Benchmarking Off-Policy Reinforcement Learning Controllers for Quadcopter Stabilisation

Wietske van der Westhuizen
vwswie001@myuct.ac.za
Department of Computer Science
University of Cape Town
Cape Town, Western Cape, South Africa

Krupa Prag
krupa.prag@uct.ac.za
Department of Computer Science
University of Cape Town
Cape Town, Western Cape, South Africa

Abstract

Quadcopters require precise control for hover stability; Proportional-Integral-Derivative (PID) controllers struggle to manage non-linear dynamics and external disturbances such as wind turbulence. Reinforcement Learning (RL) offers a model-free alternative suited to high-dimensional, continuous control tasks. This research presents a comparative analysis of three off-policy RL controllers against the PID controller for quadcopter hover stability under wind disturbances, evaluating the solution quality, algorithmic efficiency, and computational feasibility. The controllers - Soft Actor-Critic (SAC), Twin Delayed DDPG (TD3), Twin Average Delayed DDPG (TAD3), and PID - were fine-tuned using a Hyperparameter Optimisation (HPO) pipeline. Each controller was benchmarked across multiple seeds and wind conditions, using flight-specific metrics such as success rate, positional error, and inference latency. Benchmarking showed that although PID was competitive under no-wind conditions, RL controllers substantially outperformed it under disturbance while remaining computationally feasible. SAC was the superior RL controller, while TAD3's mean aggregation hypothesis was inconclusive due to insufficient evidence showing it significantly differed in performance from TD3.

Keywords

Reinforcement Learning, Quadcopter Flight Control, PID Control, Soft Actor-Critic, Twin Delayed Deep Deterministic Policy Gradient, Twin Average Delayed Deep Deterministic Policy Gradient, Sample Efficiency, Disturbance Rejection, Real-Time Control

1 Introduction

This research presents a comparative study of control methods for quadcopter hover stabilisation under wind disturbance, evaluating the PID controller against off-policy reinforcement learning algorithms. This section outlines the overarching motivation, research questions and key technical contributions of the project.

1.1 Motivation

As Unmanned Aerial Vehicles (UAVs) transition from controlled environments into high-stakes autonomous missions, such as urban logistics, search and rescue, and precision agriculture, their operational success hinges on robust low-level control [7]. While advancements in hardware miniaturisation and software sophistication have enabled rapid industry adoption, manoeuvring in unpredictable real-world conditions remains a fundamental challenge. The Proportional-Integral-Derivative (PID) controller is the dominant industrial standard for quadcopter control systems [20]. Their widespread use results from low computational overhead, well-established tuning methodologies, and deterministic stability guarantees [20]. PID systems rely on linear approximations of rigid-body dynamics that suffer from degradation when exposed to non-linear dynamics, such as rotor coupling and turbulent wind gusts [4]. RL offers a robust control alternative that can learn optimal non-linear control strategies through direct interaction with simulation physics [19]. Despite this potential, RL controllers have not replaced PID as the primary controller in commercial autopilot stacks. The lack of standardised benchmarking leads to studies conducted with incompatible simulation environments and non-flight-specific evaluation metrics, complicating comparisons [5, 10, 18]. Fair evaluation of PID vs RL controllers requires a controlled, transparent comparison under identical conditions: a shared physics engine, shared reward functions and training curriculum, matched hyperparameter optimisation (HPO) budgets, and consistent evaluation metrics.

1.2 Research Questions

To address these benchmarking gaps, this study investigates the following primary research questions:

(1) **RQ1 (Solution Quality):** Under a matched HPO tuning budget and identical simulation dynamics, do off-policy RL algorithms (SAC, TD3, TAD3) achieve higher hover success rates with lower root-mean-square position error (RMSE) and tighter attitude stability than the PID controller during hover under both nominal and composite-wind conditions?
(2) **RQ2 (Algorithmic Efficiency):** Does replacing the minimum target critic formulation in TD3 with an average twin-critic mechanism to create TAD3 reduce over-conservative value estimations as proposed by Tiong et al. [27], and how does this impact convergence reliability and disturbance rejection in quadcopter control tasks?
(3) **RQ3 (Computational Feasibility):** Can off-policy deep neural network controllers maintain inference latencies strictly below a real-time onboard feasibility limit ($\leq 4000\,\mu$s) required for high-frequency (250–1000 Hz) control loops [23]?

1.3 Contributions

The specific technical contributions of this work include:

- **Custom TAD3 Implementation:** A custom Stable-Baselines-Jax controller override implementing Twin Average Delayed Deep Deterministic Policy Gradient (TAD3) to mitigate value underestimation in continuous aerial control [27].

- **Multi-Stage HPO and Evaluation Protocol:** Implementation of an automated benchmarking framework combining Optuna-driven hyperparameter optimisation with a standardised simulation environment to ensure fair cross-algorithm comparison.
- **Rigorous Benchmark Comparison:** A head-to-head evaluation of PID against off-policy RL algorithms (SAC, TD3, TAD3) on hover success, positional RMSE, and attitude stability.
- **Disturbance Robustness and Latency Profiling:** Comprehensive stress-testing across composite-wind disturbance profiles, alongside real-time inference latency benchmarking for onboard flight loop feasibility.

2 Background and Related Work

This section outlines the fundamental knowledge and background required for the research conducted, including related literature that informed the experimentation.

2.1 Classical Control for Quadcopter Stabilisation

Quadcopters are a subset of unmanned aerial vehicles (UAVs), commonly referred to as drones, that contain 4 spinning rotors to create lift and are popularised for their ability to hover [26]. These drones are underactuated systems with six degrees of freedom and four actuators, which imposes control constraints and complicates trajectory tracking and stabilisation [8].

Control system architectures must compensate for these challenges. Strong rotor coupling dynamics between subsystems cause axes to lack independent movement, with simple input changes having a cascading effect on the drone's orientations [8]. Dynamic factors such as wind, motor noise, and varying payload continuously shift the quadcopter's equilibrium [1], introducing non-linear dynamics. Lastly, quadcopters are open-loop unstable and require constant active control to counteract divergence from the desired target. Any computational error will grow over time, resulting in compounding drift [8].

2.1.1 Proportional Integral Derivative Controller: PID controller is widely applied to classical control problems such as regulating temperature, speed, and flow. PID is cost-effective and reliable with a small hardware footprint and negligible computational overhead, meeting the real-time demands of high-frequency control loops [16]. Furthermore, the easily interpretable gain parameters make manual tuning and fault diagnosis straightforward [24]. PID controllers are designed for linear systems; thus, they are robust at hover because non-linear terms are small, as the drone is in a fixed position with small attitude changes, producing a stable and accurate control model [16].

The linear nature of PID has its limitations. Gain parameters are preset and optimised for ideal circumstances, causing degraded performance and instability during unpredictable conditions. External disturbances such as wind, centre of mass shifts or aggressive manoeuvres push the quadcopter into non-linear behaviour [16]. Furthermore, PID implements roll, pitch, yaw and altitude as decoupled subsystems. Quadcopter dynamics are inherently coupled; a corrective roll input produces simultaneous effects on pitch and translational motion [8]. This decoupling approximation introduces systematic error that PID's integral term can partially compensate for near equilibrium but cannot resolve under dynamic conditions [28].

2.2 Off-Policy Reinforcement Learning for Continuous Control

Reinforcement Learning's primary principle is maximising an agent's cumulative reward through interaction with the environment, learning from the consequences of autonomous decisions [12, 25]. This is expressed through a Markov Decision Process (MDP) where an agent in state s_t takes actions a_t to transition to state s_{t+1} and receive reward r_t [2].

The action value function $Q^\pi(s, a)$ estimates the expected reward of being in state s, taking action a under policy π [12], and is optimised through the Bellman equation (2.1),

$$Q^\pi(s, a) = \mathbb{E}\left[r + \gamma Q^\pi(s', a')\right]. \tag{2.1}$$

Training requires balancing exploration, discovering optimal policies that are unknown to the agent and exploitation, prioritising the acquired knowledge to maximise the immediate reward, with hyperparameters controlling this trade-off [21].

An off-policy algorithm's defining characteristic is its sample efficiency, due to allowing the agent to learn from previous experiences instead of discarding them after policy updates [6]. Past experiences are stored in a replay buffer, which is a fixed-capacity store of past transition tuples (s, a, r, s') to reuse across multiple policy updates [3, 17]. Buffer capacity is a critical hyperparameter as small buffers discard experiences prematurely, while large buffers dilute recent experiences with outdated ones [9].

Actor-Critic algorithms address the continuous action requirements of quadcopter control, as rotors need smooth input functions as opposed to discrete choices [2]. Actuator inputs are sensitive, with small differences across motors producing significant attitude deviations, highlighting the need for fine-grained continuous inputs [8].

2.3 Reinforcement Learning vs Classical Control Benchmarking Gaps

Despite extensive literature applying deep RL to UAVs, existing cross-study comparisons remain inconsistent. As highlighted in Table 1, prior literature exhibits three primary benchmarking gaps:

Firstly, studies fail to disclose systematic tuning procedures for both RL and PID controllers, making results difficult to reproduce [11, 18].

Second, studies evaluate algorithms across different physics platforms, such as MuJoCo [10, 18], Gazebo [11], and custom simulators [5], obscuring whether performance gains stem from algorithmic superiority or simulator artefacts.

Third, studies primarily focus on theoretical metrics like cumulative reward curves and convergence speed [10, 18] rather than domain-specific metrics like hovering stability, settling time, and steady-state error [5]. This allows RL agents to discover exploit strategies (such as "floor-camping" or sacrifice manoeuvres) that yield high mathematical rewards while failing physical flight tasks.

The conducted research resolved these gaps by enforcing a frozen environment, a strictly bounded reward, and equal HPO compute budgets across all controllers.

Table 1: Summary of continuous control and quadcopter benchmarking literature.

Study	Algorithms Evaluated	Environment Framework	Classical Baseline	Domain Flight Metrics	Primary Benchmarking Gap
Bernini et al. (2021)[5]	Custom RL attitude controllers	Custom quadcopter simulator	Partial	Yes	Lacks standardised off-policy comparison across shared PID baselines.
Hsiao et al. (2019)[14]	PPO, A2C, PID	OpenAI Gym quadcopter	Yes	Partial	Narrow algorithm coverage; unstandardised tuning procedures.
Fujimoto et al. (2018)[10]	DDPG, TD3, PPO, SAC	Standard MuJoCo locomotion	No	No	Evaluates generic continuous tasks; lacks UAV dynamics and classical control baselines.
Liu (2024)[18]	DDPG, TD3, SAC, PPO	Continuous control benchmarks	No	No	Focuses on theoretical RL metrics without quadcopter disturbance dynamics.
Proposed Study	**PID, SAC, TD3, TAD3**	**MuJoCoWarp + Gymnasium**	**Yes**	**Yes**	**Provides shared environment, reward, HPO budget, and flight metric benchmarking.**

3 Methodology

This section outlines the design methodologies used in this research. It covers the simulation environment architecture, experimental setup, theoretical background, physical implementation of the controllers, recorded metrics, and statistical analysis.

3.1 Simulation Environment

The simulation and physics environments integrated a 500 Hz instance of MuJoCoWarp with Gymnasium's vector environment API to model a Bitcraze Crazyflie 2.0 micro-quadcopter. The evaluated controllers ran at 50 Hz by importing the RL algorithms from Stable-Baselines-JAX (SBX).

The agent interacts with the environment through a continuous 17-dimensional observation space $\mathbf{s}_t \in \mathbb{R}^{17}$, standardised via a running mean/standard deviation and clipped at $[-10, 10]$, outputting a 4-dimensional normalised action space $\mathbf{a}_t \in [-1, 1]^4$ mapped to each actuator visible in Table 2. This standardisation is applied jointly across all dimensions, including the binary in-tolerance flag, rescaling it away from its native $\{0, 1\}$ representation identically to the continuous physical channels.

Each episode lasted up to 1000 control time steps, equivalent to 20 s of flight time. The quadcopter's vertical spawn height was fixed at floor level, $z = 0.05$ m (matching the Crazyflie physical body radius). Horizontal spawn positions (x, y) were randomised and sampled uniformly within ±0.1 m of the origin $(0, 0)$. Roll and pitch were perturbed within ±1°; yaw was uniformly sampled across $[-180°, 180°]$ to be independent of wind direction. The initial velocity was initialised with Gaussian noise ($\sigma = 0.01$).

Strict constraints were introduced to terminate invalid states and increase the training efficiency:

(1) **Ground Collision:** Altitude $z <= 0.0$ m.
(2) **Spherical Boundary Limit:** Distance from target $\|\mathbf{p}_{\text{target}} - \mathbf{p}\| > 2.0$ m.
(3) **Attitude Safety Cap:** Roll $|\phi| > 45°$ or pitch $|\theta| > 45°$.
(4) **Take-off Deadline:** Quadcopter must reach $z > 0.3m$ within $2s$.

Table 2: 17D Observation Space

Dimensions	State Variable / Description
1–3	Position error relative to target, $\mathbf{e}_p \in \mathbb{R}^3$
4–6	Linear velocity, $\mathbf{v} = [v_x, v_y, v_z] \in \mathbb{R}^3$
7–9	Euler orientation angles and yaw error, $[\phi, \theta, e_\psi] \in \mathbb{R}^3$
10–12	Body angular velocity, $\boldsymbol{\omega} = [p, q, r] \in \mathbb{R}^3$
13–16	Previous normalised motor action, $\mathbf{a}_{t-1} \in \mathbb{R}^4$
17	In-tolerance binary flag, $[0, 1] \in \mathbb{R}^1$ (Boolean, indicating if the drone is in the tolerance zone)

3.2 Reward Function and Curriculum

The reward function is a single-phase, bounded multiplicative function $r_t \in (0, 1]$. Proximity to the target is the main task reward, and undesirable actions are penalised, as expressed by,

$$reward = proximity \cdot \exp(-behavioural_penalty).$$

with the full reward function shown in equation (3.1),

$$r_t = F(e_t) \cdot \exp\left(-\lambda_u F(e_t)^2 \left\| \frac{\Delta \mathbf{a}_t}{\Delta u_{\text{ref}}} \right\|^2 - \lambda_\omega \left(\frac{\omega_z}{\omega_{\text{ref}}} \right)^2 \right). \tag{3.1}$$

Proximity $F(e_t)$ maps distance error $e_t = \|\mathbf{p}_{\text{target}} - \mathbf{p}\|$ to $(0, 1]$ via a hyperbolic formulation to maintain clear reward gradient signals across the full 1.45 m ($z_{target} - z_{spawn}$) ascent profile given by equation (3.2),

$$F(e_t) = \frac{\text{tol}}{\text{tol} + e_t}. \tag{3.2}$$

The tolerance zone (tol = 0.30 m) represents the distance to the target where the quadcopter receives half of the tracking efficiency reward ($F(\text{tol}) = 0.5$). The behavioural penalty is the sum of smoothness and yaw cost. Smoothness cost is expressed by equation (3.3),

$$smoothness_cost = \lambda_u F(e_t)^2 \left\| \frac{\Delta \mathbf{a}_t}{\Delta u_{\text{ref}}} \right\|^2 \tag{3.3}$$

and penalises rapid-changing control outputs exceeding a reference value, gated by proximity. This allows for aggressive actions during takeoff but penalises those commands more the closer the quadcopter is to the target. The yaw cost (equation, (3.4)) penalises unnecessary spinning about the vertical axis relative to the yaw rate reference scale $\omega_{\text{ref}} = 2.0$ rad/s, allowing the necessary roll/pitch translation tilt around the horizontal axis.

$$yaw_cost = \lambda_\omega \left(\frac{\omega_z}{\omega_{\text{ref}}} \right)^2. \tag{3.4}$$

Previous iterations used additive terms but proved prone to exploitation, such as intentional crashes or failure to take off. Because $r_t > 0$ strictly for all valid steps, early crashes forfeit future positive rewards by construction, eliminating the need for termination penalties that were heavily exploited.

Training followed a curriculum where 30% of episode spawns were drawn from full evaluation distributions and 70% used a progressive scalar $d \in [0, 1]$ to control environmental difficulty. Earlier versions utilised adaptive feedback, but it was disabled in favour of a deterministic ramp schedule to ensure reproducibility, as given by equation (3.5),

$$d(t) = \min\left(1.0,\ \frac{t}{0.85 \cdot T_{\text{budget}}}\right). \tag{3.5}$$

Difficulty scaled linearly across 85% of the total training step budget T_{budget}. Initially the drone spawned in the tolerance zone with spawn positions moving increasingly further away as the difficulty increased. Target approach speeds also scaled, reaching up to 1.0 m/s. At $d = 1.0$, wind turbulence vectors apply random forces up to a 1.2× multiplier relative to benchmark conditions.

3.3 Proportional Integral Derivative Controller

The PID controller uses one feedback loop to keep a process variable at the desired set point, calculating the error value $e(t)$ at each control step. The correction is applied through three mathematical terms as seen in equation (3.6). The proportional term reacts to the current error, the integral term accumulates past errors, and the derivative term computes the change in error to predict future errors, where K_p, K_i and K_d are tunable gains [20].

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt}. \tag{3.6}$$

3.4 Soft Actor Critic

SAC is an off-policy actor-critic algorithm that learns a stochastic policy, augmented by an entropy maximisation objective [13], encouraging exploration without noise injection [15] and can be represented by equation (3.7),

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \rho_\pi} \left[r(\mathbf{s}_t, \mathbf{a}_t) + \alpha \mathcal{H}(\pi(\cdot|\mathbf{s}_t)) \right]. \tag{3.7}$$

The entropy weight hyperparameter (α) controls the trade-off between exploration and exploitation [13]. By incorporating entropy maximisation, SAC prevents premature policy collapse and encourages broad state-space exploration, which is critical for discovering stable flight boundaries by smoothing the value landscape [15].

However, the entropy term is a training-time regularisation mechanism, shaping smoother value landscapes and encouraging exploration during learning, rather than prescribing how the policy should behave once deployed. Accordingly, all evaluation and benchmarking results (§ 4) use the deterministic mean action $\mathbf{a}_t = \mu_\theta(\mathbf{s}_t)$ rather than a sampled action, isolating learned policy quality from sampling noise and placing SAC on equal footing with the deterministic TD3/TAD3 baselines.

3.5 Twin Delayed DDPG

TD3 adapts Deep Deterministic Policy Gradient (DDPG) to address systematic overestimates of state-action values (Q-values) caused by target policy exploitation [10, 27]. TD3 mitigates this using three key mechanisms: (1) *Clipped Double Q-learning*, which computes target values using the minimum of two independent critic networks [17], as given by equation (3.8),

$$y = r + \gamma \min_{j=1,2} Q_{\theta'_j}(\mathbf{s}', \tilde{\mathbf{a}}). \tag{3.8}$$

(2) *Delayed Policy Updates*, where the actor network is updated less frequently than the critic networks ($d = 2$)[17]; and (3) *Target Policy Smoothing*, which adds clipped noise to target actions to prevent exploiting fit peaks in Q-value approximations ($\tilde{\mathbf{a}} \sim \text{clip}(\mathcal{N}(0, \sigma^2), -c, c)$) [10].

3.6 Twin Average Delayed DDPG

TD3's Clipped Double Q-learning effectively prevents Q-value overestimation, using the strict minimum ($\min(Q_1, Q_2)$) introduces a secondary flaw: Q-value *underestimation bias*[27]. In quadcopter attitude control, underestimating Q-values could cause the agent to undervalue corrective manoeuvres. The control system fails to counter the external forces on the drone, diverging further from the desired target. This will lead to compounding errors, increasing the difficulty of recovering to a stable state.

TAD3 refines TD3's update rule by replacing the minimum operator with an average of the twin critic's Q-value estimates, reducing the systematic underestimation bias whilst retaining TD3's delayed updates and target policy smoothing [27]. The proposed algorithm can be expressed as equation (3.9),

$$y_{\text{TAD3}} = r + \gamma \left(\frac{Q_{\theta'_1}(\mathbf{s}', \tilde{\mathbf{a}}) + Q_{\theta'_2}(\mathbf{s}', \tilde{\mathbf{a}})}{2} \right). \tag{3.9}$$

Evaluating TAD3 within a quadcopter control framework tests whether mitigating value pessimism translates directly into superior disturbance rejection and target tracking performance. TAD3 was implemented by subclassing the stock SBX `TD3` class and overriding the target Q-value computation method.

3.7 Evaluation Conditions and Seed Pools

The experiment used strict seed-isolation protocols to prevent data leakage across experimental stages, with distinct seed pools available in Appendix B. Furthermore, to assess policy robustness under realistic environmental conditions, evaluations were conducted across multiple dynamic wind turbulence profiles.

3.7.1 Wind Profiles. Two wind profiles were used for evaluation and excluded from training.

- **No Disturbance:** This wind profile served as the baseline, with a force magnitude fraction of 0, establishing a best-case performance.
- **Composite:** An accurate profile of real wind disturbances that aggregates constant force with periodic gusts of varying force and duration. The purpose is to test steady-state rejection of the constant force paired with adaptation and recovery to the gusts. The force envelope stretches from $[0.15, 0.60]$ of hover thrust, requiring a maximum of 88% available thrust to counter, with $3 - 4$ gusts per episode derived from the force specification in Table 3.

Parameter	Value and Description
Constant Component	0.20 of hover thrust
Gust Magnitude	uniform $[0.15, 0.40]$ of hover thrust per gust
Gust Duration	uniform $[0.5, 2.0]$ seconds
Gust Interval	uniform $[3.0, 5.0]$ seconds from gust end to next onset
Gust Start	$t = 5$ s after climb from floor
Direction	Randomised horizontal axis drawn at episode start and shared by constant wind and all gusts

Table 3: Composite Wind Model Parameters

3.8 Hyperparameter Optimisation

The Hyperparameter Optimisation (HPO) process took place over three sub-stages: calibration, search and validation. This multi-stage approach removes tuning bias and seed-dependent performance outliers, ensuring fair and standardised tuning. A full table of setup values can be found in Appendix C.

3.8.1 Calibration. This stage served as a specification probe to statistically determine the best experimental setup values for the subset of tested algorithms. Quasi-random configurations were sampled for each of the three algorithms (SAC, TD3, TAD3), trained and evaluated at predetermined checkpoints. The results were used to calculate Spearman's Rank Correlation between configuration rankings at intermediate checkpoints and the final checkpoint to compute four decisions: The objective metric for configuration selection, trial budget, feasibility of pruning bad configurations mid-trial, and the number of evaluation episodes per seed-condition pair.

3.8.2 HPO Search. The 4 controllers (SAC, TD3, TAD3, PID) were tuned under identical conditions to eliminate tuning discrepancies. Optuna's Tree-structured Parzen Estimator (TPE) was used as a Bayesian Search to find optimal configurations efficiently. The distinct search spaces are available in Appendix A.

Following the Optuna search, the top five configurations per algorithm were selected as follows: To improve fairness and penalise stochastic variance, conservative scores were calculated for each trial by subtracting the standard error from performance. The top 20 trials were normalised into the hyperparameter space and clustered into five groups using exact k-medoids enumeration. Finally, selecting the top-performing conservative trial from each cluster yielded a shortlist for the validation stage that effectively mitigates optimisation bias and noisy outliers.

3.8.3 Validation. The validation stage trained and evaluated the top 5 configurations for each algorithm identified in the Optuna search over a longer time period using new seeds. This process separates the seed-dependent configuration from truly robust candidates. The final configuration per algorithm that was used in benchmarking was determined through a four-tiered rule applied in order:

- Highest success rate under the composite wind profile.
- The lowest mean positional error over all evaluation episodes.
- The lowest interquartile range of seed-level RMSE, to favour configurations that perform reliably across seeds rather than volatile performers.
- The configuration closest to the library defaults acts as the final tie-breaker between indistinguishable performances.

3.9 Final Benchmark Protocol

Each controller's top configuration was trained on 10 fresh seeds to produce 10 models per algorithm, providing a basis for statistical analysis. Training ran for $1.5M$ steps, with 11 evaluation checkpoints benchmarking 3 episodes per seed-condition pair to produce a sample-efficiency curve. The head-to-head comparison of the controllers was computed at the final training checkpoint as outlined in §3.11.

3.10 Logged Evaluation Metrics

Every episode was evaluated based on three key categories, focusing on flight-specific metrics rather than theoretical RL metrics.

3.10.1 Solution Quality.

- Root Mean Squared Error: The mean error distance between the quadcopter and the target point.
- Control Smoothness: Variance of per-step motor command changes, measuring the smoothness of control changes, interpreted as a relative, ordinal number with lower values being more desirable.
- Attitude Oscillation: Peak-to-peak swing in roll and pitch over the final 10 seconds of flight, in degrees.

3.10.2 Convergence Reliability.

- Completion Rate: Indicates if the quadcopter triggered an early termination condition.

- Success Rate: Indicates if the quadcopter successfully took off and maintained a stable hover at the target point. All successful episodes are subsequently completed episodes.
- Time to Arrival: How long the quadcopter takes to reach the target from the spawn point.

3.10.3 Computational Overhead.

- Inference Latency: Average wall-clock time to compute one action (in microseconds).

3.11 Statistical Analysis

The pairwise comparisons between the RL and PID controllers were conducted using a Wilcoxon signed-rank test, which accounts for paired dependencies between seeds and wind profiles. Furthermore, the Signed-rank is non-parametric, requiring no assumption of normality, appropriate for success rates and RMSE distributions that may be skewed.

With the two wind profiles and six pairwise comparisons of the off-policy controllers (SAC vs TD3, SAC vs TAD3, TD3 vs TAD3, RL vs PID baseline), our resulting test matrix had twelve tests. The uncorrected $\alpha = 0.05$ is adjusted to $\alpha_{\text{Holm}} = 0.05/12 \approx 0.00417$ to control the family-wise error rate under the Bonferroni correction.

In addition to p-values, a rank-biserial correlation $r_{\text{rb}} = 1 - \frac{2U}{n_1 n_2}$ is reported alongside each test to quantify effect magnitude independent of sample size, supporting interpretation of whether significant differences are practically meaningful.

3.12 Reproducibility Infrastructure

Each episode environment was initialised with independent spawn and wind setup values, derived from the same master seed through MD5 hashing. The full reproducibility artefacts library is available in the codebase.

4 Results and Discussion

The full evaluation protocol is described in §3.9 and experiment details can be viewed in the Appendix C. Two RMSE metrics were reported: RMSE completed (default metric) was the error recorded over the last 10 seconds of a completed episode. Unconditional RMSE was calculated across all episodes, not gated by the completion flag, measuring degradation under composite conditions. Pairwise comparisons used two-sided Wilcoxon signed-rank tests with Bonferroni correction ($\alpha_{\text{Holm}} = 0.00417$). PID's low completion rate under composite wind made some RL-PID comparisons underpowered ($n = 3$ paired seeds). These results were interpreted directionally rather than statistically significant. Results are grouped by controller flight performance, training dynamics, and deployment feasibility.

4.1 Controller Flight Performance

This section discusses the flight-specific metrics recorded during benchmarking. Table 4 summarises the controller performance, with Table 5 and 7 reporting the pairwise test results. Fig. 1 and 2 visually illustrate the trajectory followed by the individual controllers under both wind conditions, using seed 202, episode 0. This episode was selected because all controllers completed the full length, allowing for fair comparison.

4.1.1 PID vs RL under no disturbance. SAC and PID successfully reached and maintained the target position during all evaluation episodes under no wind disturbance. TD3 recorded a lower episode completion rate of 96.7%; however, it succeeded at all completed episodes. TAD3 had the most early terminations, with a completion rate of 90% and a success rate of 80%. PID definitively beats TAD3, but the gap to TD3 is more ambiguous, with its confidence interval's (±3.3%) upper bound reaching the 100% success rate, allowing no conclusive verdict to be made.

All controllers achieved an RMSE that falls within the $0.3m$ tolerance zone. SAC measured the lowest RMSE at $0.006m$, with a two-sided Wilcoxon signed-rank test p-value = 0.002 calculated across all 10 benchmark seeds and paired with PID's RMSE of $0.023m$. Combined with the Rank-Biserial Correlation (r), also referred to as Effect Size, of $r = -1.00$. There is significant evidence to conclude that SAC consistently outperformed PID on every pair of seeds. TD3's RMSE ($0.016m$) was numerically lower than PID's ($0.023m$), but the paired comparison provides no significant evidence ($p = 0.106 > \alpha_{\text{Holm}}$) that TD3 outperformed PID. The effect size of $r = -0.6$ indicates that TD3 achieved a lower RMSE than PID in 6/10 evaluation episodes, suggesting the difference is due to noise rather than absolute performance. On average, PID beat TAD3's RMSE of $0.188m$ throughout the evaluated episodes. The $p = 0.496$ and $r = +0.29$ suggest weak confidence in the claim, as no reliable signal favours either controller. The RMSE analysis indicated that a subset of the controllers produced little distinction under no wind disturbances, as seen in Fig. 1, with all controllers reaching the tolerance zone comfortably. SAC produced the most effective trajectory, taking a direct path to the target, suggesting strong pathfinding ability.

Three controllers were practically motionless under no wind disturbance: SAC 0.01°, TD3 0.94° and PID producing 1.02° of oscillation. TAD3 was less settled, wobbling peak-to-peak at 7.54°. Due to no wind force being exerted on the quadcopters, it can be concluded that any oscillation was purely control effort to hold position.

The PID controller produced the most erratic control inputs (687.2), visualised by Fig. 1. It was prone to overshoot that shrunk with each pass. This could be attributed to the derivative term acting like a brake proportional to speed, bleeding off a fraction of the momentum each time it passed the target. The RL controllers produced smoother inputs; SAC was the smoothest (2.6), followed by TD3 (14.9) and then a significant jump to TAD3 (219.8). PID ranks second on RMSE but last on control smoothness, indicating different control styles tied to the underlying controller architecture. PID's linear nature caused braking to be applied identically everywhere as a result of the fixed gain structure. The RL controllers suppressed overshoot by learning non-linear policies that adapted braking to the environment.

Although PID's success rate and tracking accuracy under no wind disturbance were close to, and in TAD3's case better than, the RL controllers, this did not extend to the composite wind conditions.

4.1.2 PID vs RL under composite wind. The composite wind profile clearly differentiated the controllers, supporting the claim that PID cannot adapt to non-linear disturbances. PID's episode completion collapsed from 100% to 13.3%, while SAC remained unchanged at

Table 4: Aggregated benchmark performance of controllers evaluated over 3 episodes per seed-condition pair.

Controller	Condition	Success	Completion	RMSE final-10s (m)	Attitude Osc. (°)	Ctrl. Smoothness	Time-to-Arrival (s)
SAC	No-Disturbance	1.000 ± 0.000	1.000 ± 0.000	0.006 ± 0.001	0.01 ± 0.00	2.6 ± 0.3	1.04 (med., n=10/10)
TD3	No-Disturbance	0.967 ± 0.033	0.967 ± 0.033	0.016 ± 0.004	0.94 ± 0.34	14.9 ± 2.6	0.96 (med., n=10/10)
TAD3	No-Disturbance	0.800 ± 0.133	0.900 ± 0.100	0.188 ± 0.163	7.54 ± 3.60	219.8 ± 72.4	1.03 (med., n=8/10)
PID	No-Disturbance	1.000 ± 0.000	1.000 ± 0.000	0.023 ± 0.001	1.02 ± 0.10	687.2 ± 27.0	0.93 (med., n=10/10)
SAC	Composite	0.833 ± 0.056	1.000 ± 0.000	0.132 ± 0.011	29.0 ± 1.5	21.3 ± 1.4	6.49 (med., n=10/10)
TD3	Composite	0.333 ± 0.070	0.767 ± 0.087	0.283 ± 0.040	31.6 ± 1.9	578.8 ± 123.9	14.85 (med., n=10/10)
TAD3	Composite	0.167 ± 0.056	0.733 ± 0.109	0.434 ± 0.137	31.9 ± 4.9	1680.4 ± 321.0	17.07 (med., n=8/10)
PID	Composite	0.000 ± 0.000	0.133 ± 0.074	0.426 ± 0.035	57.3 ± 1.6	483.8 ± 15.0	19.34 (med., n=2/10)

Table 5: Pairwise comparison of RL controllers and PID for RMSE final-10s, paired by seed.

Comparison	Condition	Mean Diff. (m)	95% CI (m)	Effect Size (r)	p	Sig.	Compl. RL / PID	Seeds Dropped‡
SAC vs. PID	No-Disturbance	−0.018	[−0.020, −0.016]	−1.00	0.0020	Y	1.000 / 1.000	—
TD3 vs. PID	No-Disturbance	−0.008	[−0.014, 0.000]	−0.60	0.1055	N	0.967 / 1.000	—
TAD3 vs. PID	No-Disturbance	+0.165	[−0.003, 0.493]	+0.29	0.4961	N	0.900 / 1.000	1
SAC vs. PID	Composite	−0.324	[−0.378, −0.268]	−1.00	0.2500	N†	1.000 / 0.133	7
TD3 vs. PID	Composite	−0.206	[−0.250, −0.124]	−1.00	0.2500	N†	0.767 / 0.133	7
TAD3 vs. PID	Composite	−0.146	[−0.224, −0.098]	−1.00	0.2500	N†	0.733 / 0.133	7

† Underpowered, not merely non-significant at $n = 3$ paired seeds. ‡"Seeds Dropped" is the number of benchmark seeds present for one side but not paired into this comparison (a seed where one controller produced no completed episode).

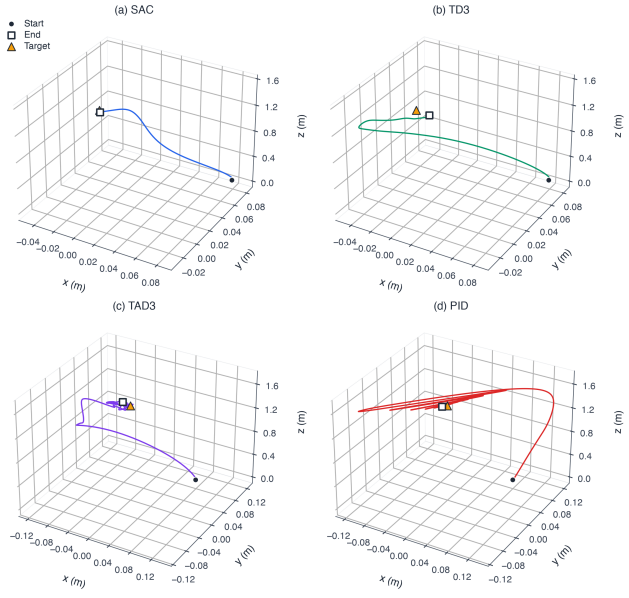


Figure 1: Flown trajectory of (a) SAC, (b) TD3, (c) TAD3 and (d) PID from start to fixed target hover point $(0, 0, 1.5)m$ (no disturbance, seed 202, episode 0). Panels (a)–(b) and (c)–(d) are drawn to different scales to make trajectories visible.

100%, TD3 dropped to 76.7%, and TAD3 to 73.3%. The success rate followed a similar pattern, with none of the PID episodes succeeding. SAC experienced the smallest decrease, from 100% to 83.3%, while TD3 and TAD3 showed larger changes, from 96.7% to 33.3% and 80% to 16.7%, respectively.

The paired comparison between PID and RL was underpowered, not non-significant. PID's low completion rate left $n = 3$ paired seeds for each test, with a two-sided p-value of $2^{1-n} = 0.25$, unreachable at α_{Holm} regardless of effect size. The mean difference recorded in Table 5 indicates a one-directional relationship between PID and the RL controllers (SAC: $-0.324m$, TD3: $-0.206m$, TAD3: $-0.146m$). These results reflect the completion and success gap, not the p-values.

Table 6: RMSE final-10s and Unconditional RMSE per controller and condition

Controller	Condition	Completion	Unconditional RMSE (all episodes, m)	RMSE final-10s (completed, m)
SAC	No-Disturbance	1.000	0.006	0.006
TD3	No-Disturbance	0.967	0.042	0.016
TAD3	No-Disturbance	0.900	0.279	0.188
PID	No-Disturbance	1.000	0.023	0.023
SAC	Composite	1.000	0.132	0.132
TD3	Composite	0.767	0.348	0.283
TAD3	Composite	0.733	0.551	0.434
PID	Composite	0.133	0.770	0.426

Unconditional RMSE, computed over all episodes, widened the reported gap. PID averaged $0.770m$ against SAC's $0.132m$, TD3's $0.348m$, and TAD3's $0.551m$. The completion-conditioned RMSE for PID averaged $0.426m$ over its 3 episodes, closer to TD3 and TAD3's completed means ($0.283m$, $0.434m$). Unconditional RMSE proved a more suitable indicator of performance under wind, as completion gating improved the RMSE of crash-dominant controllers.

Attitude oscillation increased across all controllers, influenced by the larger counter-control requirements of the composite wind. PID experienced the largest roll/pitch swing at 57.3° compared to SAC (21.3°), TD3 (31.6°) and TAD3 (31.9°). The change mirrored the increase in episode terminations, with PID exceeding the maximum roll/pitch angles more than the other pairwise comparisons.

Control smoothness did not adhere to the same pattern. PID's (483.3) control inputs were smoother than TD3 (578.8) and TAD3 (1680.4), but far from SAC (21.3). Interpreting control smoothness with completion rate indicates that PID, TAD3 and TD3 did not handle the disturbance smoothly. Under composite wind, the failure mode was losing the episode outright rather than fighting the noisy standstill. The smoothness values represent the rare episodes that did not crash.

The trajectories in Fig. 2 provided insight into the counter control actions. SAC and TD3 reached a position on the edge of the tolerance zone. Still, thrust was concentrated on maintaining position and recovering from larger gusts in the negative XY-plane that pushed them away from the target. Both controllers showed concentrated positional oscillations to the left of the target, with larger drifts corresponding to bigger gusts. TAD3 produced similar trajectory patterns to the other RL controllers, but the positional oscillations were more severe, suggesting it struggled to counter and recover from the disturbances. PID's trajectory recreated the overshoot trend, with large elliptical oscillations across the geometric bounds of the environment, rather than controlled tracking. The corrective thrust consistently trailed the disturbance instead of countering the forces due to severe phase lag. Similarly, the same phase-lag dynamic catalysed the high rate of episode terminations due to exceeding the environmental bounds during large positional swings.

PID reached the tolerance zone during 2 paired seeds with a mean time of 19.34s. SAC (6.49) and TD3 (14.85) reached it in all 10 paired seeds, and TAD3 (17.07) in 8 pairs. The increase in time to arrival suggests that wind disturbances severely affect navigational abilities, while validating SAC's robust capabilities.

4.1.3 RL Algorithm Comparison: SAC vs. TD3 vs. TAD3. The pairwise comparison between the three RL controllers in Table 7 cleanly separates the most robust controller. SAC significantly outperformed TAD3 in both evaluation conditions with a $p = 0.0039$ and effect size of $r = -1$, maintaining a mean position $0.305m$ closer to the target under wind disturbance. Furthermore, SAC outperformed TD3 under composite conditions ($p = 0.0039, r = -0.96$). The pairwise comparison of SAC and TD3 under no disturbance is not statistically significant. Although SAC has a smaller mean error difference of $0.01m$, the p-value of 0.0137 exceeds α_{Holm}.

TD3 and TAD3's results were not significantly different in either condition ($p = 0.1641$ no-disturbance, $p = 0.7344$ composite), leading to inconclusive statistical results on whether the mean aggregation hypothesis is true based on RMSE. Comparing success rate, control smoothness and flight path trajectories provided a more distinctive insight into the two controllers. TD3 had ±2× the success rate and ±3× the control smoothness of TAD3 under composite conditions while visually following a more accurate flight trajectory. TAD3's mean aggregation is less conservative than TD3's sceptical minimum selection of the twin critic's value

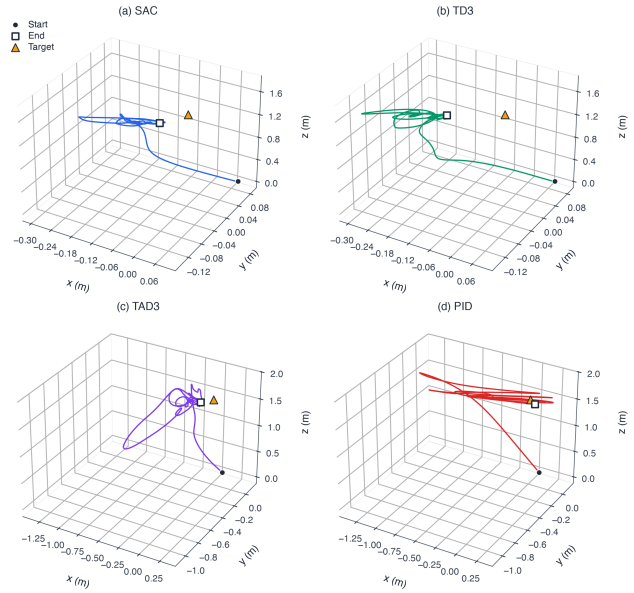


Figure 2: Flown trajectory of (a) SAC, (b) TD3, (c) TAD3 and (d) PID from start to fixed target hover point $(0, 0, 1.5)m$ (composite wind, seed 202, episode 0). Panels (a)–(b) and (c)–(d) are drawn to different scales to make trajectories visible.

estimation. Practically, TD3 was consistently biased against more aggressive flight manoeuvres, allowing longer training episodes to learn ideal policies because it experienced fewer early terminations. Conversely, TAD3's actor allowed less-verified actions over the course of training, leading to more terminations, lower success rates and a worse policy near the target point. It can be concluded that mean aggregation adversely affected the controller's flight performance.

Given all of the presented evidence in §4.1.2 and §4.1.3, it can be concluded that SAC is the superior-performing RL algorithm when evaluated on flight performance. SAC maintained lower RMSE, attitude oscillations, smoother control inputs, and more robust navigational abilities while succeeding in more evaluations under both conditions. These performance gains could be attributed to the fundamental architectural differences between SAC, TD3 and TAD3. SAC produces stochastic policies with outputs sampled from a distribution, smoothing out bad Q-estimates, as validated by the smooth control inputs. TD3/TAD3 use deterministic policy gradient updates that are more vulnerable to critic errors, which can cause action propagation in the wrong direction. Furthermore, the stochastic policy is entropy regularised, with the exploration-exploitation balance adapted over the course of training, allowing it to explore more of the state-action space and penalising narrow action distributions. TD3/TAD3 relies on a fixed exploration-noise hyperparameter, allowing for either too much or too little exploration, leading to narrow action spaces or poor policies.

4.1.4 Ranking Stability Across Conditions. Ranking the controllers by success rate as the primary metric and RMSE as a secondary

Table 7: Each pair of RL controllers, RMSE final-10s, paired by seed. "Seeds Dropped" lists benchmark seeds present for one side but not paired into this comparison (a seed where one controller produced no completed episode).

Comparison	Condition	Mean Diff. (m)	95% CI (m)	Effect Size (r)	p	Sig.	Completion A / B	Seeds Dropped
SAC vs. TD3	No-Disturbance	−0.010	[−0.018, −0.004]	−0.85	0.0137	N	1.000 / 0.967	—
SAC vs. TAD3	No-Disturbance	−0.183	[−0.511, −0.015]	−1.00	0.0039	Y	1.000 / 0.900	1
TD3 vs. TAD3	No-Disturbance	−0.172	[−0.503, 0.002]	−0.56	0.1641	N	0.967 / 0.900	1
SAC vs. TD3	Composite	−0.151	[−0.228, −0.084]	−0.96	0.0039	Y	1.000 / 0.767	—
SAC vs. TAD3	Composite	−0.305	[−0.578, −0.137]	−1.00	0.0039	Y	1.000 / 0.733	1
TD3 vs. TAD3	Composite	−0.137	[−0.448, 0.060]	−0.16	0.7344	N	0.767 / 0.733	1

metric, SAC ranked first in both, followed by TD3 and TAD3 (Table 4). The statistical ordering between TD3 and TAD3 (§4.1.3) was never concluded, but the raw ranking did not shuffle the order. TD3 led TAD3 on success rate under both no disturbance (0.967 vs 0.8) and composite (0.333 vs 0.167). Under no disturbance, PID tied SAC on success rate (1.0 each) and placed second by RMSE, ahead of TD3 and TAD3. Under composite, PID fell to last place by a considerable margin (§4.1.2). This collapse did not affect the relative position of the RL controllers. This is specific to PID and not a ranking instability under disturbance. This validates that PID is a specialist controller that is competitive in linear control tasks with a predictable environment.

4.2 Training Dynamics and Algorithm Behaviour

The following section reports the results of the calibration stage that influenced the hyperparameter tuning and training of the final RL controllers.

4.2.1 Calibration Results. The calibration stage outlined in §3.8.1 identified *Fraction in Tolerance* as the most suitable objective metric for ranking configurations during hyperparameter tuning and validation. The selection was based on the highest mean Spearman correlation between each metric and the total reward for a configuration across all 3 controllers (Fraction in Tolerance (ρ = 0.957), Success Rate (ρ = 0.91), RMSE (ρ = 0.95)).

The 750K HPO search budget in §3.8.2 was decided through a reward plateau analysis (§3.8.1). The plateau determined the minimum number of training steps needed to make informed inferences about a configuration's performance across all algorithms. TD3 plateaued at 400K time steps and SAC at 500K; TAD3 formed a knee early on but plateaued only at 750K time steps visible in Fig. 3.

4.2.2 Sample Efficiency and Training. Training sample efficiency is measured by the number of training steps needed to beat the PID's performance, in terms of RMSE and completion rate. Fig. 4 measures the final 10-s RMSE vs Training Steps for no disturbance and composite wind.

Under no disturbance, TD3 beat PID's RMSE within 400K time steps, with SAC only achieving a lower RMSE past 1.2M time steps, and TAD3 never achieving higher accuracy than PID. Under composite conditions, TD3 first reached PID's RMSE baseline (900K), with SAC following (1.1M). SAC did not produce any completed

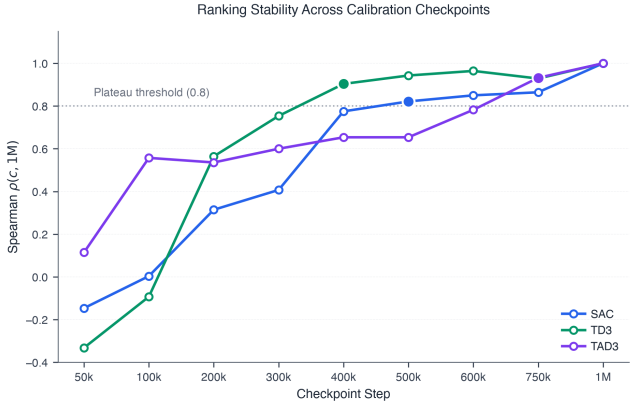


Figure 3: Spearman correlation across calibration checkpoints and final training during calibration, per algorithm.

episodes until 300K time steps into training, after which it improved rapidly. TAD3 showed the worst training efficiency, with an irregular RMSE trend before achieving its best RMSE at 1.5M time steps.

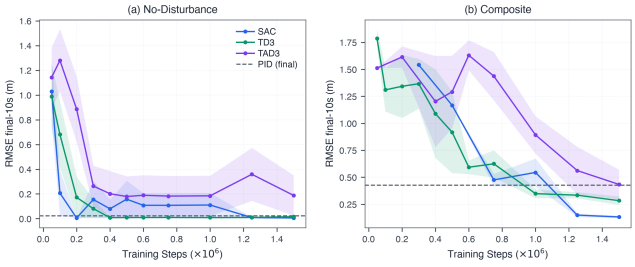


Figure 4: RMSE final-10s vs. training step for SAC, TD3, and TAD3, both conditions, with ±1 SE ribbons across the 10 benchmark seeds.

Fig. 5 models the Completion Rate vs Training Steps for both conditions. Under no disturbance, all controllers become flight-capable before the first checkpoint. SAC and TD3 equalised PID's 100% completion rate within the first 400 − 500K time steps, with TAD3 falling short. TD3 beat PID at the first checkpoint under composite wind, while SAC took longer to cross the threshold but

saw large completion-rate increases thereafter. TAD3 followed a similar pattern to SAC, crossing PID's performance at the same threshold but ending the training with a smaller completion rate.

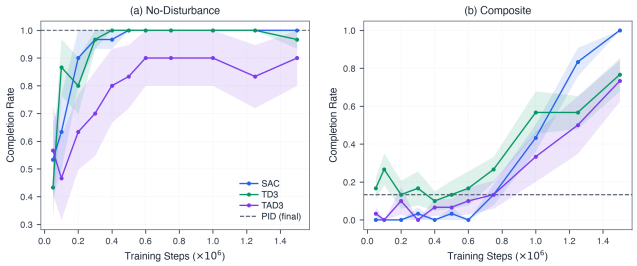


Figure 5: Completion rate vs. training step, same controllers, conditions, and PID reference convention as Fig. 4.

It can be concluded that TD3 is more sample efficient than the other RL controllers and displayed large performance gains on small time-step budgets. SAC took slightly longer to beat the PID baseline, however, it produced better policies.

TD3 and TAD3 have fixed noise decoupled from policy learning, allowing their actor to select the policy the critic determines is best. SAC's exploration is tied to its entropy-regularised stochastic policy, which tries to maintain distributional diversity rather than purely exploiting the critic's reward estimates. The exploration-exploitation trade-off allowed TD3 and TAD3 to convert early critic estimates into usable behaviour but plateaued below SAC's performance ceiling. In contrast, SAC's slower commitment to a narrower policy allowed it to improve past the point where the other controllers settled.

4.3 Deployment Feasibility and Inference Latency

The real-time feasibility budget for control decisions is set at 4,000 μs. PID used 1.1% of the budget with SAC (72%), TD3 (64%) and TAD3 (72%) all clearing the ceiling comfortably (Table 8). PID measured $\approx 60 - 65\times$ faster than the RL controllers due to its additive arithmetic ($O(1)$), validating its low overhead requirements. The RL policies require full neural-network forward passes that involve expensive matrix multiplication through multiple layers to produce outputs ($O(h^2)$, h = hidden layer width). The measured latency is an artefact of a single-sample GPU dispatch, not the algorithm itself, and was not measured on an embedded processor. Additionally, the standard deviations of the RL controllers are large relative to the headroom, posing a real threat to the feasibility measurement.

4.4 Limitations

The presented research was constrained by several limitations. **Single Airframe Specificity:** All controllers were trained on a single quadcopter model; performance may not generalise to other airframe geometries, weight distributions, or motor characteristics. Additionally, all results are simulation-only, with no results obtained from physical flight. **Limited statistical comparisons:** Three of the twelve RMSE comparisons were underpowered due to low completion rates under composite conditions. Increasing the number of benchmark seeds would allow more completed trials and enable statistically significant inference about the pairwise comparisons. Furthermore, with statistical significance testing being isolated to RMSE, it forces other metrics to be reported descriptively and not interpreted as statistically significant. **Episode boundaries limit recoverable flight trajectories:** TAD3 and PID experienced frequent episode termination under composite conditions due to exceeding the geometric boundaries of the simulation environment. Widened boundaries could give the controllers more opportunity to recover position, leading to higher completion rates and increased statistical power during analysis.

Table 8: Mean inference latency per controller, pooled across both conditions and all completed episodes.

Controller	Latency (μs)	n seeds
SAC	2,898 ± 429	10
TD3	2,570 ± 514	10
TAD3	2,861 ± 605	9
PID	44.1 ± 0.3	10

5 Conclusion and Future Work

This research benchmarked PID against three RL controllers (SAC, TD3, TAD3) to determine if they outperform PID at hover stability under wind conditions. Although PID was highly competitive under no disturbance, beating TAD3 and narrowly falling short of SAC and TD3, it struggled with the composite wind, achieving a success rate of 0%. The RL controllers all achieved higher success rates, with SAC leading at 83%, recording the lowest RMSE, attitude oscillation, and smoothest control inputs. There was little statistical confidence to distinguish between TD3 and TAD3's performance, with the results not supporting the mean aggregation hypothesis. As expected, PID produced significantly faster action outputs, validating the controller's lower computational overhead. However, the RL controllers still met the real-time feasibility bounds when measured on the benchmarking machines, indicating that the overhead of neural network passes does not hinder high-frequency decision making. Finally, the research could be improved by increasing the training and evaluation budget across more seeds, conditions, and quadcopter models to generate a more robust ordering of controllers' performance rankings.

5.1 Future Work

This research demonstrated the unique ability of RL controllers to outperform PID when exposed to non-linear disturbances. However, this work has potential continuations. Besides increasing the compute budget and scope of evaluation, cascade controller architecture could be investigated. Having multiple control loops, responsible for one aspect of flight like altitude or attitude, could allow for task specialisation, increasing potential performance.

References

[1] Ghulam E Mustafa Abro, Vijanth Sagayan Asirvadam, Saiful Azrin Bin Mohd Zulkifli, Abdul Sattar, Dileep Kumar, and Ali Anwer. 2020. Effects of unmodelled dynamic factors on an under-actuated quadrotor: A review of hybrid observer design methods. *Measurement and Control* 53, 9-10 (Nov. 2020), 1978–1987. doi:10.1177/0020294020964236

[2] Fadi AlMahamid and Katarina Grolinger. 2021. Reinforcement Learning Algorithms: An Overview and Classification. In *2021 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*. IEEE, ON, Canada, 1–7. doi:10.1109/CCECE53047.2021.9569056

[3] Gavin An and Sai Zhang. 2023. Pruning replay buffer for efficient training of deep reinforcement learning. *Journal of Emerging Investigators* (2023). doi:10.59720/23-068

[4] Ahmad Taher Azar, Anis Koubaa, Nada Ali Mohamed, Habiba A Ibrahim, Zahra Fathy Ibrahim, Muhammad Kazim, Bilel Benjdira, Adel Ammar, Ibrahim A Hameed, and Gabriella Casalino. [n. d.]. Drone Deep Reinforcement Learning: A Review. ([n. d.]).

[5] Nicola Bernini, Mikhail Bessa, Rémi Delmas, Arthur Gold, Eric Goubault, Romain Pennec, Sylvie Putot, and François Sillion. 2021. A few lessons learned in reinforcement learning for quadcopter attitude control. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*. ACM, Nashville Tennessee, 1–11. doi:10.1145/3447928.3456707

[6] Tianyue Cao. 2020. Study of sample efficiency improvements for reinforcement learning algorithms. In *2020 IEEE Integrated STEM Education Conference (ISEC)*. IEEE, Princeton, NJ, USA, 1–1. doi:10.1109/ISEC49744.2020.9397834

[7] Nourhan Elmeseiry, Nancy Alshaer, and Tawfik Ismail. 2021. A Detailed Survey and Future Directions of Unmanned Aerial Vehicles (UAVs) with Potential Applications. *Aerospace* 8, 12 (Nov. 2021), 363. doi:10.3390/aerospace8120363

[8] Bara J. Emran and Homayoun Najjaran. 2018. A review of quadrotor: An underactuated mechanical system. *Annual Reviews in Control* 46 (2018), 165–180. doi:10.1016/j.arcontrol.2018.10.009

[9] Hakan Eren, Nihat Adar, and Ahmet Yazar. 2023. The Importance of Experience Replay Buffer Size in Deep Reinforcement Learning. https://www.researchgate.net/publication/379190323_The_Importance_of_Experience_Replay_Buffer_Size_in_Deep_Reinforcement_Learning

[10] Scott Fujimoto, Herke van Hoof, and David Meger. 2018. Addressing Function Approximation Error in Actor-Critic Methods. doi:10.48550/ARXIV.1802.09477

[11] Shaswat Garg. 2025. *Reinforcement Learning Based Motion Planner and Trajectory Tracker for Unmanned Aerial Systems*. Master's thesis. University of Waterloo. https://hdl.handle.net/10012/21622

[12] Majid Ghasemi and Dariush Ebrahimi. 2024. Introduction to Reinforcement Learning. doi:10.48550/arXiv.2408.07712 arXiv:2408.07712.

[13] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 1861–1870. https://proceedings.mlr.press/v80/haarnoja18b.html

[14] Fang-I Hsiao, Cheng-Min Chiang, and Alvin Hou. 2019. Reinforcement Learning Based Quadcopter Controller. (2019).

[15] Ayush Vardhan Jha, Sai Rushendra Reddy Bommu, V.S. Shrishma Rao, and Sn Muralikrishna. 2024. Soft Actor-Critic Algorithm in High-Dimensional Continuous Control Tasks. In *2024 International Conference on Modeling, Simulation & Intelligent Computing (MoSICom)*. IEEE, Dubai, United Arab Emirates, 535–540. doi:10.1109/MoSICom63082.2024.10881478

[16] Parsa Yeganeh Khorasani. 2025. Using a proportional integral derivative (PID) system to control the quadrotor drone to ensure its reliability and robustness. doi:10.21203/rs.3.rs-6448814/v1

[17] Seung-Hyun Kong, I. Made Aswin Nahrendra, and Dong-Hee Paek. 2021. Enhanced Off-Policy Reinforcement Learning With Focused Experience Replay. *IEEE Access* 9 (2021), 93152–93164. doi:10.1109/ACCESS.2021.3085142

[18] Shijie Liu. 2024. An Evaluation of DDPG, TD3, SAC, and PPO: Deep Reinforcement Learning Algorithms for Controlling Continuous System. In *Proceedings of the 2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023)*, Badrul Hisham Ahmad (Ed.). Vol. 180. Atlantis Press International BV, Dordrecht, 15–24. doi:10.2991/978-94-6463-370-2_3

[19] Youssef Mahran, Zeyad Gamal, and Ayman El-Badawy. 2024. Reinforcement Learning Position Control of a Quadrotor Using Soft Actor-Critic (SAC). In *2024 6th Novel Intelligent and Leading Emerging Sciences Conference (NILES)*. IEEE, Giza, Egypt, 72–75. doi:10.1109/NILES63360.2024.10753187

[20] Abhishek Manalathody, Kishoor Santhana Krishnan, Jisnu Adindtra Subramanian, Sakthivel Thangavel, Jinraj V. Pushpangathan, and Haripraśanth Paranjothy. 2025. Nonlinear and Linear PID Controllers-Based Hybrid Flight Control Strategy for a Quadcopter With Slung Load. *IEEE Access* 13 (2025), 48520–48542. doi:10.1109/ACCESS.2025.3551622

[21] Mohammadamin Moradi, Zheng-Meng Zhai, Shirin Panahi, and Ying-Cheng Lai. 2024. Adaptive network approach to exploration–exploitation trade-off in reinforcement learning. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 34, 12 (Dec. 2024), 123120. doi:10.1063/5.0221833

[22] Republic of South Africa. 2013. Protection of Personal Information Act of 2013. Government Gazette No. 37067. Act of 2013.

[23] Ebad Sayed. 2026. Discrete PID in Flight Controllers: What Actually Runs at 1kHz. https://medium.com/@sayedebad.777/discrete-pid-in-flight-controllers-what-actually-runs-at-1-khz-a6efa867d189

[24] Sigurd Skogestad. 2003. Simple analytic rules for model reduction and PID controller tuning. *Journal of Process Control* 13, 4 (June 2003), 291–309. doi:10.1016/S0959-1524(02)00062-8

[25] Richard S. Sutton and Andrew Barto. 2020. *Reinforcement learning: an introduction* (second edition ed.). The MIT Press, Cambridge, Massachusetts London, England.

[26] Gary Tan and Yuhan Jin. 2025. Application and Principle of Quadcopters. *Academic Journal of Science and Technology* 18, 2 (Nov. 2025), 186–195. doi:10.54097/p5hqsj19

[27] Teckchai Tiong, Ismail Saad, Kenneth Tze Kin Teo, and Herwansyah Bin Lago. 2020. Deep Reinforcement Learning with Robust Deep Deterministic Policy Gradient. In *2020 2nd International Conference on Electrical, Control and Instrumentation Engineering (ICECIE)*. IEEE, Kuala Lumpur, Malaysia, 1–5. doi:10.1109/ICECIE50279.2020.9309539

[28] Sanxiu Wang, Yue Chen, and Guoan Zhang. 2021. Adaptive fuzzy PID cross coupled control for multi-axis motion system based on sliding mode disturbance observation. *Science Progress* 104, 2 (April 2021), 00368504211011847. doi:10.1177/00368504211011847

A Hyperparameter Tables

Table 9: Final Hyperparameter Configurations and Search Ranges

Parameter	Search Space	SAC	TD3	TAD3
Fixed				
Replay Buffer	—	1,000,000	1,000,000	1,000,000
Learning Initialisation	—	10,000	10,000	10,000
Learning Rate (η)	$[10^{-5}, 10^{-2}]$ (log)	1.49×10^{-4}	1.18×10^{-3}	2.31×10^{-3}
Discount Factor (γ)	$[0.975, 0.9975]$	0.9768	0.9771	0.9771
Polyak Soft Update (τ)	$[0.001, 0.02]$	0.0181	0.0152	0.0195
Batch Size	$\{64, 128, 256, 512\}$	128	512	512
Network Architecture (hidden dims)	$\{[64, 64], [128, 128], [256, 256]\}$	$[256, 256]$	$[256, 256]$	$[256, 256]$
Action Noise Type	SAC: {none, normal, ou}; TD3/TAD3: {normal, ou}	normal	ou	ou
Action Noise Sigma (σ)	$[0.05, 0.5]$	0.359	0.051	0.152
SAC-specific				
Entropy Coefficient (α)	{auto, 0.01}	0.01	—	—
TD3/TAD3-specific				
Policy Delay Frequency (d)	$\{1, 2, 3\}$	—	3	3
Target Policy Noise	$[0.1, 0.3]$	—	0.266	0.153
Target Noise Clip (c)	$[0.3, 0.8]$	—	0.393	0.337

B Seed Pools

Table 10: Disjoint Seeding Protocol Across HPO Stages and Benchmark Evaluation

Pool	Seeds	Count
Local Validation	0–2	3
Calibration and Training	30	1
HPO Search	10–14	5
HPO Validation	20–24	5
PID Tuning	100–104	5
Benchmark	200–209	10

C Experiment Setup Values

Table 11: Experimental Setup Values per Pipeline Stage

Parameter	Calibration	HPO Search	HPO Validation	Benchmark
Total Trials / Cells	15	100 (20 random + 80 TPE)	25 (5 configurations × 5 seeds)	1
Parallel Workers (K)	4	6	6	6
Trial Duration (time steps)	1×10^6	7.5×10^5	1×10^6	1.5×10^6
Training Seed(s)	30	30	20–24	200–209
Evaluation:				
Total Episodes (stage total)	2,700	2,000	150	660
Episodes (per seed-condition pair)	2	2	3	3
Evaluation Checkpoints	9	1	1	11
Evaluation Seeds	10–14	10–14	same as training	same as training
Evaluation Conditions	Both	Both[†]	Both	Both

[†]Both conditions are evaluated and logged every trial, but only `composite` feeds the objective score.

D Ethics Statement and AI Usage Statement

The research conducted did not require ethics clearance, nor did it use any personal information protected under the POPIA Act [22].

D.1 AI Usage Statement

Generative AI was used in the writing of this report. The summary below outlines the ML models used, their purpose, and the extent to which they influenced the final paper.

- **Grammarly** was used to aid in spelling, punctuation and fixing grammar mistakes. The tool was not used to generate any text, but to improve writing quality in line with English grammar rules.
- **Anthropic Claude** aided in the compilation of results, generation and formatting of LaTeX-based tables that are used in this report.
- **Google Gemini** was used as a feedback tool to improve the report's structure and cohesion. It was also used to ensure correct tense and scientific terminology.
- **Google Scholar Labs** helped in the research phase to find suitable peer-reviewed academic journals and papers to base my project on.