
3.2 Research Plan	39
4 Scientific Paper	36
5 Extended Methodology	66
5.1 Quadplane Dynamic Model	66
5.2 Tracking Controller	76
5.3 Time Domain Tracking Performance	87
5.4 Baseline Guidance Controller	90
5.5 RL Implementation	91
6 Verification	95
6.1 Quadplane Dynamics and Controller Performance	95
6.2 Aerodynamic Model Verification	106
6.3 RL Functionality Verification	122

7	Additional Results	128
7.1	Moving Maritime Target Landing without Localized Wind Disturbances	128
7.2	Stationary Target Landing with Localized Wind Disturbances	133
8	Conclusions	145
8.1	Research Question Answers	145
8.2	Limitations of Research	154
9	Recommendations	156
	References	157

LIST OF FIGURES

1	Deck landing mission task elements for UAVs. Retrieved from Abujoub et al. (2020).	4
2	Block diagram depicting a guidance, navigation, and control (GNC) framework. Retrieved from Kamath et al. (2020).	6
3	Representative UAV configurations considered for maritime operations.	7
4	Diagram representing the interaction between an agent and its environment in an MDP. Retrieved from Sutton and Barto (2018).	10
5	Model-free deep reinforcement learning architecture (DQN). Retrieved from Mao et al. (2016).	13
6	Actor-critic algorithm diagram illustrating how the value function approximation is used to guide policy updates. Retrieved from Fuji et al. (2018).	14
7	General framework of predefined curriculum learning, where a manually designed difficulty measurer ranks training samples, and a training scheduler determines the order in which they are presented to the learning algorithm. Retrieved from X. Wang et al. (2021).	19
8	Representative lift-and-cruise UAV configurations with varying rotor placement and airframe layouts.	22
9	Frequency domain to time domain mapping of wave characteristics. Retrieved from Bai and Bai (2005).	30
10	Time-domain heave motion trajectories generated from the JONSWAP wave spectrum for different sea-state conditions. Retrieved from Choi et al. (2025).	30
11	Schematic of the aerodynamic interaction between ship superstructure and helicopter in the air-wake region. Retrieved from Vakilabadi et al. (2023).	33
12	Contours of (a) mean streamwise velocity and (b) turbulence kinetic energy (TKE) across the ship centreline ($y = 0$), computed from wind-wave experiments. Retrieved from Setiawan et al. (2025).	34
13	Proposed thesis timeline showing the project phases and associated tasks. Purple indicates Stage 1 (quadrotor), while pink indicates Stage 2 (quadplane).	41
14	Gazebo compatible platform geometry model of a lift & cruise aircraft: the standard_vtol from the PX4-Autopilot repository. ²⁰	66
15	Free-body diagram of an X4 configuration quadrotor UAV illustrating the inertial earth-fixed and body-fixed coordinate frames. Adapted from Ho et al., 2017.	67
16	Euler rotation sequence from the UAV body-fixed frame (x, y, z) to the inertial frame (X, Y, Z) , showing the yaw (ψ), pitch (θ), and roll (ϕ) angles. Retrieved from Li and Li (2011).	67

17	Body, stability, aerodynamic, and inertial reference frames for a generic aircraft, shown in the conventional right-handed aircraft convention (x forward, y right, z down). This thesis instead uses a right-handed convention with z pointing upward and y pointing left. Retrieved from Lu et al. (2016).	68
18	Wind tunnel measurements from Z. Ma et al. (2022) for the bare wing configuration under different flap deflections. The neutral flap setting (yellow curve) is used as the reference case for aerodynamic parameter tuning in this study.	73
19	Thrust saturation and acceleration-to-attitude decomposition: vertical thrust is prioritised within the total thrust budget $F_{T_{\max}}$, with the remaining capacity allocated to horizontal acceleration.	80
20	Closed loop control system block diagram. Retrieved from Ogata (2010).	82
21	Frequency-domain analysis of the angular rate loop.	85
22	Frequency-domain analysis of the attitude loop.	86
23	Frequency-domain analysis of the velocity loop.	86
24	Frequency-domain analysis of the position loop.	86
25	Position step responses of the cascaded PID controller. (a) Horizontal (x -axis) position response. (b) Vertical (z -axis) position response.	87
26	Velocity tracking responses of the cascaded PID controller. (a) x -axis velocity response. (b) y -axis velocity response. (c) z -axis velocity response.	88
27	Position tracking response under a 10 m/s headwind.	89
28	Position tracking response under a 5 m/s vertical downwash.	89
29	Baseline guidance law position commands under sea state 6 conditions.	91
30	Reinforcement learning control architecture with direct 3D velocity command output, bypassing the position controller and feeding the velocity loop.	91
31	Schematic of the SAC framework, showing the interaction between the policy network (actor), twin Q-function networks (critics), and the entropy-regularized update mechanism.	94
32	UAV position states during undisturbed hover.	96
33	UAV attitude states during undisturbed hover.	97
34	Verification of hover equilibrium in terms of thrust under undisturbed conditions: (a) total thrust command and (b) rotor speeds.	97
35	UAV response to a positive x -axis position step.	98
36	UAV response to a positive y -axis position step.	98
37	Motor RPM allocation during x - and y -axis position step manoeuvres.	99
38	Position states during climb. The z -axis tracks the step setpoints with negligible displacement in x and y	100
39	Attitude states during climb. Roll, pitch, and yaw remain at their equilibrium values throughout the maneuver.	100

40	Verification of the vertical climb response in terms of thrust: (a) per-motor RPM trace and (b) total thrust command during the maneuver.	101
41	Yaw channel step response to a 1 rad command.	102
42	Position states during the yaw step maneuver.	103
43	Attitude states during the yaw step maneuver.	103
44	Position states during simultaneous x -, y -, z -, and yaw step commands under nominal actuator limits.	103
45	Position states during simultaneous step commands under constrained actuator limits (maximum 6000 RPM).	104
46	Yaw tracking under nominal and constrained actuator limits.	104
47	Position and pitch response during a large x -axis step command.	105
48	Kinematic consistency check: the numerical derivative of the simulated position compared against the velocity state, confirming that position and velocity are mutually consistent.	105
49	Aerodynamic response at 5 m/s headwind.	108
50	Aerodynamic response at 10 m/s headwind.	109
51	Aerodynamic response at 15 m/s headwind.	109
52	Aerodynamic response at 5 m/s vertical downwash.	112
53	Aerodynamic response at 10 m/s vertical downwash.	113
54	Aerodynamic response at 15 m/s vertical downwash.	113
55	Aerodynamic response under combined 5 m/s headwind and 5 m/s downwash.	115
56	Aerodynamic response at 2 m/s sidewind.	116
57	Aerodynamic response at 5 m/s sidewind.	117
58	Aerodynamic response during the climb manoeuvre.	118
59	Altitude response during the climb manoeuvre.	119
60	Vertical velocity during the climb manoeuvre.	119
61	Aerodynamic response during the descent manoeuvre.	119
62	Altitude response during the descent manoeuvre.	120
63	Vertical velocity during the descent manoeuvre.	120
64	Position tracking response under a 10 m/s headwind disturbance.	120
65	Altitude tracking response under a 5 m/s downwash disturbance.	121
66	Position tracking response under combined 5 m/s headwind and 5 m/s downwash.	122
67	Lateral position tracking response under a 5 m/s sidewind disturbance.	122
68	Commanded velocity and resulting UAV velocity state logged directly during a training run, over several consecutive RL updates.	124

69	Training performance of the RL guidance policy for the moving-platform landing task. (a) Landing success rate over training steps. (b) Cumulative episode reward over training steps.	129
70	Additional training metrics for the RL guidance policy in the moving-platform landing task. (a) Final distance to the landing platform at episode termination. (b) Episode length as a function of training steps.	130
71	Comparison of UAV position components (x, y, z) over time for the RL and baseline guidance solutions on the same test episode 100, alongside the corresponding landing platform position.	131
72	Vertical descent strategy of the RL guidance policy across three representative episodes, coloured by relative vertical velocity V_z	132
73	Sea state sensitivity analysis: comparison of baseline and RL guidance policy across sea states 4–7, evaluated over 100 episodes per condition under independently sampled JONSWAP wave realizations. Panels show (a) landing success rate, (b) horizontal XY landing error, (c) time to land, and (d) relative vertical speed at touchdown. Error bars denote ± 1 standard deviation.	134
74	Training performance under localized wind disturbances for the three observation configurations: (a) episode success rate and (b) cumulative episode reward.	136
75	Episode 98 trajectory analysis for the $v1$ policy under randomized wind geometry.	137
76	Representative successful episodes under randomized wind geometry ($v2$). . .	138
77	Comparison of RL guidance trajectories for the same episode and initial conditions with (a) the wind disturbance active and (b) the wind disturbance deactivated.	139
78	Representative Episode 98 trajectory for the $v3$ policy under randomized wind-wall geometry. (a) Three-dimensional trajectory showing the failed avoidance maneuver. (b) Top-down view of the same episode.	139
79	Representative trajectories under the baseline guidance law. (a) Unsuccessful episode in which the straight-line approach intersects the downwash region. (b) Successful episode in which the randomized initial geometry leaves the approach path largely clear of the disturbance.	140

LIST OF TABLES

1	Landing condition thresholds and their justification.	5
2	Overview of actor–critic algorithms used in prior work for continuous-action UAV landing problems.	15
3	Summary of RL approaches for UAV landing under disturbances	17
4	Automatic Curriculum Learning Methods and Example Applications	20
5	Curriculum stages for UAV landing on a moving UGV in ACL framework. Retrieved from C. Wang et al. (2023).	21
6	Curriculum stages for UAV landing on a moving UGV. Retrieved from Kim et al. (2023).	21
7	Measurement accuracies of commercially available sensors relevant to UAV state estimation and wind sensing.	26
8	Overview of sensor modeling in related UAV landing studies using RL with domain randomization to model sensor noise and uncertainties.	27
9	WMO Sea State Classification with Significant Wave Height and Dominant Wave Periods ^{17,18}	29
10	Beaufort wind scale with wind speed and wave characteristics. ¹⁹ Conditions in Force 3–8 are identified as relevant for maritime landing operations.(Söker et al., 2000)	32
11	Summary of tuning parameters used in the lift and drag coefficient formulas and force computation.	74
12	PID term selection across the cascade levels.	76
13	Frequency-domain design requirements for the cascaded PID control architecture.	84
14	Frequency-domain properties of the tuned cascaded PID control architecture.	85
15	PID reference step response performance metrics.	87
16	Baseline guidance law tuning parameters.	90
17	Quadplane dynamics and controller verification summary.	95
18	Motor RPM response per commanded input. M1–M4 denote front-left, front-right, rear-left, and rear-right, respectively.	99
19	Step response performance metrics per channel.	101
20	Aerodynamic model verification summary.	107
21	Steady-state aerodynamic angles and side force under headwind conditions.	108
22	Steady-state drag force and thrust compensation under headwind conditions.	110
23	Steady-state lift force, lift coefficient, and angle of attack under headwind conditions.	111

24	Steady-state aerodynamic angles and lift force under vertical downwash conditions.	112
25	Steady-state drag loading and thrust response under vertical downwash conditions.	114
26	Steady-state aerodynamic quantities under combined headwind and downwash.	115
27	Steady-state aerodynamic quantities under sidewind conditions.	116
28	Force magnitude comparison between axial drag under downwash and lateral side force under sidewind at 5 m/s.	117
29	RL functionality verification summary.	122
30	Reward component breakdown across representative episode outcomes, logged from a trained policy.	124
31	Randomized initial conditions at episodes 5, 10, and 15 across two seeded runs, confirming deterministic reproducibility.	125
32	Crash-on-pad cause attribution test results.	126
33	Sampled wind wall geometry over 100 episodes: configured range vs. observed range.	127
34	Wind activation rate over 100 episodes. With $p_{\text{wind}} = 1.0$, the wind hazard is active in every episode.	127
35	Sensitivity to sea state: comparison of baseline and RL guidance policy performance over 100 test episodes per sea state (mean $\pm$ std).	133
36	Success rates under randomized wind-wall geometry (Test 3), compared against the baseline guidance law under identical wind conditions.	141
37	Wind-avoidance metrics under active wind, evaluated on identical wind-block realizations across configurations.	142
38	Landing quality comparison considering successful landings only (mean $\pm$ std).	143

NOMENCLATURE

Abbreviations

Abbreviation	Full Term
6-DoF	Six Degrees of Freedom
A2C	Advantage Actor-Critic
ACL	Automatic Curriculum Learning
AR	Aspect Ratio
BIBO	Bounded-Input Bounded-Output
CFD	Computational Fluid Dynamics
CL	Curriculum Learning
DDPG	Deep Deterministic Policy Gradient
DMPC	Deep Model Predictive Control
DQN	Deep Q-Learning (Deep Q-Network)
EKF	Extended Kalman Filter
ESC	Electronic Speed Controller
GNC	Guidance, Navigation, and Control
GNSS	Global Navigation Satellite System
IMU	Inertial Measurement Unit
INDI	Incremental Nonlinear Dynamic Inversion
JONSWAP	Joint North Sea Wave Project
LQR	Linear Quadratic Regulator
MDP	Markov Decision Process
MPC	Model Predictive Control
NAF	Normalized Advantage Function
NDI	Nonlinear Dynamic Inversion
PID	Proportional–Integral–Derivative
PILCO	Probabilistic Inference for Learning Control
PnP	Perspective- n -Point
POMDP	Partially Observable Markov Decision Process
PPO	Proximal Policy Optimization
RAO	Response Amplitude Operator
RL	Reinforcement Learning
RMSE	Root Mean Square Error
RPM	Revolutions Per Minute
RTK	Real-Time Kinematic
SAC	Soft Actor-Critic
SARSA	State–Action–Reward–State–Action
SMC	Sliding Mode Control
SPL	Self-Paced Learning
TD3	Twin Delayed Deep Deterministic Policy Gradient
TKE	Turbulence Kinetic Energy

Continued on next page

Abbreviation	Full Term
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
UKF	Unscented Kalman Filter
UWB	Ultra-Wideband
VTOL	Vertical Take-Off and Landing
WMO	World Meteorological Organization

Greek Symbols

Symbol	Unit	Description
<i>Aerodynamics and Vehicle Attitude</i>		
α	deg	Angle of attack
β	deg	Sideslip angle
η	–	Generalized aerodynamic velocity norm
$\boldsymbol{\eta}$	rad/(rad/s)	Generalized freestream linear/angular velocity vector
Φ	–	Aerodynamic coefficient matrix (Φ -theory)
$\sigma(k, \alpha_0, \alpha)$	–	Pseudo-sigmoid aerodynamic blending function
ϕ	rad	Roll angle
θ	rad	Pitch angle
ψ	rad	Yaw angle
<i>Reinforcement Learning</i>		
γ	–	Discount factor determining the relative weighting of future rewards
ϵ	–	Exploration probability (ϵ -greedy)
θ	–	Policy parameter vector in π_θ
π	–	Policy
α	–	SAC temperature parameter
<i>Wave and JONSWAP Model</i>		
$\Delta\omega_k$	rad/s	Frequency spacing between wave components in the JONSWAP spectrum
ϕ_k	rad	Phase angle of the k th wave component in the JONSWAP spectrum
σ	–	JONSWAP spectral width parameter
ω	rad/s	Angular frequency
ω_k	rad/s	Discretized angular frequency component in the JONSWAP spectrum
ω_p	rad/s	Peak (dominant) angular frequency

Continued on next page

Symbol	Unit	Description
<i>Control and Vehicle Dynamics</i>		
α_D	–	PID derivative filter coefficient
Δt	s	Controller/simulation timestep
ε	m/s	Sideslip force smoothing parameter
$\Lambda(\omega)$	–	Skew-symmetric matrix of body angular velocity (quaternion kinematics)
τ	N·m	Torque (roll τ_ϕ , pitch τ_θ , yaw τ_ψ)
λ_d	s	Derivative filter time constant
ω_c	rad/s	Loop crossover frequency
$\boldsymbol{\omega}$	rad/s	Body-frame angular velocity vector $[p; q; r]^\top$
$\boldsymbol{\Omega}_{\text{cmd}}$	rad/s	Commanded body angular rate vector
κ	–	Weighting parameter (linear vs. angular velocity contribution)
κ_τ	–	Torque-to-thrust ratio (K_M/K_F)
ρ	kg/m ³	Air density

Roman Symbols

Symbol	Unit	Description
<i>Reinforcement Learning</i>		
$\mathbf{a}_t$	–	Current action vector sampled by the reinforcement learning policy
G_t	–	Expected cumulative discounted return
$J(\pi)$	–	Policy objective function
O	–	Observation probability function (POMDP)
o_t	–	Observation received by the agent
P	–	State-transition probability function
s_t	–	State at time step t
R, r	–	Reward function, reward components
Q	–	Action-value function
V	–	State-value function
$\mathcal{H}(\pi)$	–	Policy entropy
$\mathcal{H}_{\text{target}}$	–	SAC target entropy
<i>Wave and JONSWAP Model</i>		
A_k	m	Amplitude of the k th discretized wave component in the JONSWAP spectrum
H_s	m	Significant wave height in the JONSWAP spectrum
N	–	Number of discretized wave-frequency components in the JONSWAP spectrum

Continued on next page

Symbol	Unit	Description
$S(\omega)$	$\text{m}^2 \cdot \text{s}$	JONSWAP wave energy spectral density
T_p	s	Peak (dominant) wave period
<i>Aerodynamics</i>		
b	m	Wingspan
c	m	Mean aerodynamic chord
$\mathbf{C}_F$	–	Aerodynamic force coefficient vector
$\mathbf{C}_M$	–	Aerodynamic moment coefficient vector
$\mathbf{F}_a$	N	Aerodynamic force vector
A_{ref}	m^2	Reference wing area
V_∞	m/s	Freestream airspeed
V_a	m/s	Airspeed magnitude
$\mathbf{w}_a$	–	Aerodynamic wrench vector
x, y, z	m	Position coordinates
A_L	–	Small-angle lift slope
$a_0, a_1, a_2, a_3, a_{1,D}$	–	Lift/drag blending model coefficients
$C_{L,0}$	–	Lift coefficient at zero angle of attack
C_D, C_L, C_Y	–	Drag, lift, side-force coefficients
D	N	Drag force
L	N	Lift force
Y	N	Lateral (sideslip) aerodynamic force
$V_{\min}$	m/s	Minimum airspeed for aerodynamic force evaluation
V_{xz}	m/s	Airspeed projected on pitch plane
<i>Vehicle and Rotor Dynamics</i>		
g	m/s^2	Gravitational acceleration
$\mathbf{H}_B$	$\text{kg} \cdot \text{m}^2/\text{s}$	Angular momentum (body frame)
$\mathbf{J}$	$\text{kg} \cdot \text{m}^2$	Inertia matrix
K_F, K_T, K_M	–	Rotor thrust/moment coefficients
l, l_m	m	Motor moment arm
m	kg	Vehicle mass
N_r	–	Number of lift rotors
p, q, r	rad/s	Body-frame roll, pitch, yaw rates
$\mathbf{q}$	–	Unit quaternion $[q_0, q_1, q_2, q_3]^\top$
$\mathbf{R}$	–	Rotation matrix ($SO(3)$)
u, v, w	m/s	Body-frame velocity components
$F_{\max}$	N	Maximum motor thrust
F_T	N	Total thrust magnitude
$F_{T,i}$	N	Thrust generated by rotor i
$F_{T,\max}$	N	Maximum available total thrust
$n_i, n_{\max}$	rad/s	Rotor i angular speed / maximum rotor speed
$\mathbf{M}_{\text{rotor}}^B$	N·m	Rotor-induced body moment vector
T_{lag}	s	ESC/motor first-order lag time constant

Continued on next page

Symbol	Unit	Description
<i>Control and System Dynamics</i>		
$\mathbf{C}$	–	Screw reference (geometric scaling) matrix
$\mathbf{D}$	–	Diagonal moment-arm scaling matrix
$\mathbf{d}_k$	–	Filtered derivative term (PID)
$\mathbf{e}$	–	Generic control error vector ($\mathbf{e}_\Omega, \mathbf{e}_v, \mathbf{e}_p, \mathbf{e}_\phi$)
$G(s)$	–	Plant transfer function
GM	dB	Gain margin
K_P, T_I, T_D	–, s, s	PID proportional gain, integral time, derivative time
K_t	–	Anti-windup back-calculation gain
$L(s)$	–	Open-loop transfer function
M_S, M_T	–	Sensitivity / complementary sensitivity peak
N_f	–	Derivative filter (noise-bandwidth) coefficient
PM	deg	Phase margin
s	–	Laplace variable
$S(s), T(s)$	–	Sensitivity / closed-loop transfer function
t	s	Time
t_r, t_s	s	Rise time, settling time

Subscripts

Subscript	Description
∞	Freestream / far-field quantity
k	Index of a discretized wave-frequency component
p	Peak / dominant (wave spectrum); or propeller (when on α)
rel	Relative quantity between UAV and target
t	Target (ship) state
v	Vehicle (UAV) state
cmd	Commanded/reference quantity
meas	Measured quantity
des	Desired state
cur	Current state
sat	Saturated quantity
aw	Anti-windup correction
surf	Surface (ship deck) velocity
B, I, A, S	Body, inertial, aerodynamic, stability reference frame

Part I:

Problem Definition
and Background

1 INTRODUCTION

The growing adoption of Unmanned Aerial Vehicles (UAVs) has expanded the capability to perform missions that would otherwise be too dangerous, costly, or impractical for human operators, particularly in long-endurance and around-the-clock operations (Z. Ma et al., 2022). The researched applications include package delivery, surveillance, environmental monitoring, and disaster response (Amendola et al., 2024).

Among UAV platforms, hybrid configurations have emerged as a major research and development focus due to their ability to combine the vertical take-off and hovering capability of rotorcraft with the efficient, high-speed forward flight of fixed-wing aircraft (Z. Ma et al., 2022). Mission profiles requiring both high endurance and extended range, such as offshore wind farm inspection (Xu et al., 2025), maritime surveillance (Zhao et al., 2023), and search and rescue operations at sea (Messmer et al., 2024), are particularly well suited to the deployment of such hybrid platforms. However, maritime operations often require UAVs to autonomously land on moving vessels, which presents a significant challenge due to the combined effects of platform motion (Ibinabo & Tamunodukobipi, 2019), environmental disturbances (Söker et al., 2000), and sensor degradation (Xie et al., 2020).

Hybrid platforms, due to their winged airframes, are particularly susceptible to aerodynamic forces during the landing phase, making precise touchdown in the presence of unpredictable, highly variable wind conditions especially challenging. While heuristic guidance approaches for hybrid UAV landing exist, they typically do not explicitly account for localized turbulence or the complex, rapidly changing flow environment in the vicinity of a ship's landing deck in the loop. This limitation highlights the need for adaptive methods that can sense and incorporate wind disturbances in real time to generate robust landing trajectories. Accordingly, this thesis focuses on the design and development of a guidance controller addressing these requirements.

Reinforcement learning (RL) has emerged as a promising approach in this context (Amendola et al., 2024). By interacting directly with the environment, RL agents can learn control policies that adapt to a wide range of operating conditions without requiring explicit disturbance modeling. Furthermore, RL frameworks naturally enable multi-objective optimization through reward design, allowing simultaneous consideration of objectives such as landing accuracy, control effort, and robustness (Sutton & Barto, 2018).

Therefore, this thesis investigates RL-based guidance strategies for hybrid UAVs performing autonomous landings on maritime platforms, examining whether wind-aware agents can learn to avoid localized turbulent flow regions to improve mission safety. This thesis is structured in four parts. Part I introduces the problem framework and presents the literature review, covering the state of the art for UAV landing, including classical and RL-based methods, as well as environment modeling of ship dynamics and wind disturbances. Part II presents the scientific paper developed as part of this thesis. Part III provides extended methodology and results beyond the scope of the paper, including general modeling, verification, and additional analyses. Finally, Part IV concludes the thesis, addressing the research questions and discussing limitations and directions for future work.

2 LITERATURE REVIEW

This chapter reviews the existing literature on the maritime landing problem. First, the problem is framed within the context of autonomous systems, followed by a discussion of classical guidance and control strategies commonly adopted in prior research. RL is then introduced as an alternative approach for UAV control, as proposed in this thesis, and related work on UAV tracking and landing in maritime or comparable environments is reviewed to motivate the formulation of an RL framework. Curriculum learning (CL) is subsequently presented, together with relevant applications to UAV landing tasks. Finally, modeling approaches for lift-and-cruise quadplanes are discussed, along with methods from literature for representing ship dynamics and wake-induced wind disturbances.

2.1 Maritime Landing Problem

Landing on maritime platforms poses a significant challenge, not only due to the complex six-degree-of-freedom motion of the target ship's deck but also to severe, highly variable wind disturbances in offshore environments (Ibinabo & Tamunodukobipi, 2019). Since the 1960s, helicopter operations have been extensively studied, and various mechanical aids, such as landing hooks, stabilizing decks, and specialized guidance systems, have been developed to facilitate safe maritime landings. Despite these advances, the landing phase of offshore helicopter missions remains particularly hazardous, accounting for a substantial proportion of total accidents, among which a notable percentage result in serious injuries or fatalities to personnel (Cao et al., 2022; Setiawan et al., 2025).

In recent years, the development of Vertical Take-Off and Landing (VTOL) UAVs has enabled autonomous landings on maritime platforms, making them suitable for offshore applications. Their deployment could significantly reduce the risk of injury to human personnel and allow for specialized operations.

For both manned helicopters and unmanned VTOL vehicles, the maritime landing problem is defined to satisfy several complementary requirements that together define successful task execution:

1. **Landing accuracy:** The vehicle is required to reach the designated landing site, ensuring precise alignment of position and orientation at touchdown.
2. **Landing safety:** The landing maneuver must satisfy constraints on vehicle velocity and attitude at touchdown to prevent undesirable outcomes such as bouncing, tipping, sliding, or structural damage.
3. **Stable trajectory tracking:** The vehicle is required to track the reference landing trajectory throughout the landing window with high accuracy and smoothness, while maintaining dynamic stability and avoiding oscillatory or erratic behavior.
4. **Robustness:** Acceptable performance must be maintained in the presence of modeling uncertainties, external disturbances such as wind, and sensor noise.

The first two objectives are quantified using the absolute difference between the UAV and target platform states. Both the UAV and the ship are described using two primary reference

frames. The inertial frame ($\mathcal{F}_I$) is an Earth-fixed reference frame with axes x_I , y_I , and z_I , which is shared by both vehicles. In deriving the equations of motion and defining the states, the Earth is assumed to be flat and non-rotating.

The body frame ($\mathcal{F}_B$) is fixed to each vehicle and moves with it. For the UAV, it is attached to the center of mass, with x_b pointing forward, y_b to the left, and z_b pointing up. For the ship, the body frame is fixed to the center of the landing platform. Since rotational motion of the ship is neglected in this work, the frame can be shifted from the ship's center of gravity to the center of the landing pad without loss of generality.

A representative illustration of the UAV landing on a ship is shown in Figure 1, highlighting both the inertial and body-fixed frames.

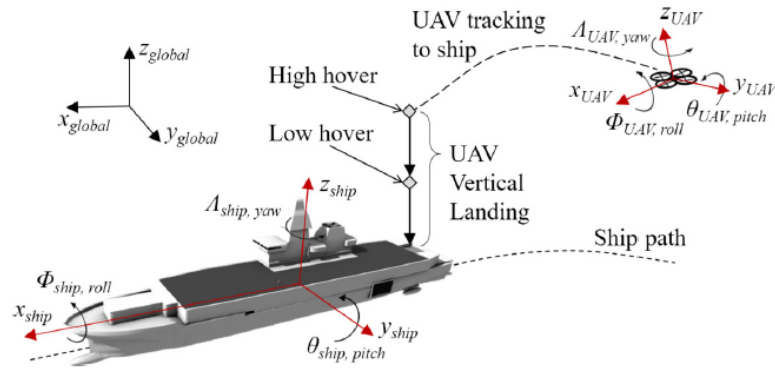


Figure 1: Deck landing mission task elements for UAVs. Retrieved from Abujoub et al. (2020).

The inertial position, translational velocity, and orientation of the ship's landing deck are given as a function of time in Equation 1. Although the ship's rotational dynamics are not modeled in this work, the instantaneous orientation of the landing deck, $\boldsymbol{\eta}_t$, is retained as part of the general target state vector. This allows the relative attitude between the UAV and the deck to be evaluated during the landing phase.

$$\mathbf{x}_t = [\mathbf{p}_t \quad \mathbf{v}_t \quad \boldsymbol{\eta}_t]^T \in \mathbb{R}^9 \quad (1)$$

$$\mathbf{p}_t = \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix}, \quad \mathbf{v}_t = \begin{bmatrix} \dot{x}_t \\ \dot{y}_t \\ \dot{z}_t \end{bmatrix}, \quad \boldsymbol{\eta}_t = \begin{bmatrix} \phi_t \\ \theta_t \\ \psi_t \end{bmatrix}$$

Similarly, the UAV's state is defined in Equation 2.

$$\mathbf{x}_v = [\mathbf{p}_v \quad \mathbf{v}_v \quad \boldsymbol{\eta}_v]^T \in \mathbb{R}^9 \quad (2)$$

$$\mathbf{p}_v = \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix}, \quad \mathbf{v}_v = \begin{bmatrix} \dot{x}_v \\ \dot{y}_v \\ \dot{z}_v \end{bmatrix}, \quad \boldsymbol{\eta}_v = \begin{bmatrix} \phi_v \\ \theta_v \\ \psi_v \end{bmatrix}$$

At the instant of landing, the objective is to satisfy the alignment and velocity matching conditions prescribed in Equation 3.

$$\begin{aligned}
\lim_{t \rightarrow t_f} |p_{rel,z}(t)| &= 0, \\
\lim_{t \rightarrow t_f} \|\mathbf{p}_{rel,xy}(t)\| &\leq p_{xy}^{\max}, \\
0 \leq \lim_{t \rightarrow t_f} v_{rel,z}(t) &\leq v_z^{\max}, \\
\lim_{t \rightarrow t_f} \|\boldsymbol{\eta}_{rel}(t)\| &\leq \eta_{\max}.
\end{aligned} \tag{3}$$

Where, the relative state between the target platform and the UAV is defined as the difference between their respective states:

$$\mathbf{state}_{rel} = \mathbf{state}_t - \mathbf{state}_v \in \mathbb{R}^9 \tag{4}$$

The thresholds for successful landing are defined in Table 1 and reflect three practical considerations. The 0.5 m horizontal tolerance corresponds to the usable landing area of the target ship deck, ensuring that the vehicle touches down within the designated zone. The 0.7 m/s vertical velocity limit is derived from the structural integrity requirements of the airframe, beyond which impact loads may risk damaging the landing gear or hull. The 10° attitude constraint ensures that the vehicle remains within its tipping stability margin at contact, preventing lateral sliding or rollover on a moving and wet deck surface.

Table 1: Landing condition thresholds and their justification.

Parameter	Symbol	Value	Basis
Horizontal position tolerance	$p_{xy}^{\max}$	0.5 m	Platform deck dimensions
Maximum vertical velocity	$v_z^{\max}$	0.7 m/s	Airframe structural limits
Maximum attitude deviation	$\eta_{\max}$	10°	Tipping stability margin

The third objective can be assessed using tracking-error metrics, such as integrated position and velocity errors during the landing phase, as well as smoothness measures based on control-input variation or oscillatory behavior. Finally, the fourth objective can be evaluated using metrics such as worst-case or root-mean-square errors across multiple scenarios with varying wind conditions, model variations, or sensor noise.

Autonomous Landing Control Framework

An autonomous landing control problem aims to achieve the prescribed landing conditions using an onboard computer that computes and executes control commands online at the available sampling rate (Kamath et al., 2020). In shipboard landing operations, multiple phases can be identified, such as approach, descent, and touchdown, which may require different control frameworks. Nevertheless, these phases are typically addressed within a common hierarchical control structure, as visualized in Figure 2.

The guidance controller operates at the highest level of the control hierarchy. Its role is to generate reference trajectories or setpoints based on the landing objective and prevailing

environmental conditions. The control framework then aims to follow these references as accurately as possible, typically formulated as a reference-tracking problem, by converting the guidance references into appropriate actuator commands.

Both the guidance and flight controllers rely on accurate knowledge of the vehicle state and its environment, which is provided by the navigation system. This system fuses measurements from onboard sensors and, when available, offboard sensors or external references to estimate the vehicle's current state and information about its surroundings.

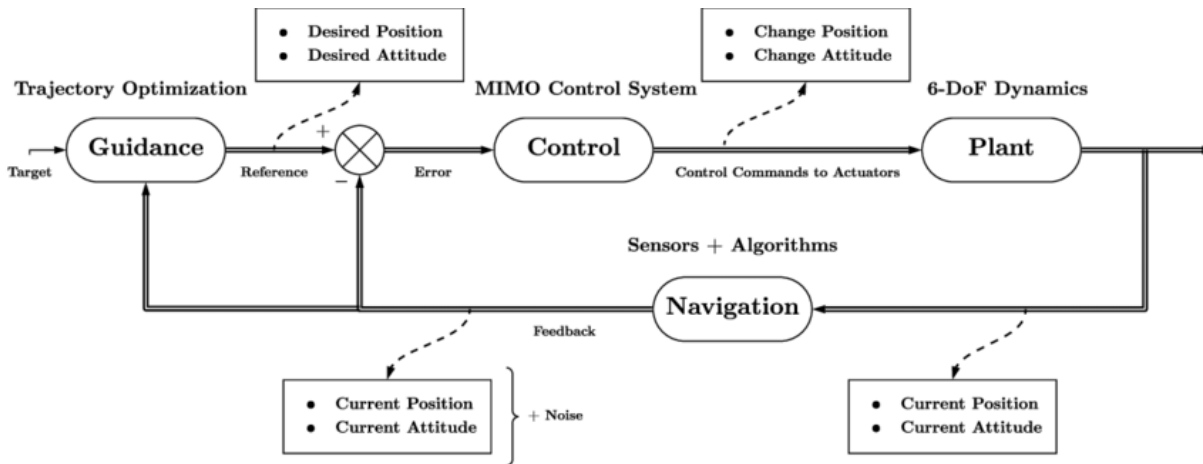


Figure 2: Block diagram depicting a guidance, navigation, and control (GNC) framework. Retrieved from Kamath et al. (2020).

UAV Platforms and Suitability

Maritime operations for UAV platforms include maritime surveillance (Zhao et al., 2023), offshore wind farm inspection and maintenance (Xu et al., 2025), and maritime search and rescue (Messmer et al., 2024). These use cases impose requirements that influence the suitability of different UAV platforms for their mission scope. The key requirements can be summarized as follows:

- **Environmental Robustness and Weather Tolerance:** UAVs must operate reliably in harsh marine environments, including high winds, salt spray, and variable temperatures. This requires adequate airframe durability and wind-resistance capabilities (United States Coast Guard, 2011).
- **Long-Range and Endurance Flight:** Missions often cover large areas far from shore, requiring UAVs with extended flight endurance, long communication ranges, and sustained performance to monitor expansive ocean areas or dispersed offshore infrastructure.¹
- **Moderate Payload Capability for Onboard Sensors:** UAVs must carry advanced sensors, such as electro-optical/infrared cameras, radar, or other payloads for surveillance, target detection, and environmental monitoring. These sensors require considerable payload capacity.¹

As a result, evaluating UAV suitability requires the definition of a set of key performance parameters. Within the scope of this thesis, these are defined as range, endurance, payload capacity, and wind susceptibility during landing. Wind susceptibility is quantified using data on maximum sustained ground-level horizontal wind speeds. Although alternative launch and recovery methods exist for non-VTOL platforms, this work considers only VTOL-capable UAVs.

The main UAV categories considered include multirotors, single-rotor helicopters, tiltrotor and tiltwing configurations, hybrid VTOL (lift-and-cruise) systems, and tailsitters as illustrated in Figure 3. A broad survey of platforms used for surveillance applications across these configuration types indicates that hybrid VTOL and tiltrotor/tiltwing systems offer particularly suitable characteristics for maritime operations. This is mainly due to their combination of vertical take-off and landing capability with efficient forward-flight performance, which significantly improves range and endurance (M. Zhou et al., 2020). Tailsitter configurations are more susceptible to headwind disturbances during transition and landing phases, which makes them less favorable for maritime operations (Ducard & Allenspach, 2021).

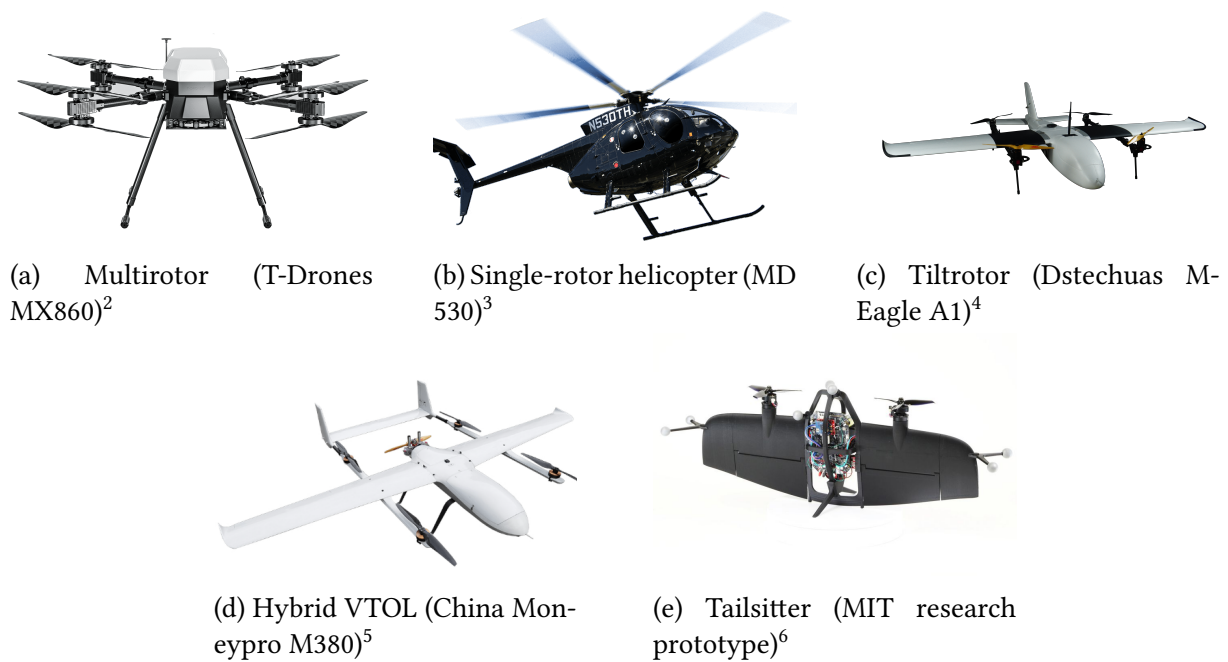


Figure 3: Representative UAV configurations considered for maritime operations.

Compared to conventional helicopters, hybrid VTOL and tiltrotor/tiltwing platforms are increasingly available at lower purchase and operational costs, contributing to their growing

¹<https://www.mmcuav.com/learning-center/coastal-and-port-surveillance>

²<https://www.unmannedsystemstechnology.com/2023/04/t-drones-mx860-coaxial-multirotor-drone/>

³<https://aeroinurance.com/aircraft-insurance/helicopter-insurance/>

⁴<https://www.dstechuas.com/-p3106531.html>

⁵<https://www.moneyprouav.com/product/china-moneypro-m380-heavy-lift-hybrid-vtol-fixed-wing-with-3-8m-wingspan-efi-engine10kg-load-capacity/>

⁶<https://news.mit.edu/2023/planning-algorithm-tailsitter-aircraft-0823>

competitiveness with more established technologies ⁷.

When comparing hybrid platforms to tiltrotor and tiltwing configurations, the presence of rotating actuators and the associated transition dynamics introduce additional mechanical and control complexity during the critical landing phase (Quan et al., 2025). In contrast, hybrid VTOL (lift-and-cruise) systems avoid in-flight actuator rotation and exhibit more straightforward transition behavior, making them a particularly promising configuration for robust maritime landing applications. Consequently, this thesis focuses on hybrid VTOL configurations as the dynamic system under investigation.

2.2 Classical Control Strategies for UAV Landing

Classical controllers comprise a broad class of control strategies that are typically based on feedback principles and may or may not rely on an explicit system model. Their design can range from simple heuristic tuning methods, such as PID control, to more structured model-based approaches derived analytically or through optimization. The primary objectives are stability, reference tracking, and disturbance rejection.

These controllers are commonly organized into a hierarchical structure consisting of two levels. The guidance level generates desired trajectories or setpoints for the vehicle, while the flight control level computes actuator commands to track these references.

Guidance Control

Commonly implemented guidance approaches for shipboard landings rely on heuristic procedures inspired by helicopter operations, as shown in Figure 1. These methods prescribe a straight-line, horizontal approach trajectory at a fixed altitude to achieve alignment with the landing platform. This reference is updated at each sampling instant to account for the relative motion of the ship. Once the descent phase is initiated, a similar approach is applied. The reference trajectory is adapted to include vertical motion while maintaining alignment with the landing pad, ensuring a smooth and controlled descent. (Abujoub et al., 2020; Ding & Huang, 2023)

In contrast to these prescribed sequences, recent studies have explored gradient-descent-based optimal trajectory planning for UAV landings on moving vessels (Tan et al., 2025). These algorithms compute trajectories by iteratively adjusting the state and control sequences to minimize a predefined cost function, using gradient information derived from the system dynamics and constraints. In Gao et al. (2023), Model Predictive Control (MPC) is used to generate the UAV's landing trajectory by predicting the future motion of the ship using its dynamic model. The UAV trajectory is then planned in a way that compensates for the predicted deck motion, allowing the vehicle to align and land safely on the moving platform. Another recently studied approach involves visual-servoing and image-based path guidance, which generates real-time approach paths by reacting directly to visual cues from landing markers or moving platforms (Keipour et al., 2022).

Tracking Control

Tracking controllers generate actuator commands to regulate the vehicle toward a desired target state. Different control methodologies vary in the extent to which explicit system

⁷<https://www.fortunebusinessinsights.com/industry-reports/unmanned-aerial-vehicle-uav-market-101603>

models are used in the controller design, whether the underlying dynamics are linearized, and the degree of robustness or adaptability to changing vehicle and environmental conditions.

PID control is one of the most widely used control strategies in industry, applied across a broad range of systems (Smeur et al., 2016). It operates by reacting to the error between the desired and current system states and is therefore not inherently model-based in structure. However, achieving satisfactory performance typically requires controller tuning based on detailed knowledge of the system dynamics, a process that is often challenging for the end user. For systems with nonlinearities, gain scheduling across the flight envelope to account for variations in the vehicle dynamics is also required. (Cárdenas R. et al., 2020) Furthermore, from a theoretical perspective, these controllers do not constitute the highest-performance control architecture (Eschmann et al., 2024).

The Linear Quadratic Regulator (LQR) is an optimal state-feedback control method that determines the feedback gain matrix (K) by minimizing a quadratic cost function (J), balancing state regulation and control effort. Unlike PID control, where gains must be tuned empirically across multiple coupled channels, LQR reduces the design problem to selecting two interpretable weighting matrices (Q and R), with the feedback gains then derived automatically through optimization.

There are also explicitly model-based control approaches that utilize the system dynamics directly in the controller design. Methods such as feedback linearization, backstepping, and sliding mode control (SMC) actively cancel or exploit nonlinearities in the vehicle model, enabling precise tracking guarantees that linear controllers cannot provide when the system operates far from an equilibrium (Smeur et al., 2016). Nonlinear Dynamic Inversion (NDI) is a classical feedback linearization technique that inverts the full nonlinear model to compute the required control input, whereas Incremental Nonlinear Dynamic Inversion (INDI) instead uses a first-order approximation combined with measured state derivatives, trading exact model dependence for greater robustness to model uncertainty (Sieberling et al., 2010).

MPC extends this further by solving a finite-horizon optimization problem at each timestep, allowing it to explicitly handle state and input constraints, such as actuator saturation and stability boundaries, that are otherwise difficult to enforce (Feng et al., 2018). This receding horizon strategy allows MPC to anticipate and correct for deviations proactively, rather than reacting only once tracking error has already accumulated, a fundamental advantage over purely reactive controllers in systems with significant lag or hard operational limits. (de Jong, 2019)

2.3 RL Based Control Strategies for UAV Landing

The problem of UAV landing has attracted increasing attention in recent years, leading to a growing body of literature on RL-based control approaches. RL is particularly well suited to this task, as it enables autonomous learning of complex landing maneuvers under a wide range of operating conditions (S. Chen et al., 2024; Xiong et al., 2025). Beyond reproducing established pilot procedures, RL-based methods can discover novel guidance and path-planning strategies that adapt to environmental disturbances such as wind and ship motion.

Introduction to RL

RL is a branch of machine learning in which an agent learns to make sequential decisions through interaction with an environment. At each discrete time step, the agent observes the environment’s current state and selects an action from a predefined action space. This action influences the environment’s evolution and leads to a new observation. In return, the environment provides a scalar feedback signal, known as the reward, which evaluates the immediate quality of the agent’s action in that state. Over time, by trial and error and by maximizing cumulative reward, the agent aims to learn an optimal policy that defines which action to take in every possible state.

A Markov Decision Process (MDP) is a mathematical framework used to formalize sequential decision-making problems in RL. An MDP is defined by a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $\mathcal{S}$ is the set of all possible states s in which the agent can be at any given time, $\mathcal{A}$ is the set of all actions a the agent can take, $P(s_{t+1} \mid s_t, a_t)$ is the probability of reaching a next state s_{t+1} when taking action a_t from state s_t , $R(s_t, a_t)$ is the scalar feedback received by the agent after taking action a_t in state s_t , and $\gamma \in [0, 1]$ is a discount factor used to determine the present value of future rewards. (Kaelbling et al., 1996; Sutton & Barto, 2018) These components inherently describe the environment in which the agent operates. In Figure 4, the interaction of an agent with the environment is visualized, where the agent observes the full state directly.

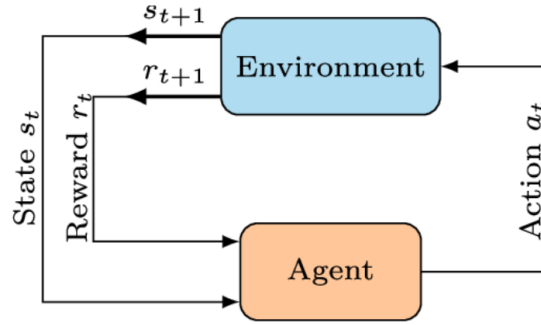


Figure 4: Diagram representing the interaction between an agent and its environment in an MDP. Retrieved from Sutton and Barto (2018).

A process satisfies the Markov property if the probability distribution of the next state depends only on the current state and action, and not on the sequence of past states or actions. Formally, for all $s_t \in \mathcal{S}$, $a_t \in \mathcal{A}$, and $s_{t+1} \in \mathcal{S}$:

$$P(s_{t+1} \mid s_t, a_t, s_{t-1}, a_{t-1}, \dots, s_0, a_0) = P(s_{t+1} \mid s_t, a_t). \quad (5)$$

This implies that the current state is a sufficient statistic for the entire history, a foundational requirement for modeling RL problems as MDPs. Under the assumptions of the Markov property and complete knowledge of the transition dynamics, an optimal policy is guaranteed to exist.

Not all RL problems can be realistically modeled as MDPs, because the MDP assumptions may not hold. In particular, there are cases where the agent cannot directly observe the full environment state that governs the transition dynamics. As a result, the agent cannot acquire complete knowledge of the environment, and the guarantee of an optimal policy no

longer holds. Such problem formulations are captured by a Partially Observable Markov Decision Process (POMDP), which extends the MDP framework to account for incomplete state information. A POMDP is typically defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \Omega, O, \gamma)$, where $\mathcal{S}$, $\mathcal{A}$, P , R , and γ retain the same meaning as in an MDP, Ω is the set of observations available to the agent, and O is the observation probability function $O(o_t \mid s_t, a_{t-1})$ that describes the likelihood of observing $o_t \in \Omega$ given the underlying state s_t and previous action a_{t-1} . In a POMDP, the agent typically maintains a belief state, which is a probability distribution over $\mathcal{S}$, to estimate the unobserved state and guide decision-making.

In RL, a policy function governs the agent’s behavior by specifying the probability of selecting an action a_t given an observation o_t at state s_t , formally denoted as $\pi(a_t \mid o_t)$. The learning objective for an agent is to approximate or find the optimal policy defined in Equation 6, which maximizes the expected cumulative discounted reward. Executing this optimal policy from the initial state yields the highest attainable expected return and characterizes the best achievable interaction with the environment under the given action space and problem formulation.

$$\pi^* = \arg \max_{\pi} \mathbb{E}[G_t] = \arg \max_{\pi} \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \right] \quad (6)$$

Here, the expected cumulative discounted reward G_t is the sum of all future rewards from time step t onward, where the discount factor γ weights immediate rewards more strongly than distant ones and ensures convergence of the infinite sum for $\gamma < 1$.

To evaluate the long-term desirability of states and actions under a given policy, value functions are defined. Since the immediate reward received at a single time step does not capture the future consequences of an action, value functions provide a quantitative measure of the expected cumulative discounted reward associated with future trajectories.

The state-value function of a policy π is defined in Equation 7, representing the expected return obtained when starting from state s and following policy π .

$$V^{\pi}(s) = \mathbb{E}_{\pi} \left[G_t \mid S_t = s \right] \quad (7)$$

Similarly, the action-value function is defined in Equation 8, which quantifies the expected return when taking action a in state s and subsequently following policy π .

$$Q^{\pi}(s, a) = \mathbb{E}_{\pi} \left[G_t \mid S_t = s, A_t = a \right] \quad (8)$$

Different mathematical approaches exist for estimating the optimal policy, and the choice of method critically influences an agent’s ability to learn under a given problem formulation. The three principal classes of RL algorithms are value-based, policy-based, and actor-critic methods.

Value-based RL methods aim to estimate the optimal value function, from which the optimal policy can be derived. In model-free settings, this is typically achieved through the action-value function $Q(s, a)$, since it directly provides a mapping from states to optimal actions. By interacting with the environment, the agent collects sample transitions that provide

information about long-term returns associated with different state–action pairs. These samples are used to iteratively update the value estimates.

In the tabular setting, the action-value function is updated using observed rewards and estimates of future values, enabling the agent to gradually refine its approximation of the optimal function $Q^*(s, a)$. Once Q^* is obtained, the optimal policy follows directly by selecting, in each state, the action that maximizes the estimated value:

$$\pi^*(s, a) = \arg \max Q^*(s, a), \quad (9)$$

A general form of the action-value function update is given by:

$$Q^\pi(s_t, a_t) \leftarrow Q^\pi(s_t, a_t) + \alpha \left[\underbrace{\text{Target}_t}_{\text{algorithm-dependent}} - Q^\pi(s_t, a_t) \right], \quad (10)$$

where α is the learning rate, and Target_t is defined according to the specific RL algorithm employed. For example:

- **Monte Carlo:** $\text{Target}_t = G_t = \sum_{k=0}^{T-t} \gamma^k r_{t+k}$, using the full return from the episode.
- **Q-Learning (off-policy):** $\text{Target}_t = r_t + \gamma \max_{a'} Q(s_{t+1}, a')$, using the greedy next action and bootstrapping from the next state’s value.
- **SARSA (on-policy):** $\text{Target}_t = r_t + \gamma Q^\pi(s_{t+1}, a_{t+1})$, using the next action actually selected by the policy and bootstrapping from the next state’s value.

A notable limitation of value-based RL methods is that they tend to produce a greedy policy when actions are selected by maximizing the estimated action-value function (Mnih et al., 2015). Such a policy exploits the current value estimates but does not naturally encourage exploration. To compensate for this, value-based algorithms typically incorporate an external exploration mechanism, most commonly the ϵ -greedy strategy, in which the agent selects a random action with probability ϵ and the greedy action with probability $1 - \epsilon$. This introduces the necessary balance between exploration and exploitation that is not inherently provided by the value-based formulation itself.

Furthermore, tabular value-based methods are effective in environments with small, discrete state and action spaces. In high-dimensional or continuous domains, however, the exponential growth of state–action combinations makes tabular value representation impractical and prone to instability, thereby preventing these methods from scaling effectively (Mnih et al., 2015). To address these limitations, neural networks are introduced to approximate value functions, giving rise to Deep Q-Learning (DQN) as visualized in Figure 5. This enables value-based methods to handle high-dimensional and continuous state spaces; however, the inherently discrete action selection in value iteration still prevents them from scaling effectively to continuous action spaces.

Policy-based methods, in contrast, directly parameterize the policy $\pi_\theta(a|s)$, often by means of using neural networks to approximate it, and optimize the parameters θ to maximize expected return (Sutton et al., 2000). Because they operate directly on the policy, these methods naturally support continuous action spaces and can represent stochastic behaviors, enabling

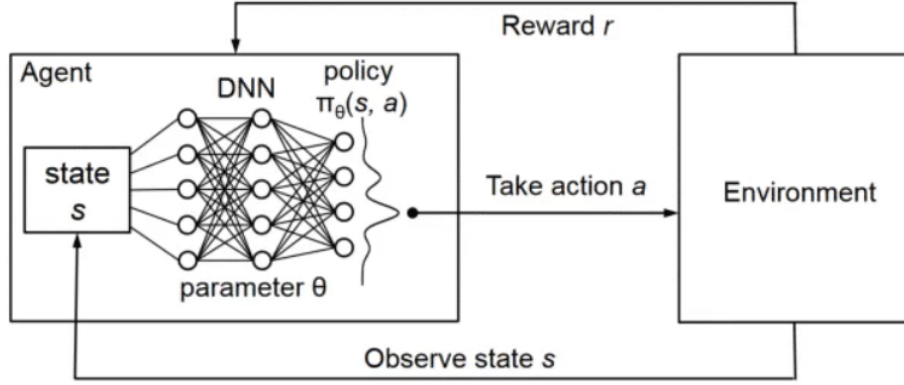


Figure 5: Model-free deep reinforcement learning architecture (DQN). Retrieved from Mao et al. (2016).

principled exploration and robustness under uncertainty (R. J. Williams, 1992). To improve performance over time, policy-based algorithms guide their search using policy gradients, adjusting parameters in directions that increase expected return by estimating an objective function and updating the policy to maximize this objective. The objective function can be formulated in many ways, for instance, by directly computing the expected return as formulated in Equation 11.

$$J(\pi_\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [G_t(\tau)]. \quad (11)$$

Since the expectation cannot generally be computed analytically in complex environments, policy-based methods approximate it by sampling trajectories from the current policy. Then, the policy updates using stochastic gradient ascent can be formulated as:

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta), \quad (12)$$

where,

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} [\nabla_\theta \log \pi_\theta(a_t | s_t) \hat{g}_t]. \quad (13)$$

A major limitation of policy gradient methods that use G_t as the estimator of the policy gradient signal $\hat{g}_t$, is the high variance of their gradient estimates (Greensmith et al., 2004). Because the return from each sampled trajectory can vary widely due to environmental and policy stochasticity, the resulting policy gradient estimates become noisy and inconsistent. This variance makes the update direction difficult to estimate reliably, leading to unstable learning dynamics and requiring large numbers of samples or small learning rates for convergence. Consequently, pure policy gradient methods such as REINFORCE tend to suffer from slow and sample-inefficient learning unless combined with variance-reduction techniques.

Variance-reduction considerations motivate the development of actor-critic methods, which form a third major category of RL algorithms. Actor-critic methods combine the strengths of

policy gradient approaches and value-based methods: the actor updates the policy parameters directly using policy gradient theory, while the critic learns an explicit estimate of a value function (typically a state-value, action-value, or advantage function) to reduce variance in the policy gradient estimates, as visualized in Figure 6. In practice, this architecture requires two neural networks: one that represents the policy by mapping states to action distributions, and another that approximates the chosen value function to provide more stable and informed updates of the policy parameters. The value network is incorporated into the objective function definition for the policy updates. (Fujimoto, van Hoof, & Meger, 2018; Haarnoja et al., 2018)

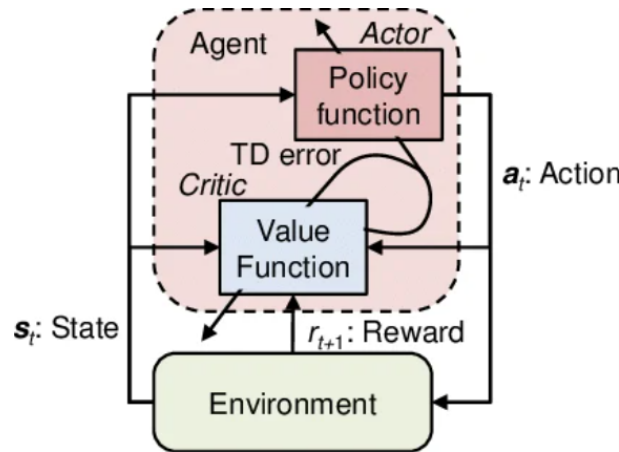


Figure 6: Actor-critic algorithm diagram illustrating how the value function approximation is used to guide policy updates. Retrieved from Fuji et al. (2018).

Both value-based and policy-based RL algorithms can be further classified by how they interact with the environment’s state-transition dynamics as model-based or model-free, and by how they use data for learning as on-policy or off-policy.

Model-based algorithms assume access to, or learn, an explicit model of the environment’s transition dynamics, enabling the agent to predict the consequences of its actions and plan ahead. In contrast, model-free algorithms do not rely on such a model; instead, they learn entirely from experience, without guarantees of recovering an accurate representation of the underlying dynamics. Examples of model-based algorithms include Deep Model Predictive Control (DMPC) (Mishra et al., 2021) and Probabilistic Inference for Learning Control (PILCO) (Deisenroth & Rasmussen, 2011), while examples of model-free algorithms include DQN (Mnih et al., 2015), SAC (Haarnoja et al., 2018), and A2C (Mnih et al., 2016).

A second distinction concerns how data is collected and used: on-policy algorithms update the policy using trajectories generated by the current version of that same policy, which improves stability but requires frequent data collection and is therefore sample-inefficient. Off-policy algorithms, on the other hand, can learn from data generated by older policies or even different behavior policies altogether, greatly improving sample efficiency and flexibility at the cost of increased challenges related to stability and distribution shift. On-policy algorithms include SARSA (Rummery & Niranjan, 1994), PPO (Schulman et al., 2017), and A2C (Mnih et al., 2016), while off-policy algorithms include DQN (Mnih et al., 2015), DDPG (Lillicrap et al., 2015), and SAC (Haarnoja et al., 2018).

RL Implementations for UAV Problems

Amendola et al. (2024) provides a comprehensive review of RL applications for multirotor UAVs, distinguishing between landings on stationary and mobile platforms. Most of these works report substantial performance improvements over baseline methods, particularly in scenarios involving landing on moving platforms whose motion exhibits spectrally correlated dynamics that the agent can learn. Actor-critic methods are commonly used to address the continuous action spaces characteristic of UAV guidance and control problems. Algorithm variants within this category are often selected based on preliminary experiments or empirical preference, rather than a systematic comparison. The algorithm choices across the literature evaluated are summarized in Table 2. It should be noted that some of the more recent actor-critic methods were introduced only in later years, with DDPG in 2015 (Lillicrap et al., 2015), PPO in 2017 (Schulman et al., 2017), TD3 in 2018 (Fujimoto, Hoof, & Meger, 2018), and SAC introduced in 2018 (Haarnoja et al., 2018), which partly explains their limited representation in earlier UAV landing studies.

Table 2: Overview of actor-critic algorithms used in prior work for continuous-action UAV landing problems.

Algo-rithm	Variants	Paper
DDPG	TD3, NAF, DDPG with prioritized replay	Saj et al., 2022, C. Wang et al., 2023, Xie et al., 2020, Peter et al., 2024, Wu et al., 2022, Kim et al., 2023
SAC	SACv2	Olivares et al., 2024
A2C	PPO, PPO-Clip, PPO-Penalty	Xiong et al., 2025, Olivares et al., 2024, Yikilmaz and Wagter, 2025,

In addition to differences in algorithmic choices, the problem formulation and overall RL model design vary substantially across studies in the literature. While some works incorporate visual cues directly into the observation space and develop end-to-end landing controllers (C. C. Chen et al., 2020; Saj et al., 2022; C. Wang et al., 2023), the most assume access to relative motion states between the UAV and the landing platform. These approaches directly use relative position, velocity, and attitude angles with respect to the landing plane for moving platforms (Abo Mosali, 2022; Amendola et al., 2023; Vankadari et al., 2018). In some studies, noise is added to the observations and subsequently filtered using a low-pass filter before being provided to the agent (Xie et al., 2020). Others incorporate explicit state estimation, typically using Kalman or extended Kalman filtering, to provide filtered relative state estimates to the learning agent.

Within the literature addressing wind disturbances, a clear distinction is made between approaches that include wind force measurements in the observation space and those that infer wind effects indirectly from the UAV state. Xiong et al. (2025) specifically studies the benefit of explicit wind sensing for quadrotor landing tasks and concludes that complex wind sensors provide limited performance gains, as comparable results are achieved when this observation is removed. However, Olivares et al. (2024) demonstrates that, in fixed-wing UAV platforms,

model-based control approaches with explicit wind sensing achieve significantly improved wind disturbance tracking performance. These findings indicate that the role of explicit wind sensing remains an open research question and may depend on the UAV type. Moreover, the practical acquisition of direct wind force measurements or accurate local wind speed estimates is challenging, which limits the feasibility of incorporating such observations with current sensing technology. Furthermore, the extensive body of work demonstrating robust performance under turbulent and stochastic wind disturbances without explicit wind-force representations suggests that direct wind measurements are not strictly necessary (Olivares et al., 2024; Peter et al., 2024).

Quite different approaches are noted from the side of action space design. Some studies simplify the task by training the policy to address only horizontal UAV movement, while vertical descent is handled by a predefined control rule. This reduces the dimensionality of the RL problem but also constrains the overall landing mission. For instance, the need to first stabilize horizontal position and then start the descent sequence, which is a widely applied landing sequence, may not work well under severe wind disturbances present at certain altitudes, as discussed by Xiong et al. (2025).

The majority of papers implement RL for guidance and path planning, with action spaces consisting of reference position, velocity, or acceleration commands. Xiong et al. (2025) addresses stationary UAV landing under stochastic wind disturbances. They develop a guidance controller that, in combination with PID or MPC tracking controllers, generates short-horizon trajectories updated at each action step based on the implicitly learned environment dynamics. Real-world tests under strong wind conditions demonstrate that the proposed planner successfully lands the quadrotor drone across all scenarios, outperforming time-optimal and conventional planners in terms of efficiency, smoothness, and control authority. Similarly, Yikilmaz and Wagter (2025) address landing of TU Delft’s variable-skew quad plane using an RL-based guidance controller that outputs optimal accelerations for an inner adaptive incremental nonlinear dynamic inversion controller. Their results indicate that the RL approach outperforms the PID benchmark, producing smoother, safer landings under wind disturbances.

There also exist papers that focus on online adaptation of PID gains for tracking tasks, demonstrating that dynamically updated gains reduce steady-state error, improve tracking accuracy, and allow quadcopters to follow trajectories more quickly and robustly under disturbances (Sönmez et al., 2025; Wu et al., 2022). While these alternative approaches perform well for moving targets and directional wind, this thesis focuses on highly localized wind gusts, where frequent path updates are often necessary to ensure successful landings, as has been shown for helicopter operations on ships (Hu et al., 2018).

The reward function in most studies is designed based on classical control objectives. Success is consistently defined as landing within a threshold around the target and touching down below a specified velocity, and rewards are structured accordingly. Collisions or deviations from a pre-defined region around the landing zone are typically penalized, while a large positive reward is granted upon landing. Assigning rewards only at success or failure creates a sparse function, leading to slow convergence and extensive exploration. To mitigate this, many works define a potential field around the landing zone and assign negative rewards proportional to the distance to the target at each transition. Some studies additionally impose a small negative reward at every step to encourage faster task completion. (Amendola et al., 2024)

In their research, Yikilmaz and Wagter (2025) investigate the impact of reward function design on landing performance for a hybrid RL guidance-based adaptive tracking controller. Multiple reward formulations are trained under different sea-state conditions that represent ship dynamics and evaluated only for heave motion. The results show that including either a velocity-dependent reward that densely encourages deceleration near the deck or a collision penalty applied at touchdown when the relative velocity is nonzero, reduces the touchdown velocity by up to 80%, significantly improving landing safety. Furthermore, increasing reward function complexity is observed to increase touchdown time.

The majority of the reviewed studies employ quadrotor platforms as representative UAV models, typically developed and tested in simulation environments such as Gazebo or PyBullet. While Gazebo provides a range of preliminary UAV models, PyBullet primarily offers the Crazyflie nanocopter, which is therefore frequently used in related work. Studies using Gazebo commonly rely on the Parrot AR model, although the Parrot ANAFI and the Iris quadrotor have also been reported (Amendola et al., 2024). These platforms are predominantly small quadcopters with limited range and endurance, which restricts their suitability for maritime missions. It should be noted that many of the RL-based landing studies reviewed in this paper are not specifically designed for maritime environments but instead focus on generic quadrotor applications such as urban navigation or package delivery, which explains the choice of platform.

Table 3 provides an overview of the papers investigating RL-based UAV landing approaches discussed in this section.

Table 3: Summary of RL approaches for UAV landing under disturbances

Paper	UAV Type	UAV Mission	Environment Model	Observation Space	Action Space
Saj et al. (2022)	Quadrotor	Autonomous landing	Moving platform (maritime), wind disturbances	Relative position and velocity	Roll and pitch commands
C. Wang et al. (2023)	Quadrotor	Autonomous landing	Moving platform (ground vehicle), wind disturbances	Visual feature vector	Velocity commands
Amendola et al. (2023)	Quadrotor	Autonomous landing	Moving platform (ground vehicle)	Relative position and velocity	Position offset commands
Abo Mosali (2022)	Quadrotor	Planar horizontal alignment for landing	Moving platform (ground vehicle)	Relative position	Roll and pitch commands

Continued on next page

Table 3 – continued from previous page

Paper	UAV Type	UAV Mission	Environment Model	Observation Space	Action Space
C. Chen et al. (2020)	Quadrotor	Marker-based autonomous take-off, forward flight, and landing	Stationary platform	Image-based grid representation	Discrete directional motion set
Xie et al. (2020)	Quadrotor	Planar horizontal alignment for landing	Moving platform (ground vehicle)	Positions and velocities of the UAV and the platform	Velocity commands
Xiong et al. (2025)	Quadrotor	Au-tonomous landing	Stationary platform, wind disturbances	Relative position and velocity, estimated wind-induced force	Acceleration commands
Olivares et al. (2024)	Fixed-wing UAV (without rudder)	Trajectory tracking	Wind disturbances	Attitude, airspeed, angular rates, aerodynamic angles, roll/pitch errors, and integrals, last sampled action	Aileron and elevator deflection
Peter et al. (2024)	Quadrotor	Au-tonomous landing	Moving platform (ground vehicle), wind disturbances	Attitude, velocity, relative position/velocity	Position offset commands
Yıkılmaz and Wagter (2025)	Variable Skew Quad Plane	Au-tonomous landing	Moving platform (maritime)	Relative position, velocity, last sampled action	Acceleration commands
Wu et al. (2022)	Quadrotor	Au-tonomous landing	Moving platform (ground vehicle)	Relative position	PID gains for the x-and y-velocity controllers
Kim et al. (2023)	Quadrotor Biplane Tailsitter	Au-tonomous landing	Moving platform (ground vehicle)	Positions and velocities of the UAV and the platform, UAV attitudes	Thrust and attitude command

2.4 Curriculum Learning Approaches

Curriculum learning (CL) is a machine learning training paradigm inspired by the progressive learning patterns observed in humans and animals. Empirical studies suggest that learning is more effective when education is structured as a curriculum, starting with simpler tasks and gradually progressing to more complex ones. CL leverages two key insights from this observation: first, the sequential organization of tasks by increasing difficulty, and second, the transferability of skills acquired in simpler tasks to support learning of more complex, target tasks. By mimicking this structured progression, CL aims to improve learning efficiency and performance in artificial agents.

X. Wang et al. (2021) evaluates CL in the context of broad machine learning applications and concludes that CL-based training strategies improve learning performance only for certain problems, depending on the nature of the problem and the way the curriculum is formulated. A CL-based training strategy is generally formulated around three key components: task generation, difficulty assessment, and task sequencing.

First, it is necessary to determine the set of intermediate or low-level tasks that can be derived from the target task. In single-task problems, this often involves parameterizing the environment to generate variations of scenarios in which a given task is performed. Second, these tasks or environment configurations must be organized by relative difficulty. Finally, the identified tasks are scheduled in a curriculum, specifying the order in which tasks are presented and when to transition to higher difficulty levels.

Figure 7 illustrates the general CL setup in its predefined form, where the training set is first processed by a difficulty measurer that evaluates the relative difficulty of each data example, and a training scheduler subsequently determines the order in which subsets of the data are presented throughout training. Here, both the difficulty measurer and the training scheduler are manually designed using human expertise. It is important to note, however, that CL approaches differ in how these components are designed: automatic curriculum learning (ACL) instead employs data-driven models or algorithms to determine difficulty and/or sequencing, enabling the curriculum to adapt dynamically to the agent’s learning progress (X. Wang et al., 2021).

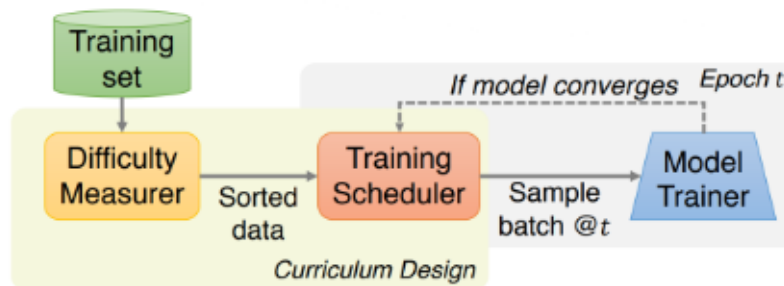


Figure 7: General framework of predefined curriculum learning, where a manually designed difficulty measurer ranks training samples, and a training scheduler determines the order in which they are presented to the learning algorithm. Retrieved from X. Wang et al. (2021).

Table 4 summarizes the main approaches to ACL. Each method differs in how difficulty assessment and task sequencing are determined, ranging from self-paced strategies that rely

on the agent’s own performance to teacher-based or optimization-driven approaches.

Table 4: Automatic Curriculum Learning Methods and Example Applications

Method	Description	Use Case / Example
Self-paced Learning (SPL)	Agent’s performance on the current tasks provides feedback to the difficulty measurer, which adjusts task weights accordingly.	Eimer et al. (2021) implements self-paced learning for simple RL tasks, including 2D ant navigation, ball-catching with a robotic arm, and a point-mass navigation system.
Transfer Teacher	Pretrained teacher model evaluates and updates task difficulty online based on the student’s learning progress.	Matiisen et al. (2017) implements a transfer teacher algorithm for navigation in a 3D Minecraft environment.
RL Teacher	A separate RL agent is trained to learn a teaching policy. This agent dynamically adjusts the task definitions, their sampling weights, or the training schedule.	Portelas et al. (2019) implements an RL teacher agent that adaptively selects environment parameters in continuous-control tasks (e.g., BipedalWalker) based on the student’s learning progress.

In RL, CL is formulated such that an agent is not trained directly on the target task, but instead leverages experience acquired from one or more source task MDPs. Knowledge gained from these source tasks can be transferred to facilitate learning in the target task in various forms, including samples, options, policies, models, or value functions. For instance, in policy transfer, the parameters of a policy learned in a source task are used to initialize the policy in the target task, thereby biasing exploration and action selection based on prior experience (Narvekar et al., 2020).

C. Wang et al. (2023) considers the problem of autonomous UAV landing on a moving unmanned ground vehicle (UGV) under wind interference and employs an ACL framework to train the landing policy. The landing task is broken down into three progressively more difficult stages with task parameters adaptively adjusted by a neural network-based Difficulty Discriminator according to the UAV’s online learning progress. These landing stages and task parameters are explained in Table 5. They compare their ACL-based method to a standard Twin Delayed Deep Deterministic Policy Gradient (TD3) RL baseline without curriculum, showing improved landing success and accuracy under varying UGV motions and wind disturbances.

Kim et al. (2023) implements an outer-loop RL controller for autonomous landing maneuvers of a quadrotor biplane tailsitter on a moving ground vehicle. The landing task is decomposed into two phases: approach and landing, with separate RL agents trained for each phase. CL is applied exclusively to the landing agent, while the approach agent is trained in a single session. The predefined CL consists of three sequential grades, detailed in Table 6, where each grade begins training from the fully converged policy of the preceding grade. The proposed RL-based method is benchmarked against a gradient-descent optimal trajectory planner. Results indicate that the RL trajectory agent achieves a significantly higher landing success rate (97% vs. 61%), with reduced computation time per step (mean 0.016 s vs. 1.603 s; standard deviation 0.002 s vs. 1.129 s), while completing the landing mission slightly slower on average (8.8 s vs.

Table 5: Curriculum stages for UAV landing on a moving UGV in ACL framework. Retrieved from C. Wang et al. (2023).

Stage	Task Parameters	Description
Basic	v_{UGV}	Maximum linear velocity of the UGV; introduces the UAV to basic landing logic on a moving target.
Default	$v_{UGV}, \omega_{UGV}, a_{UGV}$	Adds maximum angular velocity and acceleration of the UGV; trains UAV to handle rotational and accelerated motion of the target.
Frontier	$v_{UGV}, \omega_{UGV}, a_{UGV}, v_{wind}$	Introduces environmental disturbances (wind) on top of UGV motion; UAV learns robust landing under both target dynamics and external perturbations.

5.5 s) but with more consistent performance across trials (standard deviation 1.1 s vs. 0.1 s).

Table 6: Curriculum stages for UAV landing on a moving UGV. Retrieved from Kim et al. (2023).

Stage	Task Parameters	Description
Grade 1	v_{UGV}	The UAV learns to land on a UGV moving at a constant velocity sampled uniformly. This represents the simplest training scenario.
Grade 2	v_{UGV}	The UGV motion includes disturbances in longitudinal ($\dot{y}_G$) and vertical ($\dot{z}_G$) velocities. Disturbances are sampled uniformly and filtered through a low-pass filter to simulate uneven terrain.
Grade 3	v_{UGV}, w_t	The UAV is further incentivized to minimize landing time by increasing the reward weight w_t .

While these studies highlight that using CL can speed up convergence and improve policy performance, it is noted that poorly designed curricula can slow learning and destabilize the initial stages of DRL training (Narvekar et al., 2020).

2.5 Lift & Cruise Quadplane Modeling

A lift-and-cruise quadplane is a hybrid vertical takeoff and landing (VTOL) UAV that integrates characteristics of both multirotor and fixed-wing aircraft. The vehicle typically features four or more electric rotors in a multicopter configuration, providing vertical thrust for takeoff, landing, and hover, and a nose- or tail-mounted propeller delivering forward thrust during level flight. Fixed wings generate aerodynamic lift during cruise, reducing the load on the vertical lift motors, and may include control surfaces for maneuvering.

This design offers several advantages: it maintains stable, precise hover capability like a multicopter while achieving higher forward-flight efficiency and longer range than standard quadrotors due to wing-borne lift. Additionally, the configuration simplifies the transition between hover and cruise, as the wings and forward propulsion often operate independently of the vertical lift system.

Within this category, significant design variation exists. The airframe may incorporate a conventional tail, a V-tail, or a tailless configuration, depending on stability and control requirements. The vertical lift rotors may be mounted on the wings, fuselage, or booms, and can be arranged at different longitudinal or vertical offsets relative to the center of mass. In some designs, the lift rotors are mounted with a fixed pitch angle to improve aerodynamic interaction or transition performance, while in others they remain strictly vertical. These architectural choices influence the vehicle’s mass distribution, inertia properties, aerodynamic interference, and control authority across flight regimes. Representative lift-and-cruise configurations are illustrated in Figure 8.

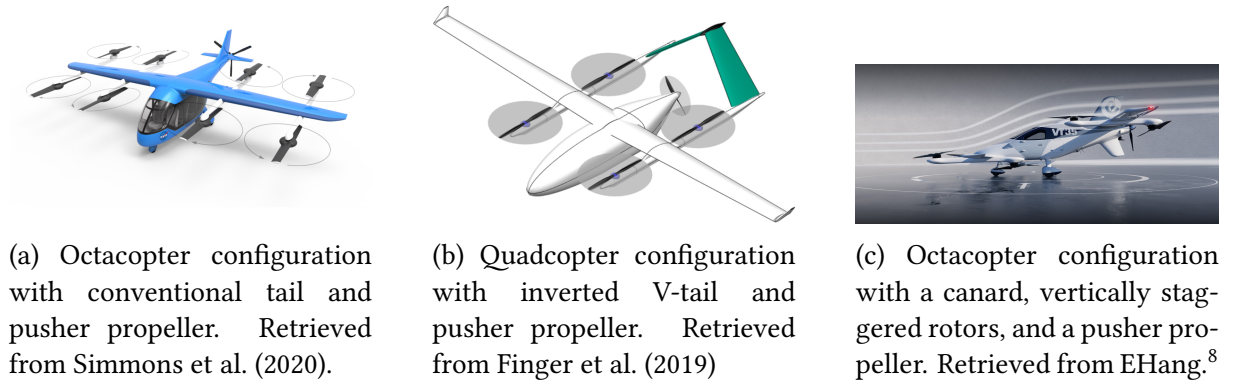


Figure 8: Representative lift-and-cruise UAV configurations with varying rotor placement and airframe layouts.

In most conventional configurations, during hover, lift, and maneuvering, authority is primarily provided by the vertical thrust of the multicopter rotors, while aerodynamic surfaces are largely ineffective due to low airspeed. In forward cruise, the fixed-wing aerodynamics dominate, with lift generated primarily by the wings and control moments achieved through conventional control surfaces (ailerons, elevator, rudder), while the vertical lift motors may be throttled down or inactive. Between these regimes, the vehicle passes through a transition phase, in which control authority is shared between the rotors and the aerodynamic surfaces, and their relative effectiveness varies with airspeed. (Pfeifle & Fichter, 2022)

Due to their fundamentally different flight envelopes, aerodynamic models developed for conventional fixed-wing aircraft are generally not suitable for accurately describing the aerodynamics of hybrid VTOL vehicles (Lustosa, 2017). A common approach for modeling aerodynamic forces and moments for conventional fixed-wing aircraft is based on the Buckingham- Π theorem, which groups the governing physical parameters into dimensionless quantities. This formulation expresses aerodynamic forces and moments in terms of dynamic pressure, characteristic geometric dimensions, and aerodynamic angles — defined relative to the aerodynamic (wind) reference frame — as given in Equation 14 and 15.

$$\mathbf{F}_a = \frac{1}{2} \rho V_\infty^2 S \mathbf{C}_F(\alpha, \beta, M, Re) \quad (14)$$

⁸<https://www.ehang.com/vt35/>

$$\mathbf{M}_a = \frac{1}{2} \rho V_\infty^2 S \mathbf{D} \mathbf{C}_M(\alpha, \beta, M, Re) \quad (15)$$

The aerodynamic force vector $\mathbf{F}_a \in \mathbb{R}^3$ and moment vector $\mathbf{M}_a \in \mathbb{R}^3$ acting on the wing are determined by the dynamic pressure $\frac{1}{2} \rho V_\infty^2$, where ρ denotes the air density and V_∞ represents the freestream airspeed. The wing planform area S scales the resulting aerodynamic forces and moments. For aerodynamic moments, the diagonal matrix $\mathbf{D}$ accounts for the characteristic moment arms of the aircraft geometry, where the wingspan b scales rolling and yawing moments, and the mean aerodynamic chord c scales pitching moments.

The force coefficient vector $\mathbf{C}_F \in \mathbb{R}^3$ and moment coefficient vector $\mathbf{C}_M \in \mathbb{R}^3$ are commonly modeled as functions of the angle of attack α and the sideslip angle β . The angle of attack α is the angle between the freestream velocity and the aircraft's longitudinal axis, formulated in Equation 16. The sideslip angle β represents the lateral deviation of the airflow from the longitudinal axis, formulated in Equation 17.

$$\alpha = \tan^{-1} \left(\frac{V_{\infty,3}}{V_{\infty,1}} \right) \quad (16)$$

$$\beta = \sin^{-1} \left(\frac{V_{\infty,2}}{\|\mathbf{V}_\infty\|} \right) \quad (17)$$

While used for conventional platforms, the Buckingham-II theorem formulation becomes problematic when applied to hybrid VTOL vehicles, as their flight envelope includes hover and vertical descent. The root cause is that this formulation relies on the aerodynamic reference frame, in which forces and moments are expressed relative to the freestream velocity vector. This reference frame is well-defined only when a meaningful freestream velocity exists. During hover, the freestream velocity approaches zero, causing the trigonometric relations used to compute aerodynamic angles to become singular. As a result, small measurement errors or numerical noise can be significantly amplified when dividing by near-zero values, leading to highly erratic estimates of the aerodynamic angles.

Furthermore, during vertical descent or ascent, the vertical velocity component dominates while the horizontal airspeed approaches zero, causing the angle of attack to approach $\pm 90^\circ$. This operating regime lies well outside the range over which conventional aerodynamic coefficient parameterizations, typically derived from small-angle forward-flight data, remain valid.

Consequently, the Buckingham-II formulation can only be applied to hybrid VTOL vehicles if the aerodynamic angles supplied to it remain well defined throughout the entire flight envelope, including hover and vertical descent, and if suitable safeguards (e.g., velocity thresholding, smoothing, or regularization of the denominator) are applied to suppress noise-driven peaks and prevent singularities as the freestream velocity approaches zero.

To address these challenges, an alternative framework for modeling hybrid VTOL hover and transition aerodynamics is presented in Lustosa (2017) through the so-called Φ -theory. This formulation introduces a singularity-free representation of aerodynamic forces and moments that does not rely on classical aerodynamic angles such as angle of attack (α) or sideslip angle

(β). Instead, all aerodynamic effects are expressed directly in the body-fixed frame as functions of the vehicle's linear and angular velocity components, as shown in Equation 18.

$$\mathbf{w}_a = \begin{bmatrix} \mathbf{F}_a \\ \mathbf{M}_a \end{bmatrix} = -\frac{1}{2}\rho S\eta \mathbf{C}\Phi(\boldsymbol{\eta})\mathbf{C}\boldsymbol{\eta}, \quad (18)$$

where $\mathbf{w}_a \in \mathbb{R}^6$ denotes the aerodynamic wrench about the center of mass, and $\boldsymbol{\eta} \in \mathbb{R}^6$ combines the vehicle's linear and angular velocity components relative to the freestream. The scalar $\eta \in \mathbb{R}_+$ represents a generalized aerodynamic norm. The air density ρ , wing planform area S , and the screw reference matrix $\mathbf{C} \in \mathbb{R}^{6 \times 6}$ are included to ensure proper dimensional scaling. The core parameters of the model, aerodynamic coefficients, are contained in the Φ -matrix, which defines the mapping $\Phi : \mathbb{R}^6 \rightarrow \mathbb{R}^{6 \times 6}$.

The velocity vector $\boldsymbol{\eta}$ consists of the freestream linear and angular velocity components, as given in Equation 19. From there, the generalized aerodynamic norm is defined as Equation 20, which remains well-defined even when the linear velocity approaches zero during hover. The tuning parameter κ weights the relative contribution of linear and angular velocity components.

$$\boldsymbol{\eta} = [u_\infty \quad v_\infty \quad w_\infty \quad p_\infty \quad q_\infty \quad r_\infty]^\top \quad (19)$$

$$\eta = \sqrt{\|\boldsymbol{\eta}_v\|^2 + \kappa\|\boldsymbol{\eta}_\omega\|^2}, \quad \kappa > 0, \quad (20)$$

The screw reference matrix $\mathbf{C}$, which incorporates geometric scaling, is defined in Equation 21. The matrix $\mathbf{D}$ is defined analogously to the scaling matrix used in the classical moment formulation and introduces geometric scaling via the vehicle's characteristic dimensions.

$$\mathbf{C} = \begin{bmatrix} \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{D} \end{bmatrix}, \quad (21)$$

The Φ -matrix is decomposed into four 3×3 sub-matrices:

$$\Phi = \begin{bmatrix} \Phi_{fv} & \Phi_{f\omega} \\ \Phi_{mv} & \Phi_{m\omega} \end{bmatrix}, \quad (22)$$

where, each block represents a distinct physical contribution:

- Φ_{fv} : forces induced by linear velocity,
- $\Phi_{f\omega}$: force contributions due to angular velocity,
- Φ_{mv} : moments induced by linear velocity,
- $\Phi_{m\omega}$: moments induced by angular velocity.

Identification of the aerodynamic coefficients in either of the previously discussed formulations can be performed by experimentally measuring the aerodynamic forces and moments across the flight envelope of the hybrid VTOL vehicle through wind tunnel testing.

The resulting measurements can then be used either to construct lookup tables for control design and simulation or to fit analytical expressions, yielding continuous closed-form aerodynamic models that are generally more suitable for control synthesis and numerical implementation (Z. Ma et al., 2022; Mathur & Atkins, 2023). As an alternative to experimental testing, the same aerodynamic data may instead be obtained through computational fluid dynamics (CFD) simulations, a cost-effective approach that substantially reduces the logistical complexity associated with extensive wind tunnel campaigns (Quan et al., 2025).

Beside the aerodynamic forces and moments acting on lift-and-cruise vehicles, thrust generation and its associated moment contributions are critical to the overall functionality and controllability of VTOL platforms. Mathur and Atkins (2023) have investigated propeller thrust behavior under both static and dynamic operating conditions using wind-tunnel and dynamometer testing. Their static-test results support classical propeller theory, where thrust is approximately proportional to the square of propeller rotational speed. This confirms the commonly used relation between rotor speed and generated thrust in many UAV dynamic models.

Under dynamic conditions, where a free-stream velocity is present, propeller performance is shown to depend strongly on both the inflow angle and airspeed magnitude. The inflow angle, denoted as propeller angle of attack α_p , represents the incidence angle at which the propeller disk encounters the incoming airflow. At low inflow angles (approximately 0°), where the airflow approaches the propeller nearly head-on, thrust decreases as airspeed increases. This reduction is attributed to increased aerodynamic drag on the rotating propeller disk and disturbed inflow conditions that reduce thrust-producing efficiency.

In contrast, at larger propeller angles of attack, where the incoming airflow approaches the propeller disk obliquely or laterally, the measured thrust increases with airspeed. This demonstrates that propeller effectiveness is highly sensitive to both vehicle orientation and flight speed, particularly during transition phases where rotor disks may experience strongly varying local flow directions.

To capture these effects, the authors fit cubic thrust models for discrete combinations of airspeed V_a and propeller angle of attack α_p . The resulting lookup-table framework provides an experimentally grounded thrust model suitable for simulation and control applications.

2.6 Estimating UAV States

State estimation is the process of computing the most accurate representation of a system's true state by combining partial and noisy measurements with a mathematical model describing the system's underlying dynamics. In the maritime landing context, reliable state estimation is particularly challenging due to the highly dynamic and adversarial offshore environment, which complicates accurate data acquisition for all operating systems (Xie et al., 2020).

Helicopter-Based State Estimation

Research on shipboard state estimation has a longer history in the helicopter domain than for UAVs, as manned rotorcraft have operated in maritime environments for decades. Both problem domains assume a cooperative ship that estimates its own state using an onboard navigation suite and broadcasts this information to the landing vehicle via a communication link.

Helicopters typically employ a layered sensor architecture for shipboard landings. Global Navigation Satellite System (GNSS) receivers provide global position and velocity for approach guidance, while Inertial Measurement Units (IMUs) supply reliable attitude and motion estimates, particularly in conditions of GNSS degradation near metallic ship structures (Yin et al., 2023). Radar altimeters and optical or infrared sensors contribute relative altitude and deck geometry information, with LiDAR or laser rangefinders enabling more accurate estimation of deck pose. These complementary sensing modalities are commonly fused to maintain robust estimates of attitude, velocity, and altitude throughout the landing sequence.

UAV State Estimation

UAVs differ from helicopters in several respects that significantly constrain their state estimation capabilities. Limitations on payload, power, and cost restrict onboard sensors to lightweight, low-cost alternatives that are individually noisier and less reliable (X. Zhou et al., 2025). Furthermore, full sensor redundancy cannot be guaranteed. As a result, UAV state estimation places greater emphasis on software-driven approaches, particularly vision-based estimation and probabilistic filtering (e.g., EKF and UKF), to compensate for reduced hardware fidelity and increased sensitivity to disturbances.

The majority of studies addressing the autonomous UAV landing problem assume that a core set of relative states is observable: the UAV’s relative position, velocity, and attitude with respect to the target landing site, as summarized in Table 3. Vision-based sensing is the most commonly adopted approach for obtaining these quantities, with fiducial markers such as AprilTags frequently employed as landing targets (C. C. Chen et al., 2020; Saj et al., 2022). In these systems, computer vision algorithms estimate relative pose by detecting marker corners and solving a Perspective- n -Point (PnP) problem using the marker’s known geometry. Beyond vision, sensors commonly assumed or explicitly used in landing-related studies include IMUs, GNSS receivers, dedicated altitude sensors (e.g., barometers or range finders), and, less frequently, airspeed sensors. Table 7 presents the UAV sensors used within the maritime operation scope, together with their associated uncertainties based on the performance specifications of off-the-shelf hardware. The use of LiDAR, laser rangefinders, and advanced sensor fusion techniques can further reduce these uncertainty margins.

Table 7: Measurement accuracies of commercially available sensors relevant to UAV state estimation and wind sensing.

Component	Measurement	Accuracy ($\pm$)
GNSS	Absolute position	1.5 m ⁹
RTK	Relative position	0.01 m + 1ppm ¹⁰
Compass	Heading angle	1° ¹¹
UWB module	Relative distance	10 cm ¹²
IMU	Angular rates	0.5% ¹³
	Accelerations	0.5% ¹³
Barometer	Altitude	0.1 m ¹⁴

Continued on next page

Table 7 – continued from previous page

Component	Measurement	Accuracy ($\pm$)
Rotary encoder	Angle of attack / sideslip	0.1° ¹⁵
Pitot tube	Airspeed	1% (0.3 m s^{-1}) ¹⁶

Sensor Uncertainty in RL-Based Landing

When developing RL-based landing controllers, accounting for sensor imperfections is essential for bridging the sim-to-real gap. Studies address this through two primary strategies: domain randomization, in which sensor noise, measurement delays, and observation uncertainty are explicitly modeled during training; and end-to-end simulation, in which visual perception is performed in real time within environments such as Gazebo by rendering landing markers directly in the simulated scene. Table 8 summarizes representative studies that explicitly consider sensor limitations in the context of RL-based UAV landing.

Table 8: Overview of sensor modeling in related UAV landing studies using RL with domain randomization to model sensor noise and uncertainties.

Paper	UAV Sensor Suite	Noise and Uncertainty Representation
C. Wang et al. (2023)	Forward-facing monocular camera (captures the gyro-stabilized horizon reference bar mounted on the ship)	Distance-dependent noise and random time delays are added to the estimated relative states.
Saj et al. (2022)	Downward-facing RGB camera (captures the UGV and its landing marker) and IMU	No explicit observation-space domain randomization is applied; the policy is trained on deterministic visual cues rendered in Gazebo.
C. C. Chen et al. (2020)	Monocular onboard camera and IMU	No explicit observation-space domain randomization is applied; the policy is trained on deterministic visual cues rendered in Gazebo.

Continued on next page

⁹https://cdn.sparkfun.com/assets/d/e/2/8/b/ZED-F9P_datasheet.pdf

¹⁰https://content.u-blox.com/sites/default/files/ZED-F9P_ProductSummary_UBX-17005151.pdf

¹¹https://www.saelig.com/supplier/isentek/ist8301c_datasheet_v10_20130712.pdf

¹²https://ftp.nooploop.com/software/products/uwb/doc/LinkTrack_Datasheet_V2.1_en.pdf

¹³<https://invensense.tdk.com/en-us/products/6-axis/icm-45686>

¹⁴<https://nl.farnell.com/seeed-studio/101020068/barometer-sensor-arduino-raspberry/dp/4007687>

¹⁵<https://witmotion-sensor.com/products/jy-me01-high-precision-18-bit-resolution-absolute-rotation-angle-sensor>

¹⁶<https://holybro.com/products/high-precision-dronecan-air-speed-sensor-dlvr?variant=44438283714749>

Table 8 – continued from previous page

Paper	UAV Sensor Suite	Noise and Uncertainty Representation
Xie et al. (2020)	Monocular onboard camera, IMU, altitude sensor	Gaussian noise is added to the relative position observations with variable standard deviation during training ($\mu = 0, \sigma = 1-4$).

2.7 Ship Dynamics

Different ship dynamics modeling approaches have been adopted in the literature for autonomous UAV ship landing applications using both classical and reinforcement learning-based controllers. In most formulations, the vessel is modeled as a rigid body with a body-fixed reference frame located at the center of the ship, as illustrated in Figure 1. The rotational motion of the ship is described using Euler angles defined analogously to those of a UAV: rotations about the x -, y -, and z -axes correspond to roll, pitch, and yaw, respectively. Translational motion along these axes is referred to as surge (longitudinal), sway (lateral), and heave (vertical).

Most ship landing models define the landing target relative to a reference point fixed to the vessel, typically the center of the landing platform. The corresponding target position is represented in inertial coordinates as (x_t, y_t, z_t) . When rotational vessel dynamics are included, the landing platform is additionally assigned an orientation to represent the instantaneous attitude of the ship during landing operations.

In seakeeping analysis, vessel motion responses are commonly characterized using Response Amplitude Operators (RAOs), which describe the amplitude of each degree-of-freedom response as a function of wave frequency. Heave, pitch, and roll are associated with gravitational restoring forces and can resonate with incoming wave energy, while surge, sway, and yaw lack restoring forces and therefore typically exhibit lower response amplitudes under comparable sea states (Aird, 2019). Both numerical and experimental RAO studies show that heave responses dominate over surge and sway in irregular waves, as heave is directly excited by the vertical wave component, whereas horizontal motions arise indirectly through hydrodynamic forcing and inertial coupling with rotational motion (Ibinabo & Tamunodukobipi, 2019).

Operational considerations further reduce horizontal motions in practice. To limit roll-induced safety risks, ships commonly adopt headings with waves approaching from the bow or at moderate oblique angles rather than in beam seas, which reduces lateral wave excitation and associated surge and sway. Heave motion, driven by the vertical wave component, remains significant for waves from all directions and therefore constitutes the dominant translational response.

Yıkılmaz and Wagter (2025) model ship motion by superimposing multiple sinusoidal wave-induced oscillations, augmented with random walk components, to generate translational motion along all three axes. The amplitudes, phases, and frequencies of the sinusoidal

components are drawn randomly from specified boundary ranges. Similarly, Saj et al. (2022) represents the ship as a single moving point in 3D space, oscillating vertically with a user-specified sinusoid to model wave-induced heave, while horizontal motion follows a user-defined position, heading, and velocity.

A number of researchers refer to the Joint North Sea Wave Project (JONSWAP) spectrum (Hasselmann et al., 1973) to model wave-induced ship motion (Ali et al., 2024; Amendola et al., 2023; Choi et al., 2025). The JONSWAP spectrum, $S(\omega)$, describes the distribution of wave energy over angular frequencies ω :

$$S(\omega) = \alpha \frac{g^2}{\omega^5} \exp \left[-\frac{5}{4} \left(\frac{\omega_p}{\omega} \right)^4 \right] \gamma^{\exp \left[-\frac{(\omega - \omega_p)^2}{2\sigma^2 \omega_p^2} \right]} \quad (23)$$

where $\omega_p = 2\pi/T_p$ denotes the peak angular frequency corresponding to the dominant wave period T_p , γ is the peak enhancement factor (commonly taken as $\gamma = 3.3$ H. Williams, 2021), σ is a piecewise spectral width parameter equal to 0.07 for $\omega \leq \omega_p$ and 0.09 otherwise, g is the gravitational acceleration, and α is a scaling factor related to the significant wave height H_s (Hasselmann et al., 1973). The significant wave height and dominant wave period are defined according to the World Meteorological Organization (WMO) sea-state classification, providing a standardized parameterization of sea conditions, as can be seen in Table 9.

Table 9: WMO Sea State Classification with Significant Wave Height and Dominant Wave Periods^{17,18}

Code	Description	Height H_s (m)	Period T_p (s)
1	Calm	0–0.1	1–3
2	Smooth	0.1–0.5	1–5
3	Slight	0.5–1.25	6–7
4	Moderate	1.25–2.5	8–9
5	Rough	2.5–4	10–11
6	Very rough	4–6	12–13
7	High	6–9	14–15

To generate a time-domain motion signal, the continuous spectrum is discretized into N frequency components, ω_k , with corresponding amplitudes

$$A_k = \sqrt{2S(\omega_k)\Delta\omega_k}, \quad k = 1, \dots, N, \quad (24)$$

where $\Delta\omega_k$ is the frequency spacing between adjacent components. The resulting heave displacement, $\eta(t)$, is then obtained by superposing all sinusoidal components:

$$\eta(t) = \sum_{k=1}^N A_k \cos(\omega_k t + \phi_k), \quad (25)$$

¹⁷<https://www.nodc.noaa.gov/gtspp/document/codetbls/wmocodes/table3700.html>

¹⁸https://www.jodc.go.jp/data_format/weather-code.html

where ϕ_k is the phase angle of the k th wave component. This frequency-domain to time-domain mapping is illustrated in Figure 9. To ensure RL policy generalization, new waveforms with randomized phases $\phi_k \in [0, 2\pi]$ are generated for each episode, preventing overfitting while preserving consistent sea-state characteristics. Figure 10 illustrates the corresponding time-domain heave motion trajectories for different sea states reported in the referenced study.

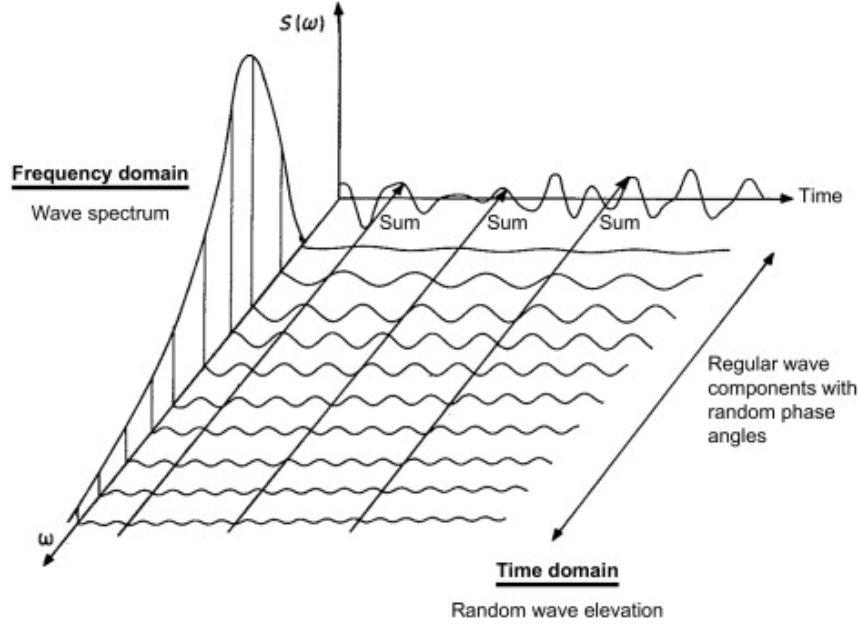


Figure 9: Frequency domain to time domain mapping of wave characteristics. Retrieved from Bai and Bai (2005).

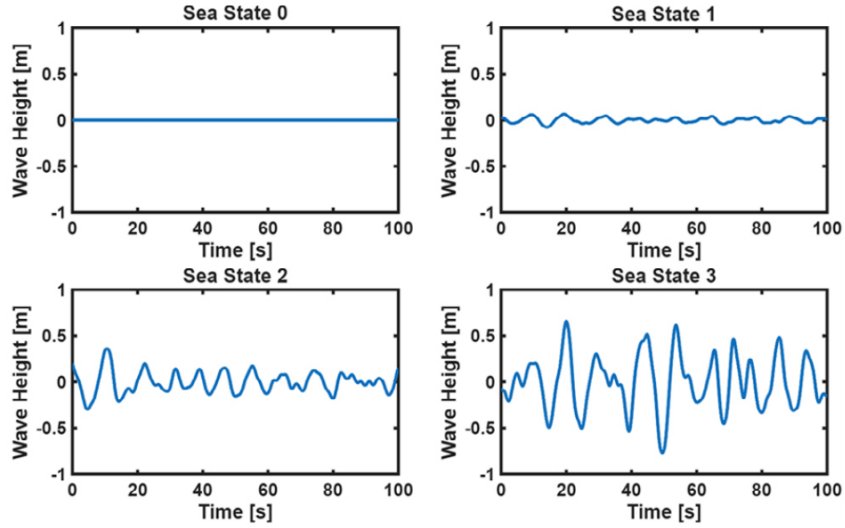


Figure 10: Time-domain heave motion trajectories generated from the JONSWAP wave spectrum for different sea-state conditions. Retrieved from Choi et al. (2025).

2.8 Wind and Wake Modeling

Operating in windy conditions is inherently challenging, and the presence of a ship superstructure further complicates the aerodynamic environment by generating unsteady, spatially heterogeneous airflow. According to Mewhinney (1998) and Aragon and Long (2005), turbulence generated by the ship air-wake is identified as the primary cause of the 120 ship-helicopter incidents documented in the U.S. Navy between the 1960s and 1998. Studies of helicopter operations on offshore oil and gas platforms further demonstrate that turbulent winds substantially increase pilot workload, with the air-wake at the landing area constituting the principal safety hazard (Setiawan et al., 2025). For lift-and-cruise UAVs, the presence of lifting surfaces introduces additional sensitivity to wind-induced forces and moments. Consequently, accurate modeling of ship-wake-induced wind disturbances is essential for generating realistically challenging simulation environments in which safe landing strategies can be developed and evaluated using RL.

Offshore Wind Characteristics

The ambient atmospheric wind environment in maritime operations is governed by the marine atmospheric boundary layer. In offshore regions away from structural influences, wind conditions develop over a relatively homogeneous sea surface characterized by low surface roughness. This differs from onshore boundary-layer flows, where terrain-induced roughness elements significantly alter turbulence structure, wind shear, and near-surface velocity profiles.

For UAV landing operations, the relevant range spans approximately Beaufort force 3 to 7, corresponding to mean wind speeds of roughly 3 to 14 m/s at the standard reference height (10 m above sea level), under which wave-induced ship motion remains operationally significant and wind loading on the vehicle is non-negligible, as shown in Table 10. This range is consistent with observational data from northern Arctic marine regions, where reported mean wind speeds lie between 7 and 15 m/s at the same reference height, alongside turbulence intensities of approximately 8%, decreasing with height and with increasing wind speed (Söker et al., 2000). This trend reflects the diminishing influence of surface-generated turbulence at elevations relevant to UAV approach and landing.

Turbulence in the offshore boundary layer is further characterized by its length scales, which govern the spatial coherence of wind fluctuations and therefore determine whether a vehicle experiences a gust as a brief impulse or a sustained lateral displacement. In the streamwise direction, integral length scales over open water are typically on the order of tens to hundreds of meters, considerably larger than over land due to the reduced surface roughness (Söker et al., 2000). These large-scale fluctuations are of particular relevance for approach trajectories, as they produce correlated wind variations over distances comparable to the vehicle's path length. Lateral and vertical length scales are generally smaller, meaning that crosswind and vertical gusts decorrelate more rapidly in space, producing localized but intense perturbations rather than sustained drift.

In the vicinity of the ship, these ambient boundary-layer conditions interact with the wake structures described above, and the resulting disturbance field reflects both contributions.

¹⁹<https://www.weather.gov/mfl/beaufort>

Table 10: Beaufort wind scale with wind speed and wave characteristics.¹⁹ Conditions in Force 3–8 are identified as relevant for maritime landing operations.(Söker et al., 2000)

Force	Speed (m/s)	Speed (kn)	Wave height (m)	Specifications for use at sea
0	0–0.5	0–1	0	Calm, sea like a mirror
1	0.5–1.5	1–3	0–0.1	Ripples without foam crests
2	2–3.5	4–6	0.1–0.3	Small wavelets, no breaking crests
3	4–6	7–10	0.5–1	Large wavelets, occasional white horses
4	6.5–8.5	11–16	1–1.5	Small waves, frequent white horses
5	8.5–10.5	17–21	2–3	Moderate waves, many white horses
6	11–13	22–27	3–4	Large waves, extensive foam crests
7	13.5–17	28–33	4–5.5	Sea heaps up, foam blown in streaks
8	17.5–20	34–40	5.5–7.5	High waves, breaking crests, spindrift
9	20–24	41–47	7–10	High waves, dense foam, reduced visibility
10	24–28	48–55	9–12.5	Very high waves, sea surface white
11	28–32	56–63	11.5–16	Exceptionally high waves, poor visibility
12	>32	>64	>14	Hurricane conditions, sea fully white

Air-Wake Structure and Local Flow Regions

In fluid dynamics, a wake is the disturbed region of flow downstream of a body moving through a fluid, characterized by a velocity deficit relative to the freestream, elevated turbulence intensity, and coherent vortical structures that interact, break down, and eventually dissipate. During shipboard operations, the combination of ambient wind and the ship’s forward motion produces a relative wind that flows over and around the superstructure. As this flow encounters the ship’s hangar and other bluff-body features, it undergoes separation at sharp aft edges, forming a shear layer between the high-speed freestream above and a region of recirculating flow in the hangar lee. Simultaneously, airflow traveling along the hull sides is deflected upward over the deck edge and feeds into this recirculation zone, rising along the hangar face (Wall et al., 2022). A representative schematic of this air-wake formation is shown in Figure 11.

Critically, the resulting flow field is not spatially uniform but is composed of distinct local regions with qualitatively different flow characteristics, each presenting different challenges

for a landing vehicle. These include:

- **Downwash regions**, where flow is directed vertically downward due to deflection over the hangar roof or other elevated structures.
- **Upwash regions**, where flow rises along the hangar face or deck edge, potentially inducing uncommanded climb and complicating descent control.
- **Lateral or crosswind regions**, where horizontal flow components from the sides of the superstructure introduce lateral drift and yaw disturbances.
- **Velocity deficit and reverse-flow zones**, where streamwise velocity can drop to below 50% of the reference wind speed or even become negative in strongly recirculating regions behind the hangar.
- **High-turbulence shear layers**, located at the boundaries between the freestream and recirculating zones, where rapid spatial gradients in velocity produce intermittent and unpredictable gusts.

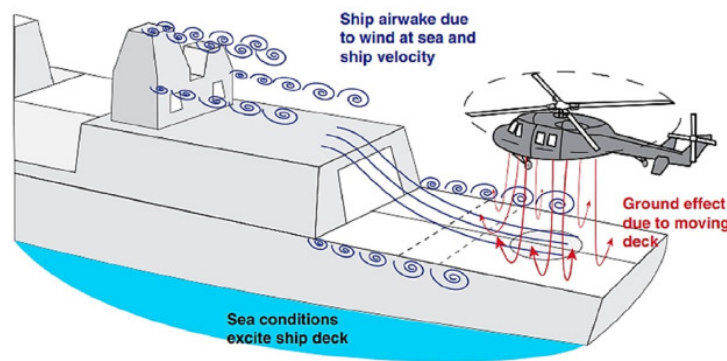


Figure 11: Schematic of the aerodynamic interaction between ship superstructure and helicopter in the air-wake region. Retrieved from Vakilabadi et al. (2023).

Experimental and Numerical Characterization

The spatial structure of the ship air-wake has been the subject of considerable experimental and numerical investigation. Setiawan et al. (2025) study the air-wake above a ship landing deck under combined wind-wave conditions, quantifying both steady flow and turbulence characteristics. Their measurements reveal that the presence of obstacles, including the ship's radar mast and hangar, generates substantial flow separation and recirculation, producing regions of velocity deficit and reverse flow. The streamwise velocity above the deck can fall below 50% of the reference wind speed, and recirculating regions can extend nearly two hangar heights downstream—approximately 18 m for the NATO-GD frigate—as illustrated in Figure 12. Turbulence kinetic energy (TKE), computed from streamwise and vertical velocity fluctuations, is elevated near the radar and hangar edges, while the interior of the recirculation zone behind the hangar exhibits comparatively lower turbulence intensities. Comparisons with baseline turbulent boundary layer flows over a flat plate indicate that the spatial characteristics of the mean velocity and TKE are largely similar within the critical

landing region, with deviations of less than 5% in mean flow and 2% in TKE. These results confirm that ship air-wake disturbances are spatially structured, with coherent regions of reduced velocity, reverse flow, and variable turbulence intensity distributed across the landing approach corridor.

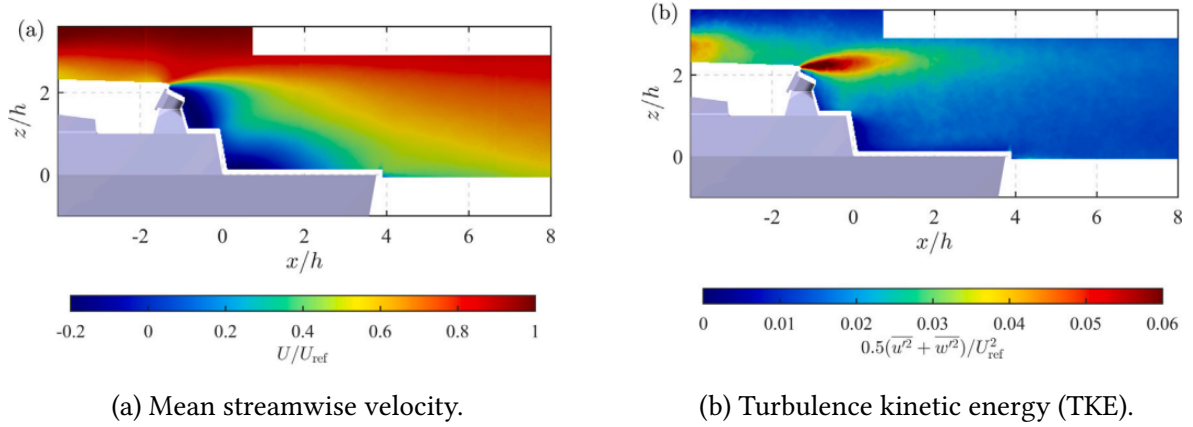


Figure 12: Contours of (a) mean streamwise velocity and (b) turbulence kinetic energy (TKE) across the ship centreline ($y = 0$), computed from wind-wave experiments. Retrieved from Setiawan et al. (2025).

The sensitivity of the wake structure to superstructure geometry is further highlighted by Montes Gomez and Sumner (2022), who experimentally investigate the mean wake of a three-dimensional surface-mounted rectangular flat plate in a low-speed wind tunnel while varying the aspect ratio ($AR = 0.5$ to 3), incidence angle ($\alpha = 0^\circ$ to 90°), and relative boundary layer thickness. Their results demonstrate that as the plate becomes less slender, the recirculation zone contracts, while both the incidence angle and aspect ratio strongly govern the formation and evolution of vortical structures. These findings are consistent with broader evidence that the shape of ship superstructures exerts a dominant influence on the resulting air-wake—an effect that often exceeds the contribution of incoming wind turbulence (Setiawan et al., 2022; Wall et al., 2022).

Taken together, these studies highlight that the ship air-wake constitutes a spatially heterogeneous disturbance field rather than a uniform reduction in wind speed, with qualitatively distinct flow regions encountered sequentially along a landing approach. This spatial complexity motivates not only a structured wind disturbance model in simulation, but also the inclusion of wind sensing in the guidance loop, enabling a controller to perceive local flow conditions, assess their safety implications, and adjust the approach trajectory accordingly.

Wind Representation in RL-Based Landing Studies

RL approaches to UAV landing can be broadly divided into two categories with respect to how wind disturbances are treated: those targeting inner-loop attitude and tracking control, and those developing outer-loop guidance policies. This distinction has important implications for how wind is modeled and what conclusions can be drawn. It is worth noting that the majority of RL-based UAV landing studies consider quadrotor platforms, for which aerodynamic sensitivity to wind is comparatively low — there are no fixed lifting surfaces to

introduce angle-of-attack sensitivity or additional moment arms. As a result, wind is often treated as a secondary concern, with simplified disturbance representations that reflect the platform’s relative robustness rather than the complexity of the actual flow environment.

Studies focused on inner-loop control commonly incorporate wind disturbances during both training and inference, typically represented as uniform wind fields with superimposed stochastic turbulence (Olivares et al., 2024; Saj et al., 2022; Wu et al., 2022). In this setting, the primary objective is to demonstrate that an RL-based tracking controller achieves superior disturbance rejection compared with a classical PID baseline. The wind representation in these cases is not intended to capture the spatial structure of a real ship’s air wake; it serves as a generic robustness test for the low-level controller.

Guidance-level RL studies, which are more directly relevant to the landing problem considered in this study, generally treat wind in a similarly simplified manner. Wind disturbances are most commonly represented as uniform ambient flow with added turbulence, applied to demonstrate that a trained guidance policy remains functional under adverse wind conditions rather than to exploit knowledge of the spatially structured wake field in trajectory generation. The localized flow regions described in the preceding subsections are rarely reflected in the simulation environments used for policy training.

This represents a meaningful gap in the literature. Although ship wake-induced wind disturbances are recognized as a primary source of flight instability during maritime landings (Aragon & Long, 2005; Mewhinney, 1998), RL studies addressing the ship landing problem have largely neglected this phenomenon. Many studies do not explicitly model wind disturbances as part of the overall system dynamics, instead focusing primarily on the ship’s motion as a moving platform (Ali et al., 2024; Frölrich, 2015; Yıkılmaz & Wagter, 2025).

A notable exception is Saj et al. (2022), who incorporate wind disturbances in Gazebo through randomized scenarios comprising constant but localized winds. By observing that headwinds predominantly affect forward drift while crosswinds influence lateral motion, they train pitch and roll controllers to represent both wind components during learning. The study considers a quadrotor platform, for which horizontal wind components constitute the dominant aerodynamic disturbance — a reasonable justification for restricting the disturbance field to the horizontal plane. For a lift-and-cruise UAV, however, the presence of fixed lifting surfaces introduces additional sensitivity to vertical flow components, making downwash, upwash, and shear-layer-induced vertical gusts aerodynamically significant disturbances that cannot be neglected in the same way.

Studies addressing landing on generic moving platforms have in some cases adopted more elaborate wind representations. Paris et al. (2020) conduct experimental trials in which turbulent wind is generated from a single side using leaf blowers, modeling the resulting drag as proportional to squared relative velocity with stochastic fluctuations spanning $\pm 2 \pm 2 \pm 2$ standard deviations of wind speed. Xiong et al. (2025) similarly assume a locally uniform, primarily unidirectional wind field with superimposed turbulence. A more comprehensive approach is proposed by Q. Ma et al. (2024), who construct a composite wind field combining logarithmic wind shear varying with altitude, stochastic turbulence via the Dryden model, discrete short-duration gusts with a half-sine profile, and a real-time mean wind component. Wind is incorporated into UAV dynamics as a drag force proportional to the relative velocity between the vehicle and the surrounding air. While this formulation captures several

physically meaningful phenomena, it remains spatially homogeneous and does not account for the directionally varying, position-dependent disturbances characteristic of a ship air wake.

Taken together, these studies indicate that wind is included in RL-based landing research, but is often represented using simplified or spatially uniform models that do not fully reflect the structured flow fields observed in experimental and numerical wake analyses. This is particularly relevant for lift-and-cruise UAVs, which are sensitive to vertical and lateral wind components as well as localized shear that are not captured by uniform wind assumptions. This motivates further investigation into whether RL controllers, when trained with spatially resolved wind information, can exploit local flow structure for improved trajectory adaptation rather than treating wind solely as an external disturbance.

3 RESEARCH FRAMEWORK

Autonomous landing of quadplane UAVs on moving maritime platforms remains an open problem due to platform motion, aerodynamic disturbances, and environmental uncertainty. In particular, ship-wake-induced wind fields can exhibit strong, rapidly varying, and spatially localized disturbances that challenge conventional control approaches, especially for quadplane platforms that are susceptible to wind disturbances. While low-level controllers may compensate for disturbances within their saturation budget, the guidance strategy governing the vehicle’s approach trajectory plays a critical role in determining the vehicle’s exposure to wind conditions that may exceed this budget. Heuristic guidance laws, which enforce fixed trajectories without adapting to the prevailing wind environment, cannot proactively route the vehicle away from regions of excessive disturbance. As a result, fixed guidance strategies may commit the vehicle to approach paths that pass through regions of severe wind disturbance, degrading landing performance in conditions where wind-aware trajectory decisions are most needed.

Thus, to overcome these limitations and ensure safe, precise landings under challenging maritime conditions, this thesis proposes a control approach in which an RL-based guidance controller generates high-level velocity commands for an inner-loop PID tracking controller. The performance of the proposed architecture is evaluated against a baseline PID-controlled guidance strategy in which the vehicle tracks a straight-line trajectory at constant altitude until it intercepts the landing plane; upon interception, descent is initiated based on go/no-go timing logic.

The central problem addressed in this thesis is formulated as a research objective as follows:

Research Objective

To develop an adaptive guidance controller which enables lift-and-cruise quadplane UAVs to autonomously land on maritime platforms under localized wind disturbances.

Building on this objective, the remainder of this section derives the research questions and outlines the project plan that guides the execution of this thesis.

3.1 Research Questions

This overarching research objective is addressed through a series of research questions. These are organized into four topic sets. It is expected that answering these questions will achieve the research objective.

The first research topic focuses on the RL problem formulation. This involves defining the maritime UAV landing task as a Markov Decision Process and formalizing all its components, including the state (observation) space, action space, and reward function. Addressing these questions provides the foundation for designing the RL guidance controller that is proposed in this research.

A. RL Problem Formulation

RQ-A1: Which control performance metrics and constraints characterize successful autonomous landings of a UAV on a moving maritime platform?

RQ-A2: How can the maritime UAV landing problem be formulated as a guidance-level RL problem that interfaces with a classical PID-based tracking controller?

RQ-A3: Which performance metrics are most suitable for quantifying convergence, robustness, and operational landing performance in RL-based maritime UAV landing tasks?

The second research theme addresses the modeling of the environment in which the RL agent operates for autonomous maritime UAV landings. This includes the representation of the moving target platform and the ship wake-induced wind disturbances that dominate the operational conditions. In addition, the interaction between the maritime environment and onboard sensing is considered, with emphasis on how environmental conditions constrain and shape the information available to the learning agent.

B. Environment and Disturbance Modeling

RQ-B1: Which characteristics of the maritime operational environment are essential to represent when learning guidance policies for autonomous UAV landings?

Simulating the quadplane UAV with sufficient fidelity for guidance-level RL requires a dedicated modeling effort, as no validated quadplane model is readily available in existing simulation frameworks. PyBullet, a Python-compatible physics engine widely used in RL research through its integration with OpenAI Gym, primarily contains validated models of simple mechanical and robotics-oriented systems. While Panerati et al. (2021) introduced gym-pybullet-drones, providing a validated dynamic model of the Bitcraze Crazyflie 2.x nano-quadrotor, this does not extend to larger multirotor platforms or hybrid quadplane configurations. The following research questions, therefore, address the modeling and verification of a quadplane simulation suitable for this work.

C. Quadplane UAV Modeling

RQ-C1: Which dynamic flight phases of the quadplane UAV (cruise, transition, and hover) are relevant for the guidance problem under investigation?

RQ-C2: How can the different dynamic flight phases of a quadplane UAV be modeled in PyBullet with sufficient fidelity for guidance-level RL?

RQ-C3: How can the PID tracking controller be designed and tuned to reliably track the guidance commands while supporting stable and safe landings in a maritime context?

RQ-C4: How can the implemented quadplane simulation model be verified prior to RL implementation?

Finally, a set of questions focuses on evaluating the performance of the developed RL-based adaptive guidance solution, which constitutes the central objective of this thesis. To this end, the following research questions are addressed:

D. Proposed Solution Performance Evaluation

RQ-D1: How does the proposed solution compare to the baseline heuristic guidance law in terms of ship landing performance?

RQ-D2: To what extent does the proposed solution exhibit evasive behavior under wind conditions that cause actuator saturation?

RQ-D3: How does the choice of observation space formulation influence the agent's ability to react to wind disturbances in real time?

3.2 Research Plan

In accordance with the research objectives and questions defined for this project, a timeline is established to structure the work over the eight-month thesis period. The main tasks are:

1. Conducting a literature review and framing the project scope;
2. Implementing and verifying the UAV and simulation environment models;
3. Designing, implementing, and tuning the cascaded PID tracking controller;
4. Implementing the RL algorithm;
5. Training the RL agent;
6. Evaluating the proposed control architecture against the heuristic guidance controller;
7. Preparing the thesis report and final presentation.

These tasks are organized into five main project phases: literature review, simulation environment modeling, RL implementation, evaluation, and thesis documentation. Figure 13 presents the proposed timeline for each phase.

To reduce development risk and manage the complexity of the project, the implementation tasks associated with the simulation environment, controller design, and RL (Tasks 2–6) are planned to be carried out using a staged development strategy. Rather than developing the complete quadplane system from the outset, the methodology is structured to progress from a simplified quadrotor model to the full quadplane dynamics.

Stage 1 – Simplified Quadrotor Model (DJI F450)

The initial development phase employs a simplified quadrotor model based on the DJI F450 platform. This model incorporates only rigid-body dynamics and rotor thrust generation, while intentionally omitting aerodynamic effects.

The objective of this stage is to establish and verify the complete control pipeline, including the simulation environment, the cascaded PID controller, the RL framework, and the interaction between the guidance policy and the low-level flight controller.

Stage 2 – Extended Quadplane Model

Following successful verification of the simplified implementation, the simulation is extended to represent the complete quadplane vehicle. The upgraded model incorporates additional physical effects, including aerodynamic force and moment models, asymmetric inertia properties, actuator first-order dynamics with propulsion time delays, and wind disturbances.

Rather than redesigning the controller and RL framework, the existing implementation is incrementally extended, requiring only gain retuning for the updated dynamics. This isolates the added complexity of the full quadplane model to a later development stage, while enabling early verification of the complete control pipeline in stage 1.

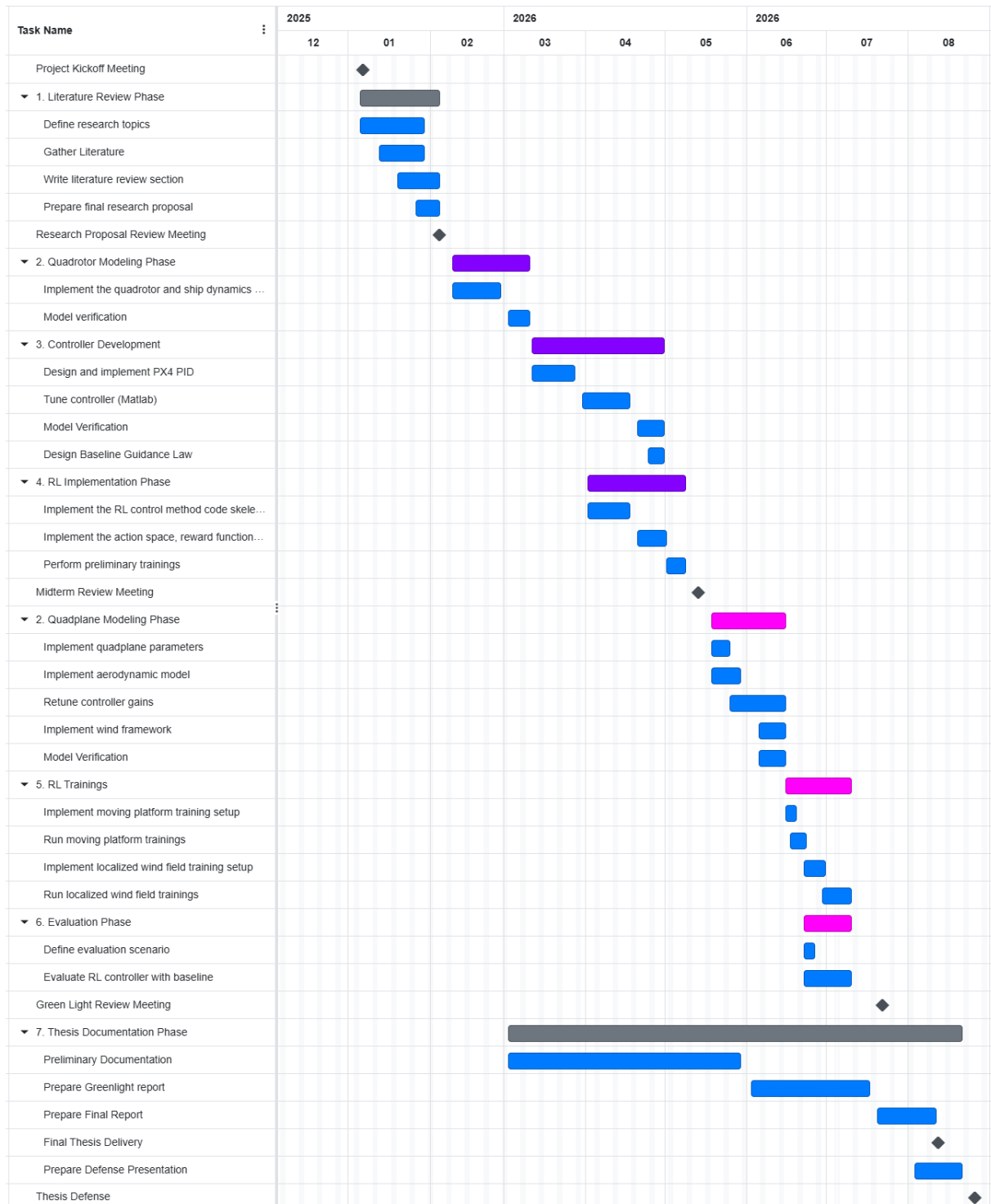


Figure 13: Proposed thesis timeline showing the project phases and associated tasks. Purple indicates Stage 1 (quadrotor), while pink indicates Stage 2 (quadplane).

Part II: Scientific Paper

Reinforcement Learning Control For Quadplane Landings

Buse Kaya

Control & Simulation Department, Faculty of Aerospace Engineering
Delft University of Technology, Delft, The Netherlands

Abstract

This thesis addresses the problem of autonomous landing of a hybrid quadplane UAV under localized turbulent wind conditions. The UAV is modeled in hover configuration, where only the upward-facing rotors are used for control allocation. A Proportional–Integral–Derivative (PID) controller is implemented as a low-level tracking controller. As a baseline, a finite-state machine (FSM)-based guidance law comprising approach, hover, and descent phases with predefined transition criteria is benchmarked against a reinforcement learning (RL)-based guidance policy. The RL agent generates velocity commands that are tracked by the PID controller. Simulation results demonstrate that the RL-based policy leverages real-time wind disturbance observations to adapt its trajectory, avoiding hazardous downwash regions. The agent additionally learns the typical geometric position and size of the localized wind field encountered during training, and incorporates this as a prior in its guidance strategy alongside its real-time sensing capability. Under a randomized wind field around the landing platform, the proposed approach achieves a landing success rate of 66%, compared to 24% for the baseline. The RL-based method additionally reduces both overall landing time and touchdown velocity.

Keywords: Reinforcement learning, maritime landing, unmanned aerial vehicle, quadplane, hybrid drones, wind disturbance, guidance control

I. Introduction

Unmanned Aerial Vehicles (UAVs) have gained significant research interest due to their expanding role in applications such as package delivery, surveillance, environmental monitoring, and disaster response (Smeur et al., 2016; Xie et al., 2020; Yıkılmaz & Wagter, 2025). Among these systems, hybrid UAVs have attracted particular attention for missions requiring both long endurance and extended range, as they combine the efficient cruise performance of fixed-wing aircraft with the vertical take-off, landing, and hovering capabilities of rotorcraft (Z. Ma et al., 2022). These combined capabilities make hybrid platforms especially suitable for maritime operations, including offshore wind farm inspection

(Xu et al., 2025), maritime surveillance (Zhao et al., 2023), and search-and-rescue missions (Messmer et al., 2024).

A key challenge in such environments is autonomous landing on moving maritime platforms, which is complicated by platform motion (Ibinabo & Tamunodukobipi, 2019), environmental disturbances (Söker et al., 2000), and sensor degradation (Xie et al., 2020). This problem is further exacerbated for hybrid UAVs due to their increased sensitivity to aerodynamic effects, particularly in the strongly turbulent and spatially varying wind fields characteristic of maritime landing environments (Setiawan et al., 2025; Söker et al., 2000). These localized flow conditions around ship superstructures can

significantly impact the overall mission safety and success.

Existing landing guidance strategies are often based on heuristic and finite-state-machine (FSM) control logic derived from helicopter landing procedures (Abujoub et al., 2020; Ding & Huang, 2023). While effective in nominal scenarios, such strategies do not adjust in real time to turbulence or rapidly varying flow near a ship’s deck.

More recently, adaptive control strategies have been explored for maritime landing problems (Amendola et al., 2024). In particular, reinforcement learning (RL) has emerged as a promising framework for learning control policies that adapt to diverse and uncertain operating conditions without requiring explicit disturbance models. Prior work has shown that RL agents can learn ship motion patterns from experience, including oscillatory and quasi-periodic behavior, and adapt their guidance strategies accordingly (Q. Ma et al., 2024; Wang et al., 2023; Yıkılmaz & Wagter, 2025). Furthermore, RL-based inner-loop controllers have demonstrated improved disturbance rejection compared to classical PID-based approaches (Saj et al., 2022; Wu et al., 2022).

Despite these advances, current approaches remain limited in two key aspects: (i) relatively little focus has been placed on evaluating the suitability of RL-based control for hybrid UAV platforms, and (ii) existing RL-based methods rarely evaluate the incorporation of localized wind-field information into the guidance and trajectory generation process.

Motivated by these gaps, the contribution of this paper is an RL-based guidance architecture for hybrid UAVs that leverages localized wind-field effects around landing platforms. In this architecture, a learned policy generates high-level velocity commands for an inner-loop PID tracking controller, trained in an environment that incorporates spatially localized, randomized wind-field disturbances.

The remainder of this paper is organized as follows. Section II reviews the relevant literature and establishes the state of the art. Section III presents the dynamic model of the hybrid UAV platform. Section IV introduces the baseline tracking controller and guidance law. Section V presents the proposed RL-based controller, including the framework and algorithmic choices. Section VI describes the training environment, including the ship motion and wind models. Section VII presents the experimental evaluation framework and compares the proposed method against the baseline. Finally, Section VIII discusses the findings, highlights limitations, and outlines directions for future work.

II. Background

This section reviews the maritime UAV landing problem, conventional guidance and control approaches, and recent RL-based methods for autonomous landing.

A. Problem Formulation

The maritime landing problem is defined by the requirement to achieve accurate position and attitude alignment at touchdown while maintaining bounded relative velocities in the presence of environmental disturbances. In this work, performance is evaluated using landing accuracy, safety constraints on velocity and attitude, trajectory-tracking quality, and robustness to disturbances and uncertainties.

The UAV (with subscript "v") and target platform (with subscript "t") are modeled in a shared inertial frame, assuming a flat, non-rotating Earth. Each system state consists of position, velocity, and Euler angles; the relative state is defined as the difference between them:

$$\mathbf{x}_{rel} = \mathbf{x}_t - \mathbf{x}_v \in \mathbb{R}^9. \quad (1)$$

A landing is considered successful if the relative position, velocity, and attitude at touchdown satisfy prescribed bounds:

$$\begin{aligned}
|p_{rel,z}(t_f)| &= 0, \\
\|\mathbf{p}_{rel,xy}(t_f)\| &\leq p_{xy}^{\max}, \\
0 \leq v_{rel,z}(t_f) &\leq v_z^{\max}, \\
\|\boldsymbol{\eta}_{rel}(t_f)\| &\leq \eta_{\max}.
\end{aligned} \tag{2}$$

where the relative state vectors are defined as $\mathbf{p}_{rel} = [x_t \ y_t \ z_t]^\top - [x_v \ y_v \ z_v]^\top$, $\mathbf{v}_{rel} = [\dot{x}_t \ \dot{y}_t \ \dot{z}_t]^\top - [\dot{x}_v \ \dot{y}_v \ \dot{z}_v]^\top$, and $\boldsymbol{\eta}_{rel} = [\phi_t \ \theta_t \ \psi_t]^\top - [\phi_v \ \theta_v \ \psi_v]^\top$, and $\mathbf{p}_{rel,xy} = [x_t \ y_t]^\top - [x_v \ y_v]^\top$ the horizontal component of $\mathbf{p}_{rel}$.

B. Classical Control Strategies

Commonly implemented guidance approaches for shipboard landings rely on heuristic procedures inspired by helicopter operations, as visualized in Figure 1. The UAV first follows a horizontal approach trajectory at a fixed altitude, recomputed at each sampling instant to account for the ship’s relative motion and align the vehicle with the landing platform. Once aligned, the descent phase begins, extending the reference trajectory with a controlled vertical component while continuing to compensate for platform motion to preserve lateral alignment throughout the descent (Abujoub et al., 2020; Ding & Huang, 2023).

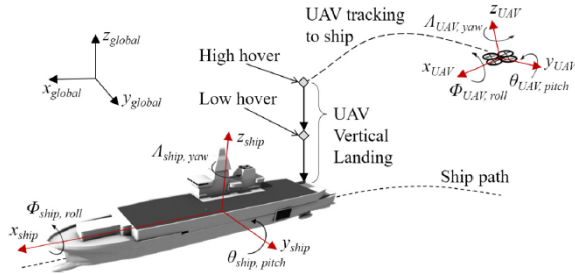


Figure 1: Deck landing mission task elements for UAVs. Retrieved from Abujoub et al. (2020).

There are also recent studies that have explored optimization-based trajectory planning for UAV landings on moving vessels. Gradient-descent methods iteratively adjust state and control sequences to minimize a predefined cost function (Tan et al., 2025). Gao et al. (2023) uses Model Predictive Control (MPC) to predict future

ship motion and plan a UAV trajectory that compensates for it, enabling safe landing on a moving platform. Another approach uses visual-servoing and image-based guidance, generating real-time approach paths by reacting directly to visual cues from landing markers or platforms (Keipour et al., 2022).

In these architectures, the guidance layer generates desired trajectories or setpoints, which are then tracked by a lower-level controller that computes the corresponding actuator commands.

PID control is among the most widely used strategies in industry for tracking (Smeur et al., 2016). It reacts to the error between desired and current states and is not inherently model-based. However, good performance typically requires tuning informed by detailed system knowledge (Cárdenas R. et al., 2020), and from a theoretical standpoint, PID is not the highest-performance architecture available (Eschmann et al., 2024).

The Linear Quadratic Regulator (LQR) is an optimal state-feedback method that computes the feedback gain matrix by minimizing a quadratic cost function. Unlike PID, where gains are tuned empirically across coupled channels, LQR reduces the design problem to selecting two interpretable weighting matrices, with gains derived automatically through optimization.

Explicitly model-based approaches—including feedback linearization, backstepping, and sliding mode control (SMC)—actively cancel or exploit system nonlinearities, enabling precise tracking guarantees that linear controllers cannot provide far from equilibrium (Smeur et al., 2016). MPC extends this further by solving a finite-horizon optimization problem at each timestep, explicitly handling state and input constraints such as actuator saturation and stability boundaries (Feng et al., 2018). Its receding-horizon strategy allows proactive correction of deviations, a key advantage over reactive controllers in systems with significant lag or hard operational limits.

C. Introduction to RL

RL is a branch of machine learning in which an agent learns to make sequential decisions through interaction with an environment. At each discrete time step t , the agent observes the environment state $s_t \in \mathcal{S}$ and selects an action $a_t \in \mathcal{A}$ from a predefined action space. The action influences the environment dynamics, leading to a new state s_{t+1} , and the environment returns a scalar reward $R(s_t, a_t)$ that evaluates the immediate quality of the action. By interacting with the environment, the agent aims to learn an optimal policy $\pi(a_t | s_t)$ that specifies the action to take in each state. (Kaelbling et al., 1996; Sutton & Barto, 2018).

This problem is commonly formulated as a Markov Decision Process (MDP), defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(s_{t+1} | s_t, a_t)$ defines the transition dynamics, $R(s_t, a_t)$ is the reward function, and $\gamma \in [0, 1]$ is the discount factor. The MDP framework relies on the Markov property, which states that the next state depends only on the current state and action, not on the history.

In many real-world applications, the agent does not have access to the full state s_t and instead receives partial observations $o_t \in \Omega$. In this case, the process is generally no longer Markov with respect to the observation sequence alone, since o_t does not uniquely determine s_t . This is modeled as a partially observable Markov decision process (POMDP), defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma, \Omega, O)$, where Ω is the observation space and $O(o_t | s_t, a_{t-1})$ is the observation function. In this setting, the agent must infer the underlying state from the history of observations and actions $\{o_0, a_0, o_1, a_1, \dots, o_t\}$ rather than directly accessing s_t .

In RL, a policy function $\pi(a_t | o_t)$ governs the agent's behavior by specifying the probability of selecting an action a_t given an observation o_t at state s_t . The objective is to identify an optimal policy that maximizes the expected cumulative discounted reward $G_t = \sum_{t=0}^{\infty} \gamma^t R_{t+1}$, ensuring the highest achievable long-term return from any

initial state. This optimal behavior is expressed as Equation 3.

$$\pi^* = \arg \max_{\pi} \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]. \quad (3)$$

Different mathematical approaches exist for estimating the optimal policy, and the choice of method critically influences an agent's ability to learn under a given problem formulation. The three principal classes of RL algorithms are value-based, policy-based, and actor-critic methods.

Value-based methods estimate the long-term value of actions and select the highest-valued action, but are less suited to continuous control (Mnih et al., 2015). Policy-based methods instead learn a direct mapping from states to actions, naturally handling continuous action spaces at the cost of higher variance in the learning signal. Actor-critic methods combine both: an actor selects actions while a critic estimates their value, yielding a more stable and sample-efficient learning process. This combined structure is widely used in modern continuous control problems, including UAV guidance and landing tasks (Fujimoto et al., 2018; Haarnoja et al., 2018).

D. Related Works

A comprehensive overview of RL applications in quadrotor UAV landing is provided by Amendola et al. (2024), which distinguishes between stationary and moving-platform scenarios. Within this body of work, some studies focus on inner-loop control, while others evaluate the potential of RL for guidance-level control. Most report substantial performance improvements over baseline methods, particularly for landing on moving platforms whose motion exhibits spectrally correlated dynamics that the agent can learn to exploit. The majority of these approaches adopt continuous-control formulations and, correspondingly, rely on actor-critic methods such as DDPG, PPO, SAC, and their variants. The selection of specific algorithms is

typically empirical rather than systematic, and is partly influenced by the study timeline relative to each algorithm’s year of introduction (DDPG: 2015, PPO: 2017, TD3: 2018, SAC: 2019).

Beyond algorithm choice, RL formulations differ substantially in state and action design. Many works assume direct access to relative UAV–platform states (position, velocity, and attitude), while others incorporate real-time vision-based observations (Chen et al., 2020; Saj et al., 2022; Wang et al., 2023). In some cases, sensor noise is explicitly modeled and filtered, or state estimation is performed using Kalman-based observers, with the resulting estimates provided to the policy (Wang et al., 2023; Xie et al., 2020).

While most prior work does not treat wind disturbance as a primary design challenge, with platform motion instead remaining the focus, a subset of studies incorporates it for robustness evaluation or as part of the broader implementation scope. As the majority of these studies address quadrotor platforms, aerodynamic effects, when modeled at all, are typically limited to drag. Studies that do account for wind disturbances face a design choice between including or excluding wind-related state information in the observation space, and evidence regarding the benefit of explicit wind sensing remains mixed: Xiong et al. (2025) reports limited performance gains from direct wind measurements in quadrotor landing tasks, whereas Olivares et al. (2024) demonstrates improved performance for fixed-wing systems when wind information is explicitly provided. Given the practical difficulty of accurately measuring local wind fields, this remains an open and significant research question.

Continuous action spaces are typically formulated in terms of position offsets, velocity, or acceleration commands within a guidance–tracking control architecture, where RL generates high-level trajectory commands that are subsequently tracked by lower-level controllers. Representative examples include learned short-horizon guidance policies (Xiong et al., 2025)

and RL-based acceleration commands combined with inner-loop nonlinear control (Yıkılmaz & Wagter, 2025). Other studies instead employ RL directly for inner-loop control, eliminating the need for a separate tracking controller; these approaches have demonstrated improved disturbance rejection compared to classical PID baselines (Abo Mosali, 2022; Kim et al., 2023; Saj et al., 2022).

Reward design is typically grounded in classical control objectives, with successful landing defined by position, velocity, and attitude thresholds at touchdown. Sparse success–failure rewards are commonly augmented with distance-based shaping or step penalties to improve convergence. Prior work shows that dense reward shaping based on velocity and collision metrics can significantly improve landing safety by encouraging early deceleration behavior, although this often comes at the cost of increased landing time (Yıkılmaz & Wagter, 2025).

These studies collectively highlight the growing interest in applying RL to UAV control, with many demonstrating improved performance in maritime landing scenarios. However, a large portion of existing studies relies on small quadrotor platforms, including models such as the Crazyflie, Parrot AR, and Iris. These platforms are well-suited for proof-of-concept validation but are not representative of long-endurance maritime operations, and their relatively low exposed (wetted) area provides limited insight into aerodynamic effects under wind disturbances. Moreover, most of these studies either do not consider wind at all or, when they do, model it as a uniform ambient wind with superimposed turbulence, so localized wind-field information is never the primary focus. Thus, a research gap remains in bridging maritime landing simulation for hybrid UAV platforms with the use of localized wind-field information within RL-based guidance and trajectory generation.

III. Quadplane Dynamic Model

A lift-and-cruise quadplane is a hybrid vertical takeoff and landing (VTOL) UAV that integrates characteristics of both multirotor and fixed-wing aircraft. The vehicle typically features four or more electric rotors in a multicopter configuration, providing vertical thrust for takeoff, landing, and hover, and a nose- or tail-mounted propeller delivering forward thrust during level flight. Fixed wings generate aerodynamic lift during cruise, reducing the load on the vertical lift motors, and may include control surfaces for maneuvering. In this work, the vehicle configuration is inspired by the `standard_vtol` model from the PX4-Autopilot Gazebo simulation package, shown in Figure 2.

The dynamics simulation is implemented using the PyBullet physics engine, which provides rigid-body dynamics, collision detection, and numerical integration for the simulation environment.

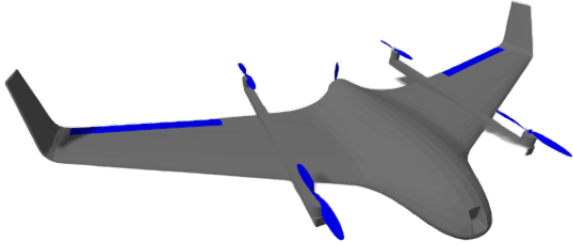


Figure 2: Gazebo-compatible platform geometry model of a lift & cruise aircraft: the `standard_vtol` from the PX4-Autopilot repository.¹

During landing, the vehicle is assumed to operate in the hover-dominated regime, with control authority provided solely by the four vertical lift rotors while the aerodynamic control surfaces remain inactive. Under these low-speed conditions, the vehicle's landing dynamics are assumed comparable to those of a quadrotor of similar scale, while retaining the inertial and aerodynamic properties of the quadplane.

¹https://docs.px4.io/main/en/sim_gazebo_gz/vehicles#standard-vtol

The UAV dynamics are described using an inertial frame $\mathcal{F}_I$ and a body-fixed frame $\mathcal{F}_B$, as illustrated in Figure 3. The inertial frame $\mathcal{F}_I$ is Earth-fixed with axes x_I , y_I , and z_I , and is assumed to be flat and non-rotating. The body-fixed frame $\mathcal{F}_B$ is attached to the UAV center of mass, with x_B pointing forward, y_B to the left, and z_B upward. Its orientation relative to $\mathcal{F}_I$ is described by the Euler angles ϕ , θ , and ψ through the rotation matrix $\mathbf{R}_B^I(\phi, \theta, \psi)$.

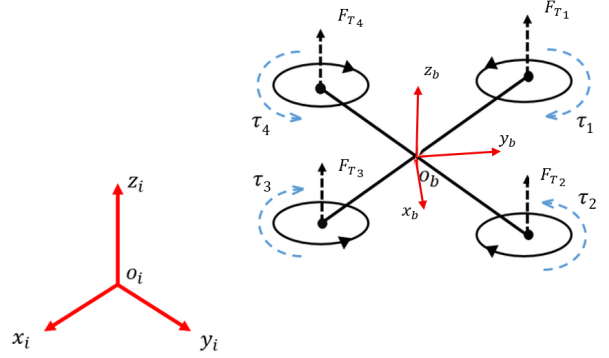


Figure 3: Free-body diagram of an X4 configuration quadrotor UAV illustrating the inertial Earth-fixed and body-fixed coordinate frames. Adapted from Ho et al., 2017.

The translational dynamics are expressed in $\mathcal{F}_I$ using Newton's second law,

$$m\ddot{\mathbf{p}} = \sum \mathbf{F}_I, \quad (4)$$

where $\mathbf{p} = [x_I \ y_I \ z_I]^T$. Rotational dynamics are expressed in $\mathcal{F}_B$ using Euler's rigid-body equation with a diagonal inertia matrix $\mathbf{J} = \text{diag}(J_x, J_y, J_z)$, justified by the assumed symmetry of the vehicle:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{J_y - J_z}{J_x} qr \\ \frac{J_z - J_x}{J_y} pr \\ \frac{J_x - J_y}{J_z} pq \end{bmatrix} + \begin{bmatrix} \frac{1}{J_x} \\ \frac{1}{J_y} \\ \frac{1}{J_z} \end{bmatrix} \mathbf{M}_B, \quad (5)$$

where $\boldsymbol{\omega} = [p \ q \ r]^T$ is the body angular velocity and $\mathbf{M}_B$ the total body-frame moment. Euler angle rates follow from the standard kinematic transformation of $\boldsymbol{\omega}$ (Quan et al., 2025).

The aerodynamic reference frame $\mathcal{F}_A$ is defined with respect to the relative airflow and is used

to express the aerodynamic forces acting on the UAV. Its x_A -axis is aligned with the relative airflow, while the z_A -axis lies in the aircraft symmetry plane and is perpendicular to x_A . The orientation of $\mathcal{F}_A$ relative to the body-fixed frame $\mathcal{F}_B$ is characterized by the angle of attack α and sideslip angle β .

Aerodynamic forces are computed in the aerodynamic frame $\mathcal{F}_A$ and subsequently transformed into the body frame. The relative air velocity $\mathbf{V}_{\text{rel}}^I = \mathbf{V}_{UAV}^I - \mathbf{V}_{\text{wind}}^I$ is rotated into $\mathcal{F}_B$ via $\mathbf{V}_{\text{rel}}^B = \mathbf{R}_I^B \mathbf{V}_{\text{rel}}^I$, from which the angle of attack α and sideslip angle β are obtained as

$$\alpha = \arctan \frac{V_{b,z}}{V_{b,x}}, \quad \beta = \arcsin \frac{V_{b,y}}{V}, \quad (6)$$

with $V = \|\mathbf{V}_{\text{rel}}^B\|$ the airspeed. The aerodynamic forces are then given by

$$\begin{aligned} D &= \frac{1}{2} \rho V^2 A_{\text{ref}} C_D, \\ Y &= \frac{1}{2} \rho V^2 A_{\text{ref}} C_Y, \\ L &= \frac{1}{2} \rho V^2 A_{\text{ref}} C_L \end{aligned} \quad (7)$$

where ρ is the air density, A_{ref} the reference wing area, and C_D , C_Y , C_L the drag, side force, and lift coefficients, respectively.

The lift and drag coefficients, $C_L(\alpha)$ and $C_D(\alpha)$, are adopted from Z. Ma et al. (2022), who performed wind tunnel measurements on a bare wing (without fuselage) over the full range of angles of attack encountered during VTOL operation ($\alpha \in [-180^\circ, 180^\circ]$). The lateral force coefficient $C_Y(\beta)$ is not measured independently in the study by Z. Ma et al. (2022), and is instead approximated by reusing the drag model evaluated at the sideslip angle. This reflects the assumption that sideslip-induced lateral forces, like drag, are dominated by crossflow separation and pressure drag rather than lift-generating effects. A scaling factor is applied to account for the smaller exposed surface area at a given sideslip angle compared to the equivalent angle of attack, since the wing presents a smaller frontal area to the flow in sideslip than in the longitudinal direction.

Although derived from a simplified wing geometry rather than a complete airframe, these coefficients are assumed to provide a reasonable approximation of the aerodynamic behavior of the considered platform, and the resulting aerodynamic forces are verified to be of appropriate magnitude for the present vehicle.

The resulting force vector $\mathbf{F}_{\text{aero}}^A = [-D \ Y \ L]^T$ is transformed to the body frame via $\mathbf{F}_{\text{aero}}^B = \mathbf{R}_A^B \mathbf{F}_{\text{aero}}^A$, where $\mathbf{R}_A^B$ is parameterized by α and β .

Following Kroo et al. (2000), which shows that hub forces, flapping, and ground effects are an order of magnitude smaller than rotor thrust and drag at wind speeds below 10 m/s, these effects are neglected in the present aerodynamic model. Aerodynamic moments are likewise neglected, consistent with the proof-of-concept scope of this model, which focuses on evaluating the RL agent's ability to infer wind disturbances rather than achieving high-fidelity force and moment prediction.

Propulsive force acts along z_b , with total thrust $F_T = \sum_{i=1}^4 K_T \omega_i^2$. Roll and pitch torques result from differential rotor thrust at arm length l ,

$$\begin{aligned} \tau_\phi &= \frac{l}{\sqrt{2}} (F_1 + F_2 - F_3 - F_4), \\ \tau_\theta &= \frac{l}{\sqrt{2}} (F_1 - F_2 - F_3 + F_4), \end{aligned} \quad (8)$$

while yaw torque arises from rotor reaction torques $\tau_i = K_M \omega_i^2$ on counter-rotating rotor pairs, $\tau_\psi = \tau_1 - \tau_2 + \tau_3 - \tau_4$. The total moment is $\mathbf{M}_B = [\tau_\phi \ \tau_\theta \ \tau_\psi]^T + [M_x \ M_y \ M_z]^T$, substituted into (5).

Combining propulsive, aerodynamic, and gravitational forces, the translational dynamics become:

$$m \dot{\mathbf{v}}_I = \mathbf{R}_B^I (\mathbf{F}_{\text{prop}}^B + \mathbf{F}_{\text{aero}}^B) + \mathbf{F}_{\text{grav}}^I. \quad (9)$$

Similarly, the nonlinear rotational equations of motion can then be expressed as follows:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{J_y - J_z}{J_x} q r \\ \frac{J_z - J_x}{J_y} p r \\ \frac{J_x - J_y}{J_z} p q \end{bmatrix} + \begin{bmatrix} \frac{1}{J_x} \\ \frac{1}{J_y} \\ \frac{1}{J_z} \end{bmatrix} \mathbf{M}_{\text{rotor}}^B. \quad (10)$$

The physical and propulsive parameters, listed in Table 1, correspond to the `standard_vtol` configuration from the PX4-Autopilot Gazebo package², with parameters unavailable in that model, including wing surface area and geometry parameters and propulsive parameters, estimated from real industry platforms such as the DeltaQuad VTOL UAV.³

Table 1: Quadplane physical and propulsion parameters

Symbol	Value [Units]
m	5 [kg]
I_{xx}	0.478 [kg·m ²]
I_{yy}	0.342 [kg·m ²]
I_{zz}	0.811 [kg·m ²]
b	2 [m]
A_{ref}	0.6 [m ²]
λ	0.2 [-]
c	0.35 [m]
l	0.495 [m]
N	4 [-]
K_T	4.05e-7 [N/RPM ²]
K_M	3.0e-8 [N·m/RPM ²]
F_T/W	2.01 [-]

Real flight systems exhibit dynamic delays between commanded actuator inputs and the physical response of the vehicle. To capture this rotor response dynamics, a first-order lag is applied between the commanded and actual rotor speed,

$$\tau \dot{\omega}(t) + \omega(t) = \omega_{cmd}(t), \quad (11)$$

with $\tau = 0.045$ s adopted from the identified motor time constant of a comparable 3.5 kg quadrotor by Eschmann et al. (2024).

IV. Controller Design

This section presents the tracking controller and baseline guidance law, which together form the basis for comparison for the RL-based guidance policy evaluated throughout this work. A cascaded PID tracking controller regulates the

vehicle’s translational and rotational dynamics and is shared between the baseline and RL configurations. The baseline guidance law itself is implemented as a finite-state sequence that generates position commands for the maritime landing task.

A. Tracking Controller

The baseline controller adopts a cascaded PID architecture comprising an outer-loop position and velocity controller, an inner-loop attitude and angular-rate controller, and a control allocation stage, as illustrated in Figure 4. This structure exploits the timescale separation between translational and rotational dynamics, with the inner loops acting as fast, well-damped actuators for the outer loops, enabling decoupled gain tuning without compromising closed-loop stability.

The attitude controller computes orientation error directly on $SO(3)$, avoiding the singularities and axis coupling associated with Euler-angle representations. Rather than tracking individual roll, pitch, and yaw angles, the controller expresses the current and desired orientations as rotation matrices $R, R_d \in SO(3)$ and computes the attitude error from their relative rotation $R_d^T R$, yielding a globally well-defined error vector regardless of the vehicle’s orientation. Another notable controller design choice concerns ship-landing operations: the deck velocity $\mathbf{v}_t$ is incorporated as a feedforward term within the velocity loop, ensuring that the closed-loop system regulates motion relative to the moving platform rather than the inertial frame.

Each PID loop is tuned using a loop-shaping framework, with crossover frequency, gain margin, phase margin, and peak sensitivity requirements assigned according to the cascade separation principle. The innermost angular-rate loop is tuned to provide the highest bandwidth, while each successive outer loop is tuned progressively more conservatively to preserve adequate bandwidth separation and robust closed-loop performance. The resulting closed-loop bandwidths are approximately 23, 9, 4, and

²https://docs.px4.io/main/en/sim_gazebo_gz/index

³<https://www.deltaquad.com/>

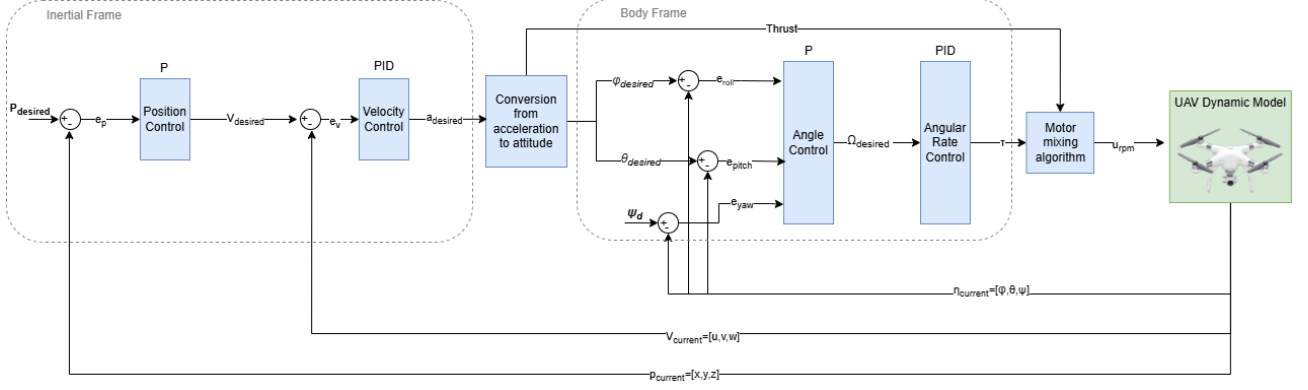


Figure 4: Cascaded PID controller architecture, inspired by the PX4 implementation.

2 rad/s for the angular-rate, attitude, velocity, and position loops, respectively, providing the frequency separation required for the cascade design assumption, whereby each outer loop treats its corresponding inner loop as effectively instantaneous. The resulting controller gains are given in Table 2.

Table 2: Final tuned controller gains for the cascaded PID architecture.

Parameter	x	y	z
<i>Position Loop (P)</i>			
K_{pos}	0.8	0.8	1.0
<i>Velocity Loop (PID)</i>			
K_P^{vel}	1.1	1.1	6.5
T_I^{vel}	1.5	1.5	1.5
T_D^{vel}	0.065	0.065	0.03
N^{vel}	5.0	5.0	5.0
<i>Attitude Loop (P)</i>			
K_{att}	4.8	4.7	2.4
<i>Rate Loop (PID)</i>			
K_P^{rate}	2.7	2.1	5.8
T_I^{rate}	1.6	1.6	1.6
T_D^{rate}	0.17	0.15	0.12
N^{rate}	10.0	10.0	10.0

B. Maritime Landing Baseline Guidance Strategy

The baseline guidance law operates above the PID tracking controller, issuing position com-

mands $\mathbf{p}_{cmd} = [x_{cmd}, y_{cmd}, z_{cmd}]^T$ to the outer-loop position controller. It is implemented as a three-phase finite-state landing sequence: approach, station keeping, and descent.

In the approach phase, the UAV is commanded to track a position at a fixed relative altitude Δz_{hover} above the geometric center of the landing platform, with the position continuously updated to account for the platform's motion and maintain safe vertical separation while achieving horizontal alignment. The UAV is considered stable once the horizontal tracking error $p_{rel,xy}$ remains below a stability threshold ϵ_{stable} for a minimum dwell time T_{hover} , at which point the system transitions to descent; if the error subsequently exceeds an abort threshold ϵ_{abort} , the system reverts to station keeping.

In the descent phase, the commanded altitude is set to the instantaneous landing platform height, offset by a fixed landing bias:

$$z_{cmd}(t) = z_{target}(t) - b, \quad (12)$$

where $z_{target}(t)$ is the time-varying landing altitude that tracks the platform's vertical motion, and b is a landing bias that prevents asymptotic convergence to the target altitude. The UAV continues to track the platform's horizontal motion throughout descent, yielding the final position setpoint $\mathbf{p}_{cmd} = [x_{target}, y_{target}, z_{cmd}]^T$.

Landing is declared when the vertical separation between the UAV and the pad surface falls within a margin $\epsilon_{land} = 0.05$ m. A landing attempt

is classified as successful if, at the moment of touchdown, the horizontal position error satisfies $|p_{rel,xy}| < 0.50$ m, the relative vertical velocity satisfies $0.0 \leq v_{z,rel} < 0.7$ m/s, and the attitude deviation satisfies $|\boldsymbol{\eta}_{rel}| < 10.0^\circ$, consistent with the criteria defined in Section II. The tuning parameters governing the guidance law are summarized in Table 3.

Table 3: Baseline guidance law tuning parameters.

Parameter	Symbol	Value
Hover altitude above pad	Δz_{hover}	3.0 m
Stable hover duration	t_{hover}	1.5 s
Stability threshold (XY)	ϵ_{stable}	0.3 m
Abort threshold (XY)	ϵ_{abort}	0.7 m
Touchdown bias offset	Δz_{bias}	0.3 m

V. RL Formulation

The landing problem is formulated as a continuous-control RL task. To decouple learning from low-level control, the RL agent operates at the guidance level, while a PID-based inner loop handles fast dynamics. Accordingly, the agent outputs 3D velocity commands that are directly tracked by the velocity controller.

The proposed framework is formulated as follows:

1. **Action Space:** The action space consists of continuous normalized velocity commands,

$$\mathbf{act}_t = [v_x, v_y, v_z]^\top \in [-1, 1]^3,$$

mapped to physical velocity references by scaling the horizontal components by $v_{xy}^{\max} = 4.0$ m/s and the vertical component by $v_z^{\max} = 3.0$ m/s, and passed directly to the inner-loop velocity controller.

2. **Observation Space:** The agent receives a 15-dimensional observation vector comprising quantities realizable by onboard sensors,

$$\mathbf{o}_t = [\mathbf{p}_{rel}, \mathbf{v}_{rel}, \boldsymbol{\eta}_v, \boldsymbol{\omega}_v, \mathbf{a}_{t-1}]^\top \in \mathbb{R}^{15},$$

where $\mathbf{p}_{rel} = \mathbf{p}_t - \mathbf{p}_v$ and $\mathbf{v}_{rel} = \mathbf{v}_t - \mathbf{v}_v$ are the relative position and velocity of the UAV with respect to the landing pad, $\boldsymbol{\eta}_v$ and $\boldsymbol{\omega}_v$ are UAV attitude and angular velocity, and $\mathbf{a}_{t-1}$ is the previous action.

To evaluate wind-awareness, two extended configurations are also considered: a 22-dimensional wind-augmented observation $\mathbf{o}_t^{\text{wind}}$, which appends normalized IMU-derived body accelerations $\tilde{\mathbf{a}}_{\text{body}}$ and hover-normalized rotor speeds $\tilde{\Omega}$ as indirect wind-sensitive cues, and a 52-dimensional full configuration $\mathbf{o}_t^{\text{full}}$, which further appends a rolling history of $H = 10$ directly measured body-frame wind velocity samples, allowing the policy to infer both the magnitude and local rate of change of the disturbance.

3. **Reward Function:** The reward function combines dense shaping terms with sparse terminal signals.

A proximity penalty continuously incentivizes convergence toward the pad:

$$r_d = -0.3 \|\mathbf{p}_{rel}\|. \quad (13)$$

When the UAV is horizontally aligned with the pad ($\|\mathbf{p}_{rel,xy}\| < 0.70$ m), a vertical speed shaping term rewards relative descent velocities within the safe touchdown window $[0, v_{z,\max}^{\text{land}}]$ and penalizes motion outside it:

$$r_z = \begin{cases} 1.5 v_z^{\text{rel}} & v_z^{\text{rel}} < 0 \\ 1.5 (v_z^{\text{rel}}/0.7) & 0 \leq v_z^{\text{rel}} \leq 0.7 \text{ m/s} \\ -1.5 (v_z^{\text{rel}} - 0.7) & v_z^{\text{rel}} > 0.7 \text{ m/s.} \end{cases} \quad (14)$$

An action smoothness penalty discourages abrupt control changes:

$$r_s = -0.3 \|\mathbf{a}_t - \mathbf{a}_{t-1}\|. \quad (15)$$

An attitude stability penalty, weighted by proximity to the pad, penalizes roll and pitch deviations:

$$\begin{aligned} r_\theta &= -0.8 \alpha_{\text{att}} \|\boldsymbol{\eta}_{rp}\|, \\ \alpha_{\text{att}} &= \max(1 - \|\mathbf{p}_{rel}\|, 0). \end{aligned} \quad (16)$$

A path-length penalty discourages inefficient trajectories by penalizing the physical distance covered by the UAV between consecutive control steps:

$$r_p = -0.5 \|\mathbf{p}_{v_t} - \mathbf{p}_{v_{t-1}}\|, \quad (17)$$

where $\mathbf{p}_{v_t}$ and $\mathbf{p}_{v_{t-1}}$ denote the UAV position at the current and previous control steps, respectively.

Terminal rewards and penalties are applied at episode end. A successful landing yields $r_{\text{land}} = +200$; an unsuccessful termination incurs $r_{\text{term}} = -100$; truncation at the maximum horizon is penalised proportionally to the remaining distance, $r_{\text{trunc}} = -\|\mathbf{p}_{\text{rel}}\|$.

4. **Success, Termination & Truncation:** A successful landing requires that the terminal relative position, velocity, and attitude constraints defined in Equation 2 are simultaneously satisfied at t_f . Termination occurs if the altitude condition is met while other constraints are violated, if ground contact occurs, or if $\|\mathbf{p}_{\text{rel}}\| > 15$ m. Episodes are truncated at 100 steps.

A. Algorithm Choice

The continuous control task, combined with stochastic disturbances and sparse rewards, motivates the use of actor-critic policy gradient methods for their training stability and efficient exploration (Olivares et al., 2024; Saj et al., 2022). Soft Actor-Critic (SAC) is selected for its strong empirical performance on continuous control (Haarnoja et al., 2018). SAC is an off-policy method that improves sample efficiency via experience replay and stabilizes learning using twin Q-networks, while its entropy-regularized objective promotes robust exploration:

$$J(\pi) = \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t))) \right], \quad (18)$$

where γ is the discount factor, $\mathcal{H}(\pi(\cdot|s)) = -\mathbb{E}_{a \sim \pi}[\log \pi(a|s)]$ is the policy entropy, and $\alpha > 0$ is the temperature parameter balancing reward and entropy maximization. SAC automatically tunes α toward a target entropy $\mathcal{H}_{\text{target}}$:

$$J(\alpha) = \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi} \left[-\alpha (\log \pi(a|s) + \mathcal{H}_{\text{target}}) \right]. \quad (19)$$

yielding higher exploration early in training and progressively more deterministic behavior as the policy improves. Transitions (s_t, a_t, r_t, s_{t+1}) are stored in a replay buffer $\mathcal{D}$ and sampled in mini-batches to update twin critic networks, whose minimum estimate mitigates value overestimation, and an actor network trained to maximize the estimated Q-value under entropy regularization. Target critic networks are updated via an exponentially weighted average of the current critic parameters for stable training targets.

The implementation in this work uses the Stable Baselines3 SAC framework⁴, configured with the hyperparameters specified in Table 4.

Table 4: SAC Hyperparameters

Hyperparameter	Value
Learning rate (α)	3×10^{-4}
Batch size	256
Replay buffer size	1×10^6
Discount factor (γ)	0.99
Target smoothing coefficient (τ)	0.005
Entropy coefficient (α_{ent})	Auto-tuned
Policy network architecture	(128, 128)
Q-network architecture	(128, 128)
Optimizer	Adam
Gradient steps per update	4
Training frequency	Every step
Target update frequency	1

B. RL Action Frequency

The RL agent is placed above the PID cascade and issues velocity commands that are tracked

⁴<https://github.com/DLR-RM/stable-baselines3>

by the velocity control loop. Its action frequency must therefore balance two competing requirements: acting too quickly leaves the underlying controller insufficient time to respond before the command is superseded, resulting in noisy state transitions and degraded credit assignment, whereas acting too slowly reduces the available control authority over the vehicle dynamics. The RL action frequency must therefore remain below the physics sampling rate while being on the order of the controlled loop's closed-loop bandwidth.

This is implemented through action repeat, where each sampled action is held constant for N consecutive physics steps:

$$f_{\text{RL}} = \frac{f_{\text{physics}}}{N} = \frac{240}{N} \text{ Hz}. \quad (20)$$

For $N = 50$, the resulting action frequency is $f_{\text{RL}} \approx 5 \text{ Hz} \approx 30 \text{ rad/s}$, exceeding the velocity-loop bandwidth of approximately 6 rad/s . The controller, therefore, may not fully reach the commanded velocity before the next action is issued; however, it responds sufficiently to generate meaningful state transitions for reward assignment and policy learning.

VI. Training Environment

The training environment is implemented in the PyBullet physics engine and wrapped in a Gymnasium-compatible interface to enable integration with standard RL frameworks. At each physics step, the UAV dynamics are propagated by evaluating the forces and moments generated by the current rotor RPM commands and disturbances. The resulting translational and rotational accelerations are integrated to update the vehicle state, consisting of position $\mathbf{p}$, velocity $\mathbf{v}$, attitude quaternion $\mathbf{q}$, Euler angles $\boldsymbol{\eta}$, body-frame angular rates $\boldsymbol{\omega}$, and Euler angle rates $\dot{\boldsymbol{\eta}}$. PyBullet is used for collision detection and visualization, while the equations of motion are integrated externally and the updated states are written back to the simulator at each timestep, based on the formulation in Section III.

The training environment incorporates, in addition to the UAV dynamics, a ship motion model and a disturbance model representing localized wind fields around the landing pad, both of which are detailed in this section, along with the episode initialization procedure.

A. Ship Motion Model

Only the translational motion of the ship is considered in this study; rotational dynamics are excluded from scope. The ship motion is defined along its body axes, which, under the rigid-body assumption and in the absence of rotations, coincide with the landing platform axes, with the origin located at the center of the landing pad.

Ship motion is modeled using the JONSWAP spectrum, which characterizes wind-generated, wave-induced vessel oscillations. The spectrum $S(\omega)$ is given by (Hasselmann et al., 1973):

$$S(\omega) = \alpha \frac{g^2}{\omega^5} \exp \left[-\frac{5}{4} \left(\frac{\omega_p}{\omega} \right)^4 \right] \gamma^{\exp \left[-\frac{(\omega - \omega_p)^2}{2\sigma^2 \omega_p^2} \right]}, \quad (21)$$

where ω is the angular frequency, $\omega_p = 2\pi/T_p$ is the peak frequency corresponding to the dominant wave period T_p , $\gamma = 3.3$ is the peak enhancement factor (Williams, 2021), and σ is the spectral width parameter defined as:

$$\sigma = \begin{cases} 0.07, & \omega \leq \omega_p, \\ 0.09, & \omega > \omega_p. \end{cases} \quad (22)$$

The scaling factor α is related to the significant wave height H_s via Equation 23 (Hasselmann et al., 1973).

$$\alpha \approx (1 - 0.287 \ln \gamma) \frac{5}{16} \frac{H_s^2 \omega_p^4}{g^2} \quad (23)$$

The significant wave height H_s and dominant wave period T_p are parameterized according to the World Meteorological Organization (WMO) sea-state classification^{5,6}.

⁵<https://www.nodc.noaa.gov/gtspp/document/codetbls/wmocodes/table3700.html>

⁶https://www.jodc.go.jp/data_format/weather-code.html

To generate a time-domain motion signal, the continuous spectrum is discretized into N frequency components ω_k with amplitudes:

$$A_k = \sqrt{2S(\omega_k)\Delta\omega_k}, \quad k = 1, \dots, N, \quad (24)$$

where $\Delta\omega_k$ is the frequency spacing, uniformly distributed over $[\omega_{\min}, \omega_{\max}]$, with $\omega_{\min} = 0.1$ rad/s and $\omega_{\max} = \max(5, 5\omega_p)$.

The time-domain ship displacement $\eta(t)$ is obtained by superposition:

$$\eta(t) = \sum_{k=1}^N A_k \cos(\omega_k t + \phi_k), \quad (25)$$

where $\phi_k \sim \mathcal{U}(0, 2\pi)$ is a random phase assigned to each component.

Heave motion is modeled using the full JONSWAP spectrum, as it constitutes the dominant translational response under wave excitation from all directions. Surge and sway, which arise indirectly through hydrodynamic forcing and inertial coupling rather than direct wave excitation, exhibit considerably lower response amplitudes in practice (Aird, 2019; Ibinabo & Tamunodukobipi, 2019). Furthermore, ships commonly maintain headings with waves approaching from the bow to limit roll-induced safety risks, which further reduces lateral wave excitation. Accordingly, surge and sway amplitudes are scaled by a factor of 0.4 relative to heave.

The spectral and synthesis parameters are summarized in Table 5, and representative time-domain responses are shown in Fig. 5.

B. Wind Disturbance Model

The wind disturbance model consists of two components: an ambient uniform wind field and one or more spatially localized wind zones. At each simulation step, the total wind velocity experienced by the UAV is computed as a function of its world-frame position, $\mathbf{p}$.

The ambient wind field, $\bar{\mathbf{v}}_0$, represents the background wind conditions present throughout the simulation domain and applies a constant

Table 5: Parameters used for JONSWAP-based ship motion generation.

Parameter	Value
N	500
Δt	0.1 s
$\omega_{\min}$	0.1 rad/s
$\omega_{\max}$	$\max(5, 5\omega_p)$
γ	3.3
ϕ_k	$\mathcal{U}(0, 2\pi)$
Sea states	WMO 1–7
Surge & sway factor	0.4

mean wind velocity at all positions. Localized wind zones supplement the ambient field by defining spatially bounded regions with distinct wind conditions. Each zone is represented as a rectangular prism with explicit bounds in the x -, y -, and z -directions, a mean wind velocity $\bar{\mathbf{v}}_i$, and, optionally, an independent Dryden turbulence model. Multiple localized wind zones may coexist within a single simulation environment.

To avoid discontinuities when transitioning between wind regions, a cosine-based blending scheme is employed. For each localized zone i , a position-dependent weighting function,

$$w_i(\mathbf{p}) \in [0, 1],$$

is computed from the UAV's distance to the zone boundaries. Outside the zone by more than a blending distance d_{blend} , the weight is zero and the zone has no influence. Within a distance d_{blend} of the boundary, the weight increases smoothly according to a cosine ramp. Once the UAV is at least d_{blend} inside the zone, the weight reaches unity and the zone wind is fully applied.

The ambient wind is assigned the complementary weight

$$w_0 = \max\left(0, 1 - \sum_i w_i(\mathbf{p})\right), \quad (26)$$

ensuring that the weighting factors sum to unity. The resulting blended mean wind velocity is

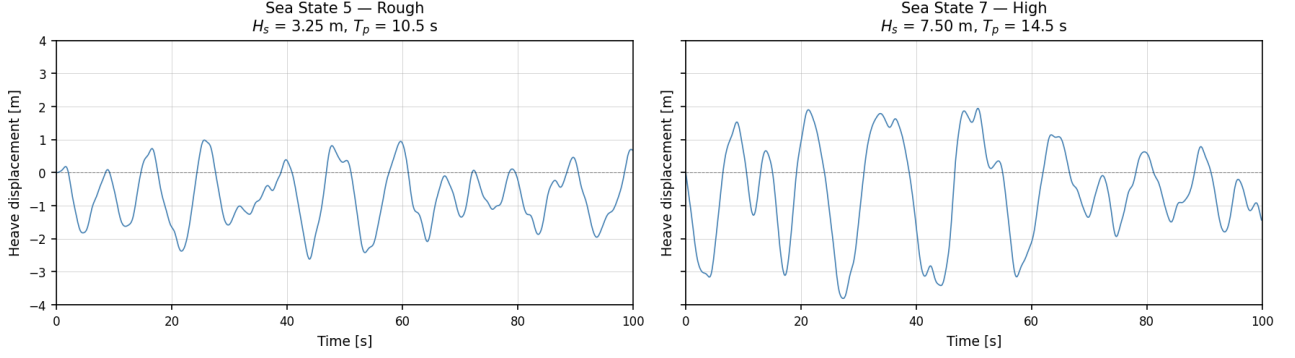


Figure 5: Representative heave motion generated using the JONSWAP model for sea states 5 and 7.

therefore given by

$$\bar{\mathbf{v}}_{\text{wind}}(\mathbf{p}) = \frac{w_0 \bar{\mathbf{v}}_0 + \sum_i w_i(\mathbf{p}) \bar{\mathbf{v}}_i}{w_0 + \sum_i w_i(\mathbf{p})}. \quad (27)$$

Atmospheric turbulence, representing the random small-scale velocity fluctuations arising from mechanical mixing, thermal convection, and wind shear, is modeled using the Dryden turbulence model. The Dryden model generates stochastic gust velocities by passing white noise through rational shaping filters parameterized by the turbulence intensity σ_i and turbulence length scale L_i along each inertial axis. (Stevens & Lewis, 2003)

The longitudinal gust component, u_g , is generated by the first-order transfer function given in Equation 28,

$$H_u(s) = \sigma_u \sqrt{\frac{2V}{\pi L_u}} \frac{1}{s + \frac{V}{L_u}}, \quad (28)$$

whereas the lateral and vertical gust components, v_g and w_g , are generated using the second-order filters given in Equation 29,

$$H_{v,w}(s) = \sigma_{v,w} \sqrt{\frac{3V}{\pi L_{v,w}}} \frac{s + \frac{V}{\sqrt{3} L_{v,w}}}{\left(s + \frac{V}{L_{v,w}}\right)^2}, \quad (29)$$

where V denotes the UAV airspeed (Stevens & Lewis, 2003). The continuous-time filters are

discretized using a first-order Euler scheme at the simulation frequency of 240 Hz.

Each localized wind zone maintains an independent Dryden filter instance with its own turbulence parameters, allowing different regions to exhibit distinct turbulence characteristics. The resulting blended gust vector $\mathbf{g}_{\text{blended}}$ is expressed directly in the inertial frame and added to the blended mean wind, giving the total wind velocity experienced by the UAV is then given by Equation 30:

$$\mathbf{v}_{\text{wind}}(\mathbf{p}) = \bar{\mathbf{v}}_{\text{wind}}(\mathbf{p}) + \mathbf{g}_{\text{blended}}, \quad (30)$$

where $\mathbf{g}_{\text{blended}}$ uses the same position-dependent weighting as the mean wind blending, ensuring smooth spatial transitions in both the mean and turbulent components across wind zone boundaries.

C. Episode Initialization

Two distinct training configurations are studied in this work, each targeting a different aspect of the autonomous landing problem. The first evaluates tracking and landing on a moving maritime platform without wind disturbances, while the second evaluates detection of and response to a localized wind disturbance encountered while approaching a stationary platform for landing.

In the wind-free configuration, the UAV initial position is sampled uniformly from $x \in [-5, 5]$ m, $y \in [-5, 5]$ m, $z \in [8, 10]$ m, and the landing platform origin from $x \in [-1.5, 1.5]$ m,

$y \in [-1.5, 1.5]$ m, $z \in [5, 6.5]$ m, with pairs exceeding a maximum UAV–target separation rejected and resampled. A new ship-motion trajectory, generated via the JONSWAP spectral model with independently randomized phases, ensures no two episodes share the same deck-motion realization even at identical sea states. Sea state is sampled uniformly from WMO sea states 1–5 (slight to medium), with no ambient or localized wind included.

For the wind-disturbance experiments, the landing platform remains stationary. The UAV and landing-pad positions are randomized within larger spawn bounds than in the wind-free configuration, subject to the constraint that the UAV is always initialized at a more negative x -coordinate than the landing pad.

A localized wind zone is subsequently positioned between the UAV and the landing platform along the nominal approach direction. The wind-zone center is defined as

$$x_c = t_x - R + \delta_x, \quad (31)$$

where t_x denotes the target landing-pad position, R is a prescribed wall offset, and δ_x is a random longitudinal displacement. The UAV is then spawned behind the upstream face of the wind zone, ensuring the spatial ordering of:

$$\text{UAV} \longrightarrow \text{Wind Zone} \longrightarrow \text{Landing Pad},$$

for every episode. This configuration provides a proof-of-concept scenario in which the aircraft must traverse a localized disturbance before reaching the landing platform.

To prevent the policy from overfitting to a fixed disturbance geometry, both the geometric dimensions and aerodynamic properties of the wind zone are randomized independently for every episode. The parameter ranges used during training are summarized in Table 6.

In both configurations, guided exploration is incorporated during the first N_{demo} episodes by initializing the UAV in close proximity to

Table 6: Wind-zone randomization ranges used during training.

Parameter	Range	Unit
Mean wind velocity (v_z)	$[-30, -5]$	m/s
Wall offset R	4	m
Wall half-width	$[1.0, 6.0]$	m
Wall height	$[3.0, 10.0]$	m
Wall thickness	4.0	m
Wall y -shift (δ_y)	$[-6.0, 6.0]$	m
Wall x -shift (δ_x)	$[-3.0, -1.0]$	m
Turbulence intensity	$[1.0, 1.0, 2.0]$	m/s
σ		
Turbulence length scale $\mathbf{L}$	$[10, 10, 5]$	m

the landing target every tenth episode, rather than at a randomized stand-off position. This periodic near-target initialization is evaluated for the facilitation of early discovery of the sparse terminal landing reward.

VI. Results

This section presents the evaluation results for the two guidance policies introduced previously. Part A reports results for the moving maritime target landing configuration, and Part B reports results for the stationary target configuration under localized wind disturbances encountered during the approach phase. In both cases, the trained RL policy is compared against the baseline guidance law in terms of landing performance.

A. Moving Maritime Target Landing (No Wind)

To assess the effect of guided exploration on training convergence, two configurations were compared: a standard random initialization and a guided-exploration variant in which, every tenth episode during the first 1000 training episodes, the UAV was initialized near the target to expose the agent to the sparse terminal landing reward early in training.

Guided exploration substantially accelerated learning and reduced its variability relative to the unguided configuration, as can be seen in Fig. 6. The guided configuration reached an 80% landing success rate after approximately 50k RL steps, compared with approximately 160k steps for the unguided configuration. Although the unguided configuration eventually exceeded an 80% success rate, learning remained oscillatory throughout training, whereas the guided configuration converged to a stable plateau in success rate.

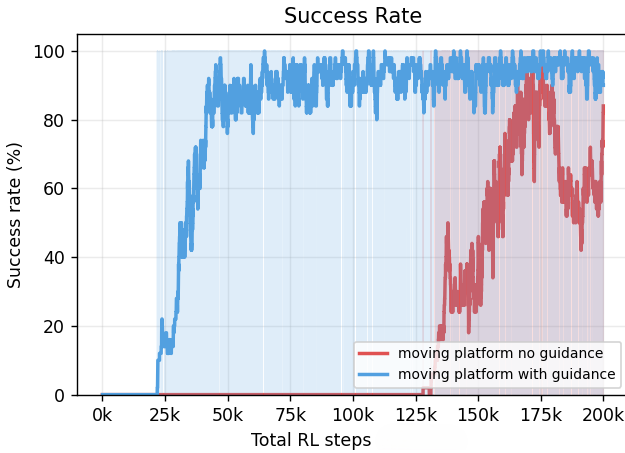


Figure 6: Landing success rate over RL training steps for the moving-platform landing task: with and without guided exploration.

For both configurations, proximity to the landing pad at episode termination converged similarly after approximately 25k–50k steps, indicating that target-tracking behavior was learned readily in either case. The key difference is that the guided policy is reliably committed to the landing maneuver, whereas the unguided policy instead learns to associate close-range proximity with elevated termination risk, a consequence of experiencing far more crashes than successful landings near the platform during early training. This local optimum, in which loitering near the target is favored over the terminal descent, accounts for the delayed emergence of successful landings. Guided exploration was therefore adopted for all subsequent trainings.

The trained guidance policy with guided ex-

ploration via initial-state randomization was evaluated over 100 test episodes under sea state 6, representing conditions beyond the maximum sea state encountered during training. Independent JONSWAP wave realizations were generated for each episode to evaluate both realization-level and severity-level generalization. The RL guidance policy achieved a 100% landing success rate, compared with 92% for the baseline controller. The corresponding landing performance metrics are summarized in Table 7.

Horizontal landing accuracy was largely preserved, with only a marginal increase in the mean XY landing error. The principal advantage of the RL guidance policy was a 67.9% reduction in landing time, reflecting a more direct landing maneuver rather than prolonged station keeping above the platform. This improvement was achieved without sacrificing touchdown quality, as the relative vertical touchdown velocity was reduced by 41.8% compared with the baseline. Furthermore, touchdown roll and pitch angles did not differ meaningfully between the two controllers relative to their standard deviations (approximately 2.5° – 3.1°), indicating that the improvements in landing speed and touchdown softness were not obtained at the expense of attitude stability or horizontal landing precision.

Figures 7 compare representative trajectories produced by the RL and baseline guidance policies for a matched test episode with identical initial conditions and wave realization. The RL policy takes a direct approach to the moving landing platform, whereas the baseline controller remains above the target longer before initiating its descent. In the final seconds before touchdown, the RL policy executes a brief flare, momentarily arresting its descent before a gradual final approach; the baseline exhibits no comparable pause.

B. Stationary Target Landing with Localized Wind Disturbance

Within this framework, the observation space is augmented with a wind-history formulation to provide the policy with explicit access to real-

Table 7: Comparison of baseline and RL guidance policy performance over 100 test episodes under sea state 6 (mean $\pm$ std).

Metric	Baseline	RL	Δ	Change (%)
Success Rate [%]	92	100	+8	+8 pp
XY Landing Error [m]	0.26 ± 0.12	0.28 ± 0.13	+0.02	+7.7%
Time to Land [s]	11.3 ± 3.4	3.64 ± 0.84	-7.66	-67.9%
Roll at Landing [deg]	0.0 ± 2.5	0.1 ± 3.1	+0.04	
Pitch at Landing [deg]	0.4 ± 2.8	0.0 ± 2.8	-0.42	
Relative V_z at Landing [m/s]	0.38 ± 0.11	0.22 ± 0.10	-0.16	-41.8%

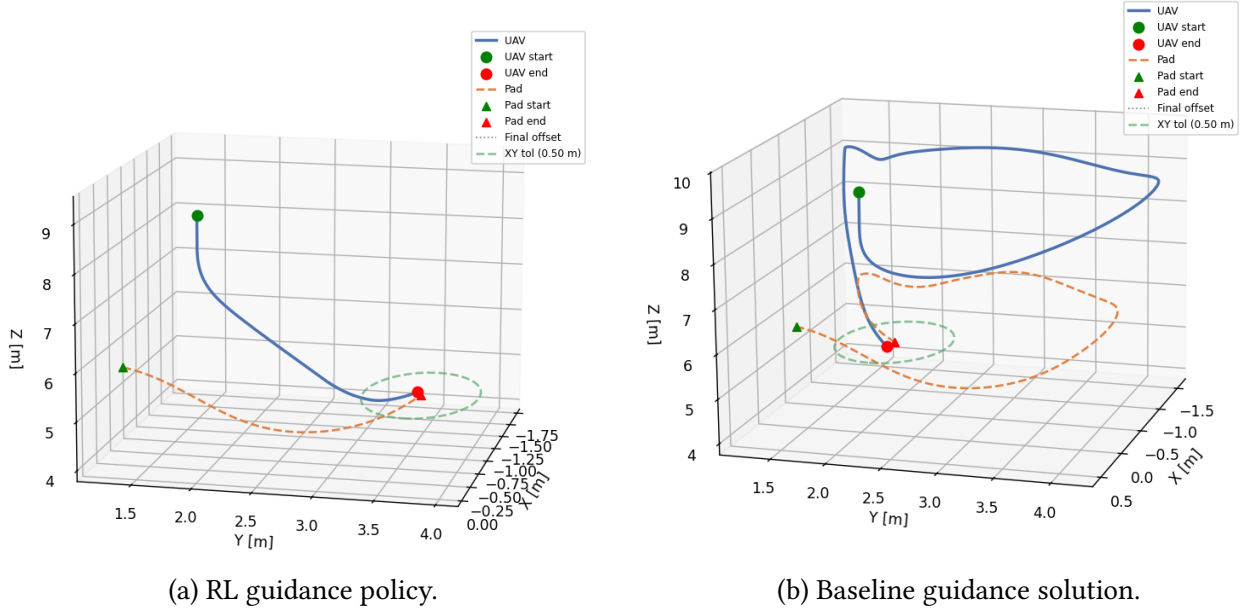


Figure 7: Comparison of three-dimensional UAV trajectories under the RL and baseline guidance solutions for the same representative test episode 100 under sea state 6.

time wind information. Specifically, the base observation space is extended with onboard IMU and rotor-speed measurements, together with a 10-step history of directly sensed body-frame wind velocity, resulting in $\mathbf{o}_t^{\text{full}} \in \mathbb{R}^{52}$. This formulation was selected among three candidates for encoding wind information – indirect inference from onboard states, wind-velocity history, and instantaneous wind-velocity gradient – as it consistently outperformed the alternatives; only the wind-history results are reported here for brevity. Guided exploration, consisting of near-target spawning every 10th episode during the first 1000 episodes, was applied throughout training following the findings presented in the

previous section.

As shown in Fig. 8, the policy converged to a success rate of 70–90% after approximately 50k RL steps, with the remaining oscillations attributable to large per-episode variation in wind-zone geometry and intensity rather than to training instability.

The trained policy is evaluated under randomized wind-wall geometry, with wall offset, half-width, height, and lateral/longitudinal position independently resampled in each episode. This prevents the disturbance region from remaining fixed relative to the nominal approach corridor, exposing the policy to partial, off-axis, or fully intercepting wind encounters at test time.

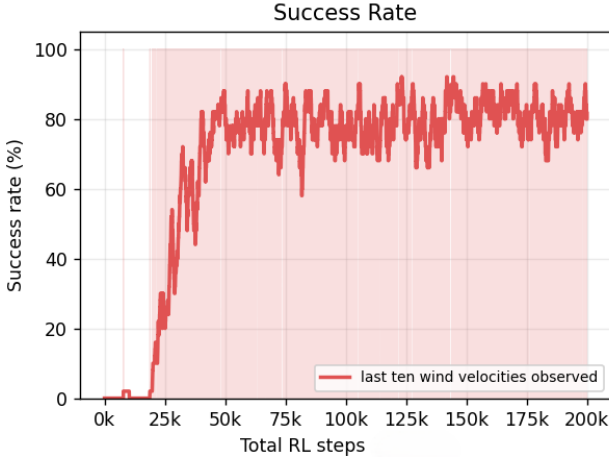


Figure 8: Training convergence of the RL guidance policy under localized wind disturbances: episode success rate.

Trajectory inspection shows that the RL policy combines real-time wind sensing with a learned prior over likely wind-zone locations from training, routing around or above the disturbance in most episodes. This anticipatory behavior manifests as a lateral circling maneuver early in the approach, initiated solely by the initial UAV-pad geometry, before any wind is sensed. In some cases, this prior works against the policy: it tends toward the wind location expected by the training distribution, only to encounter the true disturbance elsewhere, since the position is randomized at test time.

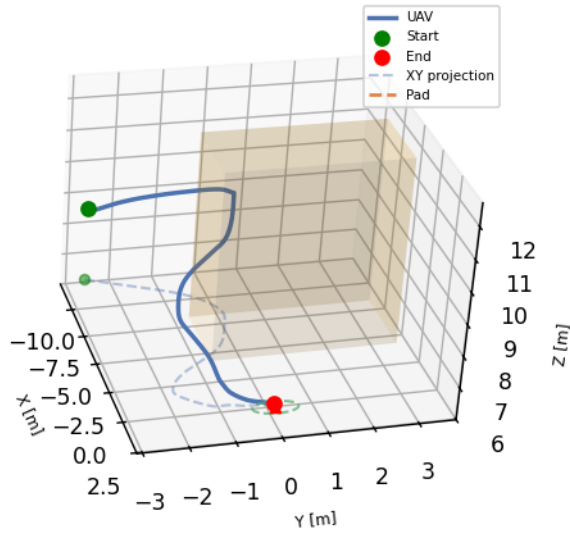
Critically, however, this anticipatory prior is not the only mechanism at play. In the majority of episodes, the policy is additionally observed to actively correct its trajectory in real time as it encounters the wind field, adjusting its path in direct response to onboard wind sensing rather than relying on the training-time prior alone. This reactive component is what ultimately allows the policy to reliably clear the disturbance even when its anticipatory maneuver alone would have been insufficient or misdirected.

To isolate these two behaviors, the same trained policy was evaluated twice under identical initial conditions: once with the wind disturbance active and once with it deactivated. Figure 9

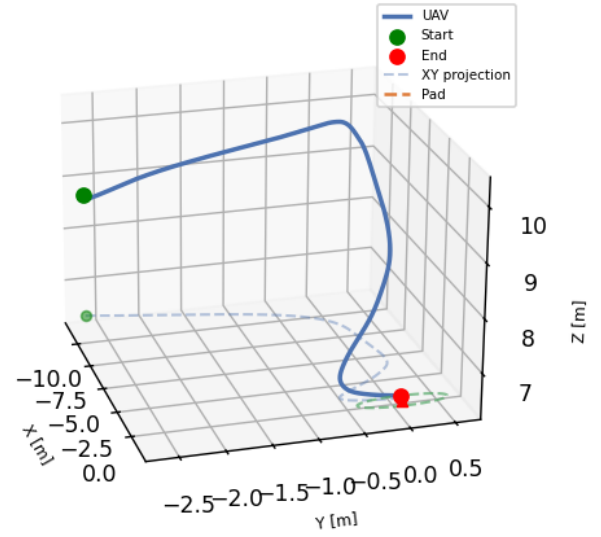
shows a representative episode with a 20 m/s downwash. With wind active, the UAV initiates an anticipatory evasive maneuver based solely on the initial UAV and pad positions, then later senses the wind field along its trajectory and makes a real-time adjustment to avoid it; with wind inactive, the anticipatory evasive circling remains, while the real-time adjustment component is absent. This confirms that the two behaviors are distinct: the anticipatory maneuver is driven by a learned prior over the initial UAV and pad geometry, while the subsequent trajectory correction is driven by the onboard wind observation.

Building on the introduced baseline policy, wind-avoidance metrics in Table 8 show that the RL policy trades path directness for reduced disturbance severity relative to the baseline, confirming the findings. It flies longer and less efficient trajectories (16.62 m versus 11.70 m; path efficiency 0.658 versus 0.819) with a larger detour (8.03 m versus 2.64 m) and greater peak lateral deviation (2.005 m versus 0.414 m), demonstrating active maneuvering rather than a straight-line approach. Despite the increased path length, the RL policy encounters the wind region less frequently (58 versus 78 exposed episodes), spends less time within the disturbance (0.292 s versus 0.472 s), and intercepts the wind wall at a larger, more oblique heading angle (23.79° versus 10.49°) in order to avoid the wind field. Peak vertical deviation remains comparable between the two configurations (2.430 m versus 2.228 m), indicating similar vertical disturbance rejection independent of the lateral avoidance strategy.

Overall, the RL policy has been shown to utilize both training experience to predict the downwash position and real-time adjustments, when feasible, to avoid the disturbance region. As this kind of awareness is not part of the baseline solution, the RL policy achieves a higher success rate over the randomized evaluation scenario.



(a) RL guidance trajectory with 20 m/s downwash wind active: anticipatory maneuver with real-time adjustment.



(b) RL guidance trajectory for the same episode and initial conditions with the wind disturbance deactivated: anticipatory maneuver only, no adjustment.

Figure 9: Comparison of RL guidance trajectories for the same episode and initial conditions with (a) the wind disturbance active and (b) the wind disturbance deactivated.

Table 8: Wind-avoidance metrics under active wind disturbance using identical wind-block realizations (mean $\pm$ std).

Metric	Baseline	RL
Success Rate [%]	24	66
Path Length [m]	11.7 ± 4.9	16.6 ± 5.0
Path Efficiency [-]	0.82 ± 0.12	0.66 ± 0.12
Detour [m]	2.6 ± 2.8	8.0 ± 3.8
Time in Wind [s]	0.47 ± 0.27	0.29 ± 0.57
Peak Lateral Deviation [m]	0.41 ± 0.45	2.0 ± 1.6
Peak Vertical Deviation [m]	2.23 ± 0.60	2.43 ± 1.02
Approach Heading vs. Wall [$^\circ$] (N)	10.5 ± 6.2 (78)	23.8 ± 9.2 (58)

VII. Conclusions and Recommendations

This work investigated RL as a guidance strategy for autonomous landing of a lift-and-cruise quadplane UAV on a moving maritime platform under localized wind disturbances. A quadplane dynamic model, including aerodynamic effects, was developed together with a cascaded PID tracking controller, against which two RL-based guidance policies were evaluated: one addressing moving-platform tracking and landing under

varying sea states, and one addressing approach and landing under a spatially localized wind disturbance.

The results show that RL is a viable method for developing guidance strategies for hybrid UAV platforms performing maritime landings. Compared to the baseline guidance law, the RL-based policy achieves better landing performance on a moving platform in sea states up to 7, with shorter touchdown times and softer touchdown velocities. Guided exploration via initial-state randomization was shown to substantially

accelerate convergence, addressing the sparse-reward, large-state-space setting characteristic of this guidance problem. When trained under localized wind exposure, the policy was found to combine anticipatory behavior, reflecting the wind field statistics observed during training, with reactive trajectory adjustments driven by real-time wind sensing.

These findings support the broader case for RL as a guidance-level alternative to heuristic control logic in maritime UAV operations. The policy’s ability to independently handle a moving landing platform and localized wind disturbances suggests that RL-based guidance could extend to scenarios involving compounding disturbances that are difficult to address through predefined guidance laws. More broadly, similar approaches may extend naturally to other hybrid platforms and operational environments involving moving targets and localized aerodynamic disturbances.

Several directions are recommended for future work. First, the sensitivity of the observed anticipatory-reactive balance in wind-avoidance behavior to the domain randomization strategy, reward function formulation, and observation space composition should be systematically characterized, with curriculum learning explored as a potential mechanism for guiding the agent towards more robust reactive behavior. Second, as the agent was found to become closely tuned to the position and characteristics of the single localized wind field employed during training, the utility of this anticipatory behavior in a maritime context should be evaluated using multiple localized wind fields arranged to represent a ship’s wake, with the number, position, and characteristics of these fields randomized to reflect the diversity of superstructure geometries that may be encountered in operation. Third, while moving-platform tracking and localized wind-disturbance rejection were addressed as separate challenges in this study, these problems could naturally be unified into a single guidance task; doing so may benefit from alternative network architectures, such as recurrent neural networks,

or from training strategies such as curriculum learning, to manage the increased complexity associated with the combined problem. Finally, independent of the RL guidance framework itself, further research should address the direct, omnidirectional wind measurement assumed in this work, which exceeds current practical UAV sensing capabilities, by investigating estimation and sensor-fusion techniques capable of providing usable wind information under realistic constraints.

REFERENCES

- Abo Mosali, N. (2022). An adaptive multi-level quantization-based reinforcement learning model for enhancing uav landing on moving targets. *Sustainability*, 14(14), 8825. <https://doi.org/10.3390/su14148825>
- Abujoub, S., McPhee, J., & Irani, R. A. (2020). Methodologies for landing autonomous aerial vehicles on maritime vessels. *Aerospace Science and Technology*, 106, 106169. <https://doi.org/10.1016/j.ast.2020.106169>
- Aird, P. (2019, January). Deepwater metocean environments. <https://doi.org/10.1016/B978-0-08-102282-5.00004-1>
- Amendola, J., Cenkeramaddi, L. R., & Jha, A. (2024). Drone landing and reinforcement learning: State-of-art, challenges and opportunities. *IEEE Open Journal of Intelligent Transportation Systems*, 5, 520–539. <https://doi.org/10.1109/OJITS.2024.3444487>
- Cárdenas R., C. A., Grisales, V. H., Collazos Morales, C. A., Cerón-Muñoz, H. D., Ariza-Colpas, P., & Caputo-Llanos, R. (2020). Quadrotor modeling and a pid control approach. In U. S. Tiwary & S. Chaudhury (Eds.), *Intelligent human computer interaction* (pp. 281–291). Springer International Publishing.
- Chen, C. C., Tsai, J., Peng-Chen, L., & Lai, C.-A. (2020). Accuracy improvement of autonomous straight take-off, flying forward, and landing of a drone with deep reinforcement learning. *International Journal of Computational Intelligence Systems*, 13(1), 914. <https://doi.org/10.2991/ijcis.d.200410.001>
- Ding, W., & Huang, H. (2023). Research on dynamic landing guidance algorithm for the maritime quadrotor. *Ocean Engineering*, 271, 113775. <https://doi.org/10.1016/j.oceaneng.2023.113775>
- Eschmann, J., Albani, D., & Loianno, G. (2024). Data-driven system identification of quadrotors subject to motor delays. *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 8095–8102. <https://doi.org/10.1109/IROS58592.2024.10801441>
- Feng, Y., Zhang, C., Baek, S., Rawashdeh, S., & Mohammadi, A. (2018). Autonomous landing of a uav on a moving platform using model predictive control. *Drones*, 2(4), 34. <https://doi.org/10.3390/drones2040034>
- Fujimoto, S., van Hoof, H., & Meger, D. (2018). Addressing function approximation error in actor-critic methods. <https://arxiv.org/abs/1802.09477>
- Gao, Y., Li, Q., & Wang, H. (2023). Model predictive control-based trajectory generation for agile landing of unmanned aerial vehicle on a moving boat. *IEEE Transactions on Control Systems Technology*, 31(4), 1550–1563. <https://doi.org/10.1109/TCST.2023.3245678>
- Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 80, 1861–1870.
- Hasselmann, K., Barnett, T. P., Bouws, E., Carlson, H., Cartwright, D. E., Enke, K., Ewing, J. A., Gienapp, H., Hasselmann, D. E., Kruseman, P., Meerburg, A., Müller, P., Olbers, D. J., Richter, K., Sell, W., & Walden, H. (1973). Measurements of wind-wave growth and swell decay during the joint north sea wave project (jonswap) [Joint North Sea Wave Project (JONSWAP) experiment]. *Deutsche Hydrographische Zeitschrift, Ergänzungsheft*, 8(12), 1–95. <http://resolver.tudelft.nl/uuid:f204e188-13b9-49d8-a6dc-4fb7c20562fc>

- Ho, D., Linder, J., Hendeby, G., & Enqvist, M. (2017). Mass estimation of a quadcopter using imu data. *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, 1260–1266. <https://doi.org/10.1109/ICUAS.2017.7991417>
- Ibinabo, I., & Tamunodukobipi, D. T. (2019). Determination of the response amplitude operator(s) of an fpso. *Engineering*, 11(9), 563–578. <https://doi.org/10.4236/eng.2019.119038>
- Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4, 237–285. <https://jair.org/index.php/jair/article/view/10166>
- Keipour, A., Pereira, G. A., Bonatti, R., Garg, R., Rastogi, P., Dubey, G., & Scherer, S. (2022). Visual servoing approach for autonomous uav landing on a moving vehicle. *Autonomous Robots*, 46(8), 1315–1332. <https://doi.org/10.1007/s10514-022-10250-7>
- Kim, J. W., McIntosh, K., Shrestha, E., Reddinger, J.-P., & Mishra, S. (2023). A reinforcement learning approach to control of a quadrotor biplane tailsitter for adaptive landing maneuvers. *Proceedings of the Vertical Flight Society 79th Annual Forum & Technology Display*, Paper 1208. <https://doi.org/10.4050/F-0079-2023-18173>
- Kroo, I., Prinz, F. B., Shantz, M., Kunz, P., Fay, G., Cheng, S., Fabian, T., & Partridge, C. (2000). *The mesicopter: A miniature rotorcraft concept phase ii interim report* (Technical Report) (NIAC Phase II study on miniature rotorcraft dynamics and design). Stanford University. Palo Alto, CA.
- Ma, Q., Wu, Y., Shoukat, M. U., Yan, Y., Wang, J., Yang, L., Yan, F., & Yan, L. (2024). Deep reinforcement learning-based wind disturbance rejection control strategy for uav. *Drones*, 8(11), 632. <https://doi.org/10.3390/drones8110632>
- Ma, Z., Smeur, E. J. J., & de Croon, G. C. H. E. (2022). Wind tunnel tests of a wing at all angles of attack. *International Journal of Micro Air Vehicles*, 14. <https://doi.org/10.1177/17568293221110931>
- Messmer, M., Kiefer, B., Varga, L. A., & Zell, A. (2024). Uav-assisted maritime search and rescue: A holistic approach [Comprehensive framework for UAV search and rescue missions]. *arXiv preprint arXiv:2403.14281*.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529–533. <https://doi.org/10.1038/nature14236>
- Olivares, D., Fournier, P., Vasishta, P., & Marzat, J. (2024). Model-free versus model-based reinforcement learning for fixed-wing uav attitude control under varying wind conditions. *Proceedings of the 21st International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 1, 79–91. <https://doi.org/10.5220/0012946600003822>
- Quan, Q., Wang, S., & Gao, W. (2025). Lifting-wing quadcopter modeling and unified control [Published online]. *Journal of Guidance, Control, and Dynamics*. <https://doi.org/10.2514/1.G008134>
- Saj, V., Lee, B., Kalathil, D., & Benedict, M. (2022). Robust reinforcement learning algorithm for vision-based ship landing of uavs. *arXiv preprint arXiv:2209.08381*. <https://arxiv.org/abs/2209.08381>
- Setiawan, H., Zheng, K., Kevin, & Monty, J. P. (2025). A probabilistic framework for assessing helicopter landing conditions within the ship air-wake. *Ocean Engineering*, 317, 120027. <https://doi.org/10.1016/j.oceaneng.2024.120027>

- Smeur, E. J., Chu, Q., & de Croon, G. C. (2016). Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles. *Journal of Guidance, Control, and Dynamics*, 39(3), 450–461. <https://doi.org/10.2514/1.G001490>
- Söker, H., Rehfeldt, K., Santjer, F., Strack, M., & Schreiber, M. (2000, October). *Offshore wind energy in the north sea: Technical possibilities and ecological considerations – a study for greenpeace* (Technical Report). Deutsches Windenergie-Institut GmbH and Schreiber Umweltplanung. https://www.offshorewindenergy.org/COD/reports/report-files/report_ort_027.pdf
- Stevens, B. L., & Lewis, F. L. (2003). *Aircraft control and simulation* (2nd ed.). John Wiley & Sons.
- Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd). MIT Press. <http://incompleteideas.net/book/the-book-2nd.html>
- Tan, L., Wu, J., Yang, X., & Song, S. (2025). Optimal landing trajectory planning method between a uav and a moving vessel. *Journal of Intelligent Robotic Systems*, 99, 123–137. <https://doi.org/10.1007/s10846-025-02288-3>
- Wang, C., Wang, J., Wei, C., Zhu, Y., Yin, D., & Li, J. (2023). Vision-based deep reinforcement learning of uav–ugv collaborative landing policy using automatic curriculum. *Drones*, 7(11), 676. <https://doi.org/10.3390/drones7110676>
- Williams, H. (2021). Jonswap waves for realistic ocean wave modelling [School of Engineering, University of Warwick]. https://www.researchgate.net/publication/354293847_JONS_WAP_waves_for_realistic_ocean_wave_modelling
- Wu, L., Wang, C., Zhang, P., & Wei, C. (2022). Deep reinforcement learning with corrective feedback for autonomous uav landing on a mobile platform. *Drones*, 6(9), 238. <https://doi.org/10.3390/drones6090238>
- Xie, J., Peng, X., Wang, H., Niu, W., & Zheng, X. (2020). Uav autonomous tracking and landing based on deep reinforcement learning strategy. *Sensors*, 20(19), 5630. <https://doi.org/10.3390/s20195630>
- Xiong, H., Li, L., Zeng, W., Li, D., Liu, Y., Raps, F., Lou, Y., & Noack, B. R. (2025). Trajectory planning for drone landing, incorporating wind-sensing capabilities, operational and safety objectives, and reinforcement learning. *Communications Engineering*, 4, 199. <https://doi.org/10.1038/s44172-025-00531-1>
- Xu, M., Deng, C., Hu, X., Lu, Y., Xue, W., & Zhu, B. (2025). Uav inspection path optimization in offshore wind farms using the option-a*-dqn algorithm. *PLoS ONE*, 20(11), e0336935.
- Yıkılmaz, C., & Wager, C. D. (2025). Reinforcement learning-based optimal guidance for landing the variable skew quad plane on a ship [Paper no. IMAV2025-7]. *16th International Micro Air Vehicle Conference and Competition*, 59–68. <http://www.ima-vs.org/papers/2025/7.pdf>
- Zhao, C., Liu, R. W., Qu, J., & Gao, R. (2023). Deep learning-based object detection in maritime unmanned aerial vehicle imagery: Review and experimental comparisons [Review of UAV object detection and surveillance in maritime environments]. *arXiv preprint arXiv:2311.07955*.

Part III:

Extended Methodology and Evaluation

5 EXTENDED METHODOLOGY

This section presents the full methodology underlying the work reported in the accompanying paper. While the paper provides a condensed account of the key design choices and results, space constraints preclude a complete treatment of all modeling assumptions, controller design decisions, and implementation details. The present section addresses this by providing extended documentation of three interconnected components: the UAV dynamics model, the classical PID controller and its tuning procedure, and the RL system design. Together, these form the complete technical foundation upon which the simulation environment and landing experiments are built.

5.1 Quadplane Dynamic Model

This section presents the dynamic model of the lift-and-cruise quadplane considered in this study. The vehicle configuration is based on the `standard_vtol` model from the PX4-Autopilot Gazebo simulation package, featuring four upward-facing rotors and a single rear pusher propeller.

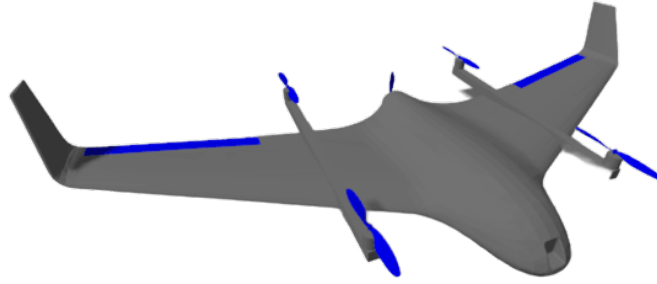


Figure 14: Gazebo compatible platform geometry model of a lift & cruise aircraft: the `standard_vtol` from the PX4-Autopilot repository.²⁰

A conventional lift-and-cruise vehicle operates in two primary flight regimes: hover and forward cruise, each dominated by distinct physical mechanisms and control allocation strategies. In hover, the dynamics are primarily thrust-driven, with control authority provided by the vertical lift rotors. In forward cruise, aerodynamic forces dominate, with lift generated mainly by the wings and control moments produced through aerodynamic control surfaces. Between these regimes lies a transition phase in which control authority is shared between the rotors and aerodynamic surfaces, and their relative contributions vary continuously with airspeed. In practical implementations, this transition is typically managed through blending strategies that interpolate between hover and cruise control laws to ensure smooth and stable performance across the flight envelope. (Pfeifle & Fichter, 2022; Saeed et al., 2018)

For the modeling purposes of this thesis, the vehicle is assumed to operate in the hover-dominated regime during landing, where aerodynamic control surfaces do not contribute to the control task. Although differences in inertia distribution and aerodynamic forces between a quadplane and a conventional quadrotor are accounted for, the low-speed landing dynamics are assumed to be largely comparable to those of a quadrotor of similar scale. This assumption forms the basis for the derivation of the equations of motion in the subsequent sections.

²⁰https://docs.px4.io/main/en/sim_gazebo_gz/vehicles#standard-vtol

Reference Frames

A quadplane UAV typically has two main reference frames. The inertial frame ($\mathcal{F}_I$) is an earth-fixed reference frame where the axes are denoted x_I, y_I, z_I . The body frame ($\mathcal{F}_B$) is fixed to the UAV's center of mass and moves along with the vehicle, with x_b pointing forward, y_b pointing to the left, and z_b pointing up away from the Earth. While these conventions apply to the quadplane, the orientation of the axes is visualized here for a quadrotor in Figure 15.

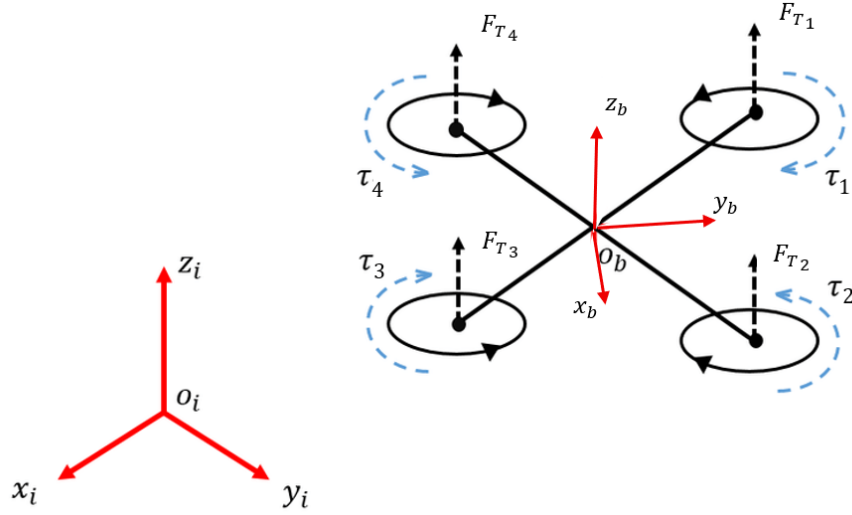


Figure 15: Free-body diagram of an X4 configuration quadrotor UAV illustrating the inertial earth-fixed and body-fixed coordinate frames. Adapted from Ho et al., 2017.

Based on these reference frames, the orientation of the quadplane can be described using Euler angles: roll (ϕ), pitch (θ), and yaw (ψ). Accordingly, the rotation matrix R_B^I transforms vectors from the body frame to the world frame using these Euler angles, formulated in Equation 26.

$$R_B^I = R_z(\psi)R_y(\theta)R_x(\phi) = \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ s_\psi c_\theta & s_\psi s_\theta s_\phi + c_\psi c_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \quad (26)$$

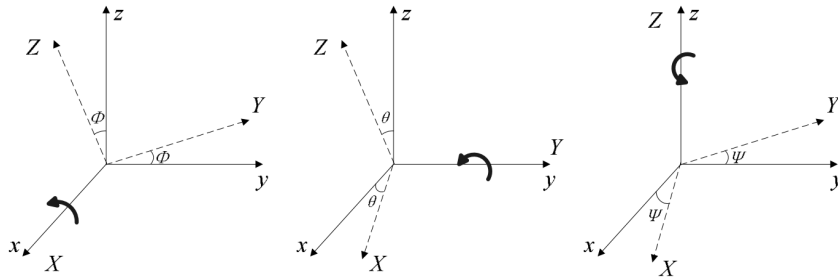


Figure 16: Euler rotation sequence from the UAV body-fixed frame (x, y, z) to the inertial frame (X, Y, Z), showing the yaw (ψ), pitch (θ), and roll (ϕ) angles. Retrieved from Li and Li (2011).

In addition to the body-fixed and inertial frames, the aerodynamic reference frame ($\mathcal{F}_A$) is commonly used to describe the aerodynamic forces and moments acting on the UAV. This

frame is typically aligned with the relative airflow. The x_a -axis points in the direction of the incoming air, the z_a -axis is perpendicular to x_a in the plane of symmetry, pointing downward. Aerodynamic forces such as lift, drag, and side force, as well as moments generated by control surfaces, are naturally expressed in this frame. The orientation of the aerodynamic frame relative to the body frame is determined by the angle of attack α and the sideslip angle β , which are defined as the angles between the body axes and the relative airflow.

There also exists the stability reference frame ($\mathcal{F}_S$), whose x_s -axis is aligned with the projection of the relative wind onto the body x - z plane (i.e., along the direction of flight velocity in the plane of symmetry). By construction, the stability frame differs from the aerodynamic frame only in the orientation of the lateral axis: the aerodynamic frame's y_a -axis points perpendicular to the relative wind, whereas the stability frame's y_s -axis lies in the plane of symmetry. In this frame, longitudinal forces and moments (e.g., lift, drag, pitching moment) are decoupled from lateral-directional effects.

The aerodynamic and stability frames are visualized in relation to the body-fixed and inertial frames in Figure 17. The figure follows the conventional aircraft convention, in which the positive z -axis points downward and the positive y -axis points to the right. This thesis instead adopts a convention in which the positive z -axis points upward and the positive y -axis points to the left, equivalent to a 180° rotation about the x -axis relative to the frames shown in the figure. The definitions of the aerodynamic and stability frames remain otherwise unchanged.

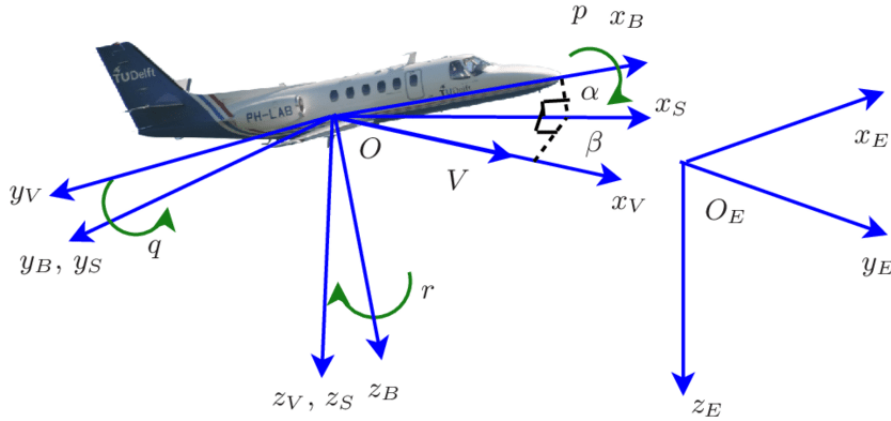


Figure 17: Body, stability, aerodynamic, and inertial reference frames for a generic aircraft, shown in the conventional right-handed aircraft convention (x forward, y right, z down). This thesis instead uses a right-handed convention with z pointing upward and y pointing left. Retrieved from Lu et al. (2016).

To go from the aerodynamic frame to the body frame, the rotation matrix $\mathbf{R}_A^B$ in Equation 27 is defined as

$$\mathbf{R}_A^B = \begin{bmatrix} \cos \alpha \cos \beta & \sin \beta & -\sin \alpha \cos \beta \\ -\cos \alpha \sin \beta & \cos \beta & \sin \alpha \sin \beta \\ \sin \alpha & 0 & \cos \alpha \end{bmatrix}. \quad (27)$$

To go from the stability frame to the body frame, the rotation matrix $\mathbf{R}_S^B$ in Equation 28 is defined as

$$\mathbf{R}_S^B = \begin{bmatrix} \cos \alpha & 0 & -\sin \alpha \\ 0 & 1 & 0 \\ \sin \alpha & 0 & \cos \alpha \end{bmatrix}. \quad (28)$$

Kinematics and Dynamics

A quadplane exhibits six degrees of freedom (6-DoF), this includes translational motion along the inertial x_I , y_I , and z_I axes, and rotational motion about the body-fixed axes, namely roll (ϕ), pitch (θ), and yaw (ψ). In this section, the kinematic and dynamic equations for translation and rotation are defined.

This thesis expresses translational motion in the inertial frame, while rotational dynamics are described in the body frame. This choice aligns with standard state estimation frameworks and control architectures (Quan et al., 2025).

Let $\mathbf{p} = [x_I \ y_I \ z_I]^T$ denote the position of the quadplane's center of mass expressed in the inertial frame $\mathcal{F}_I$. The vehicle's inertial velocity is defined as the time derivative of its position vector, and its inertial acceleration is the time derivative of its velocity, as expressed in Equation 29.

$$\dot{\mathbf{p}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}, \quad \ddot{\mathbf{p}} = \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} \quad (29)$$

Applying Newton's second law in the inertial frame $\mathcal{F}_I$, the translational dynamics of the quadplane's center of mass are governed by Equation 30.

$$m\ddot{\mathbf{p}} = \sum \mathbf{F}_I \quad (30)$$

where m denotes the mass of the vehicle and $\sum \mathbf{F}_I$ represents the sum of all external forces acting on the vehicle, expressed in the inertial frame.

A different approach is adopted for deriving the rotational kinematics and dynamics. The rotational kinematics describe the relationship between the body-frame angular velocity and the time derivatives of the chosen attitude parameterization. The UAV's angular velocity vector $\boldsymbol{\omega} = [p \ q \ r]^T$ is expressed in the body frame.

The vehicle orientation is parameterized using unit quaternions, defined as $\mathbf{q} = [q_0 \ q_1 \ q_2 \ q_3]^T$, where q_0 is the scalar part and $[q_1 \ q_2 \ q_3]^T$ is the vector part, subject to the unit-norm constraint $\|\mathbf{q}\| = 1$. Quaternions are preferred over Euler angles for numerical integration because they are globally non-singular, avoiding the gimbal lock singularity that arises in Euler representations at $\theta = \pm 90^\circ$, and are computationally efficient to propagate (Quan et al., 2025). The rotation matrix $\mathbf{R} \in SO(3)$, which maps vectors from the body frame $\mathcal{F}_B$ to the inertial frame $\mathcal{F}_I$, is computed directly from the quaternion at each timestep and is used for all force and torque projections in both the dynamics and the control loops.

The quaternion kinematics are governed by:

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\Lambda}(\boldsymbol{\omega}) \mathbf{q}, \quad (31)$$

where $\boldsymbol{\Lambda}(\boldsymbol{\omega})$ is the skew-symmetric matrix formed from the body angular velocity:

$$\Lambda(\boldsymbol{\omega}) = \begin{bmatrix} 0 & r & -q & p \\ -r & 0 & p & q \\ q & -p & 0 & r \\ -p & -q & -r & 0 \end{bmatrix}. \quad (32)$$

In discrete time, Equation 31 is propagated using the matrix exponential, which preserves the unit-norm constraint exactly without requiring renormalization at each step:

$$\mathbf{q}(t + \Delta t) = \left[\cos\left(\frac{\|\boldsymbol{\omega}\| \Delta t}{2}\right) \mathbf{I} + \frac{1}{\|\boldsymbol{\omega}\|} \Lambda(\boldsymbol{\omega}) \sin\left(\frac{\|\boldsymbol{\omega}\| \Delta t}{2}\right) \right] \mathbf{q}(t). \quad (33)$$

Although quaternions are used for integration, Euler angles (ϕ, θ, ψ) are recovered from the quaternion as a secondary output for visualization and controller interfacing. Since the Euler angle rates are not equal to the body angular velocity components, a nonlinear transformation is required. This relationship is given by the Euler rate transformation matrix in Equation 34, which maps the body-frame angular velocity to the corresponding Euler angle rates.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin \phi \tan \theta & \cos \phi \tan \theta \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi \sec \theta & \cos \phi \sec \theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \quad (34)$$

For deriving the rotational dynamics, Euler's rigid-body equation is used, as expressed in Equation 35.

$$\boldsymbol{\tau}_B = \frac{d\mathbf{H}_B}{dt} + \boldsymbol{\omega}_B \times \mathbf{H}_B, \quad (35)$$

where $\boldsymbol{\tau}_B$ denotes the torque vector expressed in the body frame, $\mathbf{H}_B$ is the angular momentum in the body frame, and $\boldsymbol{\omega}_B$ is the angular velocity of the body frame. The cross-product term arises due to the rotation of the body-fixed reference frame relative to the inertial frame.

Assuming that the body-fixed axes are aligned with the principal axes of inertia, the angular momentum simplifies to Equation 36.

$$\mathbf{H}_B = \mathbf{J}\boldsymbol{\omega}_B = \begin{bmatrix} J_{xx} & J_{xy} & J_{xz} \\ J_{yx} & J_{yy} & J_{yz} \\ J_{zx} & J_{zy} & J_{zz} \end{bmatrix} \boldsymbol{\omega}_B \approx \begin{bmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \quad (36)$$

Here, the approximation corresponds to neglecting the off-diagonal (coupling) terms of the inertia matrix. This simplification is justified for vehicles that are symmetric about their principal axes. Within the scope of this thesis, the lift-and-cruise quadplane is assumed to be symmetric along all three body axes, allowing the inertia matrix to be represented in diagonal form and significantly simplifying the rotational dynamics.

Since the inertia matrix $\mathbf{J}$ is constant in the body frame, the time derivative of the angular momentum when substituted into Equation 35 yields:

$$\boldsymbol{\tau}_B = \mathbf{J}\dot{\boldsymbol{\omega}}_B + \boldsymbol{\omega}_B \times (\mathbf{J}\boldsymbol{\omega}_B). \quad (37)$$

Rearranging the resulting expression, substituting the expanded cross-product term, and normalizing by the corresponding principal moments of inertia yields Equation 38.

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{J_y - J_z}{J_x} qr \\ \frac{J_z - J_x}{J_y} pr \\ \frac{J_x - J_y}{J_z} pq \end{bmatrix} + \begin{bmatrix} \frac{1}{J_x} \tau_\phi \\ \frac{1}{J_y} \tau_\theta \\ \frac{1}{J_z} \tau_\psi \end{bmatrix}. \quad (38)$$

Forces and Moments

In this analysis, the vehicle is assumed to operate in a hover-dominated flight regime, where its dynamics closely resemble those of a conventional quadcopter. Control authority is primarily provided by the four upward-facing rotors, while conventional aerodynamic control surfaces are assumed to be inactive.

Nevertheless, even in hover, the presence of the wing induces non-negligible aerodynamic forces under ambient wind or gust conditions. These effects are therefore explicitly accounted for in the dynamic model.

The relative air velocity in the inertial frame is defined as

$$\mathbf{V}_{\text{rel}}^I = \mathbf{V}_{\text{UAV}}^I - \mathbf{V}_{\text{wind}}^I, \quad (39)$$

and is transformed into the body frame via the rotation matrix $\mathbf{R}_I^B$:

$$\mathbf{V}_{\text{rel}}^B = \mathbf{R}_I^B \mathbf{V}_{\text{rel}}^I = [u, v, w]^\top. \quad (40)$$

The aerodynamic angles are computed from the body-frame velocity components. The angle of attack α and sideslip angle β are defined as

$$\alpha = \arctan\left(\frac{w}{u}\right), \quad \beta = \arcsin\left(\frac{v}{V}\right), \quad (41)$$

where $V = \|\mathbf{V}_{\text{rel}}^B\|$ is the total airspeed and $V_{xz} = \sqrt{u^2 + w^2}$ is the airspeed projected onto the pitch plane.

To avoid numerical singularities at low airspeed, aerodynamic forces are only evaluated when

$$V \geq V_{\text{min}}, \quad (42)$$

with threshold set to 1.0 m/s.

The dynamic pressure is given by

$$\bar{q} = \frac{1}{2} \rho V^2, \quad (43)$$

where ρ is the air density. The lift and drag forces in the pitch plane are

$$L = \bar{q} A_{\text{ref}} C_L(\alpha), \quad D = \bar{q} A_{\text{ref}} C_D(\alpha), \quad (44)$$

with A_{ref} the reference wing area.

While aerodynamic lift and drag models for small angles of attack have been studied extensively, VTOL platforms operate over a much wider flight envelope, experiencing angles of attack ranging from -180° to 180° during vertical take-off, landing, and hover. Consequently, the aerodynamic coefficients $C_L(\alpha)$ and $C_D(\alpha)$ are modeled using a blended formulation that smoothly transitions between attached-flow behavior at low angles of attack and separated-flow behavior at high angles of attack. This approach captures the post-stall aerodynamic characteristics while providing a continuous representation of the aerodynamic forces across the entire flight envelope.

The blending is governed by a pseudo-sigmoid function

$$\sigma(k, \alpha_0, \alpha) = \frac{1 + \tanh(k(\alpha_0^2 - \alpha^2))}{1 + \tanh(k \alpha_0^2)}, \quad (45)$$

which transitions from unity in attached flow ($|\alpha| \ll \alpha_0$) to zero in the separated regime ($|\alpha| \gg \alpha_0$).

The lift coefficient is defined using Equation 46,

$$C_L(\alpha) = \sigma_L (C_{L,0} + A_L \sin \alpha) + (1 - \sigma_L) a_1 \sin 2\alpha, \quad (46)$$

where $C_{L,0}$ accounts for camber-induced lift at zero incidence, A_L is the small-angle lift slope, and a_1 governs the high-angle behavior.

The drag coefficient is modeled as Equation 47,

$$C_D(\alpha) = \sigma_D \left[a_0 + \frac{a_2 a_3}{(a_2 - a_3) \cos^2 \alpha + a_3} \sin^2 \alpha \right] + (1 - \sigma_D) [a_0 + 2a_{1,D} \sin^2 \alpha], \quad (47)$$

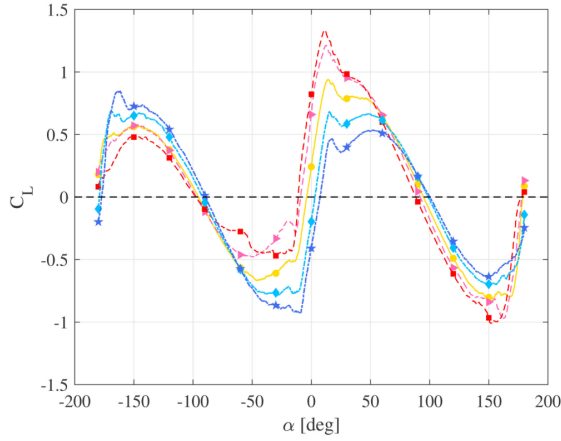
where a_0 is the parasitic drag offset, $a_{1,D}$ governs the high-angle behavior, and a_2 and a_3 primarily shape the low-angle behavior. Independent blending parameters of 15 degrees and 3 degrees are used for lift and drag, respectively.

The parameters of the lift and drag coefficient models are tuned using the wind tunnel measurements reported by Z. Ma et al. (2022). In their study, the authors conducted a comprehensive experimental campaign to characterize the aerodynamic behavior of a tailsitter aircraft over the full angle-of-attack range. Measurements were performed for various flap deflections and throttle settings to capture the influence of control surface deflections and propeller slipstream on the aerodynamic forces. To obtain the baseline aerodynamic characteristics of the airframe, the authors also tested the bare wing configuration, with the control surfaces neutral and the propellers removed. The measurements from this configuration are used to tune the aerodynamic model presented in this work, as they provide representative lift and drag characteristics of the wing independent of propulsion and control surface effects.

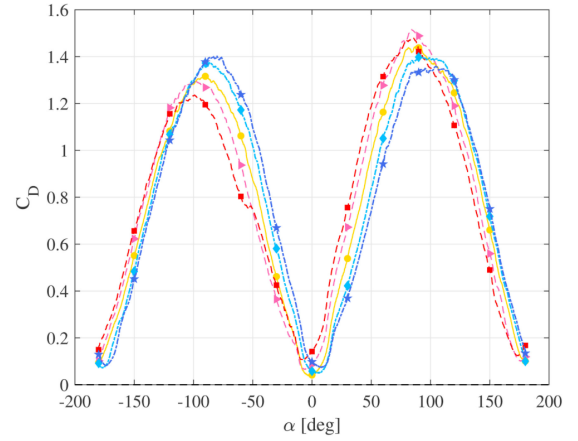
In Figure 18, the wind tunnel results from Z. Ma et al. (2022) are shown for the bare wing configuration, i.e., without the propulsion system attached, under different flap deflections. The neutral flap position (in yellow) is used in this study as the reference configuration for parameter identification.

Besides lift and drag, a lateral aerodynamic force due to sideslip is included using a smooth approximation,

$$Y = -\tanh\left(\frac{v}{\varepsilon}\right) \bar{q} A_{ref} C_Y(\beta), \quad (48)$$



(a) Lift coefficient $C_L(\alpha)$



(b) Drag coefficient $C_D(\alpha)$

Figure 18: Wind tunnel measurements from Z. Ma et al. (2022) for the bare wing configuration under different flap deflections. The neutral flap setting (yellow curve) is used as the reference case for aerodynamic parameter tuning in this study.

where A_{ref} is the reference wind area and

$$C_Y(\beta) = 0.4 C_D(\beta). \quad (49)$$

The lateral force is assumed to scale with the drag characteristics of the airframe, since sideslip primarily induces crossflow separation and pressure drag rather than lift-dominated effects. The aerodynamic information captured by the drag model is therefore reused to represent the lateral aerodynamic response in the body y -direction.

The $\tanh(\cdot)$ term ensures that the sign of the lateral force transitions smoothly with the direction of the lateral velocity, avoiding the discontinuity that would otherwise arise from directly mapping sideslip direction to a signed force near zero lateral velocity. The parameter

$$\varepsilon = 0.05 \text{ m/s},$$

sets the width of this transition region, suppressing numerical chattering at low sideslip while preserving the correct asymptotic behaviour at larger lateral velocities.

The following table summarizes all parameters used in the aerodynamic lift and drag models defined in Equation 46 and Equation 47, as well as the associated vehicle and environmental parameters required for the aerodynamic force computation.

The aerodynamic force vector in the aerodynamic frame is assembled as:

$$\mathbf{F}_{\text{aero}}^A = \begin{bmatrix} -D \\ Y \\ L \end{bmatrix}, \quad (50)$$

which is then rotated into the inertial frame:

$$\mathbf{F}_{\text{aero}}^I = \mathbf{R}_B^I \mathbf{R}_A^B \mathbf{F}_{\text{aero}}^B. \quad (51)$$

Table 11: Summary of tuning parameters used in the lift and drag coefficient formulas and force computation.

Symbol	Value	Description
ρ	1.225 kg/m ³	Air density
A_{ref}	0.6 m ²	Reference wing area
a_0	0.055	Zero-lift drag offset
a_1	0.700	High-angle lift scale
$a_{1,D}$	0.623	High-angle drag scale
a_2	13.0	Drag model shape parameter
a_3	3.3	Drag model shape parameter
C_{L0}	0.2	Lift at zero angle of attack
α_0	15°	Lift blending threshold
$\alpha_{0,D}$	3°	Drag blending threshold
k_L	38	Lift blending sharpness
k_D	48	Drag blending sharpness
ε	0.05 m/s	Sideslip smoothing parameter

Effects such as hub forces, blade flapping, and ground effect are neglected, following Kroo et al. (2000), who show these contributions to be at least an order of magnitude smaller than rotor thrust and drag below 10 m/s. Aerodynamic moments are also omitted in this proof-of-concept model, as the focus is on evaluating the reinforcement learning agent’s ability to infer wind disturbances, where force effects dominate at the considered level of fidelity.

For hover-dominated flight, the quadplane’s propulsion system provides vertical lift through the multicopter rotors, producing forces aligned with the body-fixed z_b -axis. The corresponding propulsive force vector is formulated as Equation 52. This vector is also transformed to the inertial frame through the rotation matrix $\mathbf{R}_B^I$.

$$\mathbf{F}_{prop}^B = \begin{bmatrix} 0 \\ 0 \\ F_T \end{bmatrix}, \quad (52)$$

where the total thrust magnitude F_T is defined as

$$F_T = \sum_{i=1}^{N_r} F_{T,i}, \quad (53)$$

with N_r denoting the number of lift rotors and $F_{T,i}$ the thrust generated by rotor i . The thrust produced by each rotor is approximated using the quadratic propeller relation

$$F_{T,i} = K_T n_i^2, \quad (54)$$

where n_i is the rotational speed of rotor i and K_T is the thrust coefficient, which depends on the propeller and motor configuration and is typically identified experimentally for a specific vehicle setup (Mathur & Atkins, 2023).

In addition to aerodynamic and propulsive forces, gravitational forces act in the inertial frame $\mathcal{F}_I$, as formulated in Equation 55.

$$\mathbf{F}_{grav}^I = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}. \quad (55)$$

Substituting the total force acting on the quadplane in the inertial frame $\mathcal{F}_I$ into Newton's second law gives the translational equations of motion as Equation 56.

$$m \dot{\mathbf{v}}_I = \mathbf{R}_B^I (\mathbf{F}_{prop}^B + \mathbf{F}_{aero}^B) + \mathbf{F}_{grav}^I, \quad (56)$$

Having established the forces governing the translational dynamics, the rotational equations of motion are now considered. The attitude of the quadplane, specifically roll (ϕ) and pitch (θ), is generated by differential thrust moments about the body x_b and y_b axes. These moments arise because each motor produces a thrust along the body z_b axis at a location offset from the vehicle's center of mass by a moment arm of length l , as shown in Figure 15.]

The attitude of the quadrotor, specifically roll (ϕ) and pitch (θ), is generated by rotational moments about the body x_b and y_b axes. These moments arise because each motor produces a thrust along the body z_b axis at a location offset from the vehicle's center of mass at a distance of l as shown by Figure 15. The roll and pitch moments are then given by Equation 57 and Equation 58, respectively.

$$\tau_\phi = \frac{l}{\sqrt{2}} (F_1 + F_2 - F_3 - F_4), \quad (57)$$

$$\tau_\theta = \frac{l}{\sqrt{2}} (F_1 - F_2 - F_3 + F_4), \quad (58)$$

Yaw torque operates on a different principle. It arises from the reaction torques generated by each propeller about the vertical body axis as a consequence of Newton's third law. In a quadrotor configuration, diagonally opposite rotors rotate in opposite directions (e.g., rotors 1 and 3 counterclockwise, and rotors 2 and 4 clockwise) such that the reaction torques cancel under nominal operating conditions. Controlled yaw motion is achieved by introducing an imbalance between these opposing torques through differential rotor speed commands. The resulting yaw control torque τ_ψ is given by Equation 59.

$$\tau_\psi = \tau_1 - \tau_2 + \tau_3 - \tau_4, \quad (59)$$

where each rotor reaction torque τ_i is modeled as

$$\tau_i = K_M n_i^2, \quad (60)$$

with n_i denoting the angular velocity of rotor i and K_M the propeller moment coefficient, which depends on the motor-propeller configuration and is typically identified experimentally for a given platform.

Consequently, the rotor-induced moments acting on the quadplane are represented in the body-fixed frame as:

$$\mathbf{M}_{\text{rotor}}^B = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \quad (61)$$

The nonlinear rotational equations of motion can then be expressed as follows.

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{J_y - J_z}{J_x} qr \\ \frac{J_z - J_x}{J_y} pr \\ \frac{J_x - J_y}{J_z} pq \end{bmatrix} + \begin{bmatrix} \frac{1}{J_x} \\ \frac{1}{J_y} \\ \frac{1}{J_z} \end{bmatrix} \mathbf{M}_{\text{rotor}}^B. \quad (62)$$

5.2 Tracking Controller

The baseline tracking controller follows a cascaded PID architecture, a standard design for multirotor platforms in which the control problem is decomposed into two nested loops operating at different timescales. The outer loop, running in the inertial frame, computes a desired acceleration vector from position and velocity tracking errors. This desired acceleration is then mapped to attitude commands, which serve as the reference input to the inner loop. The inner loop operates in the body frame and tracks these attitude commands by computing the torque demands required to drive attitude errors to zero. Finally, a motor mixing stage translates the desired collective thrust and body torques into individual motor speed commands.

The cascade structure is motivated by the principle of timescale separation. Each loop operates at a bandwidth matched to the dynamics it regulates, such that each inner loop presents itself as a fast, reliable actuator to the loop above it. No single loop is therefore required to manage dynamics spanning multiple timescales simultaneously, and each level can be independently tuned without compromising the stability margins of adjacent loops.

The inclusion of integral and derivative action at each level is determined by two criteria. First, whether the loop is subject to persistent disturbances that no outer loop will observe and correct — if so, an integrator is necessary to eliminate steady-state error. Second, whether the inner loop already provides sufficient damping — if so, a derivative term in the outer loop would produce redundant damping of the same motion, yielding an overdamped response. Applying these criteria across the cascade yields the controller structure summarised in Table 12.

Table 12: PID term selection across the cascade levels.

Loop	P	I	D	Rationale
Angular rate	✓	✓	✓	Motor asymmetries and bearing friction create persistent torque biases invisible to outer loops. No inner loop exists to provide damping. All three terms are required.

Continued on next page

Table 12 – continued from previous page

Loop	P	I	D	Rationale
Angle/Attitude	✓			Persistent attitude errors cause translational offset detected and corrected by outer loops. The rate loop below provides inherent damping. Only a proportional term is needed.
Velocity	✓	✓	✓	Disturbances such as wind apply persistent forces directly to velocity, invisible to the inner loops. No inner loop damps velocity overshoot. All three terms are required.
Position	✓			Persistent position errors manifest as velocity errors, which the velocity loop integrator corrects. The velocity loop provides inherent damping. Only a proportional term is needed.

In this section, the inner loop design is explained first, followed by the outer loop design. The angular rate controller occupies the innermost loop of the cascade, receiving body-frame rate commands $\Omega_{\text{cmd}} = [p_{\text{cmd}}, q_{\text{cmd}}, r_{\text{cmd}}]^\top$ from the attitude loop and producing torque demands $\tau_{\text{cmd}} \in \mathbb{R}^3$ that are subsequently allocated to individual motor speeds. The rate error is defined as:

$$\mathbf{e}_\Omega = \Omega_{\text{cmd}} - \Omega_{\text{meas}}, \quad (63)$$

and the PID control law is:

$$\tau_{\text{cmd}} = \mathbf{K}_P^{\text{rate}} \left(\mathbf{e}_\Omega + \frac{1}{T_I} \int \mathbf{e}_\Omega dt + T_D \mathbf{d}_k \right), \quad (64)$$

where $\mathbf{d}_k$ is a filtered derivative computed via a first-order low-pass differentiator to attenuate high-frequency noise expected in the angular rate state:

$$\mathbf{d}_k = \alpha \mathbf{d}_{k-1} + (1 - \alpha) \cdot N_f \cdot \frac{\mathbf{e}_k - \mathbf{e}_{k-1}}{\Delta t}, \quad \alpha = \frac{\lambda_d}{\lambda_d + \Delta t}, \quad \lambda_d = \frac{T_D}{N_f}, \quad (65)$$

with N_f governing the noise-bandwidth trade-off and Δt the controller timestep. The derivative acts on the rate error rather than the measurement directly; since Ω_{cmd} originates from the smooth attitude proportional loop, this avoids derivative kick on reference changes (Åström & Hägglund, 1995).

The integral term is subject to back-calculation anti-windup to prevent integrator wind-up during torque saturation. The saturated torque output is compared to the unsaturated demand, and the resulting discrepancy is fed back to correct the integrator at rate $K_t = 1/T_I$:

$$\dot{\mathbf{I}} = \mathbf{e}_\Omega + K_t (\tau_{\text{sat}} - \tau_{\text{raw}}), \quad (66)$$

with the integrator state further hard-clamped to ± 10 to bound accumulation under sustained large errors.

The raw torque demand is saturated against physically realisable limits, which depend on the available thrust headroom at the current operating point. Defining the per-motor thrust headroom as $b_{\text{max}} = \min(F_{\text{max}} - T_{\text{motor}}, T_{\text{motor}})$, where $T_{\text{motor}} = F_T/4$ is the thrust share per motor, the torque limits are:

$$\tau_{\text{lim},xy} = 4 b_{\text{max}} l_m, \quad \tau_{\text{lim},z} = 4 b_{\text{max}} \kappa_\tau, \quad (67)$$

where $l_m = l/\sqrt{2}$ is the effective moment arm in X-configuration and $\kappa_\tau = K_M/K_F$ is the torque-to-thrust ratio. To ensure smooth loss of control authority at low thrust, the limits are further scaled by:

$$\lambda_T = \text{clip}\left(\frac{F_T}{0.2mg}, 0, 1\right). \quad (68)$$

After torque saturation, the control allocation module maps the commanded total thrust and body torques to individual motor thrust commands using a prioritized allocation strategy. Roll and pitch torques are allocated first, consuming the available actuator authority, while yaw torque is assigned only the remaining motor headroom. This prioritization reflects the greater importance of roll and pitch stabilization compared to yaw tracking. Consequently, the yaw command may be reduced when insufficient actuator authority remains after satisfying the roll and pitch demands.

The resulting motor thrust commands are given by

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix} = \frac{F_T}{4} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + \frac{1}{4l_m} \begin{bmatrix} +\tau_x + \tau_y \\ +\tau_x - \tau_y \\ -\tau_x - \tau_y \\ -\tau_x + \tau_y \end{bmatrix} + \frac{1}{4\kappa_\tau} \begin{bmatrix} -\tau_z \\ +\tau_z \\ -\tau_z \\ +\tau_z \end{bmatrix}, \quad (69)$$

The resulting motor thrust commands are clipped according to

$$0 \leq F_i \leq F_{\max}, \quad i = 1, \dots, 4, \quad (70)$$

to ensure compliance with actuator limitations. The thrust commands are subsequently converted into motor speed setpoints using the propeller thrust model

$$n_i = \sqrt{\frac{F_i}{K_F}}, \quad i = 1, \dots, 4, \quad (71)$$

where K_F is the propeller thrust coefficient. The commanded motor speeds n_i constitute the final output of the cascaded control architecture and are supplied directly to the motor dynamics model in simulation.

One level above the angular rate loop is the attitude controller. This controller receives a desired orientation $\mathbf{R}_{\text{des}} \in SO(3)$ from the acceleration-to-attitude mapping block and computes the body-frame angular rate commands required to drive the current orientation toward the desired attitude.

The attitude error is computed directly on the rotation group $SO(3)$, thereby avoiding Euler-angle representations, which suffer from singularities (gimbal lock) and introduce artificial coupling between rotational axes at large attitude angles. The relative rotation between the current and desired body frames is defined as

$$\mathbf{R}_{\text{err}} = \mathbf{R}_{\text{cur}}^\top \mathbf{R}_{\text{des}}, \quad (72)$$

where $\mathbf{R}_{\text{cur}} \in SO(3)$ is obtained from the measured vehicle quaternion. Since $\mathbf{R}_{\text{err}}$ is expressed in the current body frame, the resulting error signal is naturally independent of the vehicle heading, unlike Euler-angle error formulations.

The rotational error vector is extracted from the skew-symmetric component of $\mathbf{R}_{\text{err}}$ using the vee operator:

$$\mathbf{e}_\phi = \frac{1}{2} \begin{bmatrix} R_{\text{err},32} - R_{\text{err},23} \\ R_{\text{err},13} - R_{\text{err},31} \\ R_{\text{err},21} - R_{\text{err},12} \end{bmatrix}. \quad (73)$$

For small attitude errors, $\mathbf{e}_\phi$ is equivalent to the axis-angle representation of the residual rotation.

A proportional attitude control law is then used to generate the desired angular rate command:

$$\boldsymbol{\Omega}_{\text{cmd}} = \mathbf{K}_P^{\text{att}} \mathbf{e}_\phi, \quad (74)$$

To ensure that the commanded rates remain physically achievable, the resulting angular rate command is saturated according to

$$\boldsymbol{\Omega}_{\text{cmd}} = \text{sat}(\boldsymbol{\Omega}_{\text{cmd}}, \pm \Omega_{\text{max}}), \quad (75)$$

before being passed to the inner rate controller, where $\Omega_{\text{max}} = 200 \text{ deg/s}$. This saturation prevents the attitude controller from requesting angular velocities that exceed actuator capabilities and preserves sufficient control authority for disturbance rejection within the rate loop.

One level above the attitude controller lies the thrust-vector-to-attitude mapping block. Since a quadplane in hover mode can only generate thrust along its body z -axis, achieving a desired translational acceleration requires tilting the vehicle such that the body z -axis aligns with the commanded force vector. This block performs three operations in sequence: thrust vector construction and saturation, geometric rotation matrix construction, and tilt saturation with anti-windup correction.

The commanded acceleration $\mathbf{a}_{\text{cmd}} \in \mathbb{R}^3$ is augmented with gravity compensation and scaled by the vehicle mass to obtain the desired thrust vector in the world frame:

$$\mathbf{f}_{\text{cmd}} = m (\mathbf{a}_{\text{cmd}} + g \hat{\mathbf{e}}_z). \quad (76)$$

The desired thrust vector is subsequently saturated to respect motor and propeller limits. The total thrust must lie within a sphere of radius $F_{T_{\text{max}}} = 4K_F n_{\text{max}}^2$, and the vertical and horizontal components share this budget. When conflicting demands arise between horizontal and vertical acceleration, a priority scheme is applied in which vertical thrust is satisfied first, visualized in Figure 19:

$$F_{z,\text{sat}} = \text{clamp}(m(a_z + g), 0, F_{T_{\text{max}}}), \quad (77)$$

$$F_{xy,\text{max}} = \sqrt{F_{T_{\text{max}}}^2 - F_{z,\text{sat}}^2}, \quad (78)$$

$$F_{xy,\text{sat}} = \text{clamp}\left(m\sqrt{a_x^2 + a_y^2}, 0, F_{xy,\text{max}}\right). \quad (79)$$

The saturated thrust vector is normalized to define the desired body z -axis. Before normalization, the vertical component is clamped to a minimum of $0.2mg$ to prevent a downward-pointing or near-singular rotation matrix, which would otherwise lead to spurious 180° roll or pitch commands.

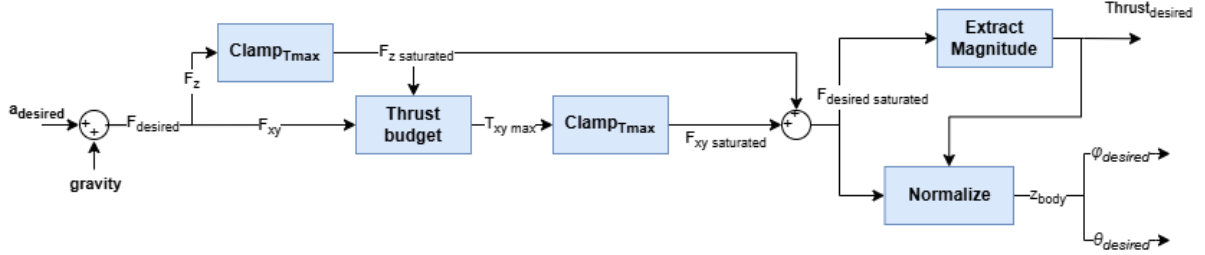


Figure 19: Thrust saturation and acceleration-to-attitude decomposition: vertical thrust is prioritised within the total thrust budget $F_{T,\text{max}}$, with the remaining capacity allocated to horizontal acceleration.

A desired rotation matrix $\mathbf{R}_{\text{des}} \in SO(3)$ is constructed geometrically using the commanded yaw angle ψ_{cmd} to resolve the remaining degree of freedom:

$$\hat{\mathbf{z}}_b = \frac{\mathbf{f}_{\text{sat}}}{\|\mathbf{f}_{\text{sat}}\|}, \quad \hat{\mathbf{y}}_b = \frac{\hat{\mathbf{z}}_b \times \hat{\mathbf{x}}_c}{\|\hat{\mathbf{z}}_b \times \hat{\mathbf{x}}_c\|}, \quad \hat{\mathbf{x}}_b = \hat{\mathbf{y}}_b \times \hat{\mathbf{z}}_b, \quad (80)$$

where $\hat{\mathbf{x}}_c = [\cos \psi_{\text{cmd}}, \sin \psi_{\text{cmd}}, 0]^\top$ defines the desired heading direction. The rotation matrix is then given by

$$\mathbf{R}_{\text{des}} = [\hat{\mathbf{x}}_b \ \hat{\mathbf{y}}_b \ \hat{\mathbf{z}}_b]. \quad (81)$$

A second saturation stage is applied after constructing the rotation matrix. The roll and pitch angles implied by $\mathbf{R}_{\text{des}}$ are extracted and clamped to $\pm\phi_{\text{max}} = \pm 20^\circ$, after which the rotation matrix is rebuilt from the saturated angles.

Because this modification alters the achievable thrust direction, the velocity integrator is corrected to prevent windup. The achievable thrust vector is recomputed as

$$\mathbf{f}_{\text{ach}} = \hat{\mathbf{z}}_{b,\text{sat}} \|\mathbf{f}_{\text{sat}}\|, \quad (82)$$

and the anti-windup correction applied to the velocity integrator is

$$\mathbf{e}_{\text{aw}} = \frac{\mathbf{f}_{\text{ach}}}{m} - \frac{\mathbf{f}_{\text{sat}}}{m}. \quad (83)$$

Here, $\hat{\mathbf{z}}_{b,\text{sat}}$ denotes the body z -axis of the tilt-clamped rotation matrix.

The scalar thrust command forwarded to the motor allocation stage is the projection of the saturated thrust vector onto the current body z -axis:

$$F_T = \max(0, \mathbf{f}_{\text{sat}}^\top \mathbf{R}_{\text{cur}} \hat{\mathbf{e}}_z). \quad (84)$$

This projection accounts for the fact that only the component of the desired force vector aligned with the current body z -axis can be produced at the present attitude, reflecting the instantaneous thrust authority available.

One loop above the acceleration-to-attitude mapping block sits the velocity controller. It receives a target velocity vector $\mathbf{v}_{\text{cmd}} \in \mathbb{R}^3$ and produces a desired acceleration $\mathbf{a}_{\text{cmd}} \in \mathbb{R}^3$ as its output. The UAV measures its ground-referenced velocity $\mathbf{v}_{\text{meas}} \in \mathbb{R}^3$ using a GNSS receiver, whose position and velocity estimates are fused with inertial measurements in the onboard state estimator.

During ship-landing operations, the target position is located on a moving vessel. A drone tracking a fixed inertial setpoint above a moving ship would drift relative to the deck as the ship translates beneath it. To compensate, the surface velocity $\mathbf{v}_{\text{surf}} \in \mathbb{R}^3$ is added as a feedforward term in the velocity error, such that the closed-loop error is expressed relative to the moving surface rather than the inertial frame:

$$\mathbf{e}_v = (\mathbf{v}_{\text{cmd}} + \mathbf{v}_{\text{surf}}) - \mathbf{v}_{\text{meas}}. \quad (85)$$

This formulation is equivalent to commanding the UAV to track $\mathbf{v}_{\text{cmd}}$ relative to the deck, which itself moves with velocity $\mathbf{v}_{\text{surf}}$ in the inertial frame. The feedforward term is active only in position-control mode; in velocity-control mode, the reinforcement learning agent already provides commands in the moving frame, and the term is set to zero.

The controller generates the acceleration command according to

$$\mathbf{a}_{\text{cmd}} = \mathbf{K}_P^{\text{vel}} \left(\mathbf{e}_v + \frac{1}{T_I} \int \mathbf{e}_v dt + T_D \mathbf{d}_k \right), \quad (86)$$

where $\mathbf{d}_k$ is a filtered estimate of the velocity error derivative. A direct finite-difference approximation is sensitive to GNSS noise; therefore, a first-order low-pass filter is applied. At each timestep k , the derivative state is updated as

$$\mathbf{d}_k = \alpha \mathbf{d}_{k-1} + (1 - \alpha) \frac{\mathbf{e}_v^{(k)} - \mathbf{e}_v^{(k-1)}}{\Delta t}, \quad \alpha = \frac{1}{1 + N_f \Delta t}, \quad (87)$$

where N_f is the derivative filter coefficient. The parameter α is chosen in a Tustin-consistent manner, ensuring that the effective filter bandwidth scales correctly with the control timestep.

The velocity integrator is susceptible to windup due to saturation in the downstream thrust and attitude allocation stages. A back-calculation anti-windup scheme is applied, where the difference between the achievable acceleration (computed from the tilt-clamped attitude) and the raw PID output is fed back to correct the integrator state. Additionally, the integrator is hard-limited to ± 5 as a secondary safeguard. A weaker back-calculation gain is used in the vertical channel to reflect the reduced control authority along the body z -axis.

The outermost loop of the cascade is the position controller. It compares the current vehicle position $\mathbf{p}_{\text{meas}} \in \mathbb{R}^3$ with the target position $\mathbf{p}_{\text{cmd}} \in \mathbb{R}^3$, yielding the position error

$$\mathbf{e}_p = \mathbf{p}_{\text{cmd}} - \mathbf{p}_{\text{meas}}. \quad (88)$$

This error is mapped to a velocity command using a proportional control law with an additive feedforward term:

$$\mathbf{v}_{\text{cmd}} = \mathbf{K}_P^{\text{pos}} \mathbf{e}_p, \quad (89)$$

The resulting velocity command is saturated on a per-axis basis before being forwarded to the velocity controller:

$$\mathbf{v}_{\text{cmd}} = \text{clamp}(\mathbf{v}_{\text{cmd}}, \pm \mathbf{v}_{\text{max}}), \quad \mathbf{v}_{\text{max}} = \begin{bmatrix} 10 \\ 10 \\ 10 \end{bmatrix} \text{ m/s}. \quad (90)$$

Controller Tuning Approach

Consider a unity feedback control structure consisting of a PID controller $C(s)$ and a plant $G(s)$, as visualized in Figure 20. The open-loop transfer function is defined according to Equation 91.

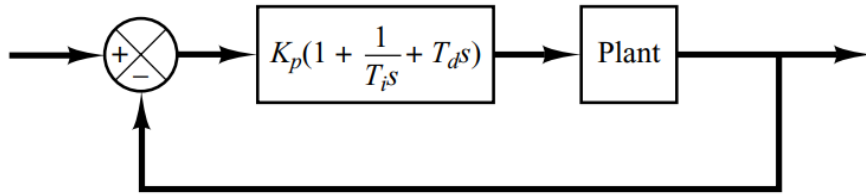


Figure 20: Closed loop control system block diagram. Retrieved from Ogata (2010).

$$L(s) = C(s)G(s) = K_p \left(1 + \frac{1}{T_i s} + T_d s \right) G(s) \quad (91)$$

The closed-loop transfer function from reference input r to output y and the sensitivity function from plant output disturbance d_{out} to output y are defined, respectively, in Equation 92 and 93. The sensitivity function quantifies the closed-loop attenuation of output disturbances and serves as the primary measure of robustness. A complementary sensitivity function is also defined to quantify the closed-loop attenuation of measurement noise. This is equivalent to the closed-loop transfer function, as there is no feedback term in the structure above. These functions serve as tools to ensure that the control objectives of the landing task defined in Section 2 are sufficiently met.

$$T(s) = \frac{L(s)}{1 + L(s)} \quad (92)$$

$$S(s) = \frac{1}{1 + L(s)} \quad (93)$$

The poles of the closed-loop transfer function determine the stability of a control system. A system is considered bounded-input bounded-output (BIBO) stable if every bounded input

produces a bounded output that eventually settles to a finite value. Mathematically, the closed-loop transfer function becomes unbounded when its denominator equals zero. The values of s that satisfy this condition are the poles of the system.

If any pole lies in the right half of the complex plane, the system response grows unbounded over time, and the system is unstable. Therefore, stability is determined by the location of the closed-loop poles, which are obtained from the denominator of the closed-loop transfer function. Since the denominator coefficients depend on the controller parameters, the PID controller gains directly affect the pole locations.

The stability margin quantifies how far the system is from becoming unstable and is commonly expressed in terms of phase margin and gain margin. The phase margin indicates the number of additional degrees of phase lag the system can tolerate at the gain crossover frequency (where $|L(j\omega)| = 1$) before the closed loop becomes unstable. The gain margin (GM) indicates the factor by which the open-loop gain can be increased or decreased before instability occurs. Both PM and GM are robustness measures that depend on systems' open-loop characteristics; therefore, controller gains directly influence these margins.

In addition to gain and phase margins, the sensitivity functions provide a frequency-domain framework for shaping closed-loop performance with respect to robustness. The magnitude of the sensitivity functions quantifies the extent to which disturbances and measurement noise are attenuated in the closed-loop output. Based on the frequency-domain characteristics of these disturbances and measurement noise, the shaping of the sensitivity transfer functions ensures that these uncertainties are sufficiently attenuated to achieve robust closed-loop performance. It is generally accepted that the peak of the sensitivity function M_s is kept below approximately 6 dB to ensure sufficient stability margin across all frequencies, while the peak of T is typically kept below 3–4 dB. Beyond limiting peak sensitivity, the crossover frequency ω_c , which determines where the sensitivity function transitions to unity gain, must be chosen deliberately: high enough to achieve the desired disturbance rejection and tracking bandwidth, but low enough to avoid amplification of unmodeled dynamics, resonances, or time-delay effects.

Building upon the frequency-domain design framework, explicit tuning requirements are derived to ensure that the control architecture satisfies the desired performance and robustness specifications. This is summarized in Table 13. The crossover frequencies for each loop are selected based on the physical bandwidth limitations of the actuator chain and the cascade separation principle. The innermost angular rate loop crossover frequency is chosen as $\omega_{c,r} \approx 18$ rad/s. Each successive outer loop is then placed at approximately half of the bandwidth of the loop directly below it, ensuring sufficient timescale separation and preventing loop interaction.

A gain margin of at least 6 dB is imposed uniformly across all loops, ensuring that the loop gain may double before instability, thereby providing a baseline robustness margin against model uncertainty and unmodeled dynamics. Phase margin targets of 50° – 70° for the inner loops and 45° – 60° for the outer loops reflect the trade-off between response speed and damping. Slightly reduced phase margins are acceptable in the outer loops, where low-frequency dynamics dominate and actuator lag is less critical.

The sensitivity peak M_S is progressively relaxed from $M_S \leq 1.6$ for the rate loop to $M_S \leq 2.0$ for the outer loops, consistent with the waterbed effect in feedback design. The constraint

$M_S \leq 1.6$ corresponds to approximately 5 dB, representing a stricter robustness requirement for the innermost loop, where unmodeled actuator dynamics and transport delays are most likely to degrade stability margins. Similarly, the complementary sensitivity peak M_T is constrained more tightly for the inner loops ($M_T \leq 1.3$) than for the outer loops ($M_T \leq 1.5$), since high-frequency gyroscope noise directly affects the rate controller. Excessive M_T would amplify IMU noise into motor torque commands, resulting in actuator chatter and increased wear.

Table 13: Frequency-domain design requirements for the cascaded PID control architecture.

Loop	ω_c (rad/s)	GM (dB)	PM ($^\circ$)	M_S (-)	M_T (-)
Angular Rate	$\omega_{c,r} \approx 18$	≥ 6	50–70	≤ 1.6	≤ 1.3
Attitude (angle)	$\omega_{c,a} \approx 9$	≥ 6	50–70	≤ 1.8	≤ 1.4
Velocity	$\omega_{c,v} \approx 4$	≥ 6	45–60	≤ 2.0	≤ 1.5
Position	$\omega_{c,p} \approx 1.5$	≥ 6	45–60	≤ 2.0	≤ 1.5

To achieve the desired frequency-domain performance, the PID parameters K_P , T_I , and T_D are determined through an iterative tuning process in MATLAB. The cascaded control architecture is analyzed sequentially from the innermost loop outward using linearized transfer function models, ensuring that the bandwidth and phase limitations of each inner loop are propagated to the outer-loop design.

The rotational plant for each axis is modeled as a pure angular integrator scaled by the corresponding moment of inertia, obtained by linearizing the rigid-body rotational dynamics about the hover equilibrium point:

$$G_{\text{rot}} = \frac{1}{J_S}, \quad (94)$$

with $J_x = 0.478 \text{ kg} \cdot \text{m}^2$, $J_y = 0.342 \text{ kg} \cdot \text{m}^2$, $J_z = 0.811 \text{ kg} \cdot \text{m}^2$, as obtained from the quadplane model. This linearization is valid in the neighborhood of hover and constitutes the primary modeling assumption underlying the frequency-domain analysis; performance away from hover is subsequently validated using nonlinear simulation.

Two additional dynamics are included to capture the dominant phase-limiting effects of the actuator chain: a lumped first-order lag representing ESC and motor dynamics with time constant $T_{\text{lag}} = 0.045 \text{ s}$, and a second-order Padé approximation of the one-step computational delay at sampling time $T_s = 1/240 \text{ s}$.

The horizontal velocity plant is derived under the small-angle assumption as

$$G_{v_x} = \frac{g}{s} T_{\text{att}}, \quad (95)$$

where T_{att} denotes the closed-loop attitude transfer function, reflecting that horizontal acceleration is generated through vehicle tilt.

The vertical dynamics are modeled as a direct thrust-to-velocity integrator:

$$G_{v_z} = \frac{1}{mS} G_{\text{delay}} G_{\text{motor}}. \quad (96)$$

Each loop is closed sequentially so that the bandwidth limitations and phase lag introduced by the inner loops are accurately propagated into the effective plant seen by the outer loops. The resulting frequency-domain properties of the tuned cascade are summarized in Table 14, with the corresponding Bode plots and sensitivity functions presented in Figures 21–24.

The open-loop Bode plots confirm that the crossover frequencies ω_c are appropriately separated throughout the cascade. Furthermore, all loops satisfy the prescribed gain and phase margin requirements, demonstrating adequate robustness to model uncertainty, parameter variations, and unmodelled dynamics.

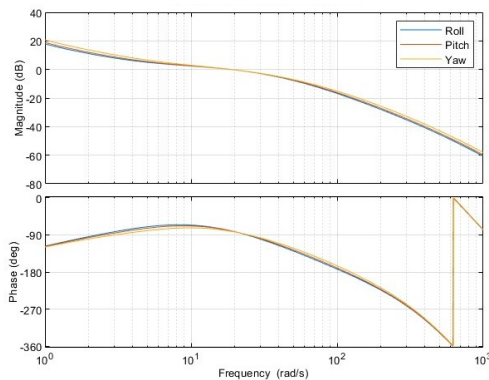
The sensitivity functions provide additional validation of the controller design. The sensitivity peak, M_S , which characterizes the worst-case amplification of output disturbances, remains below the specified limits at every cascade level. Since aerodynamic disturbances such as mean wind and slowly-varying gusts are predominantly low-frequency, they fall within the region of strong attenuation provided by the closed-loop system, well below the sensitivity peak.

Similarly, the complementary sensitivity peak, M_T , which quantifies the amplification of measurement noise, remains within the prescribed bounds for all loops. The complementary sensitivity exhibits the expected low-pass behavior, with its magnitude concentrated at low frequencies and rapidly attenuating at higher frequencies. The bandwidths ω_B and ω_{BT} are defined as the frequencies at which $|S(j\omega)|$ and $|T(j\omega)|$ first cross the -3 dB point, marking the upper limits of effective disturbance and noise rejection respectively.

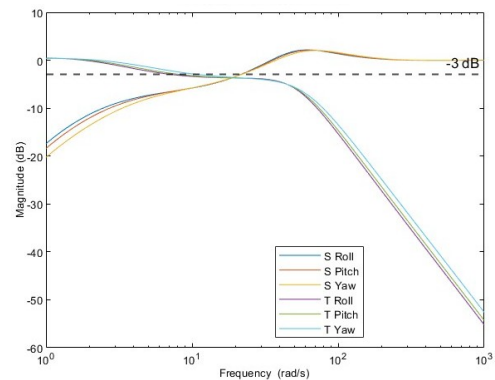
Taken together, the Bode and sensitivity analyses indicate that the tuned cascade meets the frequency-domain performance and robustness specifications of Table 13 within the assumptions of the linearized hover model.

Table 14: Frequency-domain properties of the tuned cascaded PID control architecture.

Loop	ω_c (rad/s)	GM (dB)	PM ($^\circ$)	M_S (–)	M_T (–)	ω_B (rad/s)	ω_{BT} (rad/s)
Angular Rate	18.5	18.3	99	1.27	1.05	23	10
Attitude (angle)	8.5	15.5	60	1.35	1.10	9	14
Velocity	3.9	15.8	88	1.27	1.12	4	4
Position	1.7	27.3	50	1.35	1.05	2	3

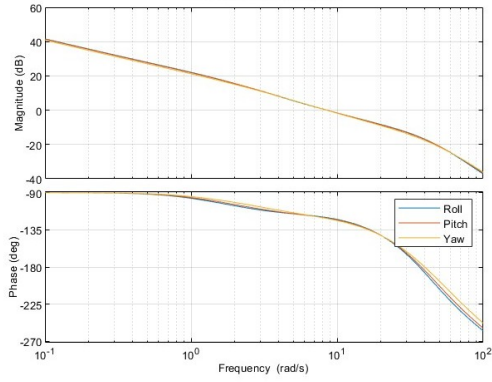


(a) Angular rate loop Bode plot

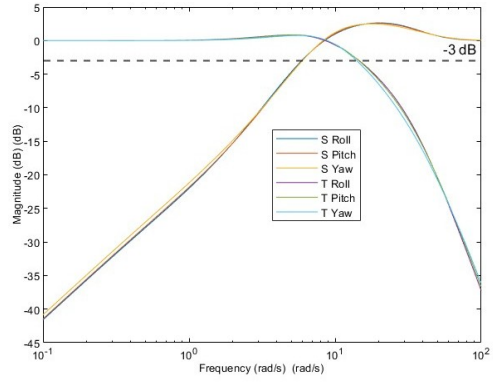


(b) Angular rate loop sensitivity

Figure 21: Frequency-domain analysis of the angular rate loop.

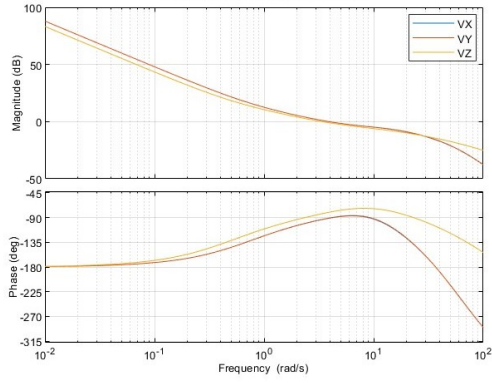


(a) Attitude loop Bode plot

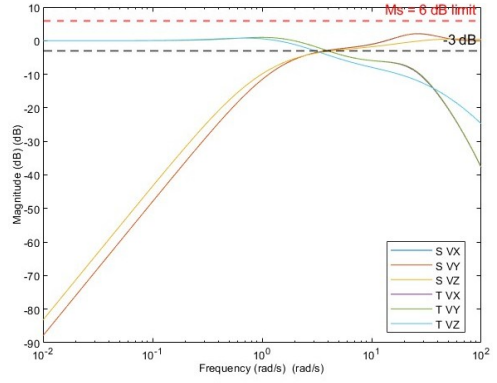


(b) Attitude loop sensitivity

Figure 22: Frequency-domain analysis of the attitude loop.

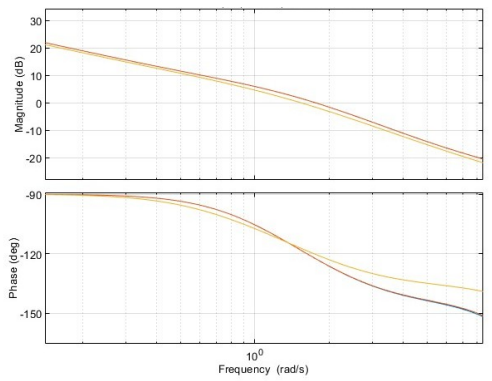


(a) Velocity loop Bode plot

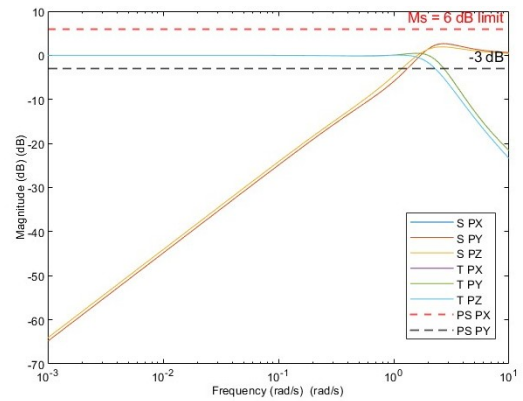


(b) Velocity loop sensitivity

Figure 23: Frequency-domain analysis of the velocity loop.



(a) Position loop Bode plot

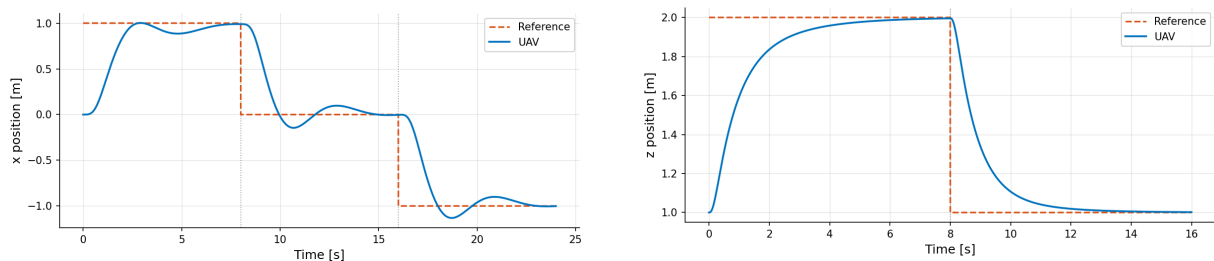


(b) Position loop sensitivity

Figure 24: Frequency-domain analysis of the position loop.

5.3 Time Domain Tracking Performance

To verify the fidelity of the simulated dynamics and the quality of the PID controller tuning, a series of reference step responses are evaluated across the position and velocity channels. Figure 25a and Figure 25b illustrate representative position tracking responses along the horizontal and vertical axes, respectively, while Figures 26a–26c present the corresponding inner-loop velocity tracking behavior.



(a) Horizontal (x -axis) position step response under PID control.

(b) Vertical (z -axis) position step response under PID control.

Figure 25: Position step responses of the cascaded PID controller. (a) Horizontal (x -axis) position response. (b) Vertical (z -axis) position response.

Figure 25a presents the x -axis position tracking response, in which the vehicle is commanded sequentially from the origin to +1 m, back to 0 m, and finally to −1 m. The UAV tracks each reference transition in a stable manner with a fast rise time; brief oscillatory transients are present at each step but remain bounded within approximately 10% of the target value. Figure 25b shows the altitude response to a climb command to 2 m followed by a descent to 1 m. Vertical tracking is well-damped with no discernible overshoot. Although the rise time is longer than that observed in the horizontal channels, the settling time remains comparable.

Quantitative step-response metrics are summarised in Table 15. Across all tested scenarios, including isolated single-axis steps, a diagonal planar step, and a full three-dimensional step, the controller achieves consistent rise times of approximately 1.4–1.5 s in the horizontal axes and 2.4 s in the vertical axis. Overshoot remains negligible ($M_p < 2\%$) in all cases, and steady-state error is bounded within ± 0.02 m, confirming integral action is effective. Settling times of approximately 5–7 s are consistent with those reported for comparable quadplane platforms (Mathur & Atkins, 2021).

Table 15: PID reference step response performance metrics.

Test	Axis	t_r [s]	M_p [%]	t_s [s]	e_{ss} [m]
Altitude	Z	2.41	0.00	5.28	0.00
X-axis	X	1.47	0.25	6.98	-0.02
Diagonal	X	1.47	0.24	6.98	-0.02
	Y	1.49	0.00	7.05	-0.02
3D	X	1.43	1.94	6.97	-0.02

Continued on next page

Table 15 – continued from previous page

Test	Axis	t_r [s]	M_p [%]	t_s [s]	e_{ss} [m]
	Y	1.43	1.61	7.02	-0.02
	Z	2.41	0.00	5.25	0.00

The velocity tracking responses are presented in Figures 26a–26c. Velocity commands are tracked without sustained oscillation across all axes, with transient overshoot bounded within approximately 20% of the target value for the horizontal channels. Vertical velocity tracking exhibits superior performance, with negligible overshoot, a rise time below 0.5 s, and a well-damped settling response. This is particularly relevant to the maneuver under investigation, as touchdown velocity is a primary determinant of landing safety; adequate closed-loop authority over the z -axis velocity channel is therefore a necessary condition for successful execution.

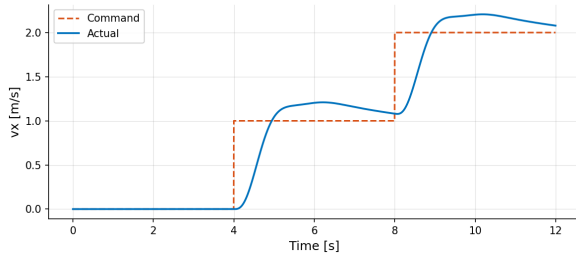
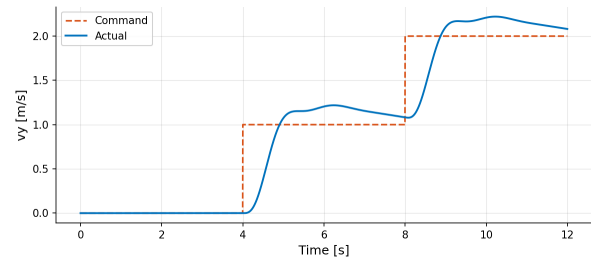
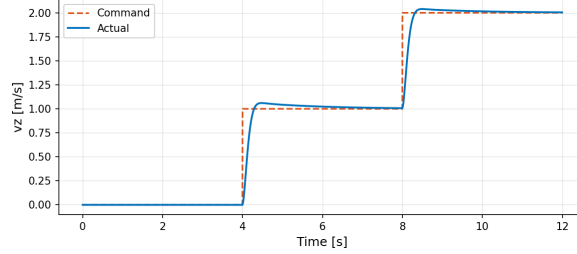
(a) x -axis velocity tracking response.(b) y -axis velocity tracking response.(c) z -axis velocity tracking response.

Figure 26: Velocity tracking responses of the cascaded PID controller. (a) x -axis velocity response. (b) y -axis velocity response. (c) z -axis velocity response.

Finally, to evaluate closed-loop position tracking under persistent aerodynamic disturbances, two representative wind conditions are examined: a 10 m/s headwind along the $-x$ direction, consistent with the ambient wind environment of the offshore operational scenario, and a 5 m/s vertical downwash along the $-z$ direction. These conditions exercise both the horizontal and vertical disturbance rejection channels of the cascade PID and allow the aerodynamic force model to be observed in closed-loop operation.

Under a 10 m/s headwind disturbance, the UAV is displaced in the negative x -direction by approximately 0.2 m, as the aerodynamic drag force opposes the vehicle's hover position. In response, the controller commands a nose-down pitch attitude to generate a horizontal thrust component that counteracts this drag force and restores the commanded x position.

The corresponding position response is shown in Figure 27, settling to steady state within approximately 6 s.

Throughout this transient, the cambered airfoil continues to generate positive lift, as the angle of attack remains above the zero-lift angle on the $C_L(\alpha)$ polar. This aerodynamic lift force partially offsets the vehicle's weight, producing a small positive displacement in the z -axis while simultaneously reducing the vertical load that must be supported by rotor thrust; consequently, the total rotor thrust settles below the nominal hover value at steady state. As the controller reduces rotor speed in response to this reduced load requirement, the resulting altitude displacement is attenuated over time, and the vehicle re-establishes the commanded altitude with no residual steady-state z -axis error.

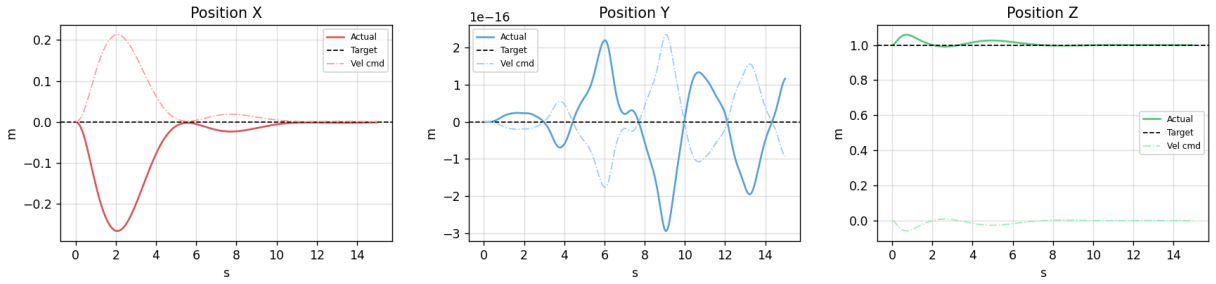


Figure 27: Position tracking response under a 10 m/s headwind.

Under a 5 m/s vertical downwash, the aerodynamic drag force acts in the negative z -direction, exerting a downward load on the vehicle of approximately 13 N at steady state. This load lies within the available thrust margin, allowing the controller to maintain the commanded altitude by increasing collective rotor thrust. As the incident flow arrives from directly above, the angle of attack approaches -90° , at which no lift or side force is generated according to the aerodynamic model. Accordingly, no coupling into the horizontal position or attitude channels is observed, confirming that the vertical aerodynamic disturbance is absorbed entirely by the collective thrust channel without exciting rotational or translational dynamics.

The corresponding position response is shown in Figure 28: the z -axis settles to steady state within approximately 6 s, while the x - and y -axis errors remain on the order of 10^{-15} – 10^{-17} m, consistent with numerical precision limits rather than any physically meaningful displacement.

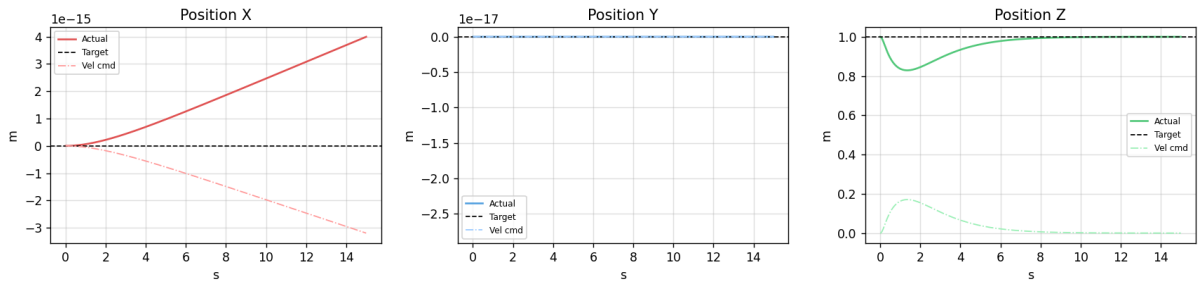


Figure 28: Position tracking response under a 5 m/s vertical downwash.

5.4 Baseline Guidance Controller

A three-phase hierarchical guidance law is implemented as the baseline controller for the autonomous shipboard landing task. The guidance law operates on top of the PID position and velocity tracking stack described in the preceding section, issuing high-level position commands $\mathbf{p}^{\text{cmd}} \in \mathbb{R}^3$ at the guidance control frequency based on the current phase of the maneuver.

Upon initialisation, the UAV is commanded to hold a fixed hover altitude of $z_{\text{hover}} = z_{\text{pad}} + \Delta z_{\text{hover}}$ above the nominal pad surface while simultaneously tracking the instantaneous horizontal position of the moving landing pad. The UAV is considered hover-stable when the horizontal distance to the pad centroid falls below a threshold ϵ_{stable} and remains so continuously for a duration t_{hover} . If the horizontal error exceeds the abort threshold ϵ_{abort} at any point during hover, the stable-time counter is reset.

Once the hover stability criterion is satisfied, the UAV transitions to the descent phase. A vertical position command is issued that steps the UAV downward toward a biased touchdown target $z_{\text{tgt}} = z_{\text{pad}} - \Delta z_{\text{bias}}$, where Δz_{bias} is a downward offset introduced to prevent asymptotic convergence to the landing pad and ensure physical contact during landing. Throughout the descent, horizontal commands continue to track the moving pad. If the horizontal error exceeds ϵ_{abort} , the descent is aborted and the guidance law reverts to hover phase.

Landing is declared when the vertical separation between the UAV and the pad surface falls within a margin ϵ_{land} . The attempt is classified as successful if, at the moment of touchdown, the horizontal position error satisfies $|p_{\text{rel},xy}| < 0.50$ m, the relative vertical velocity satisfies $|\mathbf{v}_{z_{\text{rel}}}| < 0.7$ m/s, and the attitude deviation satisfies $|\eta_{\text{rel}}| < 10.0^\circ$, as defined in Section 2.

The user-defined parameters governing the guidance law are summarised in Table 16.

Table 16: Baseline guidance law tuning parameters.

Parameter	Symbol	Value
Hover altitude above pad	Δz_{hover}	3.0 m
Stable hover duration	t_{hover}	1.5 s
Stability threshold (XY)	ϵ_{stable}	0.3 m
Abort threshold (XY)	ϵ_{abort}	0.7 m
Touchdown bias offset	Δz_{bias}	0.3 m
Landing detection margin	ϵ_{land}	0.02 m

Figure 29 illustrates the guidance law command structure over the course of a representative landing attempt under sea state 6 conditions. During the hover phase ($t < 9$ s), the horizontal position commands track the instantaneous pad centroid in x and y , updated at the controller frequency of 5 Hz, while the vertical command is held at a fixed offset of $\Delta z_{\text{hover}} = 3.0$ m above the pad surface. At $t = 9$ s, the hover stability criterion is satisfied, and the guidance law transitions to the descent phase; the z -axis command shifts to 0.3 m below the pad surface, applying the touchdown bias offset Δz_{bias} and triggering the descent maneuver.

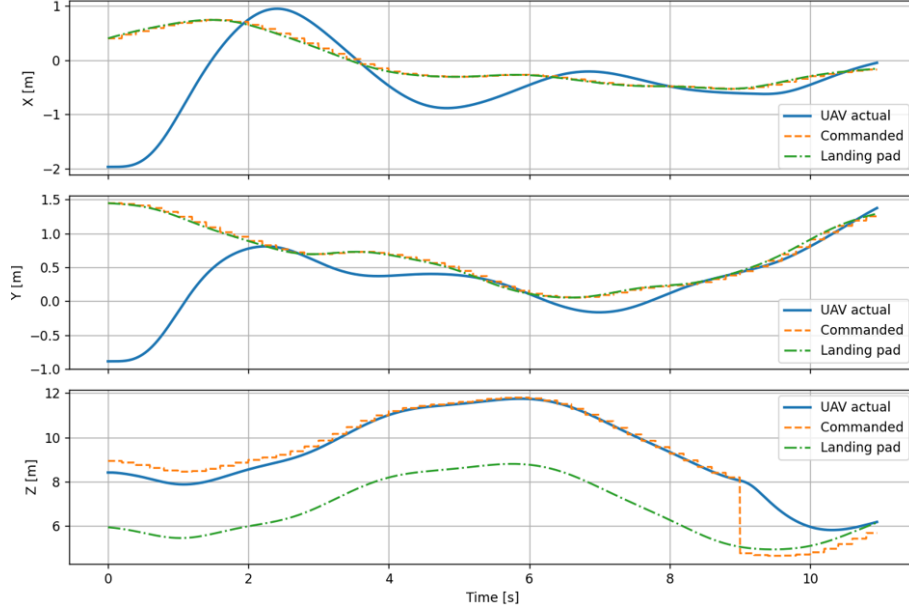


Figure 29: Baseline guidance law position commands under sea state 6 conditions.

5.5 RL Implementation

The RL agent is integrated at the guidance layer of the control stack, issuing three-dimensional velocity commands $\mathbf{v}^{\text{cmd}} \in \mathbb{R}^3$ that are passed directly to the inner-loop velocity controller, bypassing the position loop entirely. This architecture is illustrated in Figure 30. The velocity cascade of the PID controller, verified to provide satisfactory tracking performance in Section 5.3, is retained without modification; the RL agent is therefore only responsible for the mapping from the current flight state to a high-level velocity reference. Commands are issued at $f_{\text{RL}} = 5 \text{ Hz}$, obtained via action repeat from the 240 Hz physics/control loop; the full derivation and justification based on time-scale separation between the RL policy and the inner-loop controller is presented in Part II.

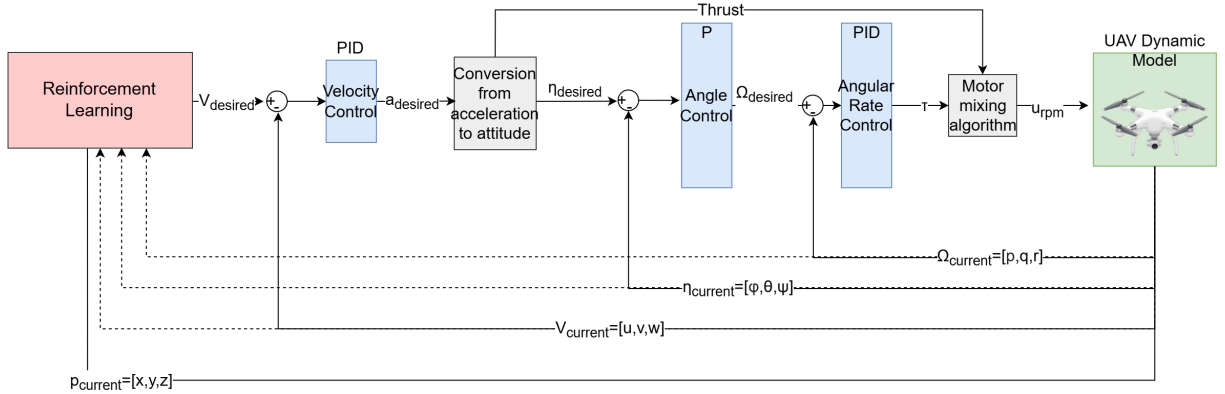


Figure 30: Reinforcement learning control architecture with direct 3D velocity command output, bypassing the position controller and feeding the velocity loop.

A continuous action space is adopted, as the vehicle dynamics evolve continuously, and precise trajectory adjustment requires fine-grained control authority. Discretizing the action space

would either constrain the expressiveness of the learned policy or necessitate an impractically large discrete set.

The full formulation of the action space, observation space, reward function, and termination/success criteria for the proposed RL-based guidance framework is presented in Part II of this thesis, and is not repeated here to avoid redundancy.

Given the need for continuous action outputs, a policy gradient approach is most appropriate for the learning model. Among these, actor–critic methods offer significant advantages over standard policy-based approaches, including improved learning stability, reduced variance in policy updates, and more efficient exploration. The choice of a specific actor–critic algorithm is guided by a careful consideration of the defined training priorities for the UAV landing task:

- **Balanced Exploration–Exploitation:** The learning process should sufficiently explore the state–action space, potentially using a dynamic exploration rate to handle its large size, while avoiding excessive randomness that could destabilize the controller or prematurely terminate episodes.
- **Stability of Training:** Policy updates should avoid large oscillations, promoting convergence and reliable learning behavior. This is particularly important in UAV simulations, where excessive randomness can lead to unsafe or terminated trajectories.
- **Robustness to Hyperparameter Tuning:** The algorithm should perform reasonably well without requiring extremely precise tuning, as the UAV environment may introduce variability due to wind and sensor noise.
- **Ease of Implementation:** Algorithms with widely available open-source implementations, clear training procedures, and moderate computational requirements are preferred for rapid experimentation and iteration.

Literature extensively evaluates actor–critic algorithms in continuous control settings, and several conclusions emerge regarding their relative performance. Among modern methods, SAC stands out (Haarnoja et al., 2018). As an off-policy algorithm, it achieves superior sample efficiency, reliably attaining higher rewards with significantly fewer environment interactions compared to PPO, DDPG, and TD3. Its entropy-regularized objective, combined with automatic entropy adjustment, provides a principled exploration mechanism that is especially valuable in high-dimensional state–action spaces and helps reduce sensitivity to reward shaping. It maintains two action-value (Q) networks, with the minimum of their values used to evaluate actions in the actor loss, thereby reducing overestimation bias and improving training stability. Furthermore, it employs delayed target Q-networks, which are used in the critic updates to compute stable Bellman targets and mitigate fluctuations in value estimates.

Consequently, SAC is selected as the RL algorithm for this work, the structure of which is visualized in Figure 31. It is formally defined as an extension of standard actor-critic frameworks that introduces an entropy-maximization term into the learning objective π as defined in Equation 97:

$$J(\pi) = \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t))) \right], \quad (97)$$

where $\gamma \in (0, 1)$ is the discount factor, $\mathcal{H}(\pi(\cdot|s)) = -\mathbb{E}_{a \sim \pi}[\log \pi(a|s)]$ is the entropy of the policy, and $\alpha > 0$ is the temperature parameter balancing reward maximization and entropy maximization. A higher value of α increases exploration by encouraging more stochastic actions, while a lower α shifts the policy toward deterministic behavior.

SAC supports automatic entropy tuning by adapting the temperature parameter α to enforce a desired target entropy. The corresponding loss function for α is defined as:

$$J(\alpha) = \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi} \left[-\alpha \left(\log \pi(a | s) + \mathcal{H}_{\text{target}} \right) \right]. \quad (98)$$

This objective adjusts α such that the expected policy entropy matches a predefined target entropy $\mathcal{H}_{\text{target}}$. In practice, this results in higher exploration during early stages of training and progressively more deterministic behavior as the policy improves, without requiring manual tuning of the entropy coefficient.

RL algorithms learn by updating their network parameters based on collected experience. Some methods perform updates using individual interaction steps with the environment, while others rely on batches of previously collected experiences. SAC is an off-policy algorithm, meaning that transitions (s_t, a_t, r_t, s_{t+1}) are stored in a replay buffer and sampled uniformly during training. This significantly improves sample efficiency compared to on-policy methods such as PPO.

At each gradient step, SAC performs the following updates:

1. **Sampling:** A mini-batch of transitions is sampled from the replay buffer.
2. **Critic update:** The critic networks are trained by minimizing the soft Bellman error. First, a target value is computed using the target networks:

$$y_t = r_t + \gamma \mathbb{E}_{a_{t+1} \sim \pi} \left[\min \left(Q_{\bar{\theta}_1}(s_{t+1}, a_{t+1}), Q_{\bar{\theta}_2}(s_{t+1}, a_{t+1}) \right) - \alpha \log \pi(a_{t+1} | s_{t+1}) \right]. \quad (99)$$

The critic loss is then defined as:

$$J_Q(\theta_i) = \mathbb{E}_{(s_t, a_t) \sim \mathcal{D}} \left[(Q_{\theta_i}(s_t, a_t) - y_t)^2 \right], \quad i = 1, 2. \quad (100)$$

The critic parameters θ_i are updated via gradient descent on J_Q .

3. **Actor update:** The policy is updated to maximize expected Q-value while encouraging entropy. This is equivalent to minimizing the actor loss:

$$J_\pi(\phi) = \mathbb{E}_{s_t \sim \mathcal{D}, a_t \sim \pi_\phi} \left[\alpha \log \pi_\phi(a_t | s_t) - \min \left(Q_{\theta_1}(s_t, a_t), Q_{\theta_2}(s_t, a_t) \right) \right]. \quad (101)$$

The actor parameters ϕ are updated via gradient descent on J_π .

4. **Target network update:** The target critic networks are updated using Polyak averaging:

$$\bar{\theta}_i \leftarrow \tau \theta_i + (1 - \tau) \bar{\theta}_i, \quad i = 1, 2, \quad (102)$$

where $\tau \in (0, 1)$ is a small smoothing coefficient.

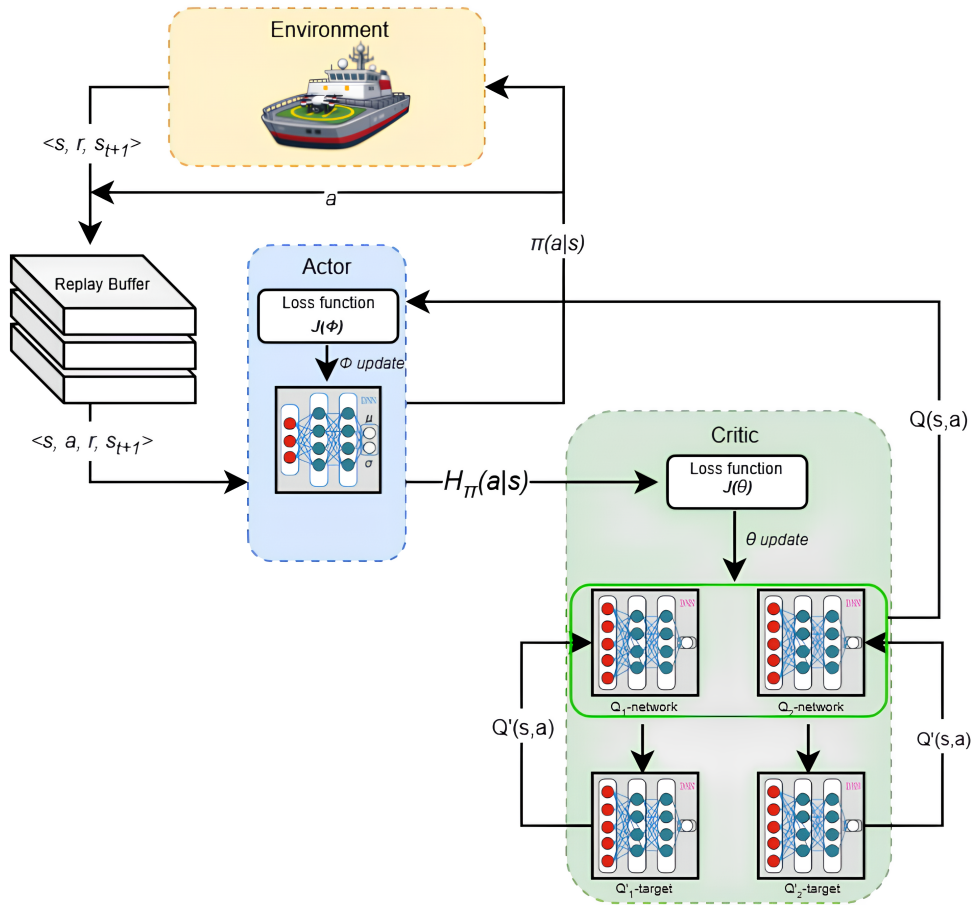


Figure 31: Schematic of the SAC framework, showing the interaction between the policy network (actor), twin Q-function networks (critics), and the entropy-regularized update mechanism.

6 VERIFICATION

The verification chapter assesses the correctness and physical plausibility of the three main components developed in this work: the quadplane dynamic model and PID controller, the aerodynamic model, and the RL implementation. Each component is evaluated through a dedicated set of targeted tests, the results of which are presented in the following sections.

6.1 Quadplane Dynamics and Controller Performance

Model verification is performed to evaluate whether the developed mathematical model reproduces the dominant dynamic behaviour of the quadplane within the intended operating regime. Since the quadplane operating in hover is an open-loop unstable system, open-loop verification is not feasible, as any perturbation from the equilibrium condition results in divergent behavior. Consequently, verification is conducted in a closed loop using the implemented PID controller. This ensures that the system remains stable throughout the simulations, allowing the model response to be compared with the expected closed-loop dynamics under representative operating conditions.

Table 17 provides a structured overview of the system-level verification tests, including the test identifier, description, applied stimulus, expected response, and the criteria used to assess success. The subsequent subsections provide a detailed discussion of each test and present the results.

Table 17: Quadplane dynamics and controller verification summary.

ID	Test	Input / Expected Output	Pass Criterion
QD-01	Hover Equilibrium	Zero position command $\rightarrow$ constant position, thrust $\approx mg$	Altitude error < 0.05 m; thrust within $\pm 5\%$ of mg
QD-02	Sign Convention	x/y step $\rightarrow$ correct pitch/roll sign	Attitude sign matches body-frame convention
QD-03	Motor Mixing	Axis step $\rightarrow$ correct differential RPM pattern	RPM matches allocation matrix; spurious excitation $< 5\%$ of hover RPM
QD-04	Climb & Descent	Altitude step $\rightarrow$ uniform RPM increase/decrease	All motors at equal RPM during vertical flight
QD-05	Step Response	Position step (all axes) $\rightarrow$ stable convergence	Overshoot $< 15\%$; rise time < 5 s; steady-state error < 0.05 m
QD-06	Yaw Decoupling	Yaw step $\rightarrow$ no roll/pitch excitation	Roll/pitch deviation $< 2^\circ$; altitude error < 0.05 m
QD-07	Cross-Coupling	Simultaneous steps; constrained actuators $\rightarrow$ yaw degrades first	Cross-axis deviation < 0.1 m; correct priority ordering confirmed
QD-08	Braking	Large step $\rightarrow$ opposing tilt during deceleration	Correct braking attitude observed; position overshoot $< 15\%$

Continued on next page

Table 17 – continued from previous page

ID	Test	Input / Expected Output	Pass Criterion
QD-09	Velocity Tracking	Velocity reference $\rightarrow$ correct tracking and position evolution	Velocity tracking error < 0.1 m/s; position consistent with integrated velocity

QD-01 Hover Equilibrium Check

The model is expected to maintain its initial position without angular commands or position error in the absence of disturbances. At the hover trim condition, thrust force must balance the vehicle weight,

$$\sum F_z = mg,$$

while all angular rates converge to zero. This test verifies the internal consistency of the equations of motion, gravitational model, and thrust model.

The simulation was initialized at the hover equilibrium condition, with the target position equal to the initial position, and was executed for 8 s.

The steady-state altitude error was found to be 0.0, the mean thrust matched mg exactly, and all angular rate channels converged to values close to zero. These results confirm that the hover trim condition is correctly represented by the model and that the controller maintains equilibrium without drift.

As shown in Figure 32, the UAV position remains constant throughout the simulation, with no deviation from the commanded hover point. Figure 33 confirms that all attitude states remain at their equilibrium values with no sustained deviation. Since no external disturbances are present, the controller commands a constant thrust equal to the vehicle weight, as shown in Figure 34a. All four rotor speeds converge to the same hover speed Ω_{hover} , as shown in Figure 34b, satisfying the thrust–weight equilibrium,

$$4k_F\Omega_{\text{hover}}^2 = mg \quad \Rightarrow \quad \Omega_{\text{hover}} = \sqrt{\frac{mg}{4k_F}} = \sqrt{\frac{5 \times 9.81}{4 \times 4.05 \times 10^{-7}}} \approx 5516 \text{ RPM} \quad (103)$$

where k_F is the rotor thrust coefficient, m is the total vehicle mass, and g is the gravitational acceleration.

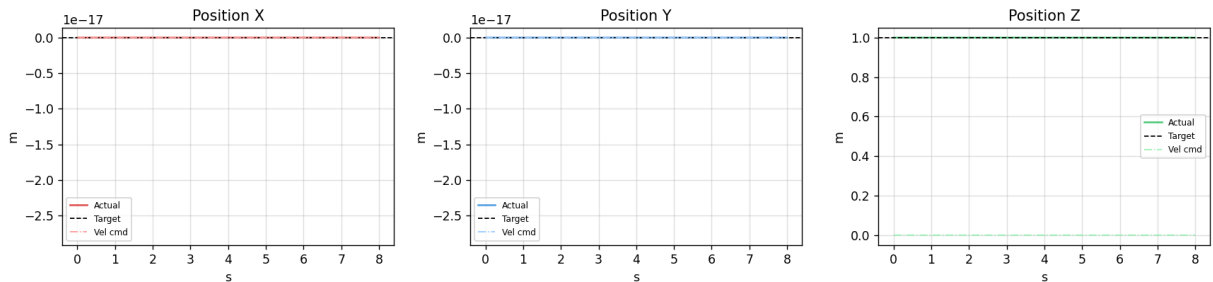


Figure 32: UAV position states during undisturbed hover.

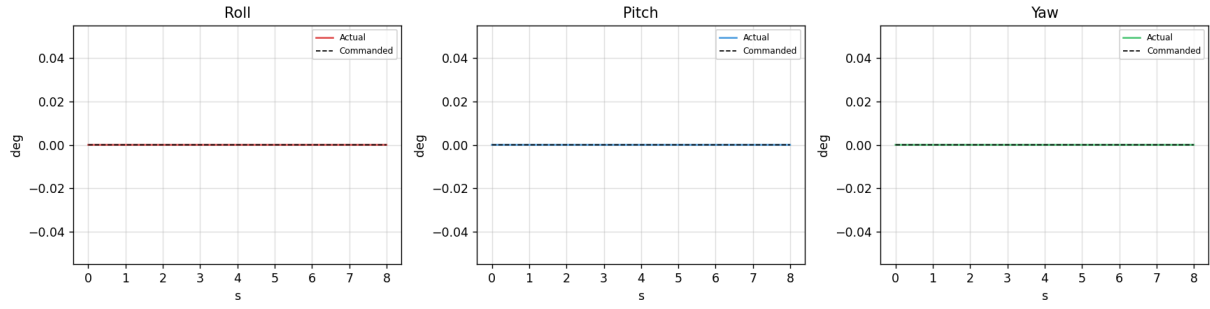
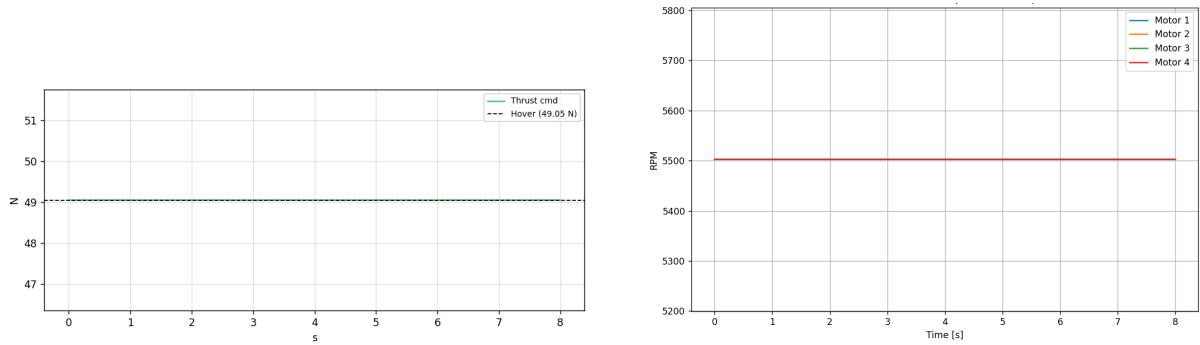


Figure 33: UAV attitude states during undisturbed hover.



(a) Total thrust command during undisturbed hover.

(b) Rotor speeds during undisturbed hover.

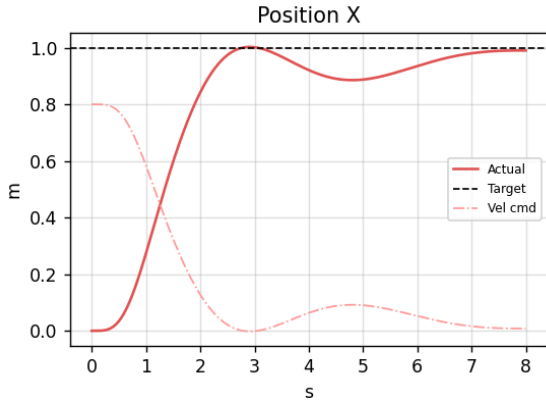
Figure 34: Verification of hover equilibrium in terms of thrust under undisturbed conditions: (a) total thrust command and (b) rotor speeds.

QD-02 Translation–Rotation Sign Convention

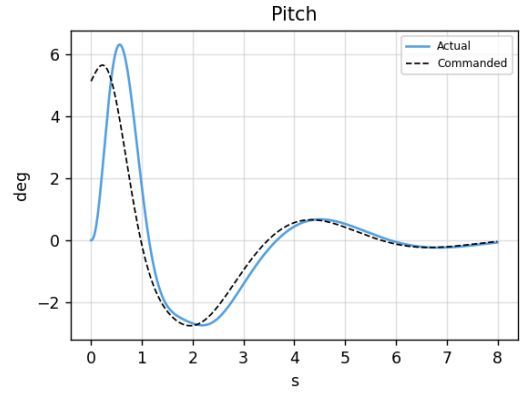
A quadplane operating in hover generates thrust only along the body z -axis. Consequently, translational motion in the horizontal plane is achieved by tilting the vehicle such that the thrust vector acquires a horizontal component. Forward motion along the x -axis therefore requires a nose-down pitch attitude, while lateral motion along the y -axis requires a roll attitude in accordance with the adopted body-frame convention. Correct implementation of this attitude–translation coupling is essential, as an incorrect sign would cause the vehicle to accelerate away from the commanded position.

This verification test consists of applying independent positive position step commands along the x - and y -axes while observing the resulting attitude response. For a positive x -axis position command, the controller is expected to generate a pitch command that tilts the vehicle toward the target. Similarly, a positive y -axis position command should produce the corresponding roll command. In both cases, the sign of the response must agree with the body-frame convention, while the magnitude should scale consistently with the applied position error.

Figure 35 shows the response to a positive position step along the x -axis, resulting in the expected pitch motion toward the commanded position, positive. The corresponding response for a positive y -axis position step is shown in Figure 36, where the expected roll motion is observed, negative. These results verify the correct implementation of the rotation matrix and the position–velocity–attitude control cascade.

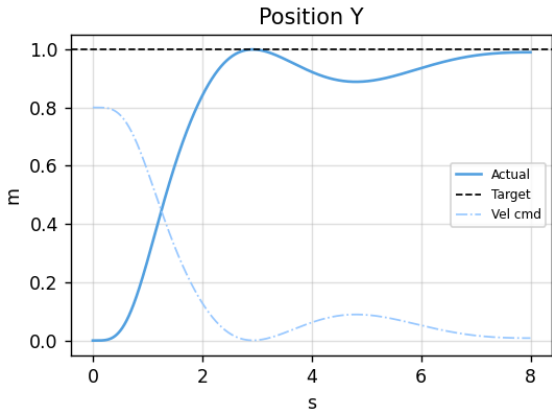


(a) Position response to an x -axis step command.

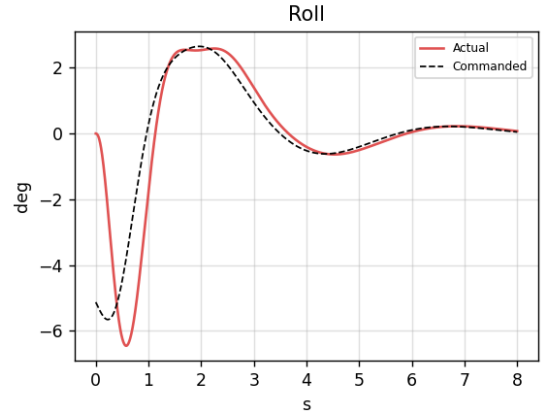


(b) Attitude response to an x -axis step command.

Figure 35: UAV response to a positive x -axis position step.



(a) Position response to a y -axis step command.



(b) Attitude response to a y -axis step command.

Figure 36: UAV response to a positive y -axis position step.

QD-03 Motor Mixing and RPM Allocation Correctness

As explained in QD-02, horizontal translation requires the vehicle to tilt, and this tilting is achieved by inducing a thrust imbalance between specific rotor pairs. To verify that this behaviour is correctly implemented by the motor allocation, the test applies separate x - and y -axis position step commands and evaluates whether the appropriate rotor pairs decrease in RPM while the opposing pairs increase.

For a positive x -step, the vehicle pitches forward, requiring the rear motor pair to increase above hover RPM while the forward pair decreases symmetrically. For a positive y -step, the equivalent roll demand produces a differential between the left and right motor pairs. For a yaw command, two diagonally opposite motors sharing the same rotation direction increase RPM while the remaining diagonal pair decreases, generating the required reactive torque without exciting roll or pitch.

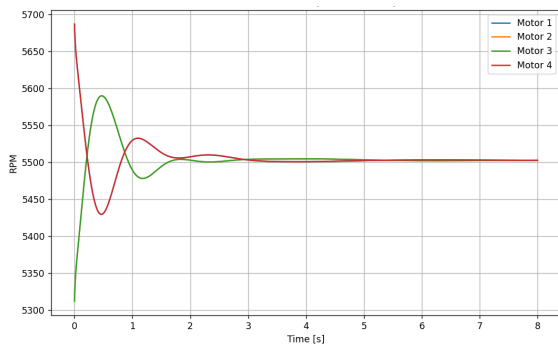
The results are summarised in Table 18, where $\uparrow$ denotes an increase above hover RPM and $\downarrow$

denotes a decrease.

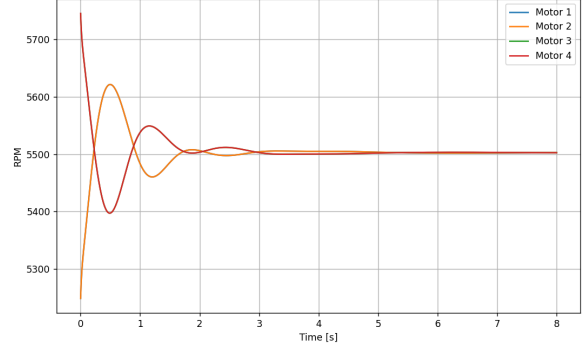
Table 18: Motor RPM response per commanded input. M1–M4 denote front-left, front-right, rear-left, and rear-right, respectively.

Input	M1 (BL)	M2 (FL)	M3 (FR)	M4 (BR)	Verified
$+x$ step (pitch forward)	↑	↓	↓	↑	✓
$+y$ step (roll left)	↓	↓	↑	↑	✓
$+\psi$ step (yaw)	↑	↓	↓	↑	✓

The RPM traces corresponding to the x - and y -axis step maneuvers are shown in Figure 37, confirming the qualitative patterns summarised in Table 18. The transient RPM differential tracks the position error visible in Figures 35 and 36, with the inter-motor spread peaking at the onset of the maneuver and decaying as the vehicle settles onto the new setpoint; the residual oscillations in rotor speed mirror the slight position oscillation observed in those plots.



(a) Rotor speeds during the $+x$ position step. The rear motor pair (M1, M4) increases above hover RPM while the forward pair (M2, M3) decreases symmetrically, producing the forward pitch required for x -axis translation.



(b) Rotor speeds during the $-y$ position step. The right motor pair (M3, M4) increases above hover RPM while the left pair (M1, M2) decreases symmetrically, producing the negative roll required for y -axis translation.

Figure 37: Motor RPM allocation during x - and y -axis position step manoeuvres.

Beyond directional correctness, the magnitude of the RPM differential is also evaluated as a consistency check. The peak deviation from the hover RPM observed during the x -step maneuver was $\Delta\Omega = 350$ RPM, resulting in a differential thrust per motor given by

$$\begin{aligned}\Delta T &= k_F (\Omega_{\text{hover}} + \Delta\Omega)^2 - k_F \Omega_{\text{hover}}^2 \\ &= 4.05 \times 10^{-7} (5516 + 350)^2 - 4.05 \times 10^{-7} (5516)^2 \approx 1.61 \text{ N}.\end{aligned}\tag{104}$$

Assuming a symmetric differential-thrust mixing strategy, in which one motor pair increases thrust by ΔT while the diagonally opposite pair decreases by the same amount, this differential thrust produces a net pitch torque of

$$\tau = 4 \Delta T \cdot \frac{L}{\sqrt{2}} = 4 \times 1.61 \times \frac{0.495}{\sqrt{2}} \approx 2.25 \text{ N}\cdot\text{m},\tag{105}$$

corresponding to a peak angular acceleration of

$$\ddot{\theta} = \frac{\tau}{J_{yy}} = \frac{2.25}{0.342} \approx 6.6 \text{ rad s}^{-2} \approx 377 \text{ deg s}^{-2}. \quad (106)$$

At this acceleration, a 6° pitch excursion would be reached in approximately $t = \sqrt{2\theta/\ddot{\theta}} = \sqrt{2 \times 6/377} \approx 0.18 \text{ s}$ under constant torque. The observed settling to 6° at $t \approx 0.4\text{--}0.5 \text{ s}$ is slower than this open-loop estimate, which is expected: the constant-torque calculation represents an upper bound on the achievable angular acceleration, whereas the closed-loop controller reduces commanded torque progressively as the attitude error decreases, and additional lags from motor spin-up dynamics and aerodynamic damping further slow the response. These results confirm that the allocation matrix output is consistent, both directionally and in order of magnitude.

QD-04 Climb and Descent

A climb and a descent command are applied to verify that vertical motion produces symmetric, collective thrust changes across all four rotors with no coupling into the attitude or horizontal position channels. The corresponding z -axis tracking with negligible horizontal displacement is presented in Figure 38 and the attitude states during the climb maneuver are shown in Figure 39, confirming that roll, pitch, and yaw remain undisturbed throughout. The per-motor RPM traces are shown in Figure 40a, where all four rotors respond identically. The descent maneuver produces the mirror response and is omitted here for brevity.

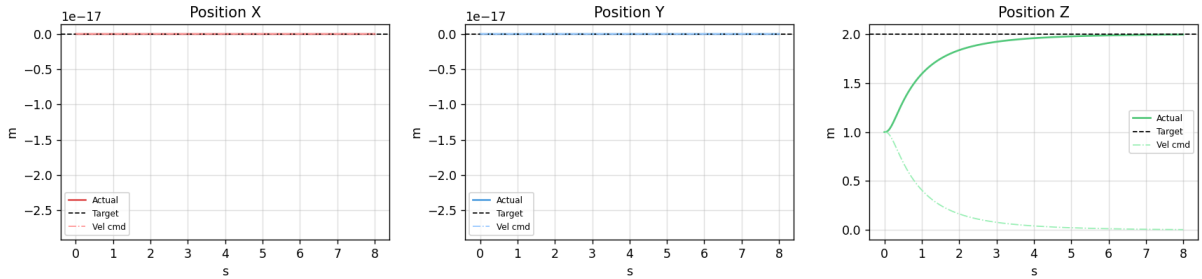


Figure 38: Position states during climb. The z -axis tracks the step setpoints with negligible displacement in x and y .

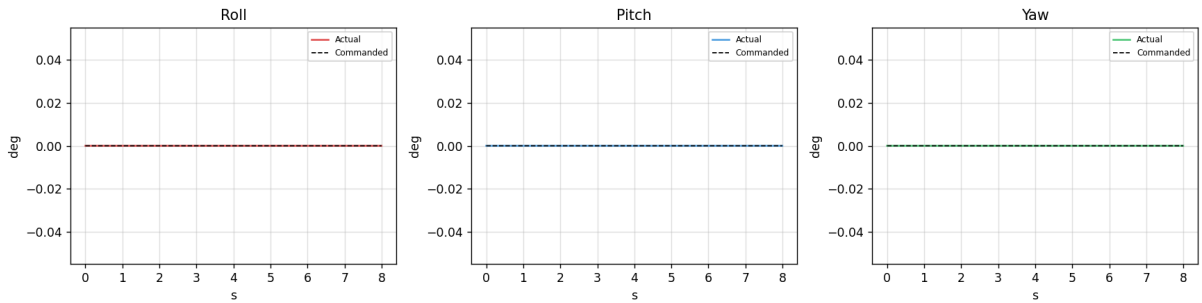
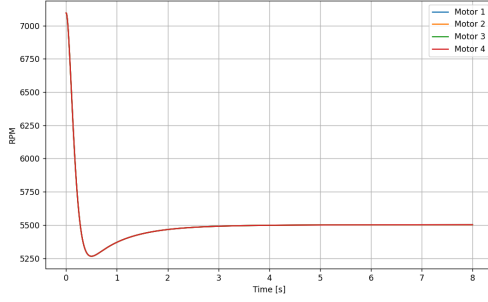
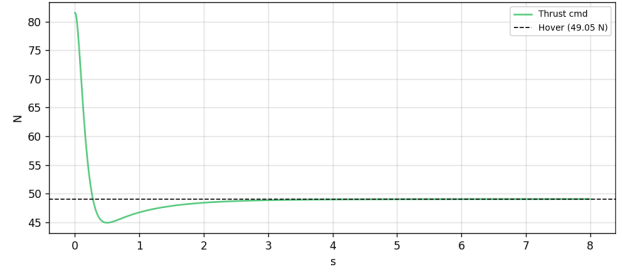


Figure 39: Attitude states during climb. Roll, pitch, and yaw remain at their equilibrium values throughout the maneuver.



(a) Per-motor RPM trace during climb.



(b) Total thrust command during climb.

Figure 40: Verification of the vertical climb response in terms of thrust: (a) per-motor RPM trace and (b) total thrust command during the maneuver.

The peak rotor speed during the climb manoeuvre increased from the hover value of $\Omega_{\text{hover}} \approx 5500$ RPM to approximately $\Omega_{\text{climb}} \approx 7000$ RPM, a differential of $\Delta\Omega = 1500$ RPM. The corresponding increase in collective thrust per motor is

$$\begin{aligned}\Delta T_{\text{motor}} &= k_F \Omega_{\text{climb}}^2 - k_F \Omega_{\text{hover}}^2 \\ &= 4.05 \times 10^{-7} (7000^2 - 5500^2) \\ &\approx 19.85 - 12.25 = 7.5 \text{ N.}\end{aligned}\tag{107}$$

so the total four-motor collective thrust increases from $4 \times 12.25 \approx 49$ N to $4 \times 19.85 \approx 79$ N, consistent with the observed thrust rising from approximately 50 N to 80 N shown in Figure 40b. The closed-loop settling time is longer than a constant-thrust kinematic estimate would suggest, as the cascade PID continuously reduces the thrust command as the altitude error decreases, with velocity damping further suppressing the peak acceleration.

QD-05 Reference Step Response and Tracking Performance

Closed-loop tracking performance is evaluated by applying independent position step inputs along the x -, y -, and z -axes, as well as a yaw step command of 1 rad. For each channel, the transient response is recorded and the following performance metrics are extracted: rise time t_r , percentage overshoot M_p , settling time t_s , and steady-state error e_{ss} . The results are compared against the control design specifications and summarised in Table 19.

Table 19: Step response performance metrics per channel.

Channel	t_r (s)	M_p (%)	t_s (s)	e_{ss} (m)
x	1.47	0.25	6.98	−0.02
y	1.45	0.19	7.01	−0.02
z	2.41	0.00	5.28	0.00
ψ	0.96	0.00	6.32	0.00

All four channels satisfy the overshoot, settling time, and steady-state error specifications, with negligible overshoot ($M_p \leq 0.25\%$) and steady-state errors of 0.02 m in translation and effectively zero in yaw. The rise-time specification is met in the x , y , and ψ channels, with

t_r ranging from 0.96 to 1.47 s. The z -channel, however, exhibits $t_r = 2.41$ s, marginally exceeding the 2 s target. Despite this, the z -channel settles fastest among all channels ($t_s = 5.28$ s) and with zero steady-state error, indicating that the slower initial rise does not compromise overall tracking accuracy.

Figures for position step responses have been presented in the preceding sections. The yaw step response to a 1 rad command is shown in Figure 41.

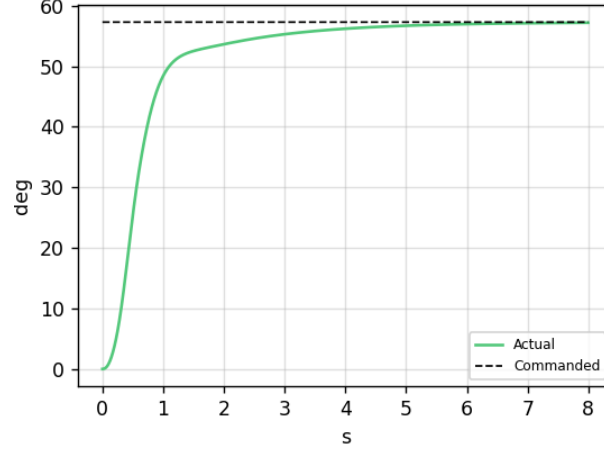


Figure 41: Yaw channel step response to a 1 rad command.

QD-06 Yaw Decoupling

A yaw command should induce rotation about the body z -axis without excitation of the roll or pitch channels. As established in QD-03, yaw torque is generated through the differential RPM between the two co-rotating diagonal motor pairs. If the allocation matrix is correctly implemented, this differential produces a net yaw moment without introducing roll or pitch moments. Any coupling into roll or pitch would indicate either an error in the allocation matrix or an unintended asymmetry in the motor model.

This test applies a yaw step command of 1 rad while holding the x -, y -, and z -references constant. The yaw channel is expected to track the reference with closed-loop behavior consistent with QD-05, while roll and pitch remain near zero and altitude is maintained. The position states during the maneuver are shown in Figure 42, confirming that the horizontal and vertical channels remain undisturbed, with residual oscillations on the order of 10^{-16} attributable to numerical error. The attitude response is shown in Figure 43, where the yaw channel tracks the reference while roll and pitch remain bounded within $\mathcal{O}(10^{-14})$ throughout, supporting the correct implementation of the motor allocation matrix.

QD-07 Cross-Coupling and Saturation Behavior

When multiple reference commands are applied simultaneously, the control architecture must respect the defined priority ordering and minimize unintended inter-axis coupling. Altitude is assigned the highest priority, such that vertical thrust is allocated before horizontal tracking, and roll and pitch are assigned higher priority than yaw in torque mapping to ensure stability. Yaw is therefore the first channel to degrade under actuator-constrained conditions.

This test applies simultaneous step inputs in the x -, y -, z -, and yaw channels. Under nominal

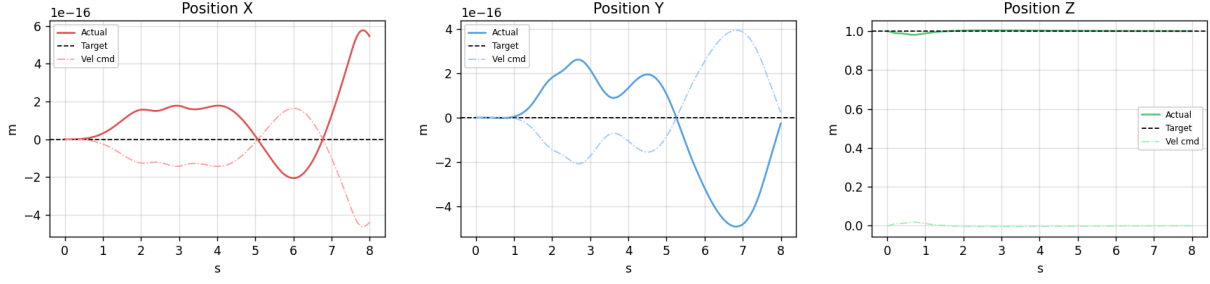


Figure 42: Position states during the yaw step maneuver.

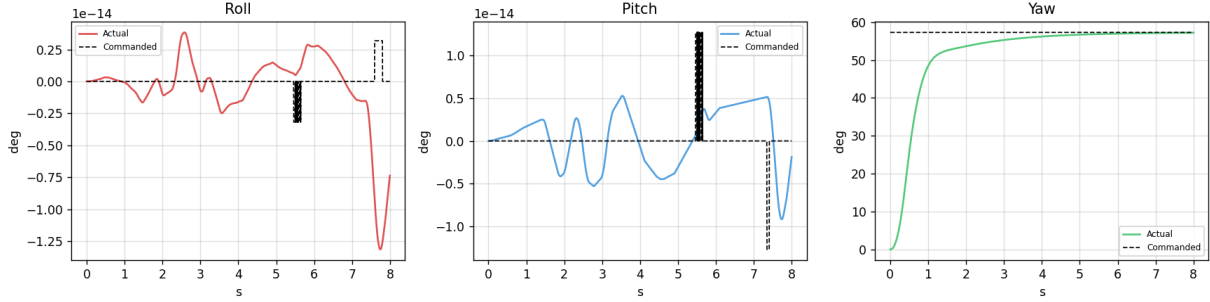


Figure 43: Attitude states during the yaw step maneuver.

conditions without actuator saturation, all channels are expected to track their respective references concurrently, with cross-axis deviations remaining within the specified tolerance. The resulting position response is shown in Figure 44, where all translational channels converge to their targets without significant inter-axis interaction.

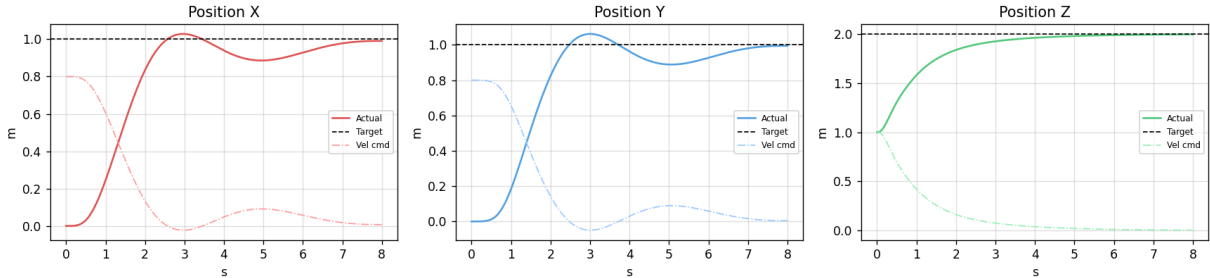


Figure 44: Position states during simultaneous x -, y -, z -, and yaw step commands under nominal actuator limits.

To evaluate the priority logic, the maximum rotor speed is constrained from 7800 RPM to 6000 RPM, just above hover, such that not all commands can be satisfied simultaneously. In this condition, yaw tracking is expected to degrade first while altitude and roll/pitch tracking are preserved. The constrained position response is shown in Figure 45, confirming that translational tracking is maintained under the reduced authority. As shown in Figure 46, yaw tracking is suppressed during the first 0.8 s of the maneuver while the position error is large and the available torque budget is consumed by the translational channels; once the position tracking demands reduce as the vehicle approaches its target, the residual torque capacity is restored and yaw converges to its reference. This behaviour confirms the correct implementation of the priority ordering in the control allocation layer.

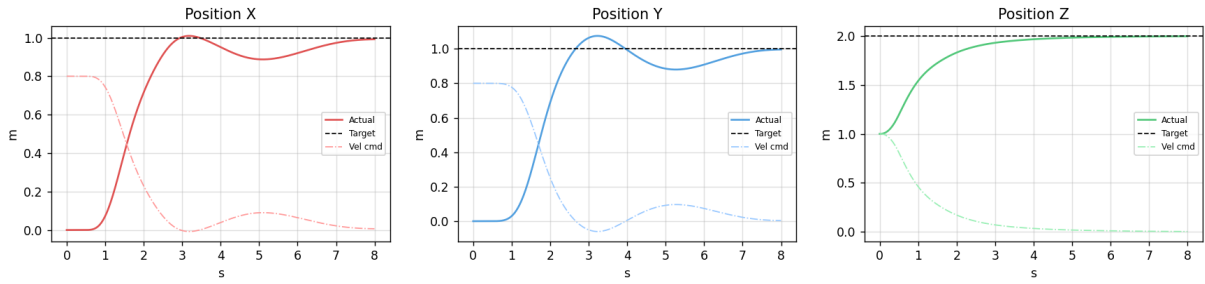
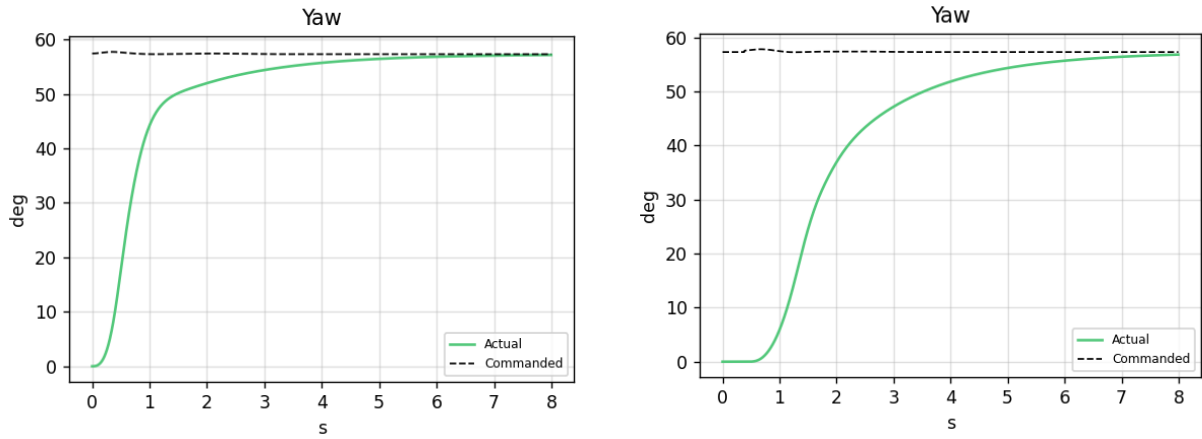


Figure 45: Position states during simultaneous step commands under constrained actuator limits (maximum 6000 RPM).



(a) Yaw response under nominal actuator limits.

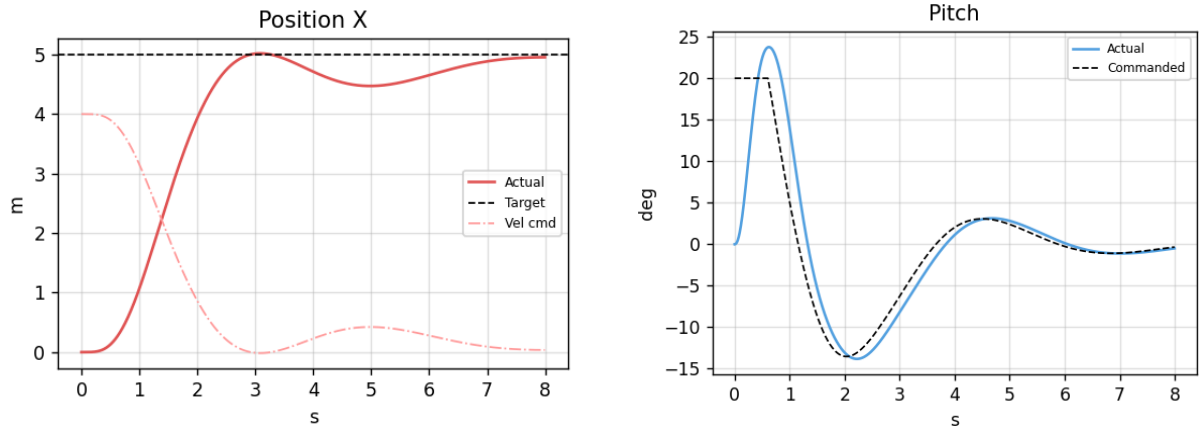
(b) Yaw response under constrained actuator limits (maximum 6000 RPM).

Figure 46: Yaw tracking under nominal and constrained actuator limits.

QD-08 Braking Behaviour and Position Derivative Action

When commanded to a target requiring a large translational displacement, the vehicle accelerates by pitching or rolling toward the target. As the position error decreases, the derivative term of the outer velocity loop begins to dominate, generating an attitude command opposing the direction of motion to decelerate the vehicle. This braking behaviour is essential to prevent overshoot and sustained oscillation around the target.

A large position step is commanded and the attitude response is evaluated during both the acceleration and deceleration phases. As shown in Figure 47, the vehicle tilts in the direction of motion during acceleration, reaching a maximum positive pitch angle of approximately 20° . From $t \approx 0.5$ s onwards the pitch command is gradually reduced as the position error decreases, transitioning to a negative pitch angle by $t \approx 1$ s as the braking phase begins. This deceleration is reflected in the position trace, where the rate of climb plateaus between $t \approx 1.2$ s and $t \approx 1.3$ s. A subsequent overshoot in the opposing pitch direction occurs as the derivative action applies excess braking, after which the vehicle pitches forward again at $t \approx 2$ s to correct the residual position error. Both pitch and position then settle to their equilibrium values, consistent with the expected behaviour of the derivative action in the outer position loop.



(a) Position response to a large x -axis step command.

(b) Pitch angle during the large x -axis step.

Figure 47: Position and pitch response during a large x -axis step command.

QD-09 Velocity Tracking

The control architecture supports two operating modes: a position-controlled mode, in which the outer position loop generates velocity references, and a velocity-controlled mode, in which velocity references are commanded directly, bypassing the position loop. This test verifies the correctness of the velocity-controlled mode by applying direct velocity step commands and comparing the simulated velocity states against the corresponding references.

As a consistency check, the simulated position state is additionally compared against the numerical integral of the commanded velocity profile. Agreement between these two quantities confirms that the kinematic relationship between velocity and position is correctly modeled, and that bypassing the position loop introduces no unintended coupling or drift. The velocity-tracking responses have been presented in the preceding sections (Section 5), while the position-integrated-velocity comparison is shown in Figure 48. Both results confirm that velocity-tracking accuracy and kinematic consistency remain within the defined thresholds, thereby verifying the correctness of the velocity-loop implementation.

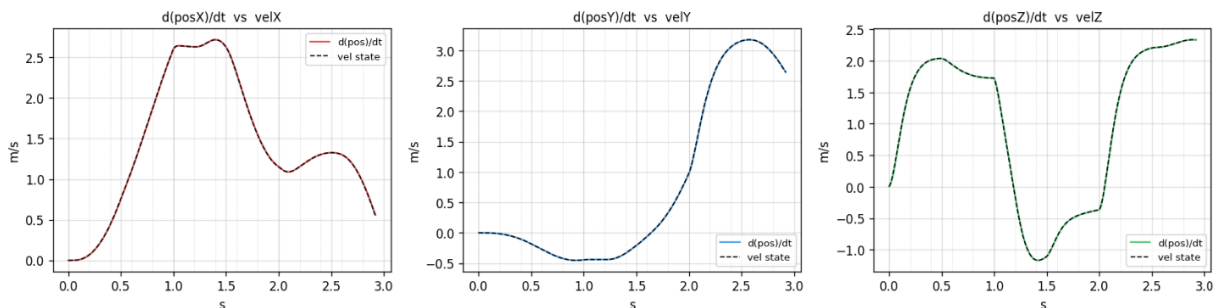


Figure 48: Kinematic consistency check: the numerical derivative of the simulated position compared against the velocity state, confirming that position and velocity are mutually consistent.

6.2 Aerodynamic Model Verification

The aerodynamic model is developed to cover the full flight envelope of the UAV, spanning angles of attack and sideslip from -180° to $+180^\circ$. The lift and drag forces are defined as:

$$L = \frac{1}{2}\rho V^2 S C_L(\alpha), \quad D = \frac{1}{2}\rho V^2 S C_D(\alpha) \quad (108)$$

where $C_L(\alpha)$ and $C_D(\alpha)$ are the lift and drag coefficients as functions of the angle of attack α , ρ is the air density, V is the airspeed magnitude, and S is the reference wing area.

In addition, a lateral aerodynamic force due to sideslip is included, modeled as

$$Y = -\tanh\left(\frac{v}{\varepsilon}\right) \bar{q} S C_Y(\beta), \quad (109)$$

where β is the sideslip angle, and $C_Y(\beta)$ is defined as a scaled function of the drag coefficient. The hyperbolic tangent term ensures a smooth transition of the force direction around zero lateral velocity, with ε preventing numerical discontinuities in the vicinity of $v = 0$.

Verification of the aerodynamic model requires confirming the correct computation of the aerodynamic angles, aerodynamic coefficients, and resulting forces, as well as their effect on tracking performance and actuator saturation across representative flow conditions. The following verification tests are therefore defined:

- **AD-01–AD-03:** Headwind at increasing magnitudes — verifying angle of attack, drag, lift, pitch compensation, and thrust–drag equilibrium.
- **AD-04–AD-05:** Vertical downwash at increasing magnitudes — verifying angle of attack near -90° , vertical drag loading, and saturation behaviour.
- **AD-06:** Combined headwind and downwash — verifying compound angle of attack, combined force decomposition, and negative lift.
- **AD-07–AD-08:** Sidewind at increasing magnitudes — verifying sideslip angle, side force, and decoupling from lift and drag.
- **AD-09–AD-10:** Climb and descent — verifying angle of attack during vertical motion phases and absence of spurious aerodynamic forces.
- **AD-11–AD-14:** Position tracking under wind disturbances — verifying closed-loop position tracking performance under various scenarios to confirm that the aerodynamic model and controller interact correctly under sustained wind loading.

Table 20 provides a structured overview of all aerodynamic verification tests.

Table 20: Aerodynamic model verification summary.

ID	Test	Input / Expected Output	Pass Criterion
AD-01	Headwind AoA	Headwind 5–15 m/s $\rightarrow \alpha \in [0^\circ, -15^\circ]$; no sideslip or side force	$\alpha \in [0^\circ, -15^\circ]$; $ \beta < 1^\circ$; side force < 0.1 N
AD-02	Headwind Drag & Pitch	Headwind 5–15 m/s $\rightarrow$ drag in $-x$; pitch into wind; thrust balances drag	Drag scales with V^2 ; thrust x -component matches drag within 5%
AD-03	Headwind Lift	Headwind 5–15 m/s $\rightarrow$ positive lift consistent with cambered aerofoil $C_L(\alpha)$	Lift sign and trend consistent with $C_L(\alpha)$ polar
AD-04	Downwash AoA	Vertical downwash 5–15 m/s $\rightarrow \alpha \approx -90^\circ$; no sideslip; lift ≈ 0	$ \alpha - (-90^\circ) < 5^\circ$; lift < 0.1 N; $ \beta < 1^\circ$
AD-05	Downwash Loading	Vertical downwash 5–15 m/s $\rightarrow$ drag in $-z$; saturation at 15 m/s	Drag scales with V^2 ; position held up to 10 m/s; saturation at 15 m/s
AD-06	Combined Wind	5 m/s headwind + 5 m/s downwash $\rightarrow \alpha \approx -45^\circ$; drag in $-x$ and $-z$	$ \alpha - (-45^\circ) < 5^\circ$; lift negative; side force < 0.1 N
AD-07	Sidewind Sideslip	Sidewind 5–15 m/s $\rightarrow \beta \approx 90^\circ$; side force in $-y$; no lift or drag	$ \beta - 90^\circ < 2^\circ$; side force scales with V^2 ; lift, drag < 0.1 N
AD-08	Sidewind Force Magnitude	Sidewind vs. matched downwash speed $\rightarrow$ side force $\approx 4\times$ lower than axial drag	Side force magnitude consistent with C_D lateral/axial ratio
AD-09	Climb AoA	Climb command $\rightarrow \alpha \approx -90^\circ$; no sideslip; lift ≈ 0	$\alpha \approx -90^\circ$; lift < 0.1 N; $ \beta < 1^\circ$
AD-10	Descent AoA	Descent command $\rightarrow \alpha \approx +90^\circ$; no sideslip; lift ≈ 0	$\alpha \approx +90^\circ$; lift < 0.1 N; $ \beta < 1^\circ$
AD-11	Tracking under Headwind	10 m/s headwind $\rightarrow$ position maintained at commanded setpoint	Steady-state position error < 0.1 m
AD-12	Tracking under Downwash	10 m/s downwash $\rightarrow$ altitude maintained up to saturation limit	Steady-state altitude error < 0.1 m
AD-13	Tracking under Combined Wind	5 m/s headwind + 5 m/s downwash $\rightarrow$ position maintained in all axes	Position error < 0.1 m in all axes
AD-14	Tracking under Sidewind	5 m/s sidewind $\rightarrow$ lateral position maintained at commanded setpoint	Lateral position error < 0.1 m

AD-01 Headwind Angle of Attack

The angle of attack α is defined as the angle between the incoming airflow and the body x -axis, computed from the wind velocity resolved in the body frame. For a pure headwind during hover, the airflow arrives along the body x -axis and α varies with the nose-down pitch attitude adopted by the controller to oppose the wind-induced drag. Since no lateral wind component is present, the sideslip angle β is expected to remain near zero and no side force should be generated.

The UAV is commanded to maintain its initial hover position under headwind speeds of 5, 10, and 15 m/s. The steady-state aerodynamic angles and side force for each condition are summarised in Table 21, and the full time histories of lift, drag, side force, lift and drag coefficients, and angle of attack are shown in Figures 49–51.

Table 21: Steady-state aerodynamic angles and side force under headwind conditions.

Headwind (m/s)	α (deg)	β (deg)	Side Force (N)
5	−0.6	0.0	0.0
10	−2.9	0.0	0.0
15	−7.2	0.0	0.0

The angle of attack remains within $[0^\circ, -10^\circ]$ across all three conditions, consistent with the increasing nose-down pitch attitude commanded by the controller as headwind magnitude grows. The sideslip angle remains at zero throughout all three tests and no side force is produced, confirming that the aerodynamic model correctly identifies a symmetric headwind as a zero- β condition with no lateral loading.

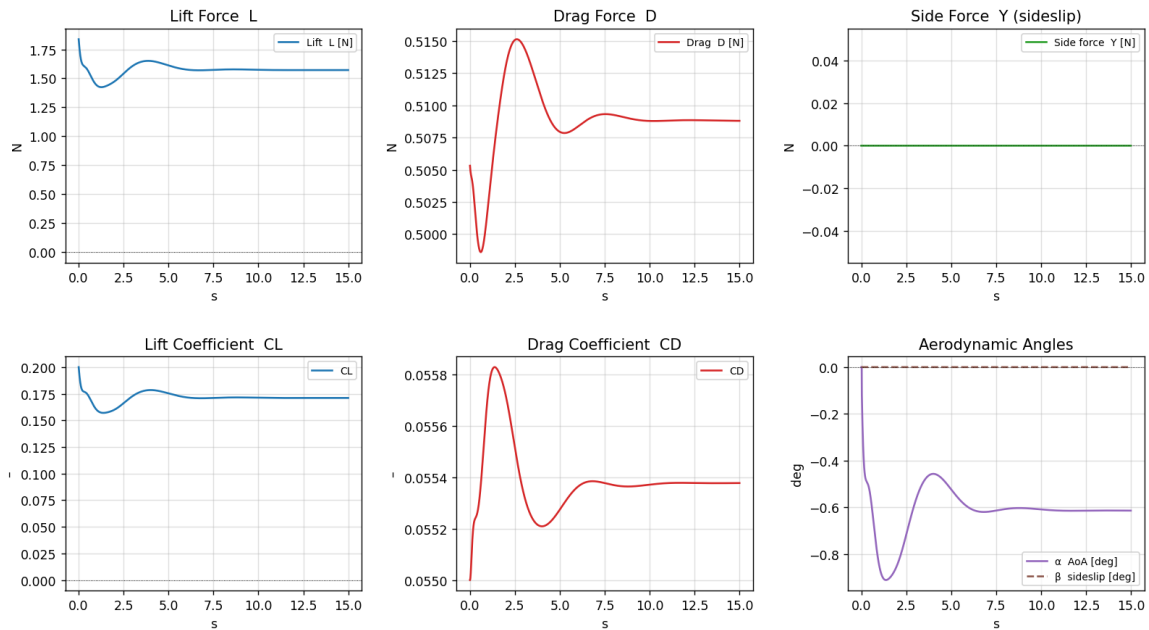


Figure 49: Aerodynamic response at 5 m/s headwind.

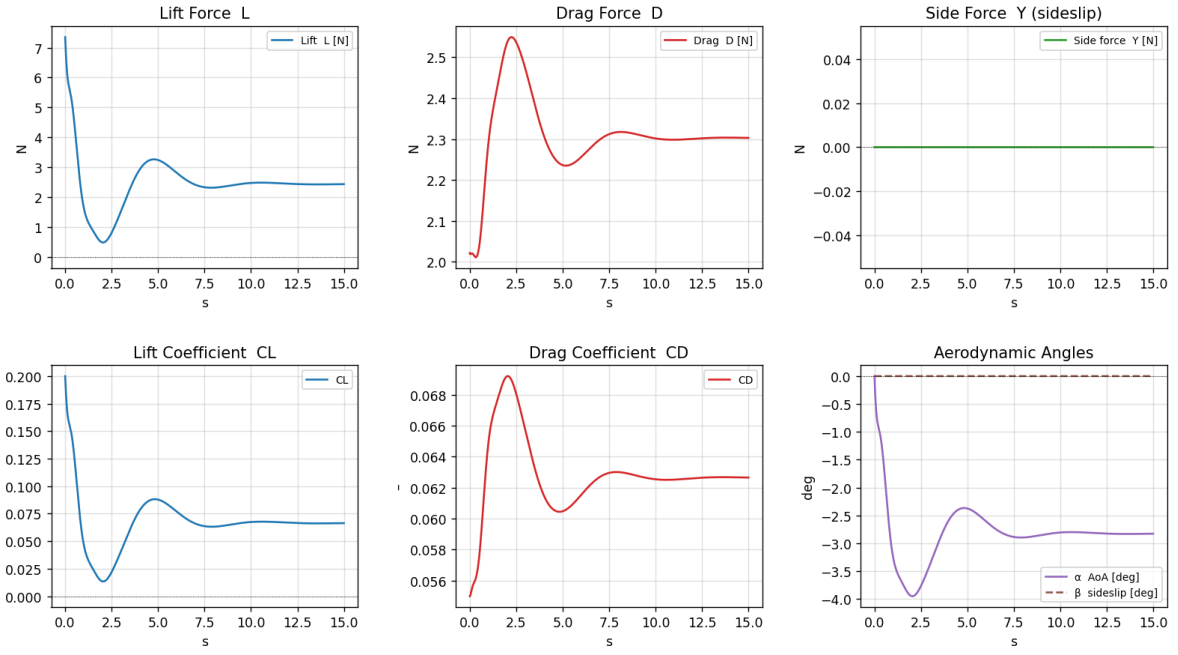


Figure 50: Aerodynamic response at 10 m/s headwind.

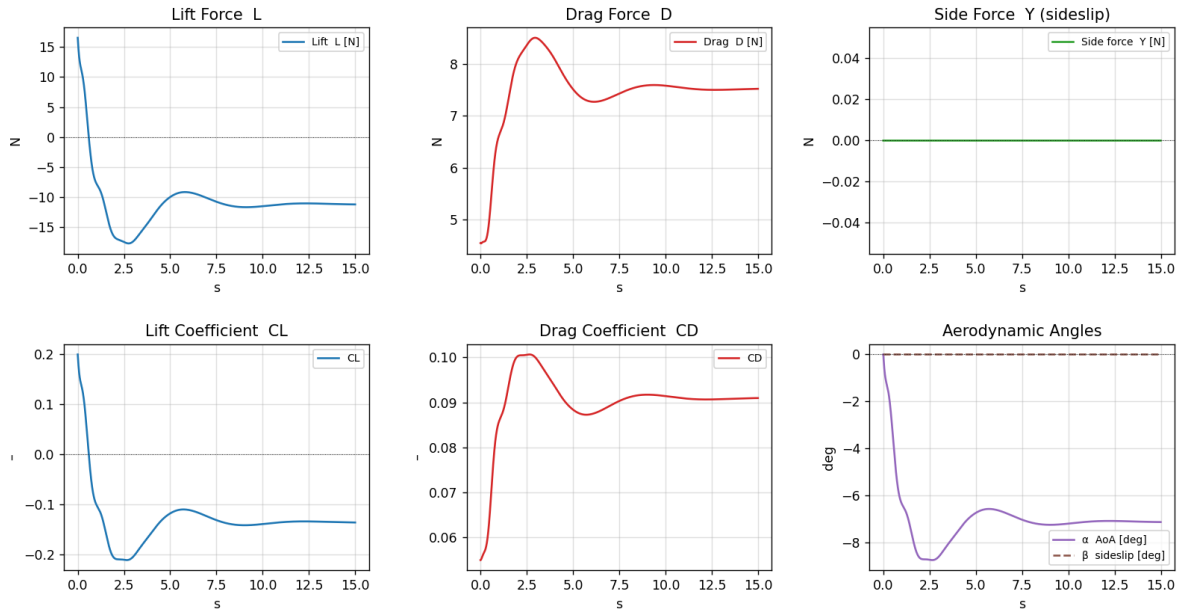


Figure 51: Aerodynamic response at 15 m/s headwind.

AD-02 Headwind Drag and Pitch Compensation

Under a headwind, the aerodynamic drag force acts along the $-x$ direction in the wind frame and opposes the vehicle's ability to maintain position. To counteract this, the controller pitches the vehicle nose-down into the wind, tilting the thrust vector such that its horizontal component balances the drag force. At equilibrium,

$$T \sin \theta = D = \frac{1}{2} \rho V^2 S C_D(\alpha), \quad (110)$$

where θ is the pitch angle, T is the total thrust, and D is the aerodynamic drag force. This equilibrium condition provides a direct consistency check between the drag model and the thrust model.

It is worth noting that the lift force behaviour differs between the low- and high-wind cases, as can be seen in Figures 49–51. At 5 and 10 m/s, the angle of attack remains small and the lift coefficient is positive, meaning the wing generates a small upward force that partially offsets the rotor thrust required to support the vehicle weight. At 15 m/s, the larger nose-down pitch drives α into a regime where the lift contribution becomes negative, adding a downward force component along the body z -axis and an additional push along the body x -axis, which the thrust model must compensate for.

The UAV is commanded to maintain hover position under headwind speeds of 5, 10, and 15 m/s. At steady state, the drag force, pitch angle, total thrust, and horizontal thrust component are recorded and compared in Table 22. The balance error is defined as the relative difference between the measured drag force and the horizontal thrust component $T \sin \theta$.

Table 22: Steady-state drag force and thrust compensation under headwind conditions.

Headwind (m/s)	D (N)	θ (deg)	$T \sin \theta$ (N)	Horiz. Err. (%)	$T \cos \theta$ (N)	Vert. Err. (%)
5	0.51	0.6	0.50	2.5	47.7	2.8
10	2.32	2.8	2.28	1.7	46.6	5.0
15	7.50	7.2	7.63	1.8	60.4	23.1

The horizontal thrust component $T \sin \theta$ matches the measured drag force to within 2.5% across all three conditions. The vertical component $T \cos \theta$ remains close to mg at 5 and 10 m/s, where the positive lift contribution partially unloads the rotors, resulting in $T \cos \theta < mg$. At 15 m/s the vertical error rises to 23.1%, consistent with the transition to negative lift of approximately -10 N, which adds a downward load on top of gravity; the expected vertical thrust in this case is $mg + |L| = 49.05 + 10 = 59.05$ N, in close agreement with the computed $T \cos \theta = 60.4$ N. The results confirm that the drag model, aerodynamic angle computation, and thrust allocation are mutually consistent and that the controller correctly pitches the vehicle to equilibrium under steady headwind loading.

AD-03 Headwind Lift Behaviour

In cruise mode, horizontal motion is achieved by the pusher/puller propeller acting along the body x -axis, eliminating the need for the rotor tilting mechanism used in hover. As a result, the UAV maintains a positive angle of attack and experiences a substantial lift force that contributes to sustaining flight. In hover, however, the dynamics differ fundamentally. The UAV must tilt to generate horizontal motion, and in the presence of a headwind the controller pitches nose-down into the wind to counteract drag, driving the angle of attack to negative values. Due to the cambered aerofoil geometry, a non-zero lift coefficient exists at zero angle of attack, and at the small nose-down attitudes required to maintain hover against low headwinds, the angle of attack remains above the zero-lift angle and lift remains positive. As the headwind increases and the controller commands larger nose-down pitch angles to generate sufficient horizontal thrust, the angle of attack decreases further and the lift force

transitions from a positive (upward) to a negative (downward) contribution. This behaviour must be correctly captured by the aerodynamic model.

The UAV is commanded to maintain a hover position under headwind speeds of 5, 10, and 15 m/s. The steady-state lift force, lift coefficient, and angle of attack are summarised in Table 23. Verification is performed by comparing the simulated lift force against the value predicted directly from the aerodynamic lift model and the overall magnitude and sign change with changing wind speeds. Lift is modeled as,

$$L = \frac{1}{2}\rho V^2 S C_L(\alpha), \quad (111)$$

where $\rho = 1.225 \text{ kg/m}^3$ is the air density and $S = 0.6 \text{ m}^2$ is the wing reference area. The predicted lift forces for each condition are computed as follows:

$$L_5 = \frac{1}{2}(1.225)(5^2)(0.6)(0.175) = 1.61 \text{ N}, \quad (112)$$

$$L_{10} = \frac{1}{2}(1.225)(10^2)(0.6)(0.065) = 2.39 \text{ N}, \quad (113)$$

$$L_{15} = \frac{1}{2}(1.225)(15^2)(0.6)(-0.130) = -10.76 \text{ N}. \quad (114)$$

Table 23: Steady-state lift force, lift coefficient, and angle of attack under headwind conditions.

Headwind (m/s)	α (deg)	C_L (-)	L_{model} (N)	$L_{\text{predicted}}$ (N)	Error (%)
5	-0.5	0.175	1.53	1.61	5.2
10	-2.3	0.065	2.40	2.39	0.4
15	-7.5	-0.130	-10.92	-10.76	1.5

At 5 m/s the angle of attack is small and negative, yet the cambered aerofoil still produces positive lift owing to the non-zero C_{L0} . At 10 m/s the lift remains positive, although the lift coefficient decreases as the nose-down pitch attitude increases, with the increasing dynamic pressure compensating for the reduced coefficient. At 15 m/s the angle of attack passes the zero-lift angle, resulting in negative lift that adds a downward aerodynamic load requiring additional rotor thrust to maintain hover.

The lift values predicted from the aerodynamic polar agree with the simulated lift force to within 5.2% across all three operating conditions. The small residual differences are attributed to transient attitude oscillations that slightly alter the instantaneous angle of attack at the sampling instant. These results confirm that the $C_L(\alpha)$ polar and the associated lift sign convention are correctly implemented throughout the headwind hover envelope.

AD-04 Downwash Angle of Attack

When the UAV is subjected to a vertical downwash, the incoming airflow arrives from directly above the vehicle, perpendicular to the body x -axis. In this condition, the angle of attack is expected to approach -90° , as the airflow is aligned with the negative body z -axis. Since the flow has no lateral component, the sideslip angle should remain near zero and no lift force should be generated, as the flow is approximately perpendicular to the wing surface. A small

deviation from the theoretical -90° is acceptable, as the controller may induce a minor pitch adjustment to maintain position under the vertical load.

The UAV is commanded to maintain its initial hover position under vertical downwash speeds of 5, 10, and 15 m/s. The angle of attack, sideslip angle, and lift force are recorded at steady state for each condition and summarised in Table 24. All three conditions produce $\alpha = -90^\circ$ exactly, with zero sideslip and zero lift force, confirming that the aerodynamic angle computation correctly identifies a purely vertical flow as a -90° angle of attack with no lateral component. The absence of lift is physically consistent. The full aerodynamic time histories for each downwash speed are shown in Figures 52–54.

Table 24: Steady-state aerodynamic angles and lift force under vertical downwash conditions.

Downwash (m/s)	α (deg)	β (deg)	L (N)
5	-90	0.0	0.0
10	-90	0.0	0.0
15	-90	0.0	0.0

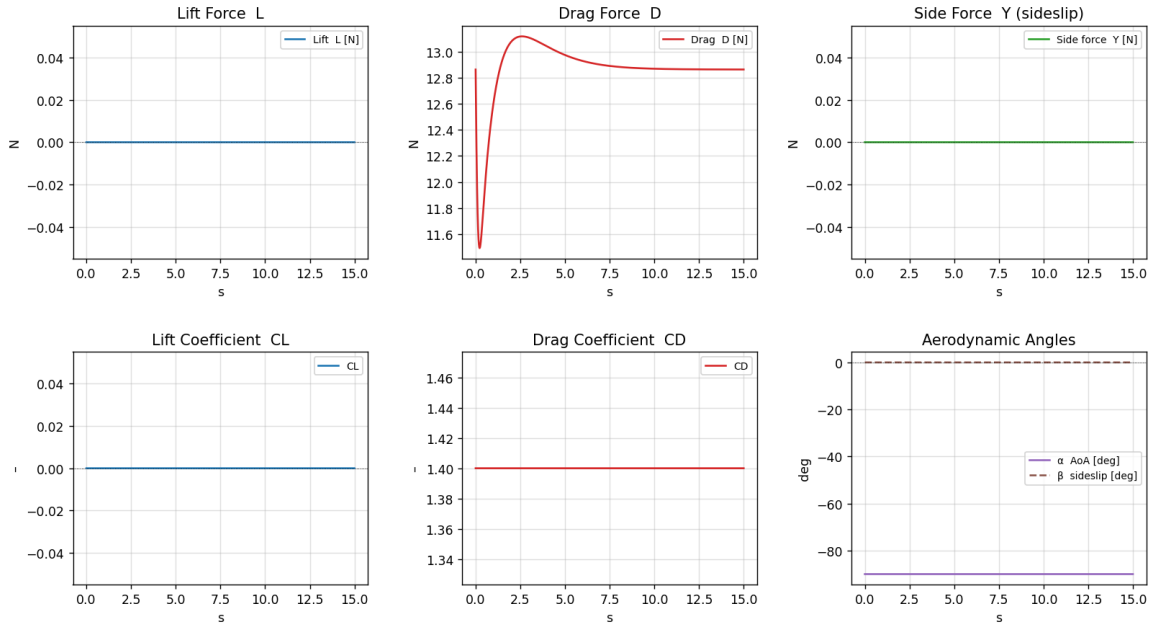


Figure 52: Aerodynamic response at 5 m/s vertical downwash.

AD-05 Downwash Drag Loading and Saturation

Under a vertical downwash, the aerodynamic drag force acts entirely along the $-z$ axis, pushing the vehicle downward. To maintain its vertical position, the controller must increase collective rotor thrust to counteract both the vehicle weight and the aerodynamic drag load. The total thrust required at equilibrium is therefore:

$$T = mg + D = mg + \frac{1}{2}\rho V^2 S C_D(\alpha), \quad (115)$$

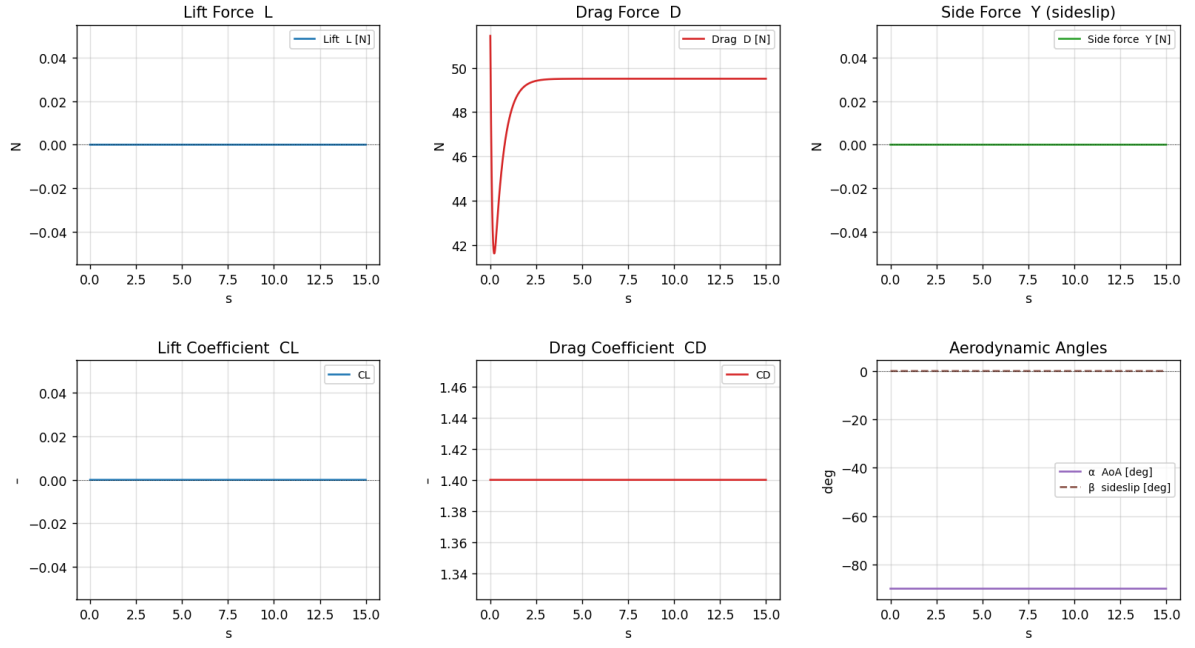


Figure 53: Aerodynamic response at 10 m/s vertical downwash.

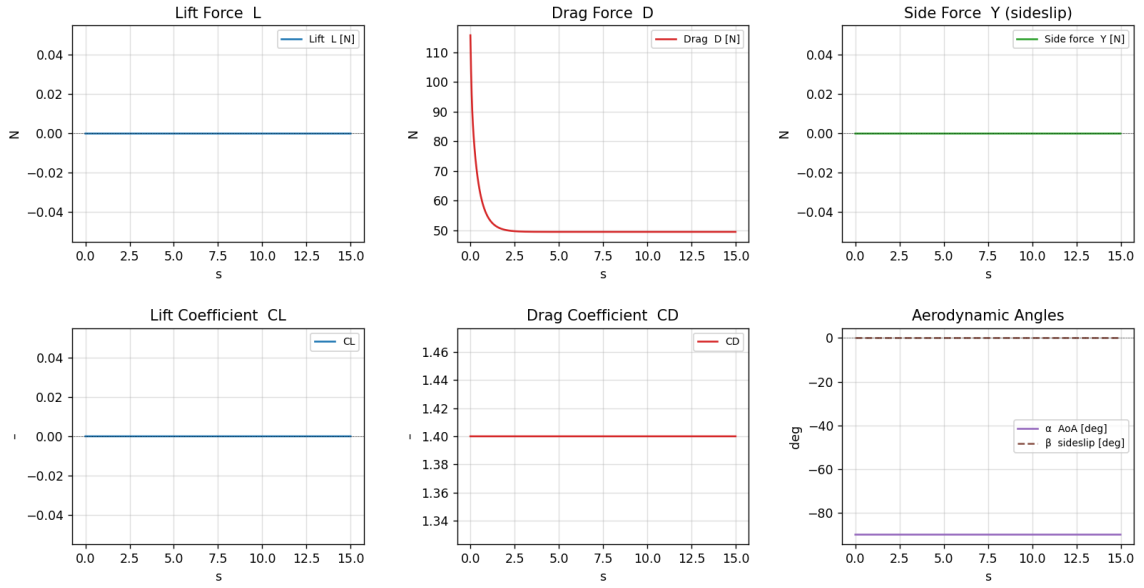


Figure 54: Aerodynamic response at 15 m/s vertical downwash.

where D is the drag force acting in the $-z$ direction and $\alpha \approx -90^\circ$ as established in AD-04. As the downwash magnitude increases, the drag load grows quadratically, eventually exceeding the available thrust budget and causing actuator saturation. Beyond this point, the controller can no longer maintain the commanded position.

The UAV is commanded to maintain hover position under downwash speeds of 5, 10, and 15 m/s. The drag force, total thrust, and saturation state are recorded at steady state and summarised in Table 25.

Table 25: Steady-state drag loading and thrust response under vertical downwash conditions.

Downwash (m/s)	D (N)	T (N)	Saturated	Position Maintained
5	12.83	61.98	No	✓
10	49.69	98.87	Yes	×
15	49.69	98.87	Yes	×

The aerodynamic time histories presented in Figures 52 and 53 both exhibit a brief reduction in drag load immediately after the downwash is applied. This transient dip occurs because the initial downward push displaces the vehicle below its commanded altitude; as the vehicle accelerates downward with the flow, the relative airspeed between the vehicle and the downwash decreases, momentarily reducing the drag load. The controller subsequently recovers the vehicle toward its setpoint, restoring the full relative airspeed and drag load.

At 5 m/s the drag load of 12.83 N is within the available thrust margin above hover ($mg \approx 49$ N), and the controller successfully maintains altitude with a total thrust of 61.98 N. At 10 m/s the drag load rises to 49.69 N, consuming the entire thrust margin and saturating the actuators at 98.87 N; the controller can no longer maintain the commanded altitude and the vehicle descends. The 15 m/s case produces identical steady-state values to the 10 m/s case, as shown in Figure 54. This is explained by the descent dynamics: the initial drag at 15 m/s is substantially higher, on the order of 110 N as visible in Figure 54, driving the vehicle downward rapidly. As the vehicle drifts down with the flow the relative airspeed reduces to approximately 10 m/s, at which point the drag force settles to the same 49.69 N observed in the 10 m/s saturated case. The actuators remain saturated throughout and position is not recovered. These results confirm that the drag model, thrust-saturation logic, and altitude controller interact correctly as vertical aerodynamic loading increases.

AD-06 Combined Headwind and Downwash

When the UAV is subjected to simultaneous headwind and downwash components, the resulting airflow arrives at an intermediate angle between the body x - and z -axes. For equal headwind and downwash magnitudes of 5 m/s, the resultant flow direction bisects the two axes, producing an angle of attack of approximately -45° . The aerodynamic force now has components acting in both the $-x$ and $-z$ world-frame directions, imposing simultaneous horizontal and vertical loads on the vehicle. As established in AD-03, negative angles of attack in the presence of a cambered aerofoil produce a negative (downward) lift force, which adds to the vertical aerodynamic load and further increases the thrust required to maintain position. Since the flow has no lateral component, the sideslip angle should remain near zero and no side force should be generated.

The UAV is commanded to maintain its initial hover position under a simultaneous 5 m/s headwind and 5 m/s downwash. The angle of attack, sideslip angle, lift force, and world-frame aerodynamic force components are recorded at steady state and summarised in Table 26.

The angle of attack of -45.80° is consistent with the theoretical prediction of -45° for equal headwind and downwash components, with the small deviation attributable to the pitch attitude adopted by the controller to counteract the horizontal load. The combined aerodynamic force in the world frame acts primarily in the $-z$ direction (-18.87 N), reflecting

Table 26: Steady-state aerodynamic quantities under combined headwind and downwash.

α (deg)	β (deg)	L (N)	$F_{x,\text{world}}$ (N)	$F_{z,\text{world}}$ (N)	Side Force (N)
-45.80	0.00	-12.92	-0.46	-18.87	0.00

the dominant downward loading from both the drag projected onto the vertical axis and the negative lift contribution from the cambered aerofoil at negative α . The horizontal component (-0.46 N) is small, as the near-symmetric flow angle distributes the drag nearly equally between the two axes and the lift partially cancels the horizontal drag projection. The sideslip angle remains at zero and no side force is produced, confirming that the aerodynamic model correctly handles compound flow conditions involving simultaneous longitudinal and vertical wind components.

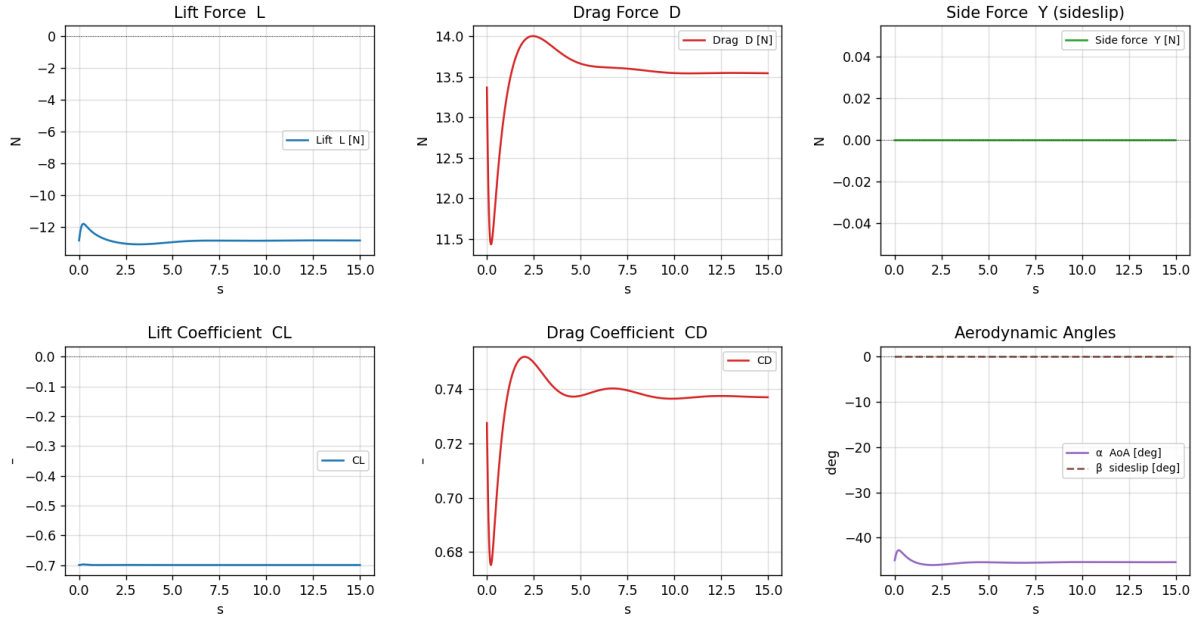


Figure 55: Aerodynamic response under combined 5 m/s headwind and 5 m/s downwash.

AD-07 Sidewind Sideslip and Side Force

When the UAV is subjected to a pure sidewind, the incoming airflow arrives perpendicular to the body x - z plane, entirely along the lateral axis. In this condition, the sideslip angle β is expected to approach 90° , as the flow is aligned with the body y -axis. Since the flow has a negligible or no longitudinal component in the world frame, the angle of attack should remain small, and no meaningful lift or drag should be generated at low sidewind speeds. The dominant aerodynamic effect is a side force acting in the $-y$ direction, which the controller counteracts by rolling, generating a horizontal thrust component along the y -axis.

The UAV is commanded to maintain its initial hover position under sidewind speeds of 2 and 5 m/s. The sideslip angle, side force, lift force, drag force, and induced roll angle are recorded at steady state for each condition and summarised in Table 27. The aerodynamic time histories for each condition are shown in Figures 56–57.

Table 27: Steady-state aerodynamic quantities under sidewind conditions.

Sidewind (m/s)	β (deg)	Side Force (N)	L (N)	D (N)	ϕ (deg)
2	89.23	−0.41	0.00	0.00	−0.52
5	85.60	−2.57	0.00	0.00	−2.91

At both 2 and 5 m/s the roll angle remains below 3° , the body-frame projection of the sidewind onto the longitudinal axis is negligible, and lift and drag forces are effectively zero. The side force scales with sidewind magnitude across the two conditions and the induced roll angle follows accordingly, confirming that the aerodynamic model correctly identifies and resolves lateral flow. Specifically, increasing the sidewind from 2 to 5 m/s increases the side force from -0.41 N to -2.57 N and the roll angle from -0.52° to -2.91° , while the sideslip angle decreases slightly from 89.23° to 85.60° . This small reduction in β reflects the increasing influence of the freestream velocity on the resultant relative airflow, whereas the absence of lift and drag confirms that the aerodynamic loading remains almost entirely confined to the lateral axis, as expected for a pure sidewind.

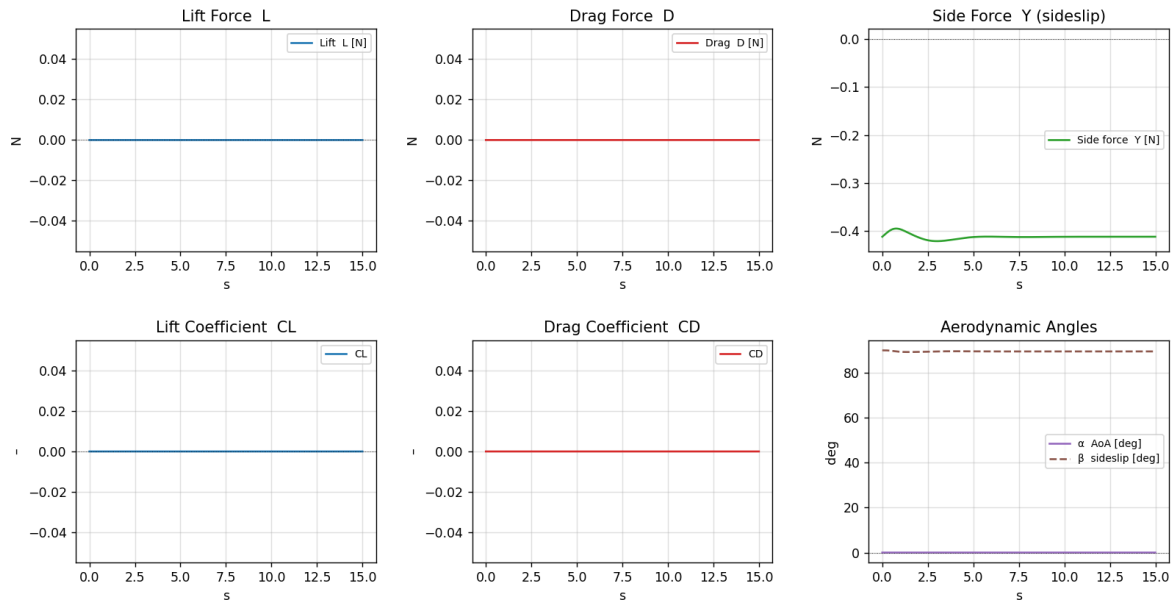


Figure 56: Aerodynamic response at 2 m/s sidewind.

AD-08 Sidewind Force Magnitude Comparison

Having confirmed the correct direction and scaling of the side force in AD-07, this test evaluates its magnitude relative to the drag force produced under equivalent downwash conditions established in AD-05. The side force is governed by the lateral force coefficient $C_{Fy}(\beta)$ at $\beta \approx 90^\circ$, which reflects the aerodynamic resistance of the fuselage and wing to lateral flow. Since the vehicle geometry is not optimised for lateral flow, this coefficient is implemented as 0.2 times the axial drag coefficient $C_D(\alpha)$ evaluated at the same dynamic pressure, and the side force magnitude is therefore expected to be approximately one fifth of the drag force produced under an equivalent downwash speed.

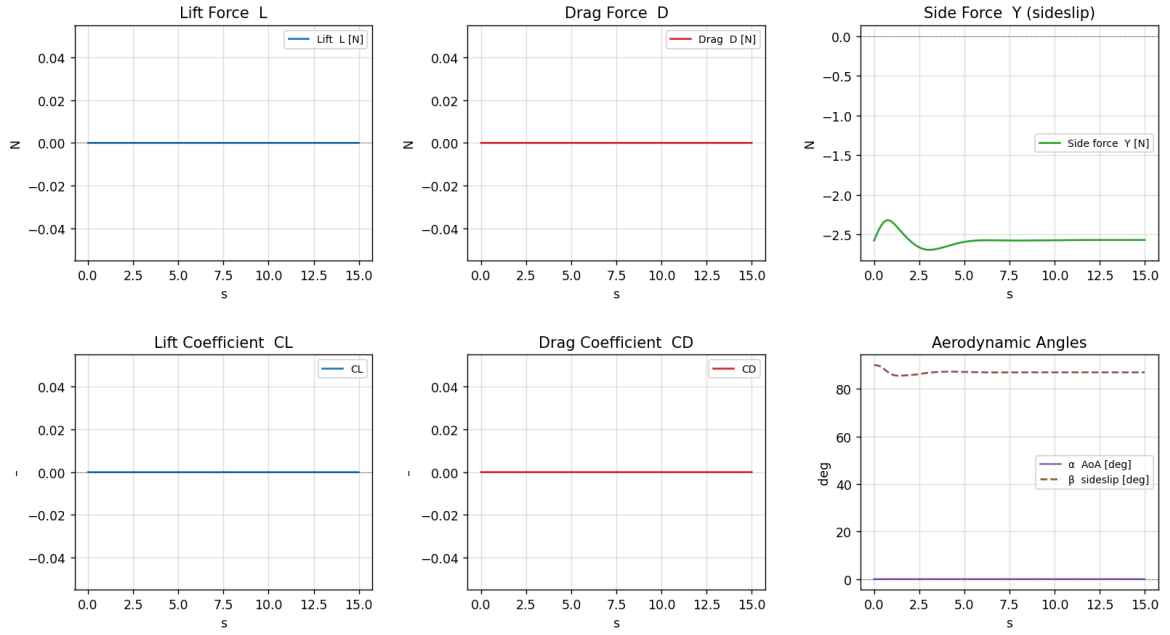


Figure 57: Aerodynamic response at 5 m/s sidewind.

The comparison is performed at a matched airspeed of 5 m/s, where both the downwash drag from AD-05 and the sidewind side force from AD-07 are available at the same dynamic pressure $q = \frac{1}{2}\rho V^2 = \frac{1}{2}(1.225)(25) = 15.31$ Pa. The results are summarised in Table 28.

Table 28: Force magnitude comparison between axial drag under downwash and lateral side force under sidewind at 5 m/s.

Condition	Airspeed (m/s)	Force (N)	Ratio	Expected Ratio
Downwash drag (AD-05)	5	12.83	—	—
Sidewind side force (AD-07)	5	2.57	$2.57/12.83 = 0.20$	0.2

The measured ratio of 0.20 matches the design intent exactly, confirming that the lateral force coefficient is correctly implemented at $0.2 C_D(\alpha)$ and that the side force magnitude scales appropriately with the aerodynamic geometry of the vehicle.

AD-09 Climb Angle of Attack

During a climb manoeuvre, the UAV moves vertically upward into still air, meaning the relative airflow arrives from above along the negative body z -axis. In this condition, the angle of attack is expected to approach -90° , as the relative wind is aligned with the negative body z -axis. Since the motion is purely vertical and no lateral velocity component is present, the sideslip angle should remain near zero and no lift force should be generated, as the flow is approximately perpendicular to the wing surface. A drag force opposing the upward motion is expected, acting in the $-z$ direction and increasing with climb rate, requiring the controller to produce additional collective thrust to overcome both gravity and this aerodynamic drag

load.

The UAV is commanded to perform a climb manoeuvre and the aerodynamic quantities are recorded throughout. The angle of attack confirms -90° for the first 1.5 s of the simulation, coinciding with the active climb phase visible in Figures 59 and 60. No lift force and no side force are produced during this period, consistent with edge-on flow over the wing at $\alpha = -90^\circ$. The drag force acts in the $-z$ direction opposing the climb and peaks at approximately 5 N, corresponding to the peak climb velocity of approximately 3 m/s. This magnitude is physically consistent: at $V = 3$ m/s the dynamic pressure is $q = \frac{1}{2}(1.225)(9) = 5.51$ Pa, and with $C_D(90^\circ) \approx 1.5$ and $S = 0.6$ m² the predicted drag is $qSC_D \approx 4.97$ N, in close agreement with the observed peak. Once the climb velocity reduces as the vehicle approaches the target altitude, the relative airspeed decreases and the drag force returns to zero. These results confirm that the aerodynamic model correctly identifies vertical upward motion and produces the expected force decomposition with no spurious lateral or lift contributions.

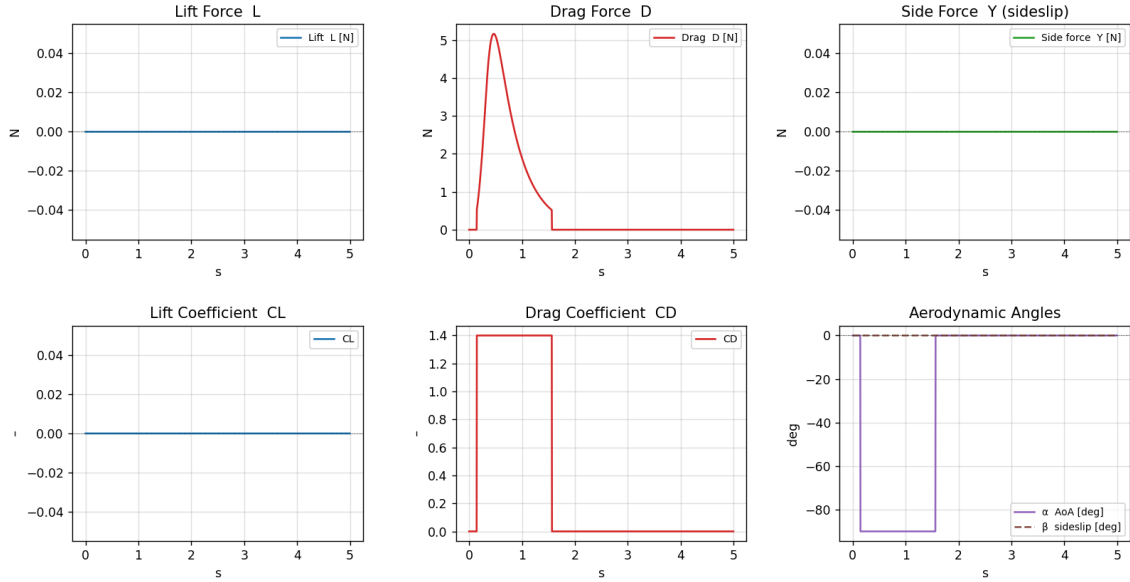


Figure 58: Aerodynamic response during the climb manoeuvre.

AD-10 Descent Angle of Attack

During a descent manoeuvre, the UAV moves vertically downward, meaning the relative airflow arrives from below along the positive body z -axis. In this condition, the angle of attack is expected to approach $+90^\circ$, as the relative wind is aligned with the positive body z -axis. Since the motion is purely vertical and no lateral velocity component is present, the sideslip angle should remain near zero and no lift force should be generated. A drag force opposing the downward motion is expected, acting in the $+z$ direction and increasing with descent rate, partially counteracting gravity and reducing the collective thrust required to control the descent.

The UAV is commanded to perform a descent manoeuvre and the aerodynamic quantities are recorded throughout. The results are the mirror image of those obtained in AD-09: the angle of attack confirms $+90^\circ$ for the active descent phase, no lift force and no side force are produced, and the drag force peaks at approximately 5 N at the maximum descent rate of

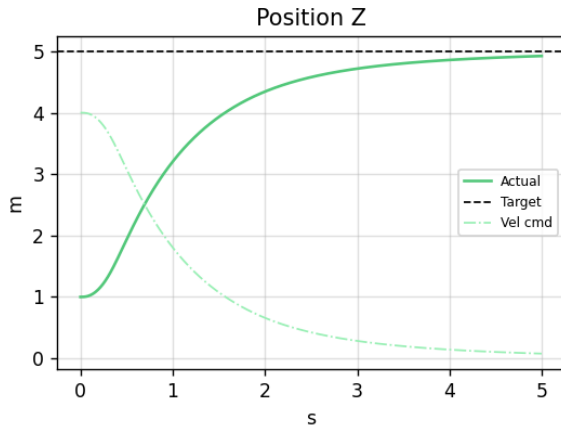


Figure 59: Altitude response during the climb manoeuvre.

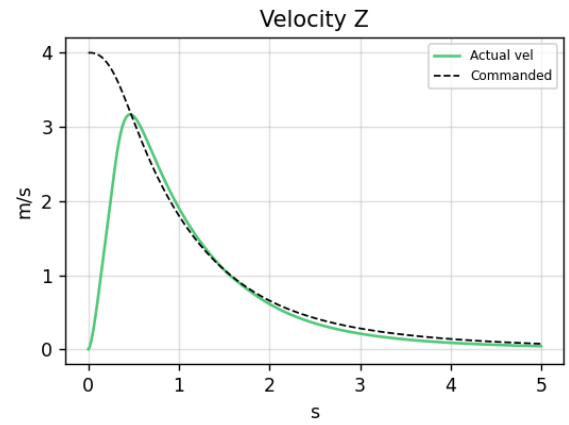


Figure 60: Vertical velocity during the climb manoeuvre.

approximately -3 m/s. The sign of the drag force is opposite to the climb case, acting in the $+z$ direction as expected for a flow arriving from below. The magnitude agrees with the same dynamic pressure argument applied in AD-09, confirming symmetry between the climb and descent aerodynamic responses. Once the descent velocity reduces as the vehicle approaches the target altitude, the relative airspeed decreases and the drag force returns to zero. These results confirm that the aerodynamic model correctly identifies vertical downward motion and produces the expected force decomposition with opposite signs and consistent magnitudes relative to the climb case.

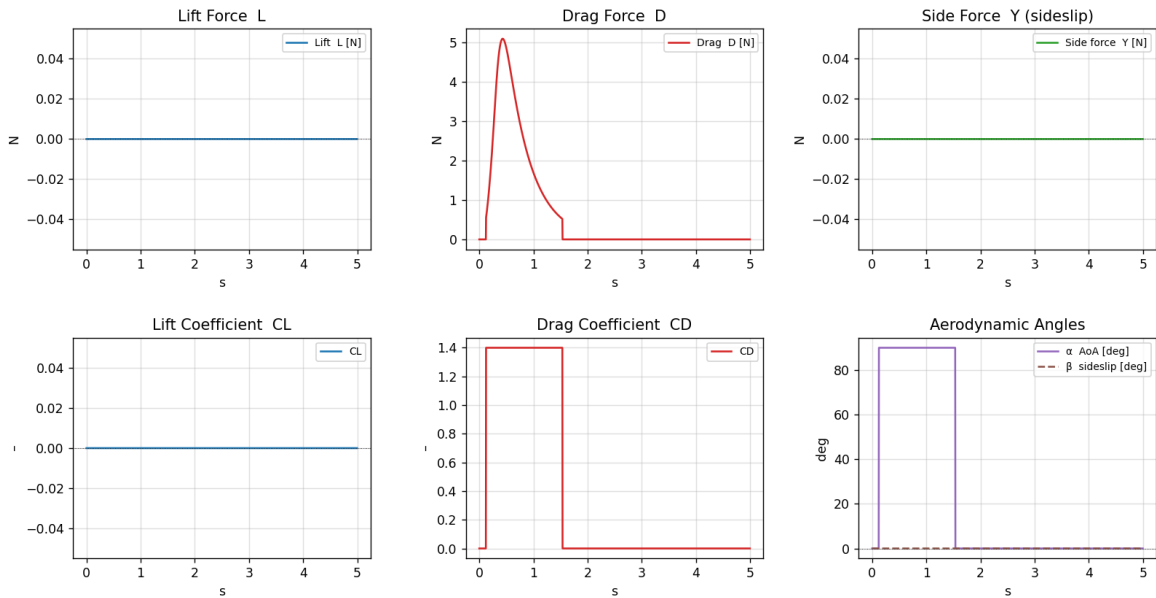


Figure 61: Aerodynamic response during the descent manoeuvre.

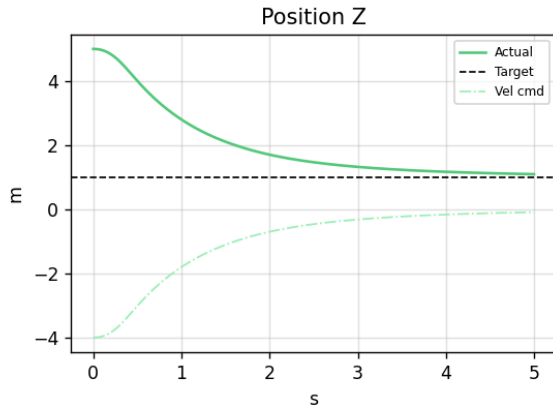


Figure 62: Altitude response during the descent manoeuvre.

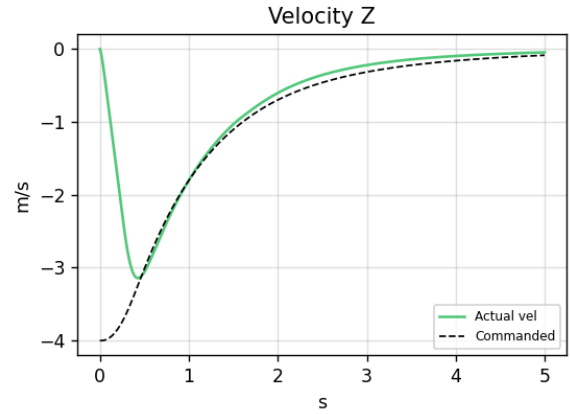


Figure 63: Vertical velocity during the descent manoeuvre.

AD-11 Position Tracking under Headwind

With the aerodynamic force decomposition under headwind already verified in AD-01 through AD-03, this test assesses whether the position controller can reject a sustained headwind disturbance and hold the commanded setpoint. A steady 10 m/s headwind is applied to a UAV hovering at a fixed position, and the resulting position error is recorded throughout the manoeuvre.

The UAV is commanded to hold position while the headwind is applied, and the position response is recorded throughout. Following an initial transient as the controller responds to the onset of the disturbance, the position error converges to a steady-state value of approximately 0.002 m, remaining below the 0.1 m threshold for the remainder of the simulation, as shown in Figure 64. The UAV is displaced considerably along the x -axis due to headwind-induced drag, an offset that the PID controller recovers once the error becomes significant, alongside a minor displacement in z consistent with the positive lift generated in this headwind configuration, as shown in Figure 50. This confirms that the controller successfully compensates for the headwind-induced drag load and maintains station-keeping within the required tolerance.

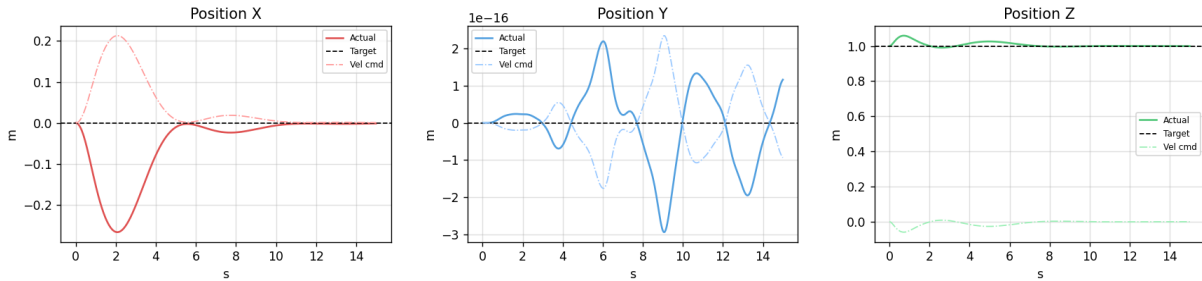


Figure 64: Position tracking response under a 10 m/s headwind disturbance.

AD-12 Position Tracking under Downwash

With the aerodynamic response to vertical downwash already verified in AD-04 and AD-05, this test assesses whether the altitude controller can reject a sustained downwash disturbance

and hold the commanded altitude. A steady 5 m/s downwash is applied while the UAV holds a fixed position, and the position error is recorded throughout the manoeuvre.

The altitude error remains below the 0.1 m threshold throughout, as shown in Figure 65, confirming that the controller successfully compensates for the added drag load and tracks the commanded altitude. The x - and y -axis errors remain at the level of numerical noise, on the order of 10^{-15} m, confirming that the downwash disturbance couples exclusively into the vertical axis with no spurious lateral response.

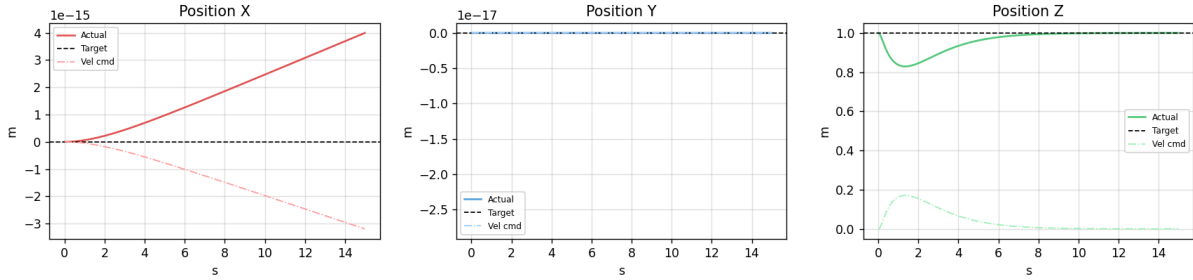


Figure 65: Altitude tracking response under a 5 m/s downwash disturbance.

AD-13 Position Tracking under Combined Headwind and Downwash

With the combined force decomposition under simultaneous headwind and With the combined force decomposition under simultaneous headwind and downwash already verified in AD-06, this test assesses whether the position controller can hold station when both disturbances act at once. A 5 m/s headwind and a 5 m/s downwash are applied simultaneously to a UAV holding a fixed position, and the position error is recorded in all three axes.

The position error in each axis converges to a steady-state value below the 0.1 m threshold, as shown in Figure 66, confirming that the controller correctly rejects the combined disturbance. However, the z -axis response does not follow a linear superposition of the individual headwind and downwash cases. The z displacement observed here is larger than in the pure 5 m/s downwash case (AD-12), despite the headwind alone contributing only a small positive lift in z (AD-11). This is a direct consequence of the angle of attack shifting to approximately -45° under combined loading, as established in AD-06. At this angle, the drag coefficient decreases relative to the pure-headwind case, while the lift component acts increasingly in the same sense as the downwash-induced drag rather than opposing it, effectively reinforcing the vertical loading. The combined disturbance therefore couples into z more strongly than a linear sum of the individual-axis responses would predict, reflecting the underlying nonlinearity of the $C_L(\alpha)$ and $C_D(\alpha)$ polars rather than any deficiency in the position controller.

AD-14 Position Tracking under Sidewind

With the aerodynamic sideslip response under sidewind already verified in AD-07 and AD-08, this test assesses whether the position controller can reject a sustained sidewind disturbance and hold the commanded lateral position. A steady 5 m/s sidewind is applied to a UAV hovering at a fixed position, and the lateral position error is recorded throughout.

The lateral position error converges to a steady-state value of approximately 0.001 m, remaining below the 0.1 m threshold, as shown in Figure 67. This confirms that the controller

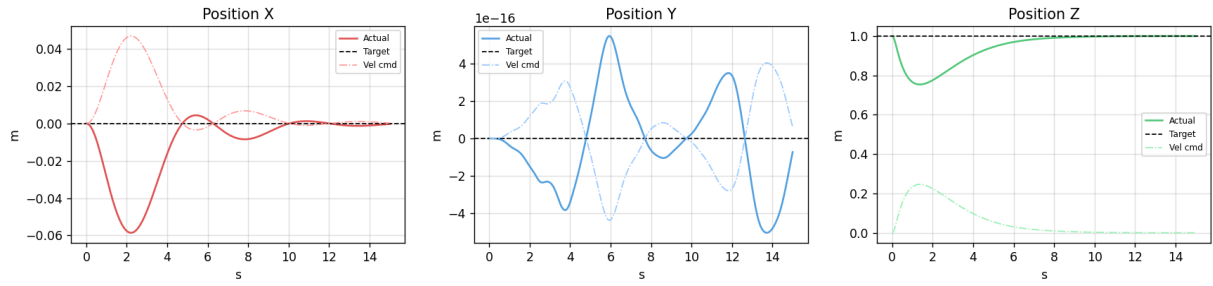


Figure 66: Position tracking response under combined 5 m/s headwind and 5 m/s downwash.

successfully compensates for the lateral force induced by the sidewind and maintains lateral station-keeping within the required tolerance.

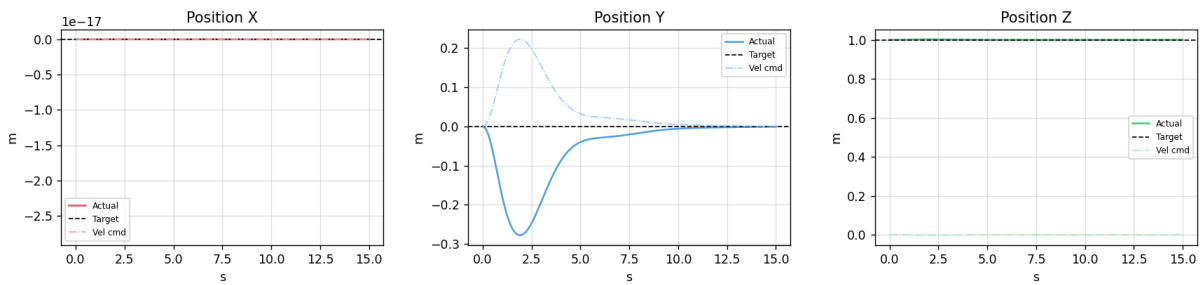


Figure 67: Lateral position tracking response under a 5 m/s sidewind disturbance.

6.3 RL Functionality Verification

The RL functionality verification confirms that the RL framework is correctly integrated with the simulation environment and the PID control architecture. These tests target the correctness of the RL pipeline itself: whether actions are correctly communicated to the controller, whether the agent's interactions with the environment at the intended frequency are suitable for the tracking controller, whether the reward signal reflects the defined objectives, and whether the environment initializes and resets consistently across episodes. The tests are conducted at inference time under controlled conditions.

Table 29 provides a structured overview of all RL functionality verification tests.

Table 29: RL functionality verification summary.

ID	Test	Input / Expected Output	Pass Criterion
RF-01	Action-to-Controller Interface & Sampling Frequency	Step change in commanded velocity, held constant between RL updates → UAV velocity state progresses in the commanded direction following each update	Velocity state moves toward the new command direction within the RL update interval; no divergence or sign mismatch

Continued on next page

Table 29 – continued from previous page

ID	Test	Input / Expected Output	Pass Criterion
RF-02	Reward Signal Verification	Simulated episodes covering landing, termination, truncation, and approach → large positive reward at landing, large negative reward at termination/truncation, increasing reward on approach	Reward behaviour matches the defined reward structure; no NaN or infinite values
RF-03	Episode Reset & Initialisation	Repeated resets under fixed and randomised configurations, varied seeds → correct state initialisation, ship motion and wind field reset, seed reproducibility	Deterministic resets reproducible; randomised resets remain within specified bounds
RF-04	Wind History Buffer Behaviour	Wind velocity buffer printed before each RL update and over the course of a single episode → buffer shifts correctly (oldest entry dropped, newest appended) and resets to zero on episode start	Printed buffer sequence matches expected shift order; buffer correctly zero-initialised on reset
RF-05	Landing Condition Boundary	UAV state driven just inside/outside each individual landing threshold (xy tolerance, vertical speed, attitude) independently	Landing flag is set true only when all thresholds are satisfied simultaneously
RF-06	Crash-on-Pad Cause Attribution	UAV reaches pad z-tolerance without satisfying all landing conditions → failure cause(s) logged	Logged cause(s) (xy/speed/attitude) correctly match the violated threshold(s)
RF-07	Floor & Max-Distance Termination	UAV altitude driven below floor height; UAV displaced beyond maximum allowed distance from target → episode terminates	Termination correctly triggered in both cases and not otherwise
RF-08	Wind Wall Geometry & Activation Probability	Wind wall geometry printed per episode; parameter distribution and active/inactive episode count evaluated over 100 episodes	Sampled geometry parameters remain within configured ranges; wind activation rate matches configured probability

RF-01 Action-to-Controller Interface and Sampling Frequency

This test verifies that velocity commands issued by the RL agent are correctly received and tracked by the underlying PID controller at the defined RL update rate, with each command held constant between updates. Verification is performed by logging data directly from within the RL training framework. A training run is initiated and terminated prematurely, during which the commanded velocity and resulting UAV velocity state are recorded at each RL update over several consecutive steps.

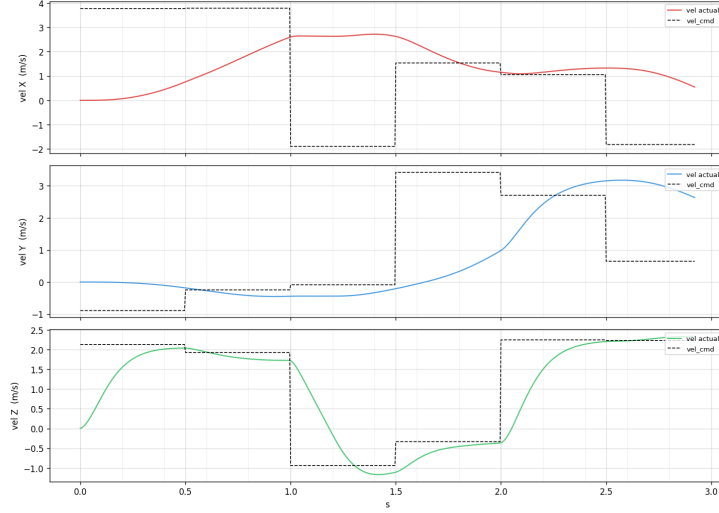


Figure 68: Commanded velocity and resulting UAV velocity state logged directly during a training run, over several consecutive RL updates.

The velocity state is observed to evolve in the commanded direction after each update, with short delays attributable to vehicle inertia and the dynamics of physical acceleration and deceleration. No sign mismatches or divergence are observed, confirming correct interfacing between the RL action output and the PID controller at the defined sampling frequency.

RF-02 Reward Signal Verification

This test verifies that the reward signal behaves as defined across the full range of episode outcomes: successful landing, termination, truncation, and the approach phase preceding these events. Verification is performed by logging the full reward component breakdown from a trained policy during evaluation rollouts, allowing each term’s contribution to be inspected directly at episode outcome.

Table 30: Reward component breakdown across representative episode outcomes, logged from a trained policy.

Component	Crashed	Landed	Truncated
Proximity (r_d)	-17.04	-22.34	-30.20
Vertical speed (r_z)	+0.00	+1.73	+0.00
Smoothness (r_s)	-1.64	-2.94	-3.10
Attitude (r_θ)	+0.00	-0.30	-0.05
Path length (r_p)	-4.51	-6.14	-8.20
Terminal	-100.00	+200.00	-3.85
Total	-123.19	+170.01	-45.40

The crashed episode shows a large negative terminal penalty offsetting a positive approach contribution, consistent with a UAV that was closing in on the pad but failed to satisfy the landing conditions at contact. The landed episode shows all dense shaping terms remaining small relative to the +200 terminal reward, and a nonzero vertical-speed contribution, confirming that r_z activates correctly once the UAV is horizontally aligned with the pad. The

truncated case illustrates the expected structure at maximum episode horizon: no terminal landing or crash bonus, but a distance-proportional penalty reflecting the UAV’s final position relative to the target. Across all inspected episodes, reward values remain finite, with no NaN or divergent terms observed, and the sign and relative magnitude of each component match the intended implementation.

RF-03 Episode Reset and Initialisation

This test verifies correct environment initialization across repeated resets, under both fixed and randomized configurations, and confirms reproducibility under fixed random seeds. Initial UAV and target states, ship motion signals, and wind field configuration were printed and inspected across multiple resets, including seeded and unseeded runs. Deterministic resets were confirmed to be exactly reproducible, and randomized resets were confirmed to remain within their configured bounds.

In practice, exact reproducibility of initial conditions across the RL policy and the PID baseline is enforced not by reseeding the random number generator directly, but through a fixture-based mechanism: each RL evaluation episode records its sampled initial conditions (UAV spawn, ship motion realization, wind configuration) to a fixture file via `record_rl_fixture()`, which the PID baseline subsequently reads and replays episode-for-episode. This guarantees that both approaches are evaluated against an identical set of randomized initial conditions. To additionally verify that the underlying seeding of the randomization itself is correctly deterministic, the sampled UAV position, ship position, and sea state were printed at episodes 5, 10, and 15 across repeated runs initialized with the same seed. The results, summarized in Table 31, confirm that identical values are produced across runs, verifying that the seeding mechanism is deterministic and reproducible.

Table 31: Randomized initial conditions at episodes 5, 10, and 15 across two seeded runs, confirming deterministic reproducibility.

Episode	Run	UAV Position [m]	Ship Position [m]	Sea State
5	Run 1	(0.835, −0.916, 8.933)	(−1.369, −1.037, 6.025)	3
	Run 2	(0.835, −0.916, 8.933)	(−1.369, −1.037, 6.025)	3
10	Run 1	(0.615, 0.842, 8.918)	(0.206, −1.081, 5.172)	6
	Run 2	(0.615, 0.842, 8.918)	(0.206, −1.081, 5.172)	6
15	Run 1	(−1.432, −1.230, 9.445)	(−0.114, −1.016, 5.752)	6
	Run 2	(−1.432, −1.230, 9.445)	(−0.114, −1.016, 5.752)	6

RF-04 Wind History Buffer Behaviour

This test verifies that the rolling wind velocity history buffer is correctly maintained across an episode. The buffer contents were printed before each RL update and inspected over the course of a full episode to confirm the expected shifting behavior (oldest entry dropped, newest entry appended) and correct zero-initialization at the start of each episode. The printed sequence confirmed the buffer updates as expected in both respects.

RF-05 Landing Condition Boundary

This test verifies that the landing condition is triggered only when all individual thresholds – horizontal position tolerance, vertical touchdown speed, and attitude – are simultaneously satisfied. The UAV state was independently overwritten to just inside and just outside each threshold, with the remaining conditions held satisfied, and the resulting landing flag was recorded. The landing flag was set to true only in the nominal case, confirming that all three conditions must be satisfied simultaneously for a successful landing to be registered.

RF-06 Crash-on-Pad Cause Attribution

This test verifies that when the UAV reaches the pad’s vertical tolerance without satisfying the full landing condition, the corresponding failure cause(s) are correctly logged. The UAV was driven to the pad’s z-tolerance under controlled violations of the xy, speed, and attitude conditions, individually and in combination.

Table 32: Crash-on-pad cause attribution test results.

Induced violation	Logged cause(s)
XY tolerance only	xy
Vertical speed only	speed
Attitude only	attitude
XY tolerance + attitude	xy + attitude

In each case, the logged cause(s) correctly match the induced violation(s), confirming correct attribution logic in `_compute_terminated`.

RF-07 Floor and Max-Distance Termination

This test verifies that episode termination is correctly triggered when the UAV altitude drops below the defined floor height, or when the UAV-target distance exceeds the maximum allowed distance. Both conditions were induced independently by direct state manipulation, and correct triggering (and non-triggering under nominal conditions) was confirmed by inspection of the termination flag and printed diagnostic output.

RF-08 Wind Wall Geometry and Activation Probability

This test verifies that the wind wall geometry is correctly sampled within its configured parameter ranges, and that the wind hazard activates at the configured probability. Wind wall geometry (offset, half-width, height) is logged at each reset and its distribution evaluated over 100 episodes; the fraction of episodes with wind active was separately counted over the same 100 episodes. The sampled geometry statistics are summarized in Table 33, and the observed activation rate is reported in Table 34.

The sampled geometry parameters remain within their configured ranges across all 100 episodes, and the observed activation rate is consistent with the configured probability.

Table 33: Sampled wind wall geometry over 100 episodes: configured range vs. observed range.

Parameter	Configured range	Observed range	Within range?
Offset R [m]	[3.5, 4.5]	[3.51, 4.48]	✓
Half-width h_w [m]	[1.0, 4.0]	[1.06, 3.98]	✓
Height H [m]	[3.0, 6.0]	[3.04, 5.94]	✓
Wind speed v_{wind} [m/s]	[−25.0, −15.0]	[−24.88, −15.13]	✓
Y shift [m]	[−3.0, 3.0]	[−2.97, 2.99]	✓
X shift [m]	[−3.0, −1.0]	[−2.99, −1.01]	✓

Table 34: Wind activation rate over 100 episodes. With $p_{\text{wind}} = 1.0$, the wind hazard is active in every episode.

Configured p_{wind}	Active episodes	Observed rate
0.8	82 / 100	0.82

7 ADDITIONAL RESULTS

This section presents the full set of results obtained in this thesis, complementing the paper provided in Part 2. The results are organized around the two central problems addressed in this study, moving-platform landing and localized wind-disturbance avoidance, and include the key performance tests together with the sensitivity analyses.

7.1 Moving Maritime Target Landing without Localized Wind Disturbances

In this set of experiments, the guidance policy is trained to perform autonomous landings on a moving maritime target in the absence of localized wind disturbances. The target motion is generated using the JONSWAP-based sea-state model described previously, with the sea state randomly sampled for each episode from a range of 1 to 5, exposing the policy to a broad spectrum of platform motion intensities, from mild to severe deck excursions. In addition, the initial positions of both the UAV and the target are randomized within their respective spawn regions at the start of every episode, subject to a maximum initial separation constraint.

For this set of experiments, the observation space is reduced to a simplified, wind-unaware formulation:

$$\mathbf{o}_t = [\mathbf{p}_{\text{rel}}, \mathbf{v}_{\text{rel}}, \boldsymbol{\eta}_v, \boldsymbol{\omega}_v, \mathbf{a}_{t-1}]^\top \in \mathbb{R}^{15}.$$

Two training configurations are compared: training with and without guided exploration via initial-state randomization. In the guided-exploration configuration, for the first 1000 training episodes, the UAV is periodically (every tenth episode) initialized in close proximity to the landing target, rather than at a randomized stand-off position. This near-target initialization allows the agent to repeatedly encounter and discover the sparse terminal landing reward early in training.

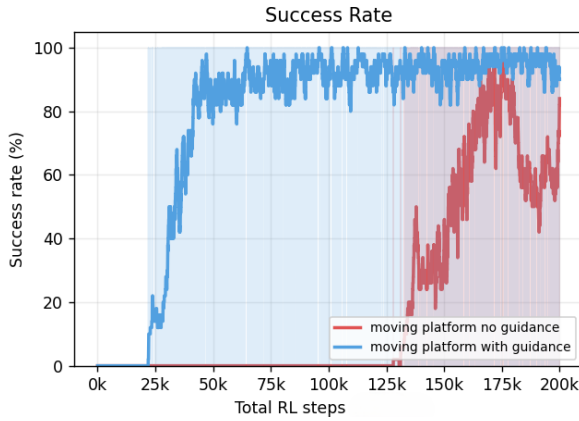
Training Performance Evaluation

Training performance is evaluated using four metrics tracked over the course of environment interaction (RL steps): landing success rate, cumulative episode reward, final distance to the landing platform at episode termination, and episode duration in RL steps. A success is recorded whenever the UAV satisfies the full landing condition (horizontal position, vertical velocity, touchdown-speed, and attitude criteria) within an episode.

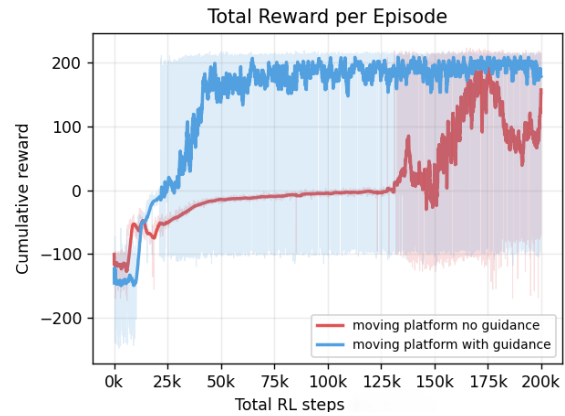
The two training configurations, with and without guided exploration via initial-state randomization, exhibit markedly different learning dynamics. The guided-exploration configuration achieves a success rate exceeding 80% substantially earlier in training, at approximately 50k RL steps, whereas the configuration without guided initialization reaches this threshold only at approximately 160k RL steps (Figure 69a). This gap is consistent with the expected role of guided initialisation in exposing the agent to the sparse landing reward early in training, thereby accelerating convergence towards a successful landing policy.

Beyond the speed of initial success-rate growth, the two configurations differ in their long-term convergence behavior. The guided-exploration policy converges to a stable solution, evidenced by a sustained plateau in both success rate (Figure 69a) and episode reward (Figure 69b), with reward consistently remaining near the sparse landing bonus of 200.

This indicates that the policy has reliably learned to commit to and execute successful landings. In contrast, the policy trained without guided initialization, despite eventually exceeding an 80% success rate, does not converge to a stable solution. Both success rate and accumulated reward remain oscillatory throughout the remainder of training, even after extended interaction with the environment. It is believed that, with additional training, the unguided configuration would eventually stabilize as well; nonetheless, these results clearly demonstrate the accelerating and stabilizing effect of guided exploration on policy convergence.



(a) Landing success rate over RL training steps.

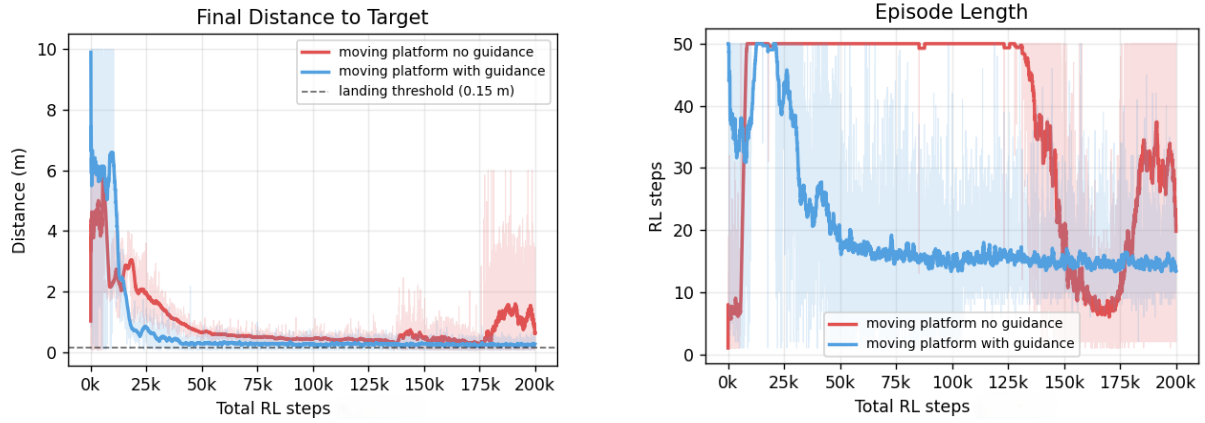


(b) Cumulative episode reward over RL training steps.

Figure 69: Training performance of the RL guidance policy for the moving-platform landing task. (a) Landing success rate over training steps. (b) Cumulative episode reward over training steps.

Notably, both configurations achieve similarly low final distances to the landing platform (Figure 70a), with convergence occurring early in training: approximately 25k steps for the guided configuration and 50k steps for the unguided configuration. This indicates that the underlying approach and tracking behavior are learned relatively early in both settings, and that the divergence in success-rate convergence is not attributable to deficiencies in target tracking. The key distinguishing factor is episode duration. The unguided configuration exhibits a substantially higher rate of truncated episodes, as shown in Figure 70b, indicating that the agent frequently fails to commit to landing despite being in close proximity to the target, as illustrated in Figure 70a.

This behavior is attributed to the inherent difficulty of discovering the sparse landing reward through unguided exploration alone. As terminal (crash) conditions are concentrated within the vertical envelope surrounding the landing platform, an agent lacking guided initialization tends to associate close-range proximity with elevated termination risk rather than with the opportunity to land successfully. In the absence of early successful landing experiences to counteract this association, the policy converges towards a local optimum in which hovering or loitering near the target is favored over execution of the terminal landing maneuver, the latter having become associated, through experience, with the crash penalty. This local optimum accounts for both the delayed emergence of landing successes and the subsequent oscillatory, non-convergent training behavior observed in the unguided configuration.



(a) Final distance to the landing platform at episode termination over RL training steps.

(b) Episode length (RL steps) over RL training steps.

Figure 70: Additional training metrics for the RL guidance policy in the moving-platform landing task. (a) Final distance to the landing platform at episode termination. (b) Episode length as a function of training steps.

Test Results

The trained guidance policy (with guided exploration via initial-state randomization) is evaluated under a fixed sea state 6 test scenario, with JONSWAP wave realizations sampled independently for each episode. Because these realizations are generated stochastically, the specific wave patterns encountered during testing are guaranteed not to have appeared during training. Furthermore, the underlying sea-state level itself lies outside the range used during training, so this evaluation additionally tests the policy’s ability to generalize beyond the sea states it was trained on.

Across the 100 test episodes, the RL policy achieves a landing success rate of 100%, compared to 92% for the baseline, confirming that the policy generalizes reliably to previously unseen wave realizations and initial conditions within the tested sea state. Landing quality is further assessed using horizontal (XY) landing error, time to land, roll and pitch attitude at touchdown, and relative vertical velocity at touchdown, summarized in Part II Table 7. The RL policy achieves horizontal landing accuracy comparable to the baseline, reduces mean time to land by 67.9%, and reduces relative vertical speed at touchdown by 41.8%, without a substantial difference in touchdown roll or pitch attitude. The mean differences between the RL and baseline roll and pitch attitudes are small relative to their standard deviations, suggesting the RL policy does not introduce a systematic change in touchdown attitude. A detailed discussion of these results is provided in the accompanying scientific paper.

When comparing the three-dimensional trajectories of a representative test episode, the RL policy follows a visibly more direct path to the landing platform, whereas the baseline exhibits a longer, less direct approach. This behavior is further examined in Figure 71, which compares the individual x , y , and z position components of the UAV over time for both solutions, alongside the corresponding landing platform position. The x - y tracking behavior is qualitatively similar between the two solutions, with both converging promptly to the platform’s horizontal position. The z -axis descent rate is comparable in slope between the two solutions; the key difference lies instead in timing and final approach behavior. In the

seconds preceding touchdown, the RL policy exhibits a brief flare-like maneuver, momentarily arresting its descent before completing a more gradual, extended final approach to the platform, whereas the baseline solution proceeds through this phase without a comparable pause.

To further characterize this descent logic, Figure 72 examines the RL agent’s vertical approach across three additional representative episodes, spanning distinct combinations of initial altitude offset and platform heave phase. In all three episodes, the UAV’s vertical trajectory converges monotonically towards the platform’s heave motion, with no evidence of anticipatory phase matching: rather than timing its descent to coincide with a trough or peak in the platform’s motion, the agent tracks the instantaneous altitude gap directly, smoothly reducing the relative vertical velocity as this gap narrows, regardless of whether the platform is ascending or descending. This is reflected in the relative V_z color gradient, which decreases from approximately 2–3 m/s to near zero only in close vertical proximity to the platform, irrespective of heave direction, suggesting that the terminal touchdown decision is resolved reactively via the instantaneous relative velocity rather than through early anticipation of the platform’s motion phase. While this strategy consistently yields low touchdown velocities in simulation, its reliance on accurate relative-velocity estimation at short range may present a limitation in real-world deployment, where sensor noise and estimation latency are typically more pronounced at close proximity.

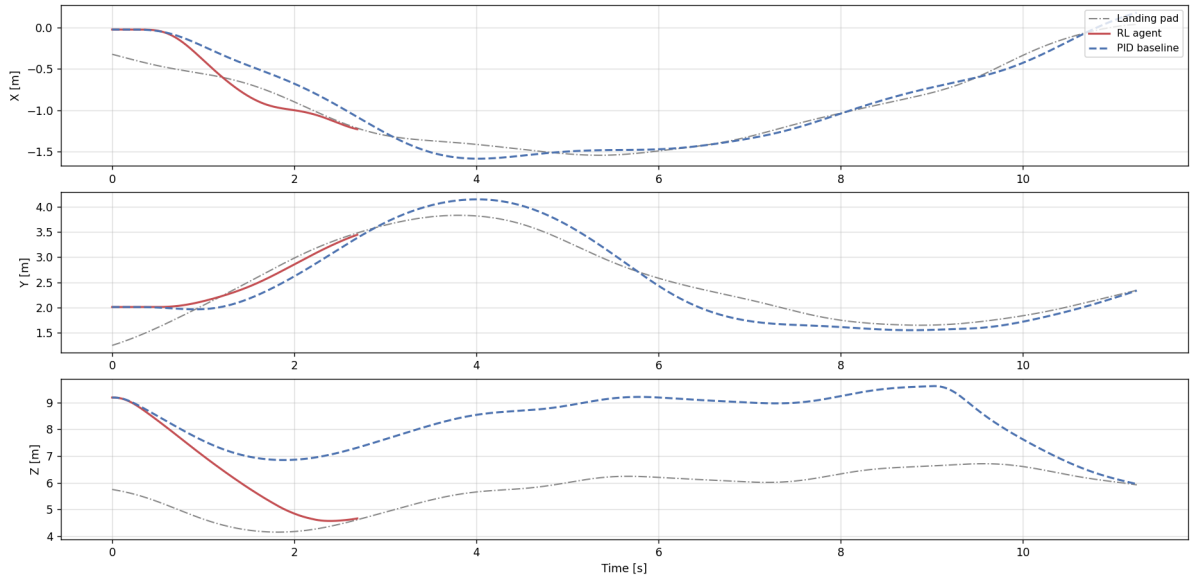


Figure 71: Comparison of UAV position components (x, y, z) over time for the RL and baseline guidance solutions on the same test episode 100, alongside the corresponding landing platform position.

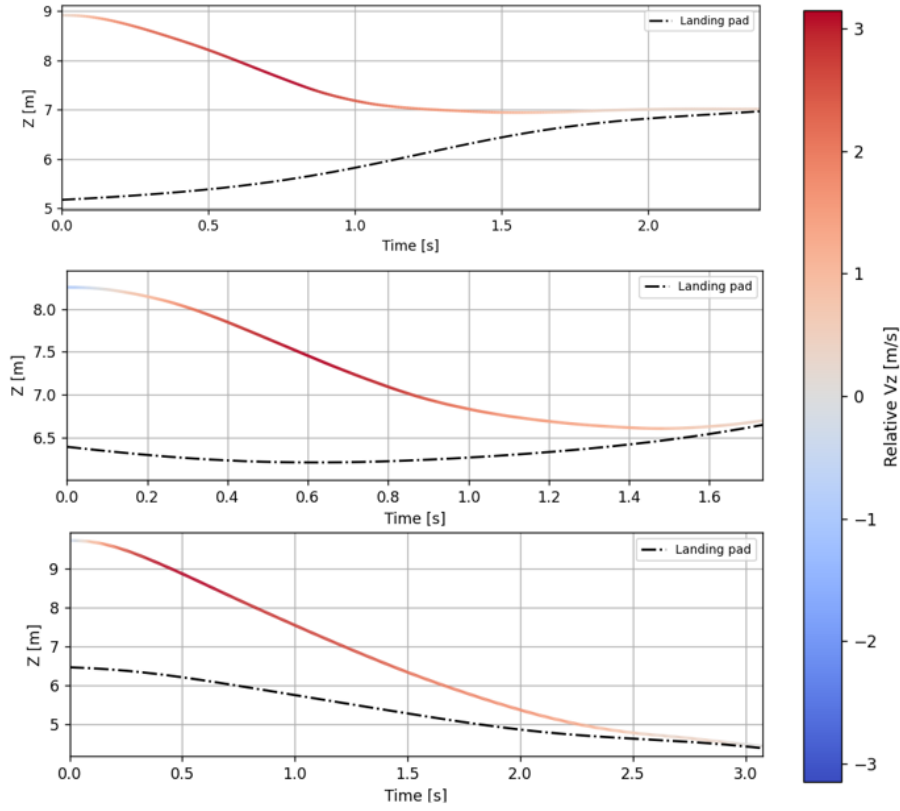


Figure 72: Vertical descent strategy of the RL guidance policy across three representative episodes, coloured by relative vertical velocity V_z .

Sensitivity Analysis

To assess the robustness of the trained policy beyond the nominal evaluation conditions, its performance is examined under increasing sea-state severity, which affects the amplitude and frequency content of the platform motion the policy must track.

Sensitivity to Sea State

To assess the robustness of the trained guidance policy beyond the sea state level used in primary evaluation, both the RL policy and baseline are tested across sea states 4, 5, 6, and 7. This range goes beyond the full training distribution (sea states 1–5) and extends into out-of-distribution conditions (sea states 6–7), allowing evaluation of generalization under increasingly severe wave disturbances. For each sea state, 100 episodes are evaluated with independently sampled JONSWAP wave realizations.

A full breakdown of per-metric performance across all four sea states is provided in Table 35. These results show that the RL policy maintains consistently high landing success rates across all tested sea states, with only a marginal decline to 97% at sea state 7, which lies outside the training distribution. This robustness is attributed to the domain randomization employed during training, whereby exposure to a range of sea states and stochastic wave realizations discourages overfitting to a specific disturbance regime. The baseline, by contrast, exhibits a monotonic decline in success rate with increasing sea state severity, from 100% at sea state 4

to 88% at sea state 7. This degradation reflects the absence of a relative-velocity scheduling mechanism in the baseline’s descent logic, leading to higher vertical touchdown velocities, especially in high sea states. The RL policy, having learned such scheduling through exposure to varying sea states during training, is not subject to this limitation.

Table 35: Sensitivity to sea state: comparison of baseline and RL guidance policy performance over 100 test episodes per sea state (mean $\pm$ std).

Metric	Policy	Sea State 4	Sea State 5	Sea State 6	Sea State 7
Success Rate [%]	Baseline	100	97	92	88
	RL	100	100	100	97
XY Landing Error [m]	Baseline	0.215 ± 0.110	0.238 ± 0.128	0.259 ± 0.124	0.280 ± 0.131
	RL	0.200 ± 0.095	0.212 ± 0.103	0.277 ± 0.128	0.271 ± 0.116
Time to Land [s]	Baseline	4.893 ± 1.436	6.775 ± 2.141	11.326 ± 3.374	13.337 ± 3.803
	RL	2.640 ± 0.400	2.748 ± 0.497	3.640 ± 0.843	3.921 ± 1.188
Roll at Landing [deg]	Baseline	-0.053 ± 2.774	-0.087 ± 2.258	0.048 ± 2.532	0.062 ± 2.876
	RL	0.641 ± 2.501	0.632 ± 2.381	0.092 ± 3.119	0.501 ± 2.900
Pitch at Landing [deg]	Baseline	-0.095 ± 2.153	-0.154 ± 2.341	0.433 ± 2.791	-0.197 ± 2.904
	RL	3.086 ± 2.493	2.907 ± 2.459	0.010 ± 2.839	2.548 ± 2.536
Rel. V_z at Touchdown [m/s]	Baseline	0.369 ± 0.109	0.384 ± 0.114	0.380 ± 0.113	0.452 ± 0.321
	RL	0.221 ± 0.099	0.235 ± 0.106	0.221 ± 0.101	0.266 ± 0.203

Among the landing quality metrics, time to land shows the most pronounced sensitivity to sea state for the baseline, increasing monotonically from 4.9 ± 1.4 s at sea state 4 to 13.3 ± 3.8 s at sea state 7. This is because the guidance system waits longer for a suitable landing window as platform motion becomes more energetic. Horizontal XY landing error also increases monotonically with sea state for the baseline, though the magnitude of degradation is modest.

Relative vertical speed at touchdown remains largely stable for the baseline across sea states 4 through 6, but exhibits a notable increase in both mean and variance at sea state 7, suggesting that the most severe wave conditions occasionally produce unfavorable deck velocities that cannot be avoided by the window-waiting strategy alone. The RL policy consistently achieves lower mean touchdown speeds across all sea states, and this advantage is reflected in the variance as well: even at sea state 7, where both policies show increased spread, the RL policy’s standard deviation (± 0.203 m/s) remains substantially below that of the baseline (± 0.321 m/s). This indicates that the RL policy not only lands more softly on average, but also avoids the occasional high-speed touchdown events that appear in the baseline distribution under the most severe wave conditions.

A comparison of key performance metrics across all four sea states is shown in Figure 73.

7.2 Stationary Target Landing with Localized Wind Disturbances

In this set of experiments, the guidance policy is trained to perform autonomous landings on a stationary target in the presence of a localized wind disturbance encountered during the approach phase. Unlike the previous configuration, platform motion is removed entirely; the primary challenge here is instead the detection and mitigation of a spatially confined aerodynamic disturbance that the UAV may traverse before reaching the landing pad. The wind zone is modeled as a localized rectangular region positioned between the UAV spawn

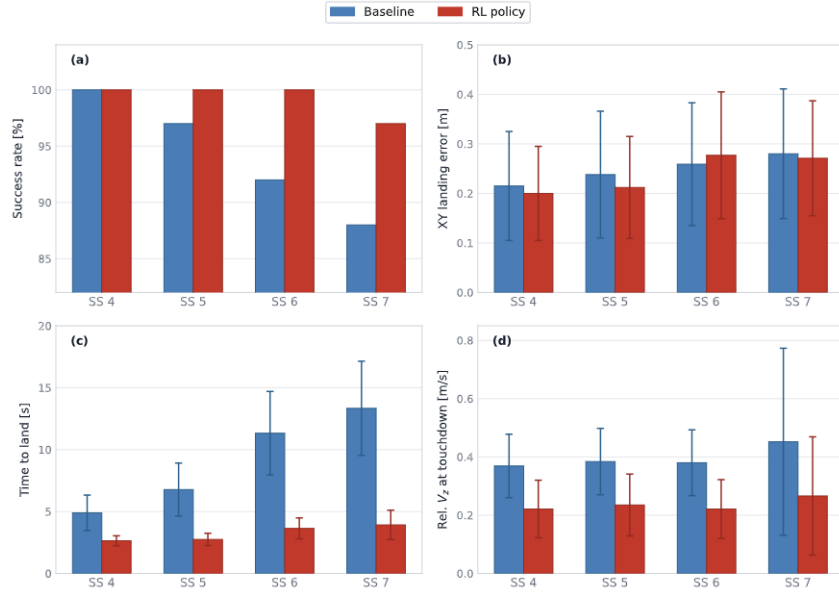


Figure 73: Sea state sensitivity analysis: comparison of baseline and RL guidance policy across sea states 4–7, evaluated over 100 episodes per condition under independently sampled JONSWAP wave realizations. Panels show (a) landing success rate, (b) horizontal XY landing error, (c) time to land, and (d) relative vertical speed at touchdown. Error bars denote ± 1 standard deviation.

location and the landing platform along the nominal approach axis, representing a localized wind disturbance; outside this region, the wind field smoothly blends to zero ambient wind, such that the UAV experiences the disturbance only within the confined zone rather than throughout the approach. To prevent the policy from overfitting to a fixed disturbance configuration, both the geometric dimensions and aerodynamic properties of the wind zone are independently randomized at the start of each episode, with the parameter ranges summarized in the accompanying scientific paper. The UAV and landing platform positions are likewise randomized within their respective spawn regions at the start of each episode, subject to the constraint that the UAV is always initialized upstream of the wind zone, preserving the spatial ordering of UAV $\rightarrow$ wind zone $\rightarrow$ landing pad throughout training.

For this configuration, three observation formulations are compared to evaluate the benefit of explicit and implicit wind state information. The first formulation, here onward v1, augments the baseline wind-unaware observation $\mathbf{o}_t \in \mathbb{R}^{15}$ with onboard accelerometer measurements $\tilde{\mathbf{a}}_{\text{body}} \in \mathbb{R}^3$ and normalized rotor speeds $\tilde{\Omega} \in \mathbb{R}^4$, providing indirect wind-sensing capability through the aerodynamic response of the vehicle:

$$\mathbf{o}_t^{\text{wind}} = [\mathbf{p}_{\text{rel}}, \mathbf{v}_{\text{rel}}, \boldsymbol{\eta}_v, \boldsymbol{\omega}_v, \mathbf{a}_{t-1}, \tilde{\mathbf{a}}_{\text{body}}, \tilde{\Omega}]^T \in \mathbb{R}^{22},$$

To establish an upper bound on wind-awareness and quantify the benefit of explicit wind state information, a second formulation, here onward v2, further augments $\mathbf{o}_t^{\text{wind}}$ with a history of the directly measured wind velocity vector. At each timestep, the body-frame wind velocity $\mathbf{v}_{w,t} \in \mathbb{R}^3$ is appended to a rolling buffer of the $H = 10$ most recent measurements, flattened

into a single vector:

$$\mathbf{V}_{\text{wind}}^{t-H+1:t} = [\mathbf{v}_{w,t-H+1}, \dots, \mathbf{v}_{w,t-1}, \mathbf{v}_{w,t}]^\top \in \mathbb{R}^{3H} = \mathbb{R}^{30}.$$

This history vector is concatenated with $\mathbf{o}_t^{\text{wind}}$ to form the full observation:

$$\mathbf{o}_t^{\text{full}} = [\mathbf{p}_{\text{rel}}, \mathbf{v}_{\text{rel}}, \boldsymbol{\eta}_v, \boldsymbol{\omega}_v, \mathbf{a}_{t-1}, \tilde{\mathbf{a}}_{\text{body}}, \tilde{\boldsymbol{\Omega}}, \mathbf{V}_{\text{wind}}^{t-H+1:t}]^\top \in \mathbb{R}^{52},$$

where the wind velocity history is measured directly onboard and expressed in the body frame.

Finally, a third observation vector is investigated, here onward v3, which aims to reduce the observation space by providing only the instantaneous gradient of the wind velocity, rather than the full history retained in $\mathbf{o}_t^{\text{wind}}$. At each timestep, the body-frame wind velocity gradient is estimated via finite differencing of consecutive measurements:

$$\dot{\mathbf{v}}_{w,t} = \frac{\mathbf{v}_{w,t} - \mathbf{v}_{w,t-1}}{\Delta t} \in \mathbb{R}^3,$$

and appended to the base observation in place of the wind history vector:

$$\mathbf{o}_t^{\text{grad}} = [\mathbf{p}_{\text{rel}}, \mathbf{v}_{\text{rel}}, \boldsymbol{\eta}_v, \boldsymbol{\omega}_v, \mathbf{a}_{t-1}, \tilde{\mathbf{a}}_{\text{body}}, \tilde{\boldsymbol{\Omega}}, \dot{\mathbf{v}}_{w,t}]^\top \in \mathbb{R}^{25}.$$

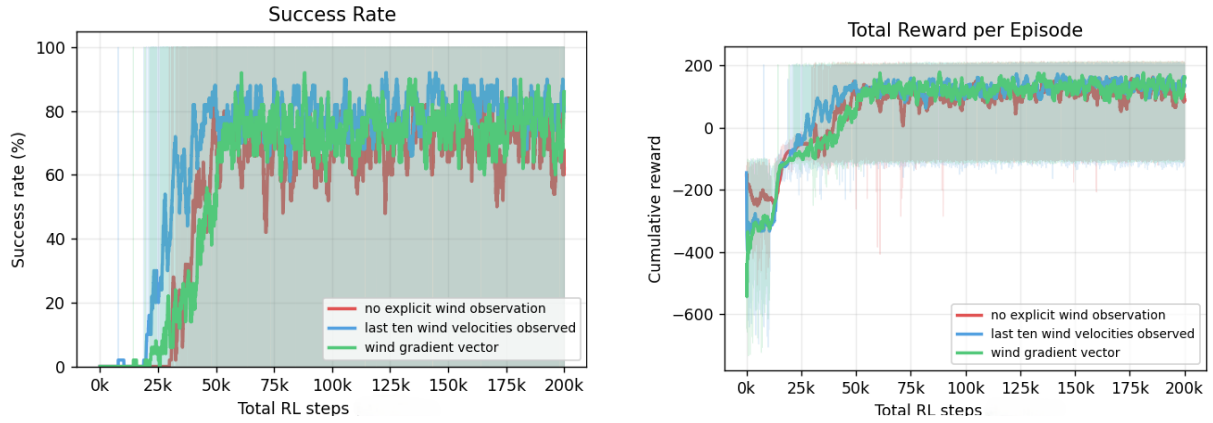
This formulation was evaluated against $\mathbf{o}_t^{\text{wind}}$ to assess whether providing the agent with comparable information about the rate of change of the wind disturbance, while reducing the observation dimensionality from 52 to 25, offered any benefit to training convergence.

As the benefit of guided exploration via initial-state randomization has already been established in the previous tests, all training runs in this set of experiments incorporate guided exploration throughout. For the first $N_{\text{demo}} = 1000$ training episodes, the UAV is periodically initialized in close proximity to the landing target every tenth episode.

Training Performance Evaluation

All three observation configurations converge to stable policies within a similar training budget, reaching success rates of 70–90% at approximately 50k RL steps (Figure 74a). Contrary to the expectation that a larger observation space would hinder convergence, v2 converges earliest and reaches the highest mean success rate, followed by v3, with v1 performing worst. Training curves remain oscillatory rather than fully stabilizing, a behavior attributed to the stochastic wind-zone randomization rather than learning instability, where the wide variation in disturbance geometry and intensity across episodes introduces sufficient episode-to-episode variance to produce noisy success-rate traces even after convergence. No systematic divergence is observed in any configuration, and the achieved success rates are considered sufficient to proceed to policy evaluation, since no further improvement is observed over an extended training horizon.

The cumulative episode reward traces agree with the success rate curves. Although the sparse landing reward of 200 is not consistently achieved, the reward settles at approximately 140–150, as shown in Figure 74b. This reflects the negative terminal penalties incurred by the residual fraction of unsuccessful episodes.



(a) Episode success rate during RL training under localized wind disturbances for all three observation configurations.

(b) Cumulative episode reward during RL training under localized wind disturbances for all three observation configurations.

Figure 74: Training performance under localized wind disturbances for the three observation configurations: (a) episode success rate and (b) cumulative episode reward.

Test Results

All trained policies are evaluated during inference under randomized wind-wall configurations. The wind-wall geometry is independently sampled for each episode, with the half-width drawn from $[1, 4]$ m, height from $[3, 6]$ m, lateral center offset $\Delta y \in [-3, 3]$ m, and longitudinal offset $\Delta x \in [-3, -1]$ m. These variations ensure that the disturbance region is not fixed relative to the nominal approach corridor, resulting in episodes with partial or complete wind-field interception and exposing the policy to off-axis disturbance encounters. A comparable form of geometric randomization, using different parameter ranges, was also applied during training to prevent overfitting to a specific wind-wall location and to encourage the policy to learn disturbance-response behavior rather than memorizing a fixed environmental configuration.

Under randomized wind-wall geometry, *v1* achieves a success rate of 31%. The policy is observed to converge on a lateral avoidance maneuver, banking left or right early in the episode, prior to any measurable signal in the wind-related observation channels. This timing indicates that the maneuver is not driven by real-time wind sensing, but instead reflects a fixed behavior learned from the average wind-field position encountered during training, which is executed independently of the actual wall placement in a given episode. As the wind-wall position is randomized during testing, this learned maneuver becomes decoupled from the true disturbance location, causing the vehicle to frequently steer directly into the wind field despite exhibiting an avoidance-oriented trajectory. This behavior is illustrated in Figure 75a, with corresponding side and top views provided in Figures 75b and 75c. The non-zero success rate is therefore attributed to incidental alignment between the fixed avoidance maneuver and the randomized wall position, rather than to a genuine reactive wind-avoidance capability. Accordingly, performance in this scenario is governed by the interaction between a training-time geometric prior and the random placement of the wind field, rather than by real-time wind observation.

For *v2*, this test yields a success rate of 66%. Inspection of individual trajectories reveals a

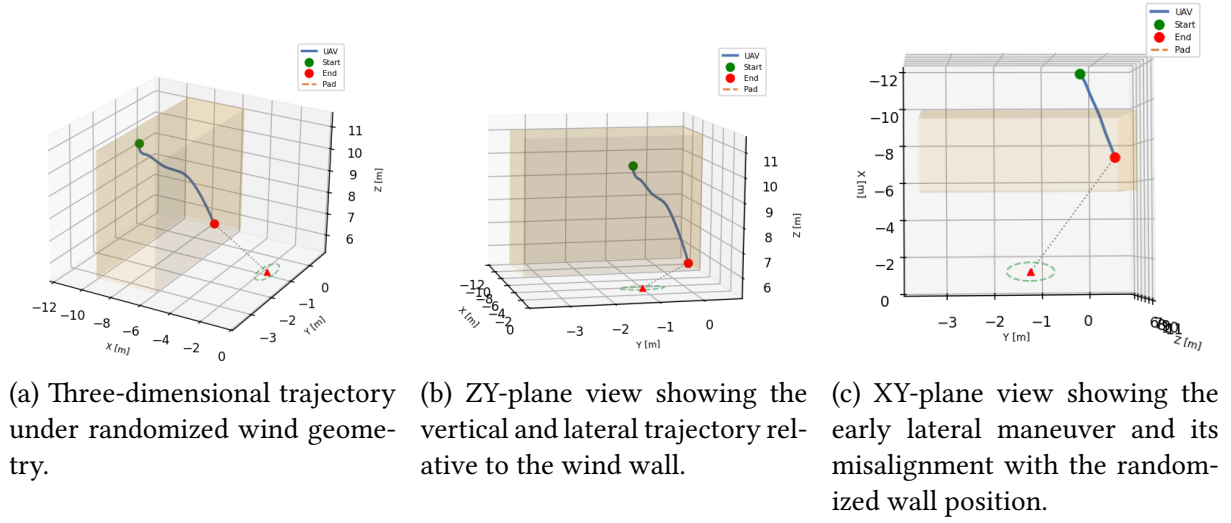
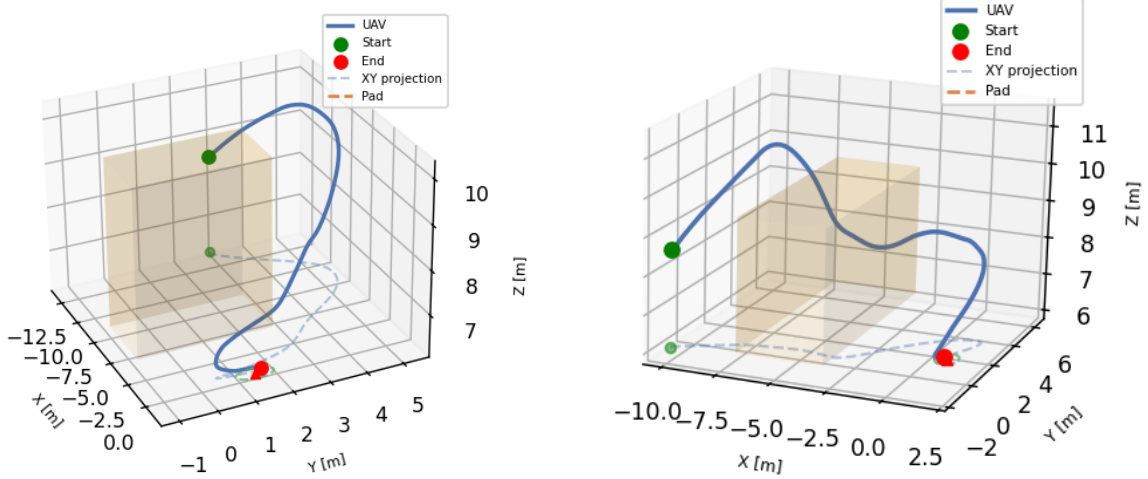


Figure 75: Episode 98 trajectory analysis for the $v1$ policy under randomized wind geometry.

recurring pattern consistent with the previously discussed anticipatory avoidance behavior, in which the vehicle's maneuver continues to reflect a geometric prior tied to the training-time wind distribution. This prior occasionally works against the vehicle: in a subset of episodes, the policy maneuvers towards the location where the wind field is implicitly expected based on training statistics, only to fly directly into the actual disturbance zone, whose randomized position was not anticipated. Compared to $v1$, the lateral offset from a direct approach path is noticeably larger here, indicating that the predictive avoidance maneuver is more aggressive: the policy commits more strongly to its learned geometric prior, producing a wider deviation from the direct start-to-end trajectory before any real-time correction based on sensed wind can take effect.

Unlike $v1$, however, the majority of $v2$ trajectories exhibit clear departures from a purely predictive approach, with lateral and vertical corrections that activate once the vehicle enters the blending zone or the wind field itself, tracking the actual, randomized position of the disturbance rather than its expected location alone. This indicates that the anticipatory component is complemented by a reactive one, in which real-time wind velocity observations are used to correct the approach as the disturbance is encountered, allowing the vehicle to route around or above the active wind zone and achieve a successful landing in most episodes. Figure 76a shows an episode in which the anticipatory component correctly predicts the most likely wind interaction zone, resulting in appropriate predictive avoidance. Figure 76b, by contrast, show clear signatures of reactive wind sensing and avoidance, evident in deviations from a smooth trajectory as the UAV intercepts the wind field. This combination of anticipatory and reactive behavior is the primary driver behind the near-doubling of the success rate relative to $v1$.

To further isolate and confirm the reactive component of this avoidance behavior, an additional controlled test was performed in which the same trained policy is evaluated twice under identical initial conditions (same UAV spawn position, same pad position, same random seed), once with the wind disturbance active and once with it deactivated. Since the initial geometric configuration is held fixed across both runs, any difference in the resulting trajectories can be attributed exclusively to the presence or absence of the wind field, rather than to variation in



(a) Episode 97 under randomized wind geometry (v2). (b) Episode 100 under randomized wind geometry (v2).

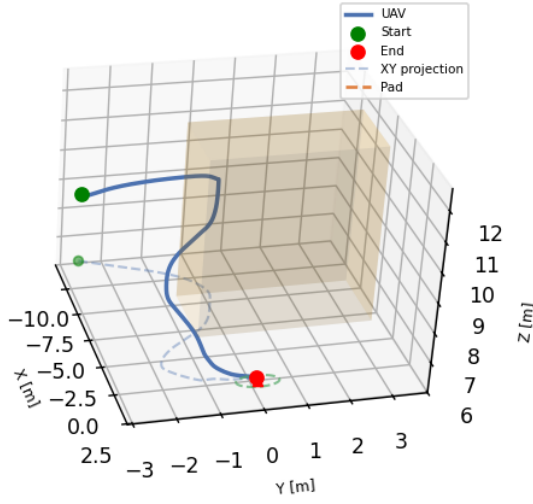
Figure 76: Representative successful episodes under randomized wind geometry (v2).

the anticipatory prior, which depends only on this shared initial conditions.

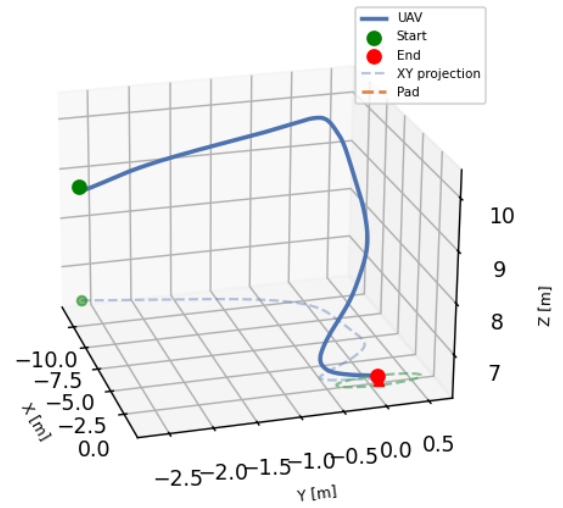
Figure 77 shows the result of this comparison for a representative episode with a 20 m/s downwash. With wind active, the UAV initiates an anticipatory evasive maneuver based solely on the initial UAV and pad positions, then later senses the wind field along its trajectory and makes a real-time adjustment to avoid it. With wind inactive, the anticipatory evasive maneuver is still clearly present and nearly identical in its early portion, while the subsequent real-time adjustment is absent, since there is no disturbance to react to. This confirms that the two behaviors are functionally distinct and separately triggered: the anticipatory maneuver is driven by a learned prior over the initial UAV-pad geometry alone, since it persists identically regardless of wind presence, while the subsequent trajectory correction is driven specifically by the onboard wind observation, since it appears if and only if the disturbance is actually present.

Returning to the randomized wind geometry evaluation introduced earlier – in which wind wall position, offset, and orientation are resampled each episode, rather than held fixed as in the isolation test above – this test results in a 37% success rate for v3. As with v1, the trajectories show a lateral avoidance maneuver, but it appears to be overfit to the expected wind position from training rather than driven by real-time wind sensing. On many occasions, this maneuver fails to correspond to the actual randomized wall placement, and the UAV ends up flying directly into the wind field despite the avoidance attempt, as shown in Figure 78a (the corresponding top-down view in Figure 78b). In some episodes, the randomized wall placement happens to align with the maneuver by chance, leaving the vehicle’s path clear of downwash, which accounts for the observed non-zero success rate.

Taken together, these results highlight that the utility of wind-related observations for this task depends strongly on their form, not merely their presence. The v1 policy, which receives no explicit wind information and must infer disturbance-related effects implicitly through IMU and RPM feedback, is left with essentially no reactive capability and consequently performs the worst of the three, at 31%. Somewhat counter-intuitively, v3, which is given direct access

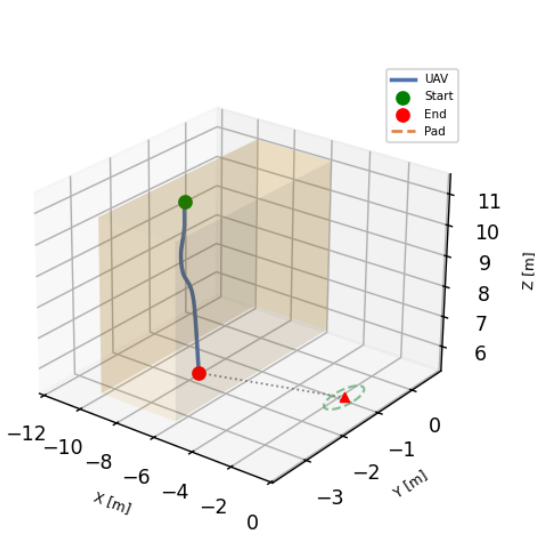


(a) RL guidance trajectory with 20 m/s downwash wind active: anticipatory maneuver with real-time adjustment.

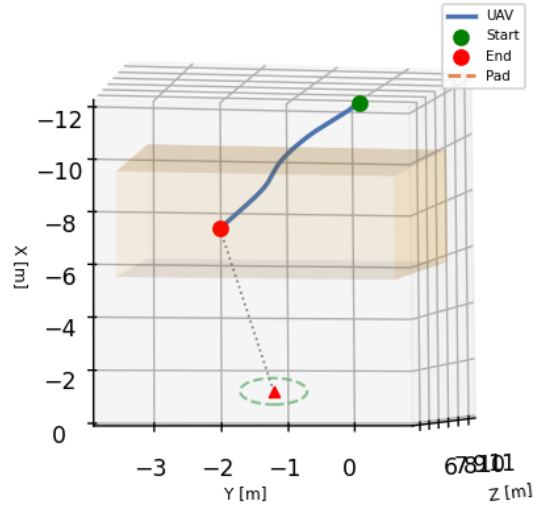


(b) RL guidance trajectory for the same episode and initial conditions with the wind disturbance deactivated: anticipatory maneuver only, no adjustment.

Figure 77: Comparison of RL guidance trajectories for the same episode and initial conditions with (a) the wind disturbance active and (b) the wind disturbance deactivated.



(a) Episode 98 trajectory under randomized wind geometry (v3).



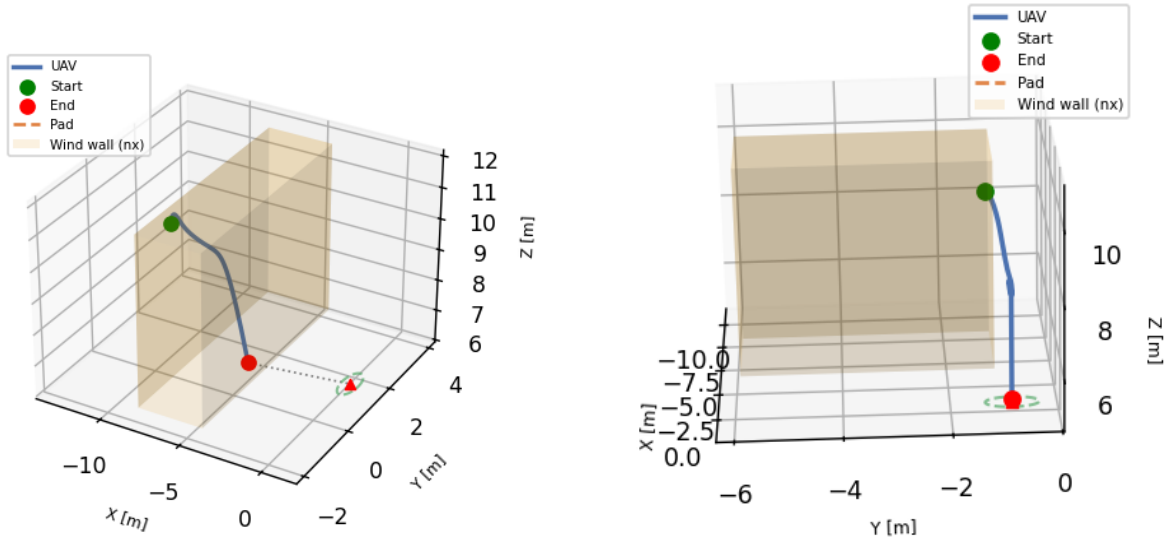
(b) Top-down view of Episode 98.

Figure 78: Representative Episode 98 trajectory for the v3 policy under randomized wind-wall geometry. (a) Three-dimensional trajectory showing the failed avoidance maneuver. (b) Top-down view of the same episode.

to the wind gradient, does not outperform v1 by a wide margin either, achieving only 37%: while gradient information is locally informative, it appears insufficient on its own to reliably disambiguate the vehicle's position relative to a randomly placed wind wall, leaving the policy still heavily reliant on its training-time geometric prior. The v2 policy, by contrast,

is given a short history of recent wind observations rather than an instantaneous reading, and clearly outperforms both alternatives at 66%. This suggests that the temporal structure of the observation, capturing how the wind field is building up or changing over the last several steps, provides a substantially stronger and more reliable signal for triggering a reactive avoidance response. On this basis, v_2 is identified as the most effective observation configuration among those tested for enabling real-time wind-avoidance behavior under randomized disturbance geometry.

By contrast, the baseline guidance law executes a straight-line trajectory at a constant altitude of 3 m above the landing plane until the horizontal position is intercepted, followed by a straight vertical descent onto the landing pad. As this approach incorporates no mechanism for disturbance avoidance, it frequently traverses the downwash region situated between the UAV and the landing pad, thereby degrading landing performance; a representative unsuccessful episode illustrating this outcome is shown in Fig. 79a. Owing to the randomized initial conditions, however, the approach path is occasionally unobstructed by the downwash region, yielding a non-zero success rate of 24% for the baseline controller in this scenario. Figure 79b illustrates such a case, in which the relative initial positions of the UAV, landing pad, and wind field result in an approach path that largely avoids the downwash region, allowing the controller to complete a successful landing despite the absence of any active disturbance-avoidance capability.



(a) Representative unsuccessful episode (Episode 97), where the approach path passes through the downwash region.

(b) Representative successful episode (Episode 99), where the randomized initialization leaves the approach path clear of the downwash region.

Figure 79: Representative trajectories under the baseline guidance law. (a) Unsuccessful episode in which the straight-line approach intersects the downwash region. (b) Successful episode in which the randomized initial geometry leaves the approach path largely clear of the disturbance.

Table 36 summarizes the success rates achieved by each policy configuration under randomized wind-wall geometry, alongside the baseline guidance law evaluated under the same wind conditions for reference.

Table 36: Success rates under randomized wind-wall geometry (Test 3), compared against the baseline guidance law under identical wind conditions.

Configuration	Observation Modality	Success Rate (%)
Baseline guidance law	—	24
v1	IMU + RPM (implicit)	31
v2	10-step wind history	66
v3	Wind gradient	37

Table 37 reports the metrics used to characterize each configuration’s ability to avoid or mitigate the influence of the wind field during the approach phase. All metrics are computed over identical wind-block realizations across configurations to ensure a fair comparison. The metrics are defined as follows:

- **Path Length [m]** – Total arc length of the flown trajectory, obtained by summing the Euclidean distance between consecutive position samples from the start of the approach to touchdown.
- **Path Efficiency [-]** – Ratio of the straight-line distance between the approach start point and the landing pad to the actual path length flown; a value closer to one indicates a more direct trajectory.
- **Detour [m]** – Additional distance flown relative to the straight-line distance (Path Length minus the straight-line distance), representing the extra travel incurred by avoidance manoeuvres.
- **Time in Wind [s]** – Duration for which the UAV remains within the defined wind-affected (downwash) region during the approach.
- **Peak Lateral Deviation [m]** – Maximum cross-track distance between the UAV and the nominal straight-line approach path.
- **Peak Vertical Deviation [m]** – Maximum deviation of the UAV altitude from its nominal approach altitude profile.
- **Approach Heading vs. Wall [deg] (N)** – Mean angle between the UAV heading and the orientation of the wind wall at the point of encounter, averaged over the same N episodes with recorded wind exposure.

The results show a clear trade-off between path directness and wind exposure severity for v2 relative to the baseline. The v2 policy flies substantially longer paths (16.62 m versus 11.70 m), resulting in lower path efficiency (0.658 versus 0.819) and a larger detour (8.03 m versus 2.64 m), reflecting active maneuvering to avoid the wind field. This behavior is further supported by the considerably larger peak lateral deviation (2.005 m versus 0.414 m), whereas the baseline’s near-zero lateral deviation confirms that it follows an essentially straight approach irrespective of the wind location.

Despite the longer trajectories, v2 exhibits more favorable interactions with the wind field. It experiences fewer episodes with measurable wind exposure ($N = 58$ versus $N = 78$ for the

Table 37: Wind-avoidance metrics under active wind, evaluated on identical wind-block realizations across configurations.

Metric	v1 (Mean $\pm$ Std)	v2 (Mean $\pm$ Std)	v3 (Mean $\pm$ Std)	Baseline (Mean $\pm$ Std)
Path Length [m]	9.854 ± 4.499	16.62 ± 5.02	9.736 ± 2.787	11.70 ± 4.86
Path Efficiency [-]	0.902 ± 0.084	0.658 ± 0.120	0.912 ± 0.040	0.819 ± 0.118
Detour [m]	0.784 ± 2.418	8.03 ± 3.77	0.712 ± 0.488	2.64 ± 2.75
Time in Wind [s]	0.383 ± 0.226	0.292 ± 0.571	0.397 ± 0.293	0.472 ± 0.268
Peak Lateral Deviation [m]	0.998 ± 1.488	2.005 ± 1.614	0.740 ± 0.496	0.414 ± 0.453
Peak Vertical Deviation [m]	0.875 ± 0.407	2.430 ± 1.022	1.135 ± 0.407	2.228 ± 0.597
Approach Heading vs Wall [deg] (<i>N</i>)	13.773 $\pm$ 11.158 (68)	23.79 $\pm$ 9.17 (58)	6.548 $\pm$ 5.063 (68)	10.49 $\pm$ 6.17 (78)

baseline), and spends less time within the wind region (0.292 s versus 0.572 s). Furthermore, v2 approaches the wind wall at a larger mean heading angle (23.79° versus 10.49°), suggesting that it intercepts the disturbance more obliquely, thereby reducing the severity of the resulting downwash. The peak vertical deviation remains similar for both configurations (2.430 m versus 2.228 m), indicating comparable rejection of vertical disturbance irrespective of the lateral avoidance strategy.

In contrast, v1 and v3 exhibit quantitatively similar behavior to each other, consistent with the trajectory-level observations discussed earlier. Both policies fly comparatively short, efficient paths (9.854 m and 9.736 m, respectively) with high path efficiency (0.902 and 0.912) and small detours (0.784 m and 0.712 m), indicating that neither policy meaningfully deviates from the shortest-path corridor between the approach start point and the landing pad. Their peak lateral deviations (0.998 m and 0.740 m) are also substantially smaller than that of v2 (2.005 m). Notably, v3’s approach heading relative to the wind wall is the smallest of all configurations (6.548°), indicating a near head-on intersection with the disturbance, which is consistent with a policy that maneuvers according to an expected wind location rather than reacting to the true wind position. Both v1 and v3 also experience a relatively high number of wind-exposure episodes ($N = 68$ for both) compared with v2 ($N = 58$), reinforcing that their limited lateral deviation does little to reduce the frequency of wind encounters. These patterns support the qualitative observation that v1 and v3 follow a largely shortest-path approach strategy, with only minor deviations that are insufficiently coupled to the true wind location to meaningfully reduce exposure, resulting in worse overall performance than v2.

Taken together, these results identify v2 as the best-performing configuration in this scenario. It achieves the highest success rate among the evaluated policies, and its trajectory-level metrics indicate that this performance, while not consistent, stems from deliberate, wind-informed avoidance behavior rather than favorable geometry alone. Although v2 accepts longer, less direct trajectories, this trade-off allows it to reduce both the frequency and severity of wind exposure – distinguishing it from v1, v3, and the baseline, whose limited or purely geometry-driven avoidance leaves them consistently exposed to disturbances they cannot reliably survive.

Beyond wind-avoidance behavior, landing quality was compared between the two approaches using successful landings only, with results summarized in Table 38.

Table 38: Landing quality comparison considering successful landings only (mean $\pm$ std).

Metric	Baseline ($N = 24$)	RL / v2 ($N = 66$)
XY Landing Error [m]	0.159 ± 0.080	0.231 ± 0.079
Time to Land [s]	10.650 ± 1.224	6.525 ± 1.300
Roll at Landing [$^\circ$]	-0.004 ± 0.168	1.305 ± 3.301
Pitch at Landing [$^\circ$]	0.301 ± 0.459	-0.221 ± 1.903
Relative V_z at Touchdown [m/s]	0.341 ± 0.009	0.187 ± 0.060

Consistent with the results obtained for the previously evaluated moving-platform landing configuration, the RL policy lands substantially faster and more softly than the baseline, reducing mean landing time by 38.7% and relative touchdown vertical velocity by 45.2%. These improvements come at the expense of horizontal landing precision, with mean XY landing error increasing by 45.3%. Under localized wind training, the agent is also observed to favor side approach landings over the aligned descent logic seen in earlier configurations, suggesting that a curriculum-learning approach may have helped the policy discover a more robust final touchdown maneuver. Furthermore, touchdown attitude exhibits noticeably greater variability for the RL policy, with roll and pitch standard deviations of 3.301° and 1.903° , respectively, compared with 0.168° and 0.459° for the baseline. This behavior is consistent with the more aggressive, actively corrective maneuvers required to avoid the localized wind disturbance, in contrast to the baseline’s undisturbed, straight-line approach.

Part IV:

Closing Remarks

8 CONCLUSIONS

This thesis investigated RL as a guidance strategy to enable lift-and-cruise quadplane UAVs to autonomously land on moving maritime platforms under localized wind disturbances. To this end, a quadplane model was first developed in a PyBullet simulation environment that included aerodynamic effects. A PID-based inner-loop tracking controller was then developed, with control allocation assigned to the four hover propellers, assuming the vehicle operates in hover configuration during the terminal landing phase. A heuristic finite-state machine guidance law, inspired by current rotorcraft ship landing procedures, was implemented as a performance baseline. Two distinct RL training configurations were evaluated: one targeting moving-platform tracking and landing under varying sea states, and one targeting approach and landing under spatially localized wind disturbances on a stationary platform. This work demonstrated the following key findings:

- Reinforcement learning is a viable method for developing guidance strategies for hybrid platforms performing maritime landings.
- Compared to the baseline, the RL-based guidance law performs better at landing on a moving platform in sea states up to 7, achieving shorter touchdown times and softer touchdown velocities.
- Guided exploration via initial-state randomization substantially accelerates convergence, effectively addressing the sparse-reward, large-state-space nature of the guidance problem and enabling the agent to learn effective landing behavior that was otherwise difficult to discover through random exploration alone.
- When trained in scenarios that expose the agent to localized wind fields, the policy learns the approximate position of the wind field from training statistics, while also displaying avoidance behavior driven by real-time wind sensing.

To structure the development efforts in this thesis and guide the achievement of the research objective, research questions were defined. The following sections address each of these research questions, formulated in Section 3, and outline the key limitations of the work.

8.1 Research Question Answers

The research questions were organized into three topic sets, with answers emerging across different phases of the work. Collectively, they are concluded to have contributed to achieving the research objective. Each question is explicitly answered below, with references to the thesis sections where it is primarily addressed, aiming to provide a useful reference for future researchers on the different design aspects involved.

A. RL Problem Formulation

This set of research questions was formulated to frame the research problem and establish the fundamental basis of the study. Addressing these questions provided the necessary theoretical foundation for the remainder of the research. Their detailed answers are presented in the literature review, while a concise overview is provided here.

A. RL Problem Formulation

RQ-A1: Which control performance metrics and constraints characterize successful autonomous landings of a UAV on a moving maritime platform?

This question is addressed in detail in Section 2.1 (Maritime Landing Problem). The maritime landing problem, for both manned helicopters and unmanned VTOL vehicles, requires satisfying four complementary objectives: accuracy in reaching the designated landing site, safety in maintaining touchdown velocity and attitude within prescribed bounds, smoothness in tracking the approach trajectory while preserving dynamic stability, and robustness to modeling uncertainties, external disturbances, and sensor noise.

The first two objectives are quantified through the relative states between the UAV and the target platform. Within this thesis, a successful landing is defined as one satisfying all of the following conditions simultaneously at touchdown time t_f :

$$\begin{aligned}\lim_{t \rightarrow t_f} |p_{\text{rel},z}(t)| &= 0, \\ \lim_{t \rightarrow t_f} \|\mathbf{p}_{\text{rel},xy}(t)\| &\leq 0.5 \text{ m}, \\ \lim_{t \rightarrow t_f} |\dot{p}_{\text{rel},z}(t)| &\leq 0.7 \text{ m/s}, \\ \lim_{t \rightarrow t_f} \|\boldsymbol{\eta}(t)\| &\leq 10^\circ.\end{aligned}$$

where $\mathbf{p}_{\text{rel},xy}$ denotes the horizontal position error, $\dot{p}_{\text{rel},z}$ the relative vertical velocity at contact, and $\boldsymbol{\eta}$ the vehicle roll and pitch angles. These thresholds were established in consultation with industry experts and reflect three practical considerations. The 0.5 m horizontal tolerance corresponds to the usable landing area of the target ship deck, ensuring that the vehicle touches down within the designated zone. The 0.7 m/s vertical velocity limit is derived from the structural integrity requirements of the airframe, beyond which impact loads may risk damaging the landing gear or hull. The 10° attitude constraint ensures that the vehicle remains within its tipping stability margin at contact, preventing lateral sliding or rollover on a moving and wet deck surface.

A. RL Problem Formulation

RQ-A2: How can the maritime UAV landing problem be formulated as a guidance-level RL problem that interfaces with a classical PID-based tracking controller?

The formulation is motivated by a survey of RL-based UAV landing research presented in Section 2.3, which found that guidance-level approaches, where the RL policy operates in the outer loop and outputs position, velocity, or acceleration commands to a classical tracking controller, consistently outperform baseline solutions for moving-platform landing tasks, primarily demonstrated on quadrotor platforms. Adopting this architecture preserves the physics-based tracking controller and motor allocation logic unchanged, while the learned policy focuses solely on high-level trajectory decisions.

The controller architecture is chosen to replicate the PX4 quadrotor cascade structure, consisting of nested position, velocity, attitude, and rate loops. For the RL formulation, the

outermost position loop is bypassed, and the policy directly outputs 3-dimensional velocity commands that are passed to the velocity tracking controller. This required a dedicated bypass in the control pipeline and reduces the action space to a compact, physically meaningful 3-dimensional continuous vector.

The observation space was designed incrementally. An initial minimal state vector — capturing relative position, relative velocity, attitude, body rates, and previous action — was sufficient for stationary and moving platform landing, validating the core formulation and the suitability of the method for the control task. Wind disturbances motivated two successive extensions: first adding IMU-derived accelerations and rotor RPMs as implicit wind indicators, then adding the wind velocity vector expressed in the body frame as an explicit observation. The complete formulation is detailed in Section 5.5.

The reward function combines dense shaping terms derived from the landing performance metrics defined in RQ-A1 with sparse terminal bonuses and penalties, following established practice in similar work. The full reward structure is presented in Section 5.5.

A. RL Problem Formulation

RQ-A3: Which performance metrics are most suitable for quantifying convergence, robustness, and operational landing performance in RL-based maritime UAV landing tasks?

Training convergence is assessed by tracking several learning signals against total RL steps: success rate, cumulative episode reward, minimum distance to the landing platform, and episode duration. Convergence is considered achieved when the success rate plateaus at a stable level, and the mean episode reward stabilizes in the vicinity of the sparse landing bonus. This approach is consistent with the convergence analysis methodology observed in comparable RL-based UAV landing studies.

Operational landing performance is evaluated during inference against the control objectives established in RQ-A1. At the moment of touchdown, the following metrics are recorded: horizontal position error $\|\mathbf{p}_{\text{rel},xy}\|$, relative vertical velocity $\dot{p}_{\text{rel},z}$, and vehicle attitude $\|\boldsymbol{\eta}\|$, assessed against the thresholds defined in Equation 8.1. Flight quality during the approach phase is quantified separately through the root mean square error (RMSE) of roll and pitch angles, body angular rates, and velocity command smoothness, measured as the mean magnitude of successive action differences. Together, these metrics provide a comprehensive characterization of both task success and the behavioral quality of the learned policy.

B. Environment and Disturbance Modeling

B. Environment and Disturbance Modeling

RQ-B1: Which characteristics of the maritime operational environment are essential to represent when learning guidance policies for autonomous UAV landings?

The maritime landing environment presents three compounding layers of difficulty: degraded sensing conditions due to the offshore environment; a moving target driven by wave-induced platform motion; and spatially localized aerodynamic disturbances arising from ship superstructure wake interactions. Each of these is characterized in detail in the literature

review: state estimation constraints in Section 2.6, platform dynamics in Section 2.7, and wind and wake structure in Section 2.8, including how prior RL-based landing studies have represented them.

For this thesis, the essential characteristics to include were identified as platform motion and spatially localized wind disturbances, as these directly determine the guidance-level decisions the agent must learn to make. Sensor noise and estimation uncertainty, while operationally relevant, were excluded from training to avoid confounding policy convergence; robustness to measurement noise was instead evaluated separately at inference time.

Two further simplifying assumptions were made to maintain a manageable scope:

- Platform attitude, including the periodic roll, pitch, and yaw induced by wave action, was excluded, retaining only translational heave, surge, and sway motion.
- The ship airwake is a complex, geometry-dependent, and turbulent flow field that is difficult to model without dedicated CFD data for a specific vessel. Accordingly, simplified rectangular localized wind zones with parameterized downwash and Dryden turbulence were used in place of a physics-based localized wind field model. This simplification is justified by the proof-of-concept nature of the study: demonstrating that an RL guidance policy can learn to detect and respond to spatially localized wind disturbances does not require a high-fidelity reproduction of a ship wake.

Future work may improve on this by incorporating ship attitude into the platform motion model, positioning wind zones to more closely resemble experimentally or numerically characterized wake environments, and including sensor noise in training. Such extensions would narrow the sim-to-real gap and elevate the conclusions to an operationally representative level.

C. Quadplane UAV Modeling

These research questions focus on the modeling of the quadplane UAV. A detailed description of the design steps is provided in Section 5.1 (Quadplane Dynamic Model). The resulting quadplane model is controlled using a PID controller, whose design and tuning are presented in Section 5.2 (Tracking Controller). While the detailed answers to these research questions are presented in the subsequent sections, a brief overview is provided here to summarise the key conclusions.

C. Quadplane UAV Modeling

RQ-C1: Which dynamic flight phases of the quadplane UAV (cruise, transition, and hover) are relevant for the guidance problem under investigation?

Quadplane operations span three distinct flight phases: cruise, transition, and hover, each requiring a different control allocation strategy. The transition between these phases—i.e., determining when and how the vehicle switches from fixed-wing to multirotor authority—remains an open research problem without a universally accepted solution and is therefore typically treated as a design choice rather than a prescribed procedure.

Given that the scope of this thesis focuses on guidance-level landing control rather than transition management the problem is restricted to the hover phase throughout. This

assumption simplifies the control architecture, avoids the considerable complexity associated with transition dynamics, and aligns with current operational practice.

One important limitation of this assumption should be noted. Aerodynamic analysis of the hover phase shows that headwind conditions require nose-down pitching to maintain position, causing the wing to generate lift with a downward component relative to the vehicle body, thereby degrading hover performance. In offshore maritime environments, where persistent and directional winds are common, this represents a significant constraint. It is therefore suggested that evaluating the landing manoeuvre within the transition phase—where the pusher or puller propeller remains active and provides additional aerodynamic authority—may yield improved disturbance rejection and is recommended as a direction for future work.

C. Quadplane UAV Modeling

RQ-C2: How can the different dynamic flight phases of a quadplane UAV be modeled in PyBullet with sufficient fidelity for guidance-level RL?

Since the RL policy operates at guidance level—producing velocity commands that are tracked by a classical inner-loop controller—the simulation is not required to reproduce every dynamic detail of the full flight envelope. Instead, sufficient fidelity is achieved when the vehicle’s translational and rotational response to velocity commands, as well as its sensitivity to external disturbances, are represented with enough realism for the learned guidance policy to be meaningful and potentially transferable.

The PyBullet simulation implements rigid-body dynamics with full six-degree-of-freedom translational and rotational equations of motion. Quadrotor thrust and torque allocation are modelled explicitly, with a first-order actuator lag applied to rotor RPM to capture motor dynamics. A blended aerodynamic model spanning the full flight envelope is incorporated, in which lift, drag, and lateral side force are computed as continuous functions of angle of attack and sideslip angle. A sigmoid blending function is used to transition smoothly between attached-flow and separated-flow regimes. Since PyBullet does not provide a native atmospheric model, wind disturbances are introduced through a custom local wind field module incorporating Dryden turbulence, enabling realistic stochastic wind exposure during training. The assumption of hover-only operation implies that cruise and transition control allocation—such as the pusher or puller propeller—is not activated during the guidance task; however, aerodynamic forces associated with cruise flight are still captured by the model.

The following physical phenomena are not modelled and represent the primary sources of the sim-to-real gap for this platform:

- **Transition and cruise control allocation:** the pusher/puller propeller is not engaged, and fixed-wing control surface authority is not modeled.
- **Aerodynamic moments:** the aerodynamic model produces forces only; pitching, rolling, and yawing moments resulting from aerodynamic loading are neglected.
- **Rotor-to-rotor aerodynamic interactions:** wind tunnel experiments reported in the literature show that inter-rotor interference generates significant asymmetric moments, particularly under lateral wind conditions; these effects are not captured.

- **Ground effect:** the increase in rotor thrust efficiency in close proximity to the deck surface is not modeled, which may influence hover authority and descent dynamics near touchdown.
- **Blade flapping:** aeroelastic rotor blade dynamics are neglected.
- **System identification:** since no physical platform was available, actuator lag and dynamic parameters are assumed from prior work rather than identified from flight data, introducing parametric uncertainty.
- **Aerodynamic model scope:** the lift and drag coefficients are adopted from studies of isolated wing aerodynamics under wind loading and do not account for the influence of the fuselage, rotor downwash, or other quadplane body components on the aerodynamic polar.

These simplifications are considered acceptable within the scope of this research, which aims to evaluate whether a guidance-level RL policy can learn to perform autonomous landing under maritime conditions rather than to produce a flight-ready controller. At the guidance level, the policy is required to learn approach and descent strategies rather than precise attitude regulation, and the dominant source of task difficulty stems from platform motion and wind exposure, both of which are represented in the simulation. The modeling fidelity is therefore deemed sufficient for the research objective. Sim-to-real transfer of the trained policy to a physical platform would require revisiting the omitted effects, particularly rotor-to-rotor interactions, aerodynamic moments, and system-identified actuator dynamics, and is left as future work.

C. Quadplane UAV Modeling

RQ-C3: How can the PID tracking controller be designed and tuned to reliably track the guidance commands while supporting stable and safe landings in a maritime context?

The tracking controller is designed as a four-level cascaded PID architecture, replicating the PX4 quadrotor control structure. It consists of nested angular rate, attitude, velocity, and position loops operating at progressively separated bandwidths. The inclusion of proportional, integral, and derivative action at each level is determined systematically: integral action is introduced where persistent disturbances act directly on the loop state and are not corrected by an outer loop, while derivative action is omitted where the inner loop already provides sufficient damping. This reasoning yields full PID controllers at the angular rate and velocity levels, and proportional-only controllers at the attitude and position levels.

Controller gains are obtained through iterative frequency-domain tuning in MATLAB, proceeding sequentially from the innermost loop outward using linearised transfer function models. Crossover frequencies are selected according to the cascade separation principle, resulting in approximate bandwidths of 18, 9, 4, and 1.5 rad/s for the rate, attitude, velocity, and position loops, respectively. All loops are verified to satisfy the prescribed gain margin (≥ 6 dB), phase margin ($45^\circ - 70^\circ$), and sensitivity peak ($M_S \leq 2.0$) requirements. Maintaining strict bandwidth separation between cascade levels proved critical as insufficient frequency separation between adjacent loops introduced inter-loop coupling and oscillatory responses that degraded tracking performance.

The linearised frequency-domain analysis provides a strong initial gain design, but small manual adjustments are needed in the nonlinear simulation due to linearisation errors that become evident under large-angle maneuvers and actuator saturation. The tuning process is furthermore sensitive to the accuracy of the inertial parameters, particularly the moments of inertia, where modest changes produce measurable shifts in the achievable phase margin and crossover frequency. Moreover, representing the controller bandwidth realistically in simulation—rather than assuming an idealised instantaneous inner loop—is important for reducing the sim-to-real gap and ensuring that the RL policy learns guidance commands that remain within the trackable envelope of the physical controller.

The tuned gains are subsequently validated in the nonlinear PyBullet simulation environment under representative operating conditions, including velocity step tracking, wind disturbance rejection, and closed-loop landing manoeuvres on both stationary and moving platforms. This confirms that the controller performs reliably as a tracking controller and is ready for RL interfacing. It is noted, however, that the manual, sensitivity-driven tuning process is time-consuming, particularly for complex platforms with uncertain or difficult-to-estimate dynamic parameters. Within this context, replacing the classical tracking controller with a learned low-level policy remains a promising direction for future research, particularly for adapting the control strategy to nonlinearities and uncertainties that are difficult to capture accurately in the model. Another direction here would be investigating more systematic tuning strategies, for example, robust control approaches such as H_∞ control to improve robustness to parameter uncertainty and unmodelled dynamics.

C. Quadplane UAV Modeling

RQ-C4: How can the implemented quadplane simulation model be verified prior to RL implementation?

Simulation verification is conducted at two levels: unit testing of individual model components and system-level testing of the integrated simulation environment. Results of the latter are reported in detail in Chapter 6. Verification is treated as a foundational activity rather than a final check, serving as the primary diagnostic tool throughout model development and enabling the identification and correction of several implementation errors that would otherwise have corrupted the RL training process. The verification effort constitutes one of the most substantial contributions of this research in terms of time and rigor. It is strongly recommended that future research involving custom simulation environments for RL adopt a similarly structured verification strategy from the outset, as undetected modeling errors at this stage propagate directly into the learned policy and are difficult to diagnose retroactively.

D. Proposed Solution Performance Evaluation

Building on the development and evaluation of the proposed RL-based adaptive guidance solution presented throughout this thesis, the corresponding research questions can now be directly addressed.

D. Proposed Solution Performance Evaluation

RQ-D1: How does the proposed solution compare to the baseline heuristic guidance law in terms of ship landing performance?

This question is answered in detail in Part II, with additional results presented in Part III. The results are organized around the two central problems addressed independently in this study: moving-platform landing and localized wind-disturbance avoidance. Across both cases, the RL-based guidance policy achieves higher success rates than the baseline, consistently producing landings within the defined thresholds.

For the moving-platform case without wind, the RL policy adopts a direct descent strategy toward the platform, with final touchdown corrections made only in close proximity to the pad. The RL-based approach achieves comparable XY landing error statistics, indicating that touchdown precision is largely preserved relative to the baseline approach. The most substantial improvement is observed in time to land, where the RL policy reduces the mean landing duration by 67.9%, reflecting a more direct landing maneuver rather than prolonged station-keeping above the target. Notably, this reduction in landing time does not come at the cost of touchdown softness: the RL policy also achieves a 41.8% reduction in relative vertical speed at touchdown.

With the wind field present, the RL-based solution successfully employs two complementary avoidance mechanisms, predictive and reactive, to reduce wind exposure during approach. This raises the success rate, evaluated under identical environment realizations, from 24% to 66%. The reactive avoidance mechanism is verified to arise specifically from real-time wind observation, a capability absent from the baseline heuristic approach. Furthermore, considering successful landings only, the RL-based solution again shows faster landings on average, reducing mean time to land from 10.65 ± 1.22 s to 6.53 ± 1.30 s, alongside a more effective braking strategy that yields lower touchdown speeds (0.187 ± 0.060 m/s vs. 0.341 ± 0.009 m/s for the baseline). Horizontal landing accuracy shows a small increase relative to the baseline (0.231 ± 0.079 m vs. 0.159 ± 0.080 m), though touchdown precision remains within the same order of magnitude.

D. Proposed Solution Performance Evaluation

RQ-D2: To what extent does the proposed solution exhibit evasive behavior under wind conditions that cause actuator saturation?

This research question stemmed from the hypothesis that evasive behavior could emerge through environmental exposure of localized wind fields during training, without being explicitly engineered into the guidance law. This is confirmed by the analysis, which identifies two complementary avoidance mechanisms: a predictive component and a reactive one.

The predictive component appears as an anticipatory maneuver driven by the initial UAV-pad geometry alone, present even when no wind disturbance is active, reflecting the distribution of localized wind-field placements experienced during training. The reactive component, by contrast, is triggered by real-time onboard wind observations and produces trajectory corrections only once the vehicle encounters the wind field itself, allowing it to route around or above the active wind zone. Together, these mechanisms are the primary driver behind the

substantial increase in success rate from 24% to 66% relative to the baseline solution.

The RL-based solution is exposed to the wind field in 58 of the 100 evaluation episodes, compared with 78 episodes for the baseline, demonstrating that the predictive avoidance logic substantially reduces wind exposure. Among the 58 wind-exposed episodes, the RL solution achieves 24 successful landings; qualitative inspection of these episodes reveals clear reactive avoidance behavior, consistent with the fact that exposure is typically confined to the periphery of the localized wind field, where maneuvering around the disturbance remains feasible. By contrast, only 2 of the baseline’s 78 wind-exposed episodes result in successful landings, displaying a lack of reactive avoidance capability.

It should be noted that the predictive avoidance logic arises entirely from the distribution of wind-field randomizations experienced during training, and therefore does not constitute a meaningful predictive capability in the physical sense. If the wind-field randomization and placement were further refined to better represent a realistic ship wake, the predictive avoidance behavior could plausibly become more reliable and practically useful.

D. Proposed Solution Performance Evaluation

RQ-D3: How does the choice of observation space formulation influence the agent’s ability to react to wind disturbances in real time?

This question is answered in detail in Part III, Additional Results Chapter. Three observation formulations are compared to isolate the contribution of explicit and implicit wind-state information: v1, which infers wind effects only indirectly through IMU and rotor-speed feedback; v2, which augments this with a 10-step history of directly measured wind velocity; and v3, which instead provides the instantaneous wind-velocity gradient, trading full temporal history for a lower-dimensional observation. All three configurations converge to stable policies within a comparable training budget, indicating that the added observation dimensionality of v2 does not impede learning.

Under randomized wind-wall geometry, the choice of observation formulation is found to have a substantial effect on real-time reactive capability. v1 achieves a 31% success rate and relies on a fixed avoidance maneuver learned from the average wind-field position seen during training, executed before any wind-related signal could be sensed; this maneuver is therefore decoupled from the true, randomized disturbance location, and its non-zero success rate is attributable to incidental alignment rather than genuine avoidance. v3 performs similarly, at 37%: although it has direct access to the wind gradient, this instantaneous signal proves insufficient to reliably disambiguate the vehicle’s position relative to a randomly placed wind wall, leaving the policy still heavily reliant on its training-time geometric prior. v2, by contrast, achieves the highest success rate at 66%, with trajectories exhibiting clear reactive corrections that activate as the vehicle enters the wind field and track its actual, randomized position rather than an expected one. A controlled isolation test, comparing identical initial conditions with and without an active wind disturbance, confirms that this reactive behavior is functionally distinct from, and additive to, an anticipatory maneuver present in all configurations, and that it is triggered specifically by real-time wind observation.

These results indicate that the benefit of wind-related observations depends strongly on their temporal structure as well as their presence: a short history of wind measurements (v2)

provides a substantially stronger and more reliable signal for triggering reactive avoidance than either implicit sensing (v1) or an instantaneous gradient estimate (v3). Accordingly, v2 is identified as the most effective observation configuration among those tested for enabling real-time wind-avoidance behavior under randomized disturbance geometry, nearly tripling the baseline guidance law’s success rate of 24% under identical wind conditions.

8.2 Limitations of Research

The UAV platform modeled in this research was not explicitly validated as a complete, integrated system against a physical counterpart; however, several of its key parameters are grounded in real-world data rather than purely theoretical estimates. The rotor time delay, for instance, was adopted from an experimentally identified motor time constant reported for a comparable quadrotor platform, and the aerodynamic coefficients were derived from wind tunnel testing rather than analytical or simulation-only approximations. Other parameters, such as moments of inertia, mass distribution, and geometric properties, were instead taken from simulation models (e.g., Gazebo) or estimated from comparable real-world platforms. Confidence in the overall model was assessed indirectly through system-level step-response comparisons with other studies. This combination of experimentally derived and simulation-based parameters is considered appropriate for proof-of-concept research of this kind, though a fully integrated validation campaign against a physical platform would be needed before pursuing higher-fidelity, real-world-ready applications.

The aerodynamic model is subject to further limitations, despite its grounding in wind tunnel data. This data was obtained from an isolated wing section rather than the full quadplane geometry, meaning the aerodynamic influence of components such as the fuselage and landing gear remains unaccounted for. Furthermore, side force was approximated from drag via a scaling assumption rather than derived from independent measurements, while aerodynamic moments were omitted entirely, both of which reduce confidence in the fidelity of the resulting representation. Collectively, these simplifications may widen the simulation-to-reality gap and should be addressed in higher-fidelity follow-on work.

A further limitation concerns the wind-field representation used during training. Only a single, localized wind field was simulated, with downwash magnitudes ranging from -5 to -20 m/s, selected to expose the agent to both survivable and critical conditions given that the quadplane’s control authority saturates at downwash velocities exceeding approximately 10 m/s. This configuration does not capture the complexity of a realistic maritime landing scenario, which involves multiple, spatially distributed wind fields arising from wake interference behind a ship’s superstructure. Moreover, as the agent was shown to learn the spatial location of the wind field rather than relying exclusively on real-time observations, a higher-fidelity representation of ship airwake dynamics would likely be required to obtain policies capable of performing in the intended maritime application. Consequently, the conclusions drawn from the present experimental setup should not be extrapolated directly to real maritime operations, where the agent’s behavior and performance may differ substantially.

A related limitation concerns the observability assumptions underlying the wind disturbance experiments. Specifically, the agent’s observation space included direct measurements of the wind velocity experienced by the UAV. This type of sensor data is not readily

available in practice, as accurate omnidirectional wind sensing remains a significant challenge. Consequently, the experimental setup and the conclusions drawn from it rely on an observation that may not be realistically obtainable onboard, further constraining the applicability of these findings to real-world systems.

Finally, this study found that the RL agent’s wind-avoidance strategy combines an anticipatory component, informed by wind statistics encountered during training, with a reactive component driven by real-time wind observations, and that the balance between the two was inconsistent across episodes. As this balance depends on the specific training configuration explored in this work, including the domain randomization strategy, reward function formulation, and observation space composition, the conclusions drawn from the present setup should not be assumed to generalize to other training configurations.

9 RECOMMENDATIONS

Building on the conclusions and limitations of this work, the following recommendations for future research are presented in order of priority:

1. **Further investigation into anticipatory and reactive wind-avoidance behavior.** This study found that the RL agent’s avoidance strategy combines an anticipatory component, informed by wind statistics encountered during training, with a reactive component driven by real-time wind observations. This balance was found to be sensitive to the domain randomization strategy, reward function formulation, and observation space composition. Future work should systematically characterize this sensitivity, and explore curriculum learning as a means of guiding the agent towards more robust reactive behavior.
2. **Extension to realistic ship-wake environments.** The agent was observed to become closely tuned to the position and characteristics of the single localized wind field employed during training, effectively treating it as a fixed environmental feature rather than a variable disturbance. The utility of this anticipatory behavior in a maritime context should therefore be investigated using multiple localized wind fields arranged to represent a ship’s wake, with the number, positions, and characteristics of these fields randomized to reflect the diversity of superstructure geometries that may be encountered during operation.
3. **Joint treatment of moving-platform tracking and wind-disturbance rejection.** This study addressed moving-platform landing and localized wind disturbance rejection as separate challenges. These problems could naturally be unified into a single guidance task, and doing so may benefit from alternative network architectures, such as recurrent neural networks, or from training strategies such as curriculum learning, to manage the increased complexity associated with the combined problem.
4. **Realistic wind-sensing assumptions.** The direct, omnidirectional wind velocity measurement assumed in this work exceeds current practical UAV sensing capabilities. Independent of the RL guidance framework, further research should investigate wind approximation, estimation, and sensor fusion techniques that provide usable wind velocity information to the guidance policy under realistic sensing constraints.
5. **RL as an inner-loop controller.** This thesis employed RL as a guidance strategy layered above a classical cascaded PID tracking controller, the development and tuning of which proved time-consuming. Future work should investigate the use of RL as an inner-loop controller, commanding rotor speeds or inner-loop states directly and thereby bypassing the PID architecture entirely. Such an approach would avoid reliance on a linearized model and may improve robustness to unforeseen disturbances. Its feasibility has already been demonstrated for a quadrotor, including real-world validation, by Saj et al. (2022).
6. **Model validation for sim-to-real transfer.** Provided that results are obtained that justify further pursuit towards real-world deployment, dedicated system identification and wind tunnel testing on the specific quadplane geometry should be carried out and implemented in training to reduce the simulation-to-reality gap.

REFERENCES

- Abo Mosali, N. (2022). An adaptive multi-level quantization-based reinforcement learning model for enhancing uav landing on moving targets. *Sustainability*, 14(14), 8825. <https://doi.org/10.3390/su14148825>
- Abujoub, S., McPhee, J., & Irani, R. A. (2020). Methodologies for landing autonomous aerial vehicles on maritime vessels. *Aerospace Science and Technology*, 106, 106169. <https://doi.org/10.1016/j.ast.2020.106169>
- Aird, P. (2019, January). Deepwater metocean environments. <https://doi.org/10.1016/B978-0-08-102282-5.00004-1>
- Ali, A. M., Gupta, A., & Hashim, H. A. (2024). Deep reinforcement learning for sim-to-real policy transfer of vtol-uavs offshore docking operations. *Applied Soft Computing*, 162, 111843. <https://doi.org/10.1016/j.asoc.2024.111843>
- Amendola, J., Cenkeramaddi, L. R., & Jha, A. (2023). Single reinforcement learning policy for landing a drone under different ugv velocities and trajectories. *Proceedings of the 11th International Conference on Control, Mechatronics and Automation (ICCMA)*, 115–120.
- Amendola, J., Cenkeramaddi, L. R., & Jha, A. (2024). Drone landing and reinforcement learning: State-of-art, challenges and opportunities. *IEEE Open Journal of Intelligent Transportation Systems*, 5, 520–539. <https://doi.org/10.1109/OJITS.2024.3444487>
- Aragon, C. R., & Long, K. R. (2005). Airflow hazard visualization for helicopter pilots: Flight simulation study results [Discusses helicopter airflow hazards including turbulence and unsteady ship airwakes]. *61st Annual Forum Proceedings of the American Helicopter Society*. <https://ntrs.nasa.gov/citations/20050184140>
- Åström, K. J., & Hägglund, T. (1995). *Pid controllers: Theory, design, and tuning* (2nd ed.). ISA – The Instrumentation, Systems, Automation Society.
- Bai, Y., & Bai, Q. (2005). Chapter 6 - hydrodynamics around pipes. In *Subsea pipelines and risers* (pp. 89–100). Elsevier Science Ltd. <https://doi.org/10.1016/B978-008044566-3.50008-7>
- Cao, Y., Qin, Y., Tan, W., & Li, G. (2022). Numerical simulation of fully coupled flow-field and operational limitation envelopes of helicopter-ship combinations. *Journal of Marine Science and Engineering*, 10(10). <https://doi.org/10.3390/jmse10101455>
- Cárdenas R., C. A., Grisales, V. H., Collazos Morales, C. A., Cerón-Muñoz, H. D., Ariza-Colpas, P., & Caputo-Llanos, R. (2020). Quadrotor modeling and a pid control approach. In U. S. Tiwary & S. Chaudhury (Eds.), *Intelligent human computer interaction* (pp. 281–291). Springer International Publishing.
- Chen, C. C., Tsai, J., Peng-Chen, L., & Lai, C.-A. (2020). Accuracy improvement of autonomous straight take-off, flying forward, and landing of a drone with deep reinforcement learning. *International Journal of Computational Intelligence Systems*, 13(1), 914. <https://doi.org/10.2991/ijcis.d.200410.001>
- Chen, S., Mo, Y., Wu, X., Xiao, J., & Liu, Q. (2024). Reinforcement learning-based energy-saving path planning for uavs in turbulent wind. *Electronics*, 13(16), 3190. <https://doi.org/10.3390/electronics13163190>
- Choi, B., Jung, W. J., Byeon, M. S., Jung, S. Y., Song, J. W., & Kim, Y. H. (2025). Reinforcement learning reward design for robust autonomous shipboard drone landing. *Journal of Positioning, Navigation, and Timing*, 14(4), 421–429. <https://doi.org/10.11003/JPNT.2025.14.4.421>

- de Jong, J. M. (2019). *Model-based flight control for a vtol aircraft with independently tilting rotors* [Master's thesis]. Delft University of Technology.
- Deisenroth, M., & Rasmussen, C. E. (2011). Pilco: A model-based and data-efficient approach to policy search. *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 465–472.
- Ding, W., & Huang, H. (2023). Research on dynamic landing guidance algorithm for the maritime quadrotor. *Ocean Engineering*, 271, 113775. <https://doi.org/10.1016/j.oceaneng.2023.113775>
- Ducard, G. J., & Allenspach, M. (2021). Review of designs and flight control techniques of hybrid and convertible vtol uavs. *Aerospace Science and Technology*, 118, 107035. <https://doi.org/10.1016/j.ast.2021.107035>
- Eimer, T., Biedenkapp, A., Hutter, F., & Lindauer, M. (2021). Self-paced context evaluation for contextual reinforcement learning. *Proceedings of the 38th International Conference on Machine Learning*, 139, 3152–3162.
- Eschmann, J., Albani, D., & Loianno, G. (2024). Data-driven system identification of quadrotors subject to motor delays. *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 8095–8102. <https://doi.org/10.1109/IROS58592.2024.10801441>
- Feng, Y., Zhang, C., Baek, S., Rawashdeh, S., & Mohammadi, A. (2018). Autonomous landing of a uav on a moving platform using model predictive control. *Drones*, 2(4), 34. <https://doi.org/10.3390/drones2040034>
- Finger, D. F., Braun, C., & Bil, C. (2019). Impact of electric propulsion technology and mission requirements on the performance of vtol uavs [Published online 11 December 2018]. *CEAS Aeronautical Journal*, 10(3), 827–843. <https://doi.org/10.1007/s13272-018-0352-x>
- Frølich, M. (2015). *Automatic ship landing system for fixed-wing uav* [Master's thesis]. Norwegian University of Science and Technology. <https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2352446>
- Fuji, T., Ito, K., Matsumoto, K., & Yano, K. (2018). Deep multi-agent reinforcement learning using dnn-weight evolution to optimize supply chain performance. *Proceedings of the 51st Hawaii International Conference on System Sciences (HICSS)*. <https://doi.org/10.24251/HICSS.2018.157>
- Fujimoto, S., Hoof, H., & Meger, D. (2018). Addressing function approximation error in actor-critic methods. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 1587–1596.
- Fujimoto, S., van Hoof, H., & Meger, D. (2018). Addressing function approximation error in actor-critic methods. <https://arxiv.org/abs/1802.09477>
- Gao, Y., Li, Q., & Wang, H. (2023). Model predictive control-based trajectory generation for agile landing of unmanned aerial vehicle on a moving boat. *IEEE Transactions on Control Systems Technology*, 31(4), 1550–1563. <https://doi.org/10.1109/TCST.2023.3245678>
- Greensmith, E., Bartlett, P. L., & Baxter, J. (2004). Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5, 1471–1530.
- Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 80, 1861–1870.
- Hasselmann, K., Barnett, T. P., Bouws, E., Carlson, H., Cartwright, D. E., Enke, K., Ewing, J. A., Gienapp, H., Hasselmann, D. E., Kruseman, P., Meerburg, A., Müller, P., Olbers, D. J.,

- Richter, K., Sell, W., & Walden, H. (1973). Measurements of wind-wave growth and swell decay during the joint north sea wave project (jonswap) [Joint North Sea Wave Project (JONSWAP) experiment]. *Deutsche Hydrographische Zeitschrift, Ergänzungsheft*, 8(12), 1–95. <http://resolver.tudelft.nl/uuid:f204e188-13b9-49d8-a6dc-4fb7c20562fc>
- Ho, D., Linder, J., Hendebay, G., & Enqvist, M. (2017). Mass estimation of a quadcopter using imu data. *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, 1260–1266. <https://doi.org/10.1109/ICUAS.2017.7991417>
- Hu, H., Wu, Y., Xu, J., & Sun, Q. (2018). Path planning for autonomous landing of helicopter on the aircraft carrier. *Mathematics*, 6(10), 178. <https://doi.org/10.3390/math6100178>
- Ibinabo, I., & Tamunodukobipi, D. T. (2019). Determination of the response amplitude operator(s) of an fpso. *Engineering*, 11(9), 563–578. <https://doi.org/10.4236/eng.2019.119038>
- Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4, 237–285. <https://jair.org/index.php/jair/article/view/10166>
- Kamath, A., Assadian, F., & Robinson, S. (2020). Multivariable robust control for the powered-descent of a multibody lunar landing system.
- Keipour, A., Pereira, G. A., Bonatti, R., Garg, R., Rastogi, P., Dubey, G., & Scherer, S. (2022). Visual servoing approach for autonomous uav landing on a moving vehicle. *Autonomous Robots*, 46(8), 1315–1332. <https://doi.org/10.1007/s10514-022-10250-7>
- Kim, J. W., McIntosh, K., Shrestha, E., Reddinger, J.-P., & Mishra, S. (2023). A reinforcement learning approach to control of a quadrotor biplane tailsitter for adaptive landing maneuvers. *Proceedings of the Vertical Flight Society 79th Annual Forum & Technology Display*, Paper 1208. <https://doi.org/10.4050/F-0079-2023-18173>
- Kroo, I., Prinz, F. B., Shantz, M., Kunz, P., Fay, G., Cheng, S., Fabian, T., & Partridge, C. (2000). *The mesicopter: A miniature rotorcraft concept phase ii interim report* (Technical Report) (NIAC Phase II study on miniature rotorcraft dynamics and design). Stanford University. Palo Alto, CA.
- Li, J., & Li, Y. (2011). Dynamic analysis and pid control for a quadrotor. *Proceedings of the 2011 IEEE International Conference on Mechatronics and Automation (ICMA)*, 573–578. <https://doi.org/10.1109/ICMA.2011.5985724>
- Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.
- Lu, P., van Kampen, E., de Visser, C., & Chu, Q. (2016). Aircraft fault-tolerant trajectory control using incremental nonlinear dynamic inversion. *Control Engineering Practice*, 57, 126–141. <https://doi.org/10.1016/j.conengprac.2016.09.010>
- Lustosa, L. R. (2017). *La phi-théorie : Une approche pour la conception de lois de commande de vol des véhicules convertibles* [PhD Thesis]. Université de Toulouse, Institut Supérieur de l’Aéronautique et de l’Espace (ISAE-SUPAERO).
- Ma, Q., Wu, Y., Shoukat, M. U., Yan, Y., Wang, J., Yang, L., Yan, F., & Yan, L. (2024). Deep reinforcement learning-based wind disturbance rejection control strategy for uav. *Drones*, 8(11), 632. <https://doi.org/10.3390/drones8110632>
- Ma, Z., Smeur, E. J. J., & de Croon, G. C. H. E. (2022). Wind tunnel tests of a wing at all angles of attack. *International Journal of Micro Air Vehicles*, 14. <https://doi.org/10.1177/17568293221110931>

- Mao, H., Alizadeh, M., Menache, I., & Kandula, S. (2016). Resource management with deep reinforcement learning. *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, 50–56.
- Mathur, A., & Atkins, E. (2023). Experimental aerodynamic analysis of a QuadPlane unmanned aircraft system [Article in Advance]. *Journal of Aircraft*. <https://doi.org/10.2514/1.C036916>
- Mathur, A., & Atkins, E. M. (2021). Design, modeling and hybrid control of a quadplane. *AIAA Scitech 2021 Forum*, 1–16. <https://doi.org/10.2514/6.2021-0374>
- Matiisen, T., Oliver, A., Cohen, T., & Schulman, J. (2017). Teacher-student curriculum learning. *arXiv preprint arXiv:1707.00183*. <https://arxiv.org/abs/1707.00183>
- Messmer, M., Kiefer, B., Varga, L. A., & Zell, A. (2024). Uav-assisted maritime search and rescue: A holistic approach [Comprehensive framework for UAV search and rescue missions]. *arXiv preprint arXiv:2403.14281*.
- Mewhinney, M. (1998, October). *Flow over helicopter carriers* (tech. rep.) (Report discussing ship airwake effects on helicopter operations and related safety incidents). NASA Ames Research Center, Ames Astrogram. https://history.arc.nasa.gov/Astrogram/Astrogram_1998_10_30.pdf
- Mishra, P. K., Gasparino, M. V., Velasquez, A. E. B., & Chowdhary, G. (2021). Deep model predictive control. *arXiv preprint arXiv:2104.07171*.
- Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 1928–1937.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529–533. <https://doi.org/10.1038/nature14236>
- Montes Gomez, E., & Sumner, D. (2022). The wake of a rectangular flat plate. *Fluid Dynamics Research*, 54(6), 065504. <https://doi.org/10.1088/1873-7005/acabe3>
- Narvekar, S., Peng, B., Leonetti, M., Sinapov, J., Taylor, M. E., & Stone, P. (2020). Curriculum learning for reinforcement learning domains: A framework and survey (G. Konidaris, Ed.). *Journal of Machine Learning Research*, 21(181), 1–50.
- Ogata, K. (2010). *Modern control engineering* (5th ed.). Prentice Hall.
- Olivares, D., Fournier, P., Vasishta, P., & Marzat, J. (2024). Model-free versus model-based reinforcement learning for fixed-wing uav attitude control under varying wind conditions. *Proceedings of the 21st International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 1, 79–91. <https://doi.org/10.5220/0012946600003822>
- Panerati, J., Zheng, H., Zhou, S., Xu, J., Prorok, A., & Schoellig, A. P. (2021). 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)*.
- Paris, A., Lopez, B. T., & How, J. P. (2020). Dynamic landing of an autonomous quadrotor on a moving platform in turbulent wind conditions. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 9577–9583. <https://doi.org/10.1109/ICRA40945.2020.9197081>

- Peter, R., Ratnabala, L., Aschu, D., Fedoseev, A., & Tsetserukou, D. (2024). Tornadodrone: Bio-inspired drl-based drone landing on 6d platform with wind force disturbances. <https://doi.org/10.48550/arXiv.2406.16164>
- Pfeifle, O., & Fichter, W. (2022). Minimum power control allocation for incremental control of over-actuated transition aircraft. *Journal of Guidance, Control, and Dynamics*, 46. <https://doi.org/10.2514/1.G006929>
- Portelas, R., Colas, C., Hofmann, K., & Oudeyer, P.-Y. (2019). Teacher algorithms for curriculum learning of deep reinforcement learning agents. *arXiv preprint arXiv:1910.07224*.
- Quan, Q., Wang, S., & Gao, W. (2025). Lifting-wing quadcopter modeling and unified control [Published online]. *Journal of Guidance, Control, and Dynamics*. <https://doi.org/10.2514/1.G008134>
- Rummery, G. A., & Niranjan, M. (1994). *On-line q-learning using connectionist systems* (tech. rep. No. CUED/F-INFENG/TR 166). Cambridge University Engineering Department.
- Saeed, A. S., Younes, A. B., Islam, S., Dias, J., Seneviratne, L., & Cai, G. (2018). A review on the platform design, dynamic modeling and control of hybrid UAVs. *IEEE Access*, 6, 10571–10591.
- Saj, V., Lee, B., Kalathil, D., & Benedict, M. (2022). Robust reinforcement learning algorithm for vision-based ship landing of uavs. *arXiv preprint arXiv:2209.08381*. <https://arxiv.org/abs/2209.08381>
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- Setiawan, H., Kevin, Philip, J., & Monty, J. P. (2022). Turbulence characteristics of the ship air-wake with two different topside arrangements and inflow conditions. *Ocean Engineering*, 260, 111931. <https://doi.org/10.1016/j.oceaneng.2022.111931>
- Setiawan, H., Zheng, K., Kevin, & Monty, J. P. (2025). A probabilistic framework for assessing helicopter landing conditions within the ship air-wake. *Ocean Engineering*, 317, 120027. <https://doi.org/10.1016/j.oceaneng.2024.120027>
- Sieberling, S., Chu, Q. P., & Mulder, J. A. (2010). Robust flight control using incremental nonlinear dynamic inversion and angular acceleration prediction. *Journal of Guidance, Control, and Dynamics*, 33(6), 1732–1742. <https://doi.org/10.2514/1.49978>
- Simmons, B. M., Buning, P. G., & Murphy, P. C. (2020). *Full-envelope aero-propulsive model identification for lift+cruise aircraft using computational experiments* (tech. rep.) (NASA Technical Report). NASA Langley Research Center. Hampton, VA, USA.
- Smeur, E. J., Chu, Q., & de Croon, G. C. (2016). Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles. *Journal of Guidance, Control, and Dynamics*, 39(3), 450–461. <https://doi.org/10.2514/1.G001490>
- Söker, H., Rehfeldt, K., Santjer, F., Strack, M., & Schreiber, M. (2000, October). *Offshore wind energy in the north sea: Technical possibilities and ecological considerations – a study for greenpeace* (Technical Report). Deutsches Windenergie-Institut GmbH and Schreiber Umweltplanung. https://www.offshorewindenergy.org/COD/reports/report-files/report_027.pdf
- Sönmez, S., Montecchio, L., Martini, S., Rutherford, M. J., Rizzo, A., Stefanovic, M., & Valavanis, K. P. (2025). Reinforcement learning-based pd controller gains prediction for quadrotor uavs. *Drones*, 9(8). <https://doi.org/10.3390/drones9080581>
- Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd). MIT Press. <http://incompleteideas.net/book/the-book-2nd.html>

- Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems (NeurIPS)*, 12.
- Tan, L., Wu, J., Yang, X., & Song, S. (2025). Optimal landing trajectory planning method between a uav and a moving vessel. *Journal of Intelligent Robotic Systems*, 99, 123–137. <https://doi.org/10.1007/s10846-025-02288-3>
- United States Coast Guard. (2011, February). *Shipboard helicopter operational procedures manual, comdtinst m3710.2e* [Contains operational limits for relative wind and wind direction considerations during shipboard helicopter operations, including launch and recovery envelopes.]. Washington, DC, USA. <https://studylib.net/doc/26974404/helo-ops-manual-cim-3710-2e>
- Vakilabadi, K. A., Ghafari, H. R., & Ghassemi, H. (2023). Experimental and numerical investigation on a trimaran airwake, geometry modification. *Heliyon*, 9(11), e21144. <https://doi.org/https://doi.org/10.1016/j.heliyon.2023.e21144>
- Vankadari, M. B., Das, K., Shinde, C., & Kumar, S. (2018). A reinforcement learning approach for autonomous control and landing of a quadrotor. *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*, 676–683.
- Wall, A., Thornhill, E., Barber, H., McTavish, S., & Lee, R. (2022). Experimental investigations into the effect of at-sea conditions on ship airwake characteristics. *Journal of Wind Engineering and Industrial Aerodynamics*, 223, 104933. <https://doi.org/10.1016/j.jweia.2022.104933>
- Wang, C., Wang, J., Wei, C., Zhu, Y., Yin, D., & Li, J. (2023). Vision-based deep reinforcement learning of uav–ugv collaborative landing policy using automatic curriculum. *Drones*, 7(11), 676. <https://doi.org/10.3390/drones7110676>
- Wang, X., Chen, Y., & Zhu, W. (2021). A survey on curriculum learning. *IEEE Transactions on Neural Networks and Learning Systems*, 32(10), 4235–4253. <https://doi.org/10.1109/TNNLS.2021.3079436>
- Williams, H. (2021). Jonswap waves for realistic ocean wave modelling [School of Engineering, University of Warwick]. https://www.researchgate.net/publication/354293847_JONS_WAP_waves_for_realistic_ocean_wave_modelling
- Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4), 229–256. <https://doi.org/10.1007/BF00992696>
- Wu, L., Wang, C., Zhang, P., & Wei, C. (2022). Deep reinforcement learning with corrective feedback for autonomous uav landing on a mobile platform. *Drones*, 6(9), 238. <https://doi.org/10.3390/drones6090238>
- Xie, J., Peng, X., Wang, H., Niu, W., & Zheng, X. (2020). Uav autonomous tracking and landing based on deep reinforcement learning strategy. *Sensors*, 20(19), 5630. <https://doi.org/10.3390/s20195630>
- Xiong, H., Li, L., Zeng, W., Li, D., Liu, Y., Raps, F., Lou, Y., & Noack, B. R. (2025). Trajectory planning for drone landing, incorporating wind-sensing capabilities, operational and safety objectives, and reinforcement learning. *Communications Engineering*, 4, 199. <https://doi.org/10.1038/s44172-025-00531-1>
- Xu, M., Deng, C., Hu, X., Lu, Y., Xue, W., & Zhu, B. (2025). Uav inspection path optimization in offshore wind farms using the option-a*-dqn algorithm. *PLoS ONE*, 20(11), e0336935.

- Yin, Y., Zhang, J., Guo, M., Ning, X., Wang, Y., & Lu, J. (2023). Sensor fusion of gnss and imu data for robust localization via smoothed error state kalman filter. *Sensors*, 23(7), 3676. <https://doi.org/10.3390/s23073676>
- Yıkılmaz, C., & Wagter, C. D. (2025). Reinforcement learning-based optimal guidance for landing the variable skew quad plane on a ship [Paper no. IMAV2025-7]. *16th International Micro Air Vehicle Conference and Competition*, 59–68. <http://www.ima-vs.org/papers/2025/7.pdf>
- Zhao, C., Liu, R. W., Qu, J., & Gao, R. (2023). Deep learning-based object detection in maritime unmanned aerial vehicle imagery: Review and experimental comparisons [Review of UAV object detection and surveillance in maritime environments]. *arXiv preprint arXiv:2311.07955*.
- Zhou, M., Zhou, Z., Liu, L., Huang, J., & Lyu, Z. (2020). Review of vertical take-off and landing fixed-wing uav and its application prospect in precision agriculture. *International Journal of Precision Agricultural Aviation*, 3(4), 8–17. <https://doi.org/10.33440/j.ijpaa.20200304.130>
- Zhou, X., Chen, L., Sun, C., Jia, W., Yi, N., & Sun, W. (2025). Highly accurate attitude estimation of unmanned aerial vehicle payloads using low-cost mems. *Micromachines*, 16(6), 632. <https://doi.org/10.3390/mi16060632>