UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Instituto de Ciência e Tecnologia de Sorocaba

Aline Gabriel de Almeida

Reinforcement Learning-Based Control for eVTOLs

SOROCABA
2024

Aline Gabriel de Almeida

Reinforcement Learning-Based Control for eVTOLs

Dissertation presented to the Institute of Science and Technology of Sorocaba in partial fulfillment of the requirements for the degree of Master in Electrical Engineering.

Concentration Area: Automation

Supervisor: Prof. Dr. Alexandre da Silva Simões

SOROCABA
2024

A447r

Almeida, Aline Gabriel de
Reinforcement Learning-Based Control for eVTOLs / Aline Gabriel de Almeida. -- Sorocaba, 2024
68 p. : tabs., fotos

Dissertação (mestrado) - Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba
Orientador: Alexandre da Silva Simões

1. Inteligência artificial. 2. Controle de voo. 3. Drone. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Sorocaba

CERTIFICADO DE APROVAÇÃO

TÍTULO DA DISSERTAÇÃO: **Reinforcement Learning-Based Control for eVTOLs**

AUTORA: ALINE GABRIEL DE ALMEIDA
ORIENTADOR: ALEXANDRE DA SILVA SIMÕES

Aprovada como parte das exigências para obtenção do Título de Mestra em Engenharia Elétrica, área: Automação pela Comissão Examinadora:

Prof. Dr. ALEXANDRE DA SILVA SIMÕES (Participaçao Virtual)
Departamento de Engenharia de Controle e Automação / Instituto de Ciência e Tecnologia - UNESP - Câmpus de Sorocaba

Prof. Dr. LEOPOLDO ANDRÉ DUTRA LUSQUINO FILHO (Participaçao Virtual)
Engenharia de Controle e Automação / ICTS - Intituto de Ciência e Tecnologia de Sorocaba - UNESP

Prof. Dr. PAULO ROBERTO FERREIRA JUNIOR (Participaçao Virtual)
Ciência da Computação / Universidade Federal de Pelotas

Sorocaba, 03 de junho de 2024

"Whatever you do, work at it with all your heart."
Colossians 3:23

"Tudo quanto fizerdes, fazei-o de todo coração."
Colossenses 3:23

Resumo

Esta dissertação de mestrado explora o controle de aeronaves de decolagem e pouso vertical elétricas (eVTOL), com foco na utilização de Aprendizado por Reforço (RL) para aprimorar métodos de controle. A Otimização de Política Proximal (PPO) é utilizada como estratégia principal de controle, enquanto que uma comparação é feita com o Gradiente de Política Determinística Profunda com Atraso Duplo (TD3). As estratégias são testadas em diversos tipos de UAVs com formas e níveis de complexidade de controle diferentes. A integração dessas estratégias com modelos de aeronaves ocorre dentro do ambiente de simulação MuJoCo. Além disso, é realizada uma análise sobre a capacidade das aeronaves eVTOL de lidar com falhas. Cenários simulados incluem casos onde um ou mais rotores param de funcionar temporariamente, semelhante apossiveis situações da vida real. Por meio dessas avaliações, o objetivo é explorar a eficácia dessas estratégias de controle em contextos práticos. Essa compreensão aprofundada pode contribuir com avanços na aplicação real de técnicas de RL para o controle de aeronaves eVTOL.

Palavras-chave: Aeronaves Elétricas de Decolagem e Pouso Vertical, Aprendizado por Reforço, Otimização de Política Proximal, Gradiente de Política Determinística Profunda com Atraso Duplo, Teste de robustez, Mobilidade Urbana.

Abstract

This master's dissertation explores the control of electric vertical take-off and landing (eVTOL) aircraft, with a focus on utilizing Reinforcement Learning (RL) to enhance control methods. Proximal Policy Optimization (PPO) serves as the main control strategy, while a comparison is made with the Twin-Delayed Deep Deterministic Policy Gradient (TD3). The strategies are tested across various types of UAVs with differing shapes and levels of control complexity. Integration of these strategies with aircraft models occurs within the MuJoCo simulation environment. Furthermore, an examination is conducted regarding the eVTOL aircraft's ability to manage malfunctions. Simulated scenarios include instances where one or more rotors cease functioning temporarily, akin to possible real-life scenarios. Through these evaluations, the aim is to gain insights into the effectiveness of these control strategies in practical contexts. This understanding could potentially lead to advancements in the real-world application of RL techniques for eVTOL aircraft control.

Keywords: eVTOL, Reinforcement Learning, Proximal Policy Optimization, Twin-Delayed Deep Deterministic Policy Gradient, Control Strategies, Robustness Testing, Urban Air Mobility.

List of Figures

List of Tables

List of Abbreviations and Acronyms

A2C	Advantage Actor-Critic
CO_2	Carbon dioxide
DRL	Deep Reinforcement Learning
eVTOLs	Electric Vertical Takeoff and Landing
FCTS	Flying Car Transportation System
MTOW	Maximum takeoff weight
NGS	Near Ground Space
PPO	Proximal Policy Optimization
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
tanh	Hyperbolic tangent
TD3	Twin-Delayed Deep Deterministic Policy Gradient
TRUAV	Tilt-rotor UAV
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
VTOL	Vertical Takeoff and Landing

Contents

1 Introduction

1.1 Urban Mobility in Conventional Transportation System

Cities and urban areas worldwide have experienced rapid growth in recent decades. Today, 55% % of the world's population lives in urban areas, which is expected to increase to 68% by 2050 (United Nations Department of Economic and Social Affairs, 2018). This substantial growth leads to increased demand for mobility – people making their way to work, farmers bringing their crops to market, and raw materials being delivered.

Transportation is pivotal in driving economic activity and ensuring human welfare. However, the transport sector has significant environmental and public health impacts. Transportation gas emission is one of the major contributors to carbon dioxide (CO_2) and other harmful pollutants (GOTA, 2023). Specifically, in 2020, global CO_2 emissions reached 31.9 billion tonnes CO_2, with the transportation sector contributing 7.1 billion tonnes CO_2, comprising approximately one-fifth of global CO_2 emissions. Passenger vehicle transportation stands out as the largest contributor in the transportation sector. Figure 1.1 provides a visual representation of emissions by sectors (World Resources Institute, 2023).

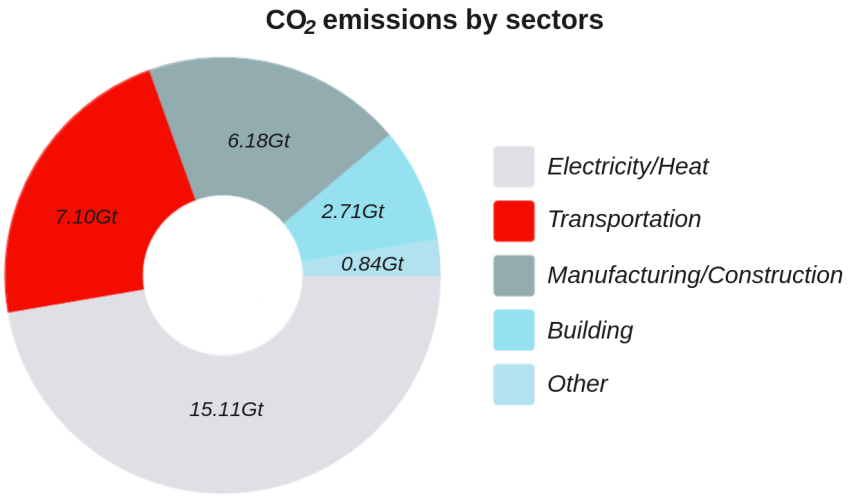


Figure 1.1 – Carbon dioxide emissions by sectors.

Conventional transportation systems exploit underground, waterborne, ground-level, and high-altitude spaces, as illustrated in Figure 1.2.

Waterborne transport, including maritime and fluvial transportation, is primarily reserved for long-distance freight transport and is unsuitable for most urban settings. The high-altitude system is also usually reserved for long-distance human/goods delivery but comes with signifi-

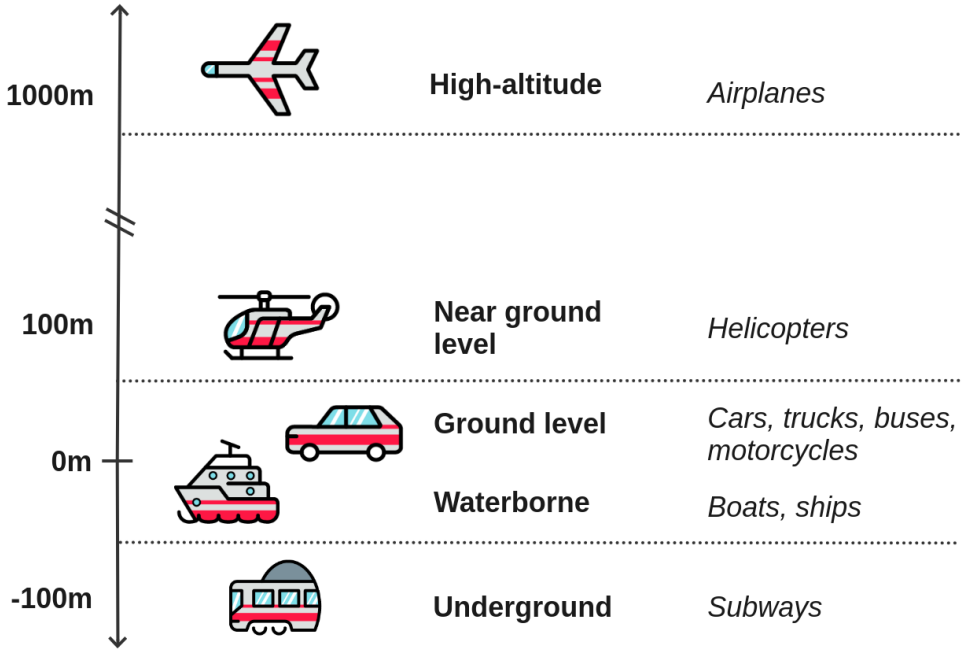


Figure 1.2 – Main transportation systems spaces.

cantly higher costs, making it impractical for urban applications. Thus, underground, ground level, and near ground level spaces best serve urban scenarios (PAN; ALOUINI, 2021).

Ground-level transportation is the predominant transportation system, up to three-quarters of all passenger mobility. Although very popular, it faces limitations due to track and road constraints, resulting in poor flexibility and traffic congestion, particularly in urban areas. As we look to the future, addressing traffic-congested cities and yet moving towards safe, affordable, accessible, integrated, and sustainable transport systems and unleashing the potential of emerging technologies to reduce congestion and air pollution is forthcoming as a world challenge to solve (United Nations, 2021).

1.2 The Potential of Flying Car Transportation Systems

Automated driving and electrification are disruptive technologies that have the potential to address the challenges associated with the growing demand for convenient passenger mobility, reduce congestion, enhance safety, and mitigate emissions. However, driving is constrained by traffic congestion on existing roadways and land-use restrictions. Electric vertical takeoff and landing aircraft (eVTOLs), on the other hand, have the potential to overcome these limitations by enabling urban and regional aerial travel services.

The Flying Car Transportation System (FCTS) concept has been introduced, reviewed, and discussed in many recent publications (PAN; ALOUINI, 2021; AHMED et al., 2020; POSTORINO; SARNE, 2020; MOFOLASAYO, 2020).

FCTS differs from traditional transportation systems by exploiting the underused Near Ground Space (NGS), which enhances travel times' reliability and accelerates the movement of people and goods, particularly in urban areas. Simultaneously, it reduces dependence on road infrastructure, traffic congestion, and air pollution. (PAN; ALOUINI, 2021; POSTORINO; SARNE, 2020).

While FCTS offers superior transportation capabilities, its widespread adoption will be largely influenced by public perception. Recent studies have shown that people's perception toward potential benefits and concerns of the future use of flying cars are influenced by a willingness for low and reliable travel times and enhanced safety and security (AHMED et al., 2021; EKER et al., 2022). Specifically, Figure 1.3 demonstrates public concerns for flying cars safety.

In this context, the ongoing development of flying cars' safety and security will play a pivotal role in the full-scale introduction of FCTS.

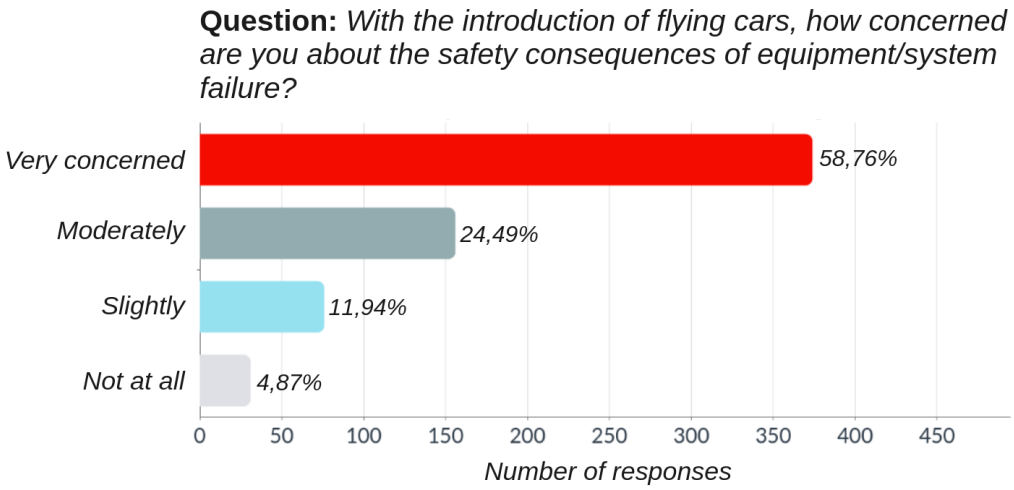


Figure 1.3 – Survey on public perception towards flying cars safety. Data source: (EKER et al., 2022)

.

1.3 Pioneering Developments in Flying Cars

Within the evolving field of flying cars, numerous companies have introduced innovative models and designs that challenge traditional aviation concepts, with Joby, Volocopter, and EHang emerging as the most prominent players in the electric aircraft industry. These companies represent innovative approaches to urban air mobility, and each offers unique features and advantages.

The Joby S4 is an eVTOL aircraft designed and focused on urban air mobility. As shown in Figure 1.4, its design characteristics include a unique configuration with six tilting rotors, allowing for vertical takeoff and landing and transitioning to more efficient fixed-wing flight for cruising. The S4 is engineered to achieve a maximum cruise velocity of around 300km/h and an operational altitude of up to 3,000 meters. With a maximum takeoff weight (MTOW) of about 1,800kg, it can carry four passengers and a pilot. The aircraft is intended to be an air taxi, facilitating urban transportation by offering fast, efficient, and environmentally friendly commuting options. Joby emphasizes safety through redundant systems and a focus on vertical takeoff and landing capabilities, making it suitable for urban environments with helipads or designated landing zones (Joby Aviation, 2023; The Vertical Flight Society, 2023b).

Figure 1.4 – Joby S4 (Joby Aviation, 2023)

The VoloRegion is another eVTOL aircraft designed for urban air mobility applications, shown in Figure 1.5. Its distinctive design features eight propellers distributed across two wings. It is engineered to reach a cruise velocity of approximately 180km/h and operate at up to 1,500m altitudes. With an estimated MTOW of 2,500kg, the VoloRegion can transport up to five passengers. Its primary use case is for urban air mobility, offering efficient and rapid transportation within cities. VoloRegion emphasizes reliability and safety through redundant systems and advanced flight control technologies, ensuring that it meets the stringent safety requirements of urban aerial transportation (Volocopter, 2023; The Vertical Flight Society, 2023c).

The Ehang 216-S is a fully electric autonomous aerial vehicle with an octocopter configuration designed for air taxis and short-haul urban transport. It features a total of 16 rotors distributed across eight arms, which provide redundancy and enhanced stability. It is engineered for a maximum cruise velocity of around 100km/h and an operational altitude of up to 3,000 meters. With an MTOW of approximately 650kg, it can carry two passengers. Ehang's primary focus is on providing autonomous passenger transport, making it an urban mobility solution of the future. Safety and reliability are central to Ehang's design, featuring multiple redundant

Figure 1.5 – Volocopter VoloRegion (Volocopter, 2023)

systems, fail-safes, and advanced flight control technologies to ensure passenger safety during autonomous flights, even in emergencies (eHang, 2023; The Vertical Flight Society, 2023a).

Figure 1.6 – Ehang 216-S (eHang, 2023)

1.4 Addressing eVTOL Control Challenges

Controlling eVTOL aircraft poses a significant technical challenge due to their highly nonlinear dynamics. These dynamics arise from the necessity to tilt the rotors, wings, or even the entire airframe to enable efficient cruising at high speeds and the ability to take off and land like a helicopter. They also complicate developing a unified control system, as eVTOLs operate in multiple flight modes, where the aircraft's behavior and dynamics significantly vary depending on the operating point or mode of operation. Additionally, operating eVTOLs in these modes poses a challenge for designing control algorithms that can seamlessly transition between them, ensuring safe and efficient operation.

In the subsequent sections, the challenges associated with Classical Control methods

will be explored and how Deep Reinforcement Learning emerges as a promising approach for addressing these limitations and enhancing the control of eVTOLs.

1.4.1 Classical Control Approach

Electric vertical takeoff and landing vehicles have relied on traditional control methods. However, the intricate nature of eVTOL operations requires a comprehensive evaluation of the advantages and limitations of these classical control strategies.

The significant control challenges of an eVTOL are due to the highly nonlinear dynamics, particularly during transition maneuvers between helicopter and airplane modes. Traditional methods for capturing aerodynamic characteristics often fail to provide analytical expressions to fine-tune controllers for such complex operations.

To address these challenges, Scheduled Control approaches have been widely adopted. These methods involve linearizing the system around specific conditions and designing dedicated controllers for each operating scenario. This approach simplifies controller synthesis and offers adaptability to distinct operational scenarios. Two primary implementations in this context are Divide and Conquer and Control Authority Weighting.

The Divide and Conquer approach discretely switches between different control laws, ensuring that only one controller operates at any given time. While this approach simplifies controller synthesis, it has some downsides. It is often limited to specific operational conditions, making it less suitable for handling abrupt changes or wind perturbations. Furthermore, it may not provide optimal performance in cases where a smooth transition between control laws is required (NAKAMURA et al., 2018; SHEN et al., 2018; YU; ZHANG; ZHANG, 2017; CETINSOY, 2015; HERNANDEZ-GARCIA; RODRIGUEZ-CORTES, 2015).

In contrast, the Control Authority Weighting approach continuously combines commands from two scheduled controllers based on scheduling-variable-dependent weights. Tuning the individual controllers' dwell time smaller than the parameter-variation time constant generally results in good local performance. However, it's essential to exercise caution regarding the global closed-loop stability and performance when employing Control Authority Weighting because this approach can be sensitive to certain conditions, and careful consideration is required to ensure overall system stability (BAUERSFELD; DUCARD, 2020; ANGLADE et al., 2019).

These scheduling policies offer valuable tools for managing the complexity of eVTOL control, but they come with trade-offs. While they simplify controller synthesis and adapt to specific operational scenarios, they may struggle with sudden changes or disturbances. The choice between Divide and Conquer and Control Authority Weighting should be made with a thorough understanding of their advantages and limitations.

Unified Control approaches represent an alternative strategy that aims to overcome the limitations of Scheduled Control approaches. Instead of relying on multiple controllers for distinct

operating conditions, Unified Control deploys a single controller designed to handle the entire range of flight conditions for hybrid UAVs. This approach is more versatile and can address the nonlinear dynamics and aerodynamic effects encountered during transitions. In unified control, tilt angles and attitude reference computations for thrust vector alignment are often integrated, allowing online optimization to enhance flight performance and efficiency. Adaptive or nonlinear control methodologies are frequently used to manage changing aerodynamic effects and control authorities, ensuring stability and robustness throughout the spectrum of flight conditions (BAUERSFELD et al., 2021; ALLENSPACH; DUCARD, 2021; LIU et al., 2019; CHIAPPINELLI et al., 2019; HEGDE; GEORGE; NAYAK, 2019).

However, it is important to note that Unified Control approaches have drawbacks. Unified Control approaches can be limited by their reliance on a single controller for the entire range of operating conditions. The control system must be designed to adapt quickly to changing conditions to avoid sudden variations in aerodynamic effects and control authorities, potentially affecting stability. Moreover, the choice of control parameters in a unified approach must be carefully optimized to ensure optimal performance across all flight scenarios. This can be a challenging task; if not done correctly, it may result in suboptimal control performance.

Additionally, the nonlinear behaviors of eVTOLs, driven by complex aerodynamic and mechanical interactions, especially during rapid transitions, challenge traditional control methods. Classical techniques, often reliant on linear approximations, struggle to accurately model these dynamics, leading to potential instability and poor performance in varied operational scenarios. Furthermore, unexpected behaviors that arise from the interplay within the system complicate control efforts, as classical methods cannot easily predict or adapt to these spontaneous changes. These challenges are particularly pronounced during critical transitions, such as switching from vertical takeoff to forward flight, where minor variations can trigger significant shifts in the system's state. Classical control methods, designed primarily for gradual changes, may not manage these transitions effectively, risking performance degradation or instability.

While, classical control techniques remain relevant in the management of eVTOLs, their application is often limited by the inherent constraints. The dynamic and varied flight modes of eVTOLs demand more adaptive and robust control paradigms. In the subsequent section, the emerging Deep Reinforcement Learning will be explored as a potential tool to enhance eVTOL control.

1.4.2 Deep Reinforcement Learning Control Approach

The emerging field of Reinforcement Learning (RL) offers a novel perspective on eVTOL control. Deep Reinforcement Learning (DRL) has been gaining prominence as a promising avenue for addressing complex control challenges due to its ability to learn and adapt control strategies from experience rather than relying on predefined rules or models (ARULKUMARAN et al., 2017).

DRL stands out as an appealing approach due to its ability to learn and adapt control strategies from experience rather than relying on predefined rules or models. It can leverage vast data to fine-tune control policies and optimize eVTOL performance across various operating conditions. This adaptability is especially crucial in eVTOLs, which frequently encounter dynamic changes in aerodynamic effects and control authorities during transitions and flight modes.

Unlike classical control methods, controllers are often designed and tuned manually; DRL allows for a more autonomous and data-driven approach. Using trial and error, DRL algorithms can explore various control strategies, identify the best results, and continuously refine the control policy. This adaptive nature can improve control performance and better handling of unexpected disturbances or changing conditions. It can be particularly advantageous as RL algorithms can adapt and optimize control policies in real time, making them well-suited for handling complex and dynamic scenarios.

The critical strengths of DRL in Unmanned Aerial Vehicles (UAV) control include:

Adaptability and Autonomous Learning: DRL allows UAVs to learn and adapt control strategies from experience rather than relying on predefined rules or models. This adaptability is crucial, especially in eVTOLs that frequently encounter dynamic changes in aerodynamic effects and control authorities during transitions and various flight modes.

Data-Driven Approach: Unlike traditional control methods that often require manual design and tuning of controllers, DRL algorithms can autonomously explore and optimize control strategies through trial and error. This data-driven approach can improve control performance and better handling of unexpected disturbances or changing conditions.

Real-Time Adaptation: DRL algorithms excel in real-time control. They continuously adapt and optimize control policies, making them well-suited for managing complex and dynamic scenarios, ensuring stability, and maximizing performance.

DRL's innovative features open doors to new and challenging tasks in UAV control, such as safely controlling hybrid UAVs and coordinating swarms of UAVs to achieve specific tasks with minimal resources. It's important to note that DRL is not without its challenges and considerations. Training DRL algorithms can be data-intensive, and ensuring safety and stability during learning is paramount. DRL algorithms can potentially learn suboptimal or unsafe control policies if not appropriately guided (AZAR et al., 2021).

1.5 Research Questions

Considering the scenario previously presented, the aim of this research is to answer the following questions:

1. Can eVTOL geometry be adapted to simulated environments considering critical factors like degree of freedom, joint configuration, and other key characteristics?

2. Can the flight of eVTOLs be controlled to meet distinct operational demands using RL-based control?

3. How do these control policies respond to potential challenging scenarios associated with flight, such as rotor failures?

1.6 Research Objectives

This study outlines its primary research objectives, centered on enhancing control strategies for eVTOL aircraft through the application of Reinforcement Learning. Specifically, utilizing the PPO algorithm as the base approach, the study aims to seamlessly integrate aircraft models with the simulator, including various aspects such as degrees of freedom, joint configuration, and other characteristics. Subsequently, the study will explore control using the TD3 algorithm, comparing the results to PPO, particularly in handling UAVs with increasing control complexity. Additionally, this study will explore the robustness of UAV control under various failure conditions, using PPO algorithm.

1.7 Text Organization

The remaining of this work is organized as follows:

Chapter 2: Background and Theory: This chapter lays the foundation for the research by exploring the theoretical underpinnings of control strategies for electric Vertical Takeoff and Landing (eVTOL) vehicles. It discusses fundamental concepts in Reinforcement Learning (RL) and their relevance to eVTOL control.

Chapter 3: Analysis of Related Work in eVTOL Control Using DRL Approaches : In this chapter, a systematic review of existing literature related to eVTOL control strategies is conducted, employing Deep Reinforcement Learning (DRL) approaches. Methodologies, network architectures, training processes, and performance metrics from relevant studies are analyzed.

Chapter 4: Materials and methods: This chapter outlines the methodology employed in this research for the development and evaluation of eVTOL control strategies. It details the simulation setup, including physics simulation environments, reward function parameters, and failure analysis methodologies.

Chapter 5: Results and Discussion: In this pivotal chapter, the results of experiments are presented. The performance of different control strategies under various conditions is analyzed,

and the robustness of policy optimization algorithms is discussed. Additionally, the results of failure tests, examining the responses of UAVs to simulated failure scenarios.

Chapter 6: Conclusion and Future Directions: The concluding chapter summarizes the key findings of the research and discusses their significance in the context of eVTOL control. Potential future directions for further exploration and development in this field are also outlined.

2 Background and Theory

In this chapter, the background knowledge supporting this work will be addressed.

2.1 Overview of Reinforcement Learning

Reinforcement Learning (RL) is a subfield of machine learning that focuses on training intelligent agents to make decisions to maximize a cumulative reward. The fundamental components of RL include the agent and environment, Policy, Reward, and value function.

Agent: In RL, an interactive agent interacts with an environment and learns to map different states to actions to achieve a predefined goal. The agent is responsible for taking actions in the environment based on observations and past experiences. The primary objective of any RL algorithm is to enable the agent to learn an optimal policy that accurately achieves the desired task, resulting in the highest possible reward (SUTTON; BARTO, 2018). The interaction between an agent and its environment is a central concept in RL. The agent selects an action from a set of available actions at a specific time and executes it in the environment. This action leads to a change in the state, either bringing the agent closer to its goal or taking it further away, with the corresponding reward signal indicating the quality of the action taken.

Environment: The environment represents the external system or domain within which an RL agent operates and interacts. It is the setting in which the agent takes actions, observes the consequences of those actions, and receives rewards or feedback based on its behavior. The environment is characterized by its dynamics and state transitions, which influence the agent's decision-making process and, in turn, the agent's actions impact the environment. This interaction between the agent and the environment is illustrated in Figure 2.1. It forms the core of the RL learning process, enabling the agent to adapt and optimize its behavior to maximize cumulative rewards.

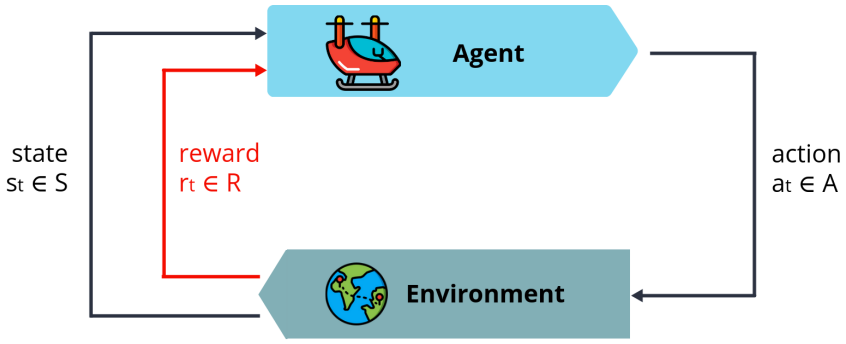


Figure 2.1 – The agent-environment interaction.

Policy: The Policy determines the agent's strategy, specifying how it responds to different

environmental states. Rewards are numerical values provided by the environment in response to the agent's actions, reflecting the desirability of those actions.

Reward: The Reward is a numerical signal provided to an agent at each time step by the environment, serving as immediate feedback to evaluate the desirability of the agent's current state and action.

Value Function: The value function calculates the long-term return, considering the cumulative rewards achieved by following a specific policy from a given state. The environment model represents the environment's behavior, aiding the agent in decision-making (SUTTON; BARTO, 2018).

In this scenario, the agent, compound by both the drone and the learning algorithm, makes decisions a_t regarding actions at time t, selecting from a set of available actions. Subsequently, the chosen action is executed within the environment, leading to a transformation in the state, denoted as s_t. This state change either brings the agent closer to its intended target, such as reaching a specific location or moves it further away. The quality of the action taken is quantified through a reward, r_t, which can represent, for example, a positive reward if the agent gets closer to its goal or a negative reward if it moves farther away from it (SUTTON; BARTO, 2018).

Deep Reinforcement Learning (DRL) is a sub-field of RL that combines deep learning techniques with reinforcement learning to solve complex problems. DRL approaches are typically categorized into model-based and model-free, with the latter further sub-categorized into policy-based or value-based methods. Value-based DRL distinguishes between on-policy and off-policy strategies, as visually represented in Figure 2.2.

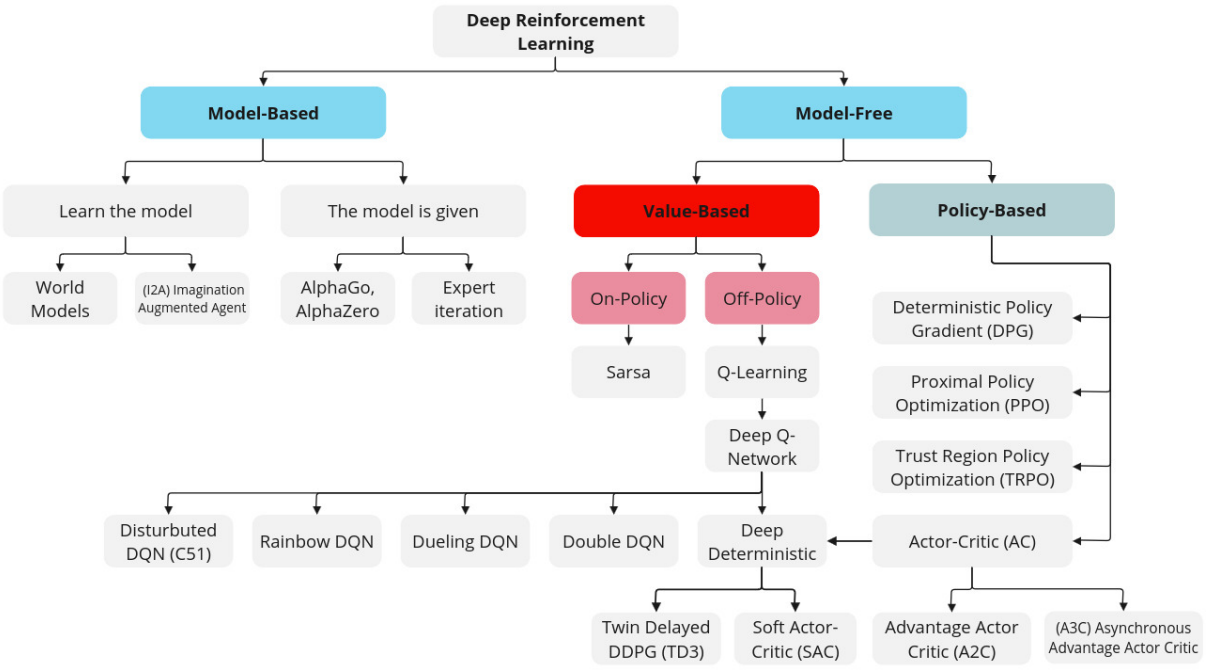


Figure 2.2 – Taxonomy of Reinforcement Learning Algorithms. Source: Adapted from (AZAR et al., 2021).

Value-Based: Value-Based Reinforcement Learning is an approach that centers around estimating the value of states or state-action pairs within an environment. The value represents the expected cumulative rewards an agent can obtain by following a particular policy, focusing on learning and updating value functions to guide decision-making. The agent selects actions that maximize the estimated value, ultimately striving to improve its Policy iteratively. This approach is particularly effective when the agent's primary goal is to identify the best actions to take in various states of the environment (SUTTON; BARTO, 2018).

Policy Based: Policy-based reinforcement Learning emphasizes learning the optimal Policy directly, which maps states to actions without explicitly estimating the value of states or state-action pairs. Instead of relying on value functions, the algorithm focuses on iteratively improving the Policy itself. This approach is well-suited for problems where the action space is large or continuous, and the agent's objective is to find the best strategy or set of actions to maximize cumulative rewards without explicitly estimating values (SUTTON; BARTO, 2018).

On-Policy: On-Policy Reinforcement Learning refers to a class of RL algorithms where the agent learns and updates its Policy while actively interacting with the environment. The Policy used for learning is the same as the Policy used for generating data. This means that the agent's actions during exploration are based on its current, potentially suboptimal, Policy. On-policy methods, like SARSA (State-Action-Reward-State-Action) and A3C (Asynchronous Advantage Actor-Critic), are valuable when dealing with problems that require careful exploration and when a balance between exploration and exploitation is crucial (SUTTON; BARTO, 2018).

Off-Policy: Off-Policy Reinforcement Learning, on the other hand, involves decoupling the Policy used for exploration from the Policy used for learning. In off-policy methods like Q-learning and DDPG (Deep Deterministic Policy Gradients), the agent can use one Policy to collect data (exploration), while learning and improving a different policy that maximizes cumulative rewards (exploitation). This separation of policies allows for more effective and efficient learning by reusing previously collected data, making it well-suited for scenarios where data efficiency and optimization are paramount (SUTTON; BARTO, 2018).

Model-based: Model-based RL involves the creation of an internal model of the environment, enabling the agent to simulate and predict the consequences of different actions. This approach enhances decision-making by providing a predictive understanding of the environment's behavior and is particularly advantageous in scenarios where real interactions are resource-intensive (ARULKUMARAN et al., 2017; SUTTON; BARTO, 2018).

Model-free: On the other hand, Model-Free RL focuses on direct learning from real-world interactions, eliminating the need for an explicit environment model. Instead, it learns optimal policies and value functions directly from observed states and rewards. This approach excels in complex, uncertain, or dynamic environments where constructing an accurate model is challenging and offers versatility for various applications due to its simplicity and adaptability. The choice between these two approaches depends on the specific problem and the practical

considerations at hand (SUTTON; BARTO, 2018).

Despite these classifications, DRL implementations share common characteristics, including closed-loop control, delayed rewards, and sequential decision-making, often involving a credit assignment problem due to the dependence on actions over time (ARULKUMARAN et al., 2017).

2.2 Key Reinforcement Learning Algorithm for eVTOL Control

In the context of Deep Reinforcement Learning (DRL) for eVTOL control, the Proximal Policy Optimization (PPO) algorithm stands as significant milestone, offering robust and effective approaches for optimizing policies in complex decision-making environments.

PPO, known for its ability to manage the exploration-exploitation trade-off and its stable policy updates, has been widely adopted in various reinforcement learning tasks, including eVTOL control.

An alternative approach can be use TD3 optimization algorithm, that introduces twin critic networks and delayed updates to enhance stability and performance in continuous action spaces, making it a promising choice for eVTOL control scenarios.

This section delves into the intricacies of the PPO and TD3 algorithms, outlining their core concepts and methodologies, and their application in optimizing policies for eVTOL control tasks.

2.2.1 Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a model-free reinforcement learning algorithm renowned for its ability to optimize policies by effectively managing the trade-off between exploration and exploitation. In reinforcement learning, exploration refers to the process of trying new actions to discover their potential benefits and gather more information about the environment. This is essential for uncovering new strategies and ensuring the agent does not get stuck in suboptimal behavior. Exploitation, on the other hand, involves leveraging the agent's current knowledge to maximize immediate rewards by choosing actions that have been effective in the past. Balancing these two aspects is crucial: excessive exploration can lead to inefficient learning and prolonged search times, while too much exploitation can prevent the agent from finding better strategies and adapting to new situations.

PPO addresses this balance by introducing flexible penalty terms instead of relying on rigid constraints. These penalties are managed through the Kullback-Leibler (KL) divergence, which measures the difference between the updated policy and the previous one. By limiting

the KL divergence, PPO ensures that policy updates are significant enough to allow exploration but not so drastic that they destabilize the learning process, supporting effective exploitation. This approach simplifies the optimization process by allowing controlled policy updates at each time step, maintaining a balance between exploring new actions and exploiting known good strategies.

PPO offers a balanced approach between implementation simplicity, sample efficiency, and ease of hyperparameter tuning. It was developed to address the limitations of Deep Q-learning and Trust Region Policy Optimization (TRPO) and aims to minimize the cost function while maintaining policy stability. It is based on an actor-critic architecture, facilitating rapid learning and adaptation by considering the critic's feedback (SCHULMAN et al., 2017).

In the actor-critic architecture, an agent interacts with an environment, receiving rewards based on its actions. Simultaneously, a critic evaluates the expected rewards that another actor's actions would have produced, allowing for a comparison between the chosen actions and the optimal ones. PPO uses gradient descent to adjust the policy and adapt it to the critics' feedback.

Policy gradient methods involve estimating and applying the policy gradient in a stochastic gradient ascent algorithm. The policy gradient estimator takes the following form:

$$\hat{g} = \hat{E}_t \left[\nabla_\theta \log \pi_\theta(a_t|s_t) \hat{A}_t \right]$$

Where π_θ is a stochastic policy, $\hat{A}_t$ is the estimator for the advantage function at time step t, and $\hat{E}_t$ denotes the expectation, computed over a batch of samples.

To prevent huge policy updates, a clipping method is applied to the advantage estimator, constraining the Policy from deviating too far from the previous policy. This method uses a hyperparameter, ϵ, to determine the allowable policy deviation, typically set to 10-20%. The clipping function is defined as:

$$L_{\text{CLIP}}(\theta) = \hat{E}_t \left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right]$$

Where the first term inside the min function represents conservative policy iteration, the second term, $clip$ modifies the surrogate objective by limiting the probability ratio, ensuring it stays within the interval $[1-\epsilon, 1+\epsilon]$. The min function selects the lower bound between the clipped and unclipped objectives, providing robustness against hyperparameter sensitivity and outlier data during training (SCHULMAN et al., 2017).

PPO aims to maintain smooth gradient updates for continuous improvement, avoiding unrecoverable policy crashes. The surrogate policy loss function, as described, plays a pivotal role in PPO's efficient learning process by leveraging the ratio of new and old probabilities and the advantages provided by the advantage function, which estimates the relative value of selected actions.

Algoritmo 2.1: Proximal Policy Optimization

for *iteration = 1,2,...* **do**
  for *actor = 1,2,...,N* **do**
    Run policy $\pi_{\theta_{\text{old}}}$ in the environment for T time steps
    Compute advantage estimates $A_1, \ldots, A_T$
  Optimize surrogate loss L with respect to θ, using K epochs and a minibatch size $M \leq NT$
  $\theta_{\text{old}} \leftarrow \theta$

The pseudocode in Algorithm 2.1 outlines the high-level structure of the Proximal Policy Optimization algorithm. It iteratively updates the Policy by running multiple actors in the environment, estimating advantages, and optimizing the surrogate loss with clipping constraints to ensure policy stability.

Proximal Policy Optimization is a well-balanced RL algorithm that combines ease of implementation, sample efficiency, and stable policy updates. PPO achieves state-of-the-art performance in RL tasks by utilizing the surrogate policy loss function and the clipping mechanism, making it a popular choice for various applications.

2.2.2 Twin Delayed Deep Deterministic Policy Gradient (TD3)

TD3 is a model-free reinforcement learning algorithm designed for continuous action spaces, offering improved stability and performance by leveraging twin critics and delayed updates. It addresses challenges in traditional Deep Q-learning and TRPO by incorporating twin networks and target networks.

The actor-critic architecture in TD3 involves an agent interacting with an environment, receiving rewards based on its actions. Concurrently, twin critic networks evaluate the expected rewards of alternative actions, reducing estimation bias. TD3's objective is to minimize the cost function while maintaining policy stability, achieved through gradient descent adjustments guided by critic feedback.

The key difference in TD3 lies in its use of twin critic networks and delayed updates. The twin critics mitigate overestimation bias by considering the minimum value of the two critics' estimates, enhancing the accuracy of action-value function evaluations. Additionally, TD3 employs target networks for both the actor and critics, gradually updating them to stabilize training and improve convergence.

The pseudocode in Algorithm 2.2 outlines the iterative process of TD3, where the actor's policy and twin critics are updated based on sampled transitions from the replay buffer. Target networks are updated periodically to stabilize training and reduce variance in value estimates.

TD3's twin critic approach and delayed updates contribute to its stability and performance,

making it a viable choice for continuous action reinforcement learning tasks.

Algoritmo 2.2: Twin Delayed Deep Deterministic Policy Gradient (TD3)

Initialize actor π_θ, twin critics Q_{ϕ_1} and Q_{ϕ_2}, target networks $\pi_{\theta_{\text{target}}}$, $Q_{\phi_{\text{target}}}$;
for *episode = 1,2,...* **do**
    Initialize random process for action exploration;
    Receive initial observation state s_1;
    for *timestep = 1,2,...* **do**
        Select action $a_t = \pi_\theta(s_t) + \epsilon_t$ with exploration noise ϵ_t;
        Execute action a_t, observe reward r_t and new state s_{t+1};
        Store transition (s_t, a_t, r_t, s_{t+1}) in D;
        Sample a random minibatch of transitions (s_i, a_i, r_i, s_{i+1}) from D;
        Compute target Q-values:
        $y_i = r_i + \gamma \min(Q_{\phi_{\text{target}}}(s_{i+1}, \pi_{\theta_{\text{target}}}(s_{i+1})), Q_{\phi_{\text{target}}}(s_{i+1}, \pi_{\theta_{\text{target}}}(s_{i+1})))$;
        Optimize twin critics Q_{ϕ_1} and Q_{ϕ_2} using MSE loss with target values y_i;
        Update actor policy π_θ using the deterministic policy gradient:;
        $\nabla_\theta J \approx \hat{E}t\left[\nabla\theta\pi_\theta(s_t)\nabla_a Q_{\phi_1}(s_t, a)|a = \pi\theta(s_t)\right]$;
        Update target networks $\pi_{\theta_{\text{target}}}$ and $Q_{\phi_{\text{target}}}$ periodically;

3 Analysis of Related Work in eVTOL Control Using DRL Approaches

The objective of this section is to provide a comprehensive overview of the most recent advancements in eVTOL control using DRL, shedding light on the diverse approaches taken by different researchers and the corresponding performance outcomes. By understanding the successes and challenges these approaches face, the groundwork can be laid for developing robust and efficient control strategies for eVTOLs, contributing to the broader objective of safe and reliable eVTOL transportation systems.

The selected academic works for comparison have been chosen based on their relevance, methodological rigor, and significance in the aircraft control domain. Each work offers a unique perspective on how RL can be leveraged to enhance the capabilities of eVTOLs. The key aspects of these studies, such as the RL algorithms employed, simulation environments, evaluation metrics, and the performance of the resulting control strategies, will be explored. Through this analysis, a better understanding of the current landscape of eVTOL control research can be facilitated, highlighting opportunities for further exploration and improvement.

3.1 PPO-Based Attitude Controller Design for a Tilt Rotor UAV in Transition Process (2023) (YANG et al., 2023)

In the investigated work, the authors address the challenges presented by the aerodynamic changes in the tilt-rotor UAV (TRUAV) during the transition process, which substantially impacts the vehicle's attitude stability.

Their study is motivated by the objective of developing a PPO-based Reinforcement Learning controller specifically tailored for managing the attitude during this transitional stage. The authors facilitate learning control strategies through direct environmental interaction by utilizing RL, specifically the PPO approach. Furthermore, they have devised and fine-tuned a reward function optimized for the transitional process. Through simulations, they assess the performance of the proposed controller and conduct a comparative analysis with alternative methods.

The research underscores the efficacy and potential of RL in addressing the intricacies of TRUAV transition process attitude control, offering valuable insights for applying RL algorithms in autonomous flight systems.

3.1.1 Aircraft Layout

The layout of the Tilt-Rotor UAV is shown in Figure 3.1. In the eVTOL configuration, a flying wing layout is adopted with the four motors positioned at a distance from the wing, and the center of gravity of the TRUAV is situated at the center of the four motors. Throughout the transition process, the tilt angles of the two front motors, referred to as r_1 and r_2, can progressively transition from 0° to 90°.

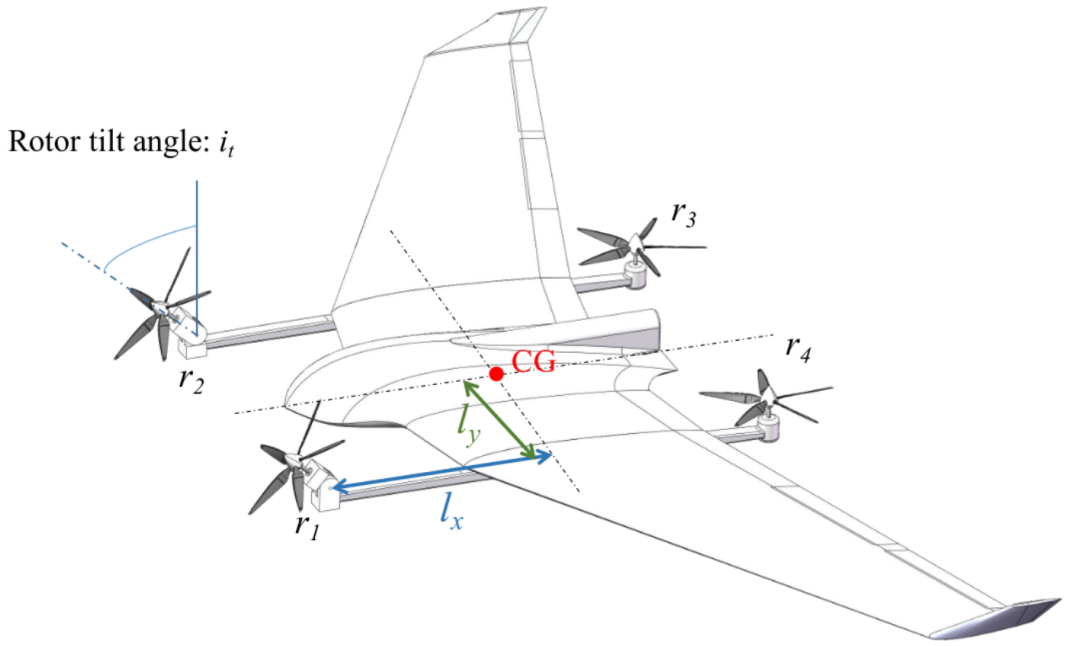


Figure 3.1 – Layout of the Tilt-Rotor UAV. Source: (YANG et al., 2023).

3.1.2 Algorithm Framework

In the paper, a standard reinforcement learning structure is used for controller training. The algorithmic framework is shown in Figure 3.2.

The applied method employs the policy proximal optimization (PPO) algorithm. In the context of the study, the PPO algorithm is applied to the task of controlling the attitude of the TRUAV during the transition phase, encompassing the acceleration from hover to cruise speed. To ensure stability and mitigate divergence, the algorithm incorporates a clip function, restricting the disparity between the previous and updated policies.

3.1.3 Network Architecture

The neural network employed in the study consists of policy and value networks, illustrated in Figure 3.3. Both networks share a similar architecture, featuring two fully connected layers comprising 128 units. The activation function utilized is the Rectified Linear Unit (ReLU). The value network produces the value for the current state vector, denoted as s, which, in turn,

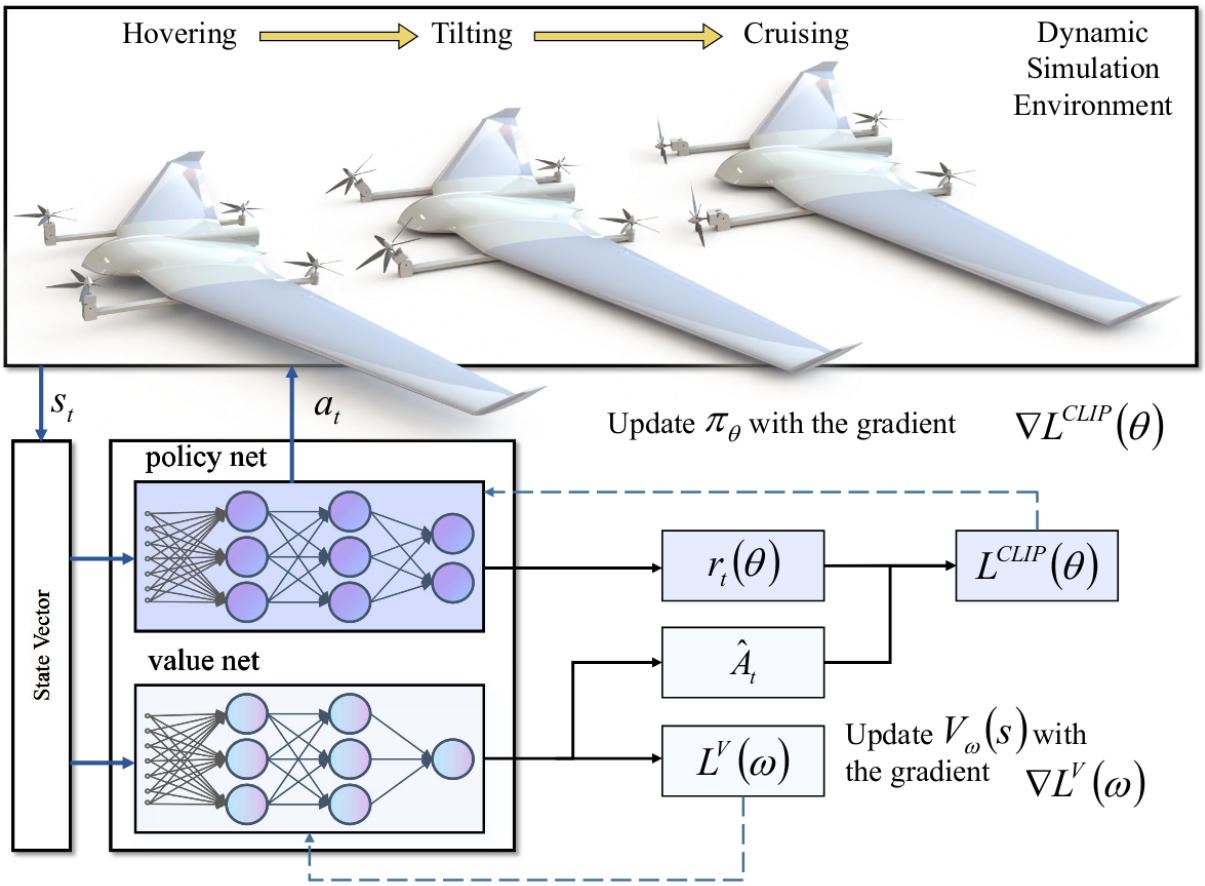


Figure 3.2 – Schematic diagram of the algorithmic framework. Source: (YANG et al., 2023).

is employed to train the output of the policy network, represented as $\mathbf{a} = (a_1, a_2, a_3, a_4)$. The action space of the policy network output is confined within a narrow, symmetric range, typically $(-1, 1)$. This design choice enhances versatility by restricting action outputs within a specific range that can be adapted to the actuator's action range through scaling operations. Following this operation, the motor speed vector, $\mathbf{r} = (r_1, r_2, r_3, r_4)$, is determined.

The input to the neural network is a 12-dimensional state vector $\mathbf{s}$. It is expressed as:

$$\mathbf{s} = (y, z, u, v, w, \phi, \theta, \psi, p, q, r, i_t) \in \mathbb{R}^{12}$$

In this formulation, $\mathbf{P} = (y, z)$ represent position, $\mathbf{v} = (u, v, w)$ correspond to velocity, $\boldsymbol{\psi} = (\phi, \theta, \psi)$ denote the Euler angles, $\boldsymbol{\omega} = (p, q, r)$ is the angular velocity, and (i_t) represent the current rotor tilt angle. According to the authors, given the transition process's predominantly horizontal and forward nature, the unnormalized x-axis position significantly impacts estimation errors and training time. To mitigate these effects, their study excluded the x-axis displacement as a state input. Furthermore, they state that restricting the state input to only the pitch angle would necessitate the neural network to simultaneously control both motor output a and tilt angle i_t and that this would increase the dimension of the action space and render the learning of an optimal strategy challenging in the absence of expert data.

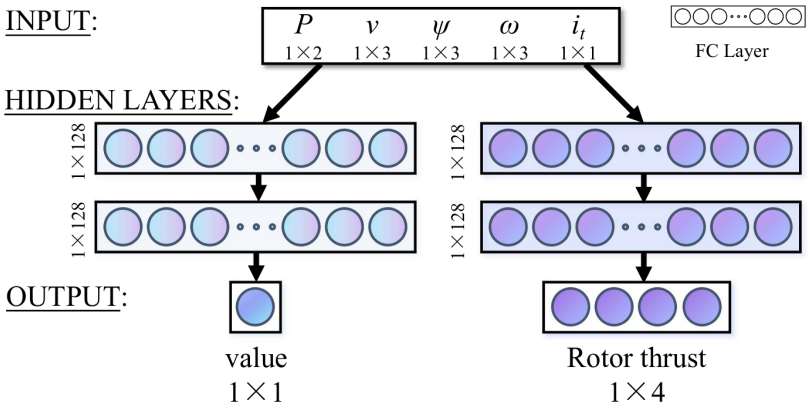


Figure 3.3 – Neural network structure diagram. Source: (YANG et al., 2023).

3.1.4 Reward Function

In the context of this work, the reward function r_t is designed to address the unique characteristics of the TRUAV transition process while taking into account various factors:

$$r_t = d + c_p \left\|\Delta \mathbf{P}_t\right\| + c_{vx} \times \min(vx_t, v_{\max}) + c_{vyz} \left\|\Delta \mathbf{v}_t\right\|$$

$$+ f(i_t) \times c_\Psi \left\|\Delta \mathbf{\Psi}_t\right\| + c_\omega \left\|\Delta \boldsymbol{\omega}_t\right\| + c_a \left\|\mathbf{a}_t\right\| + c_{da} \left\|\Delta \mathbf{a}_t\right\|$$

The reward function utilized in this study is designed to guide the TRUAV through its transition process, with a particular focus on ensuring flight stability, maintaining altitude, and achieving forward propulsion. It includes a baseline survival reward (d). These terms account for position and velocity deviations ($\Delta \mathbf{P}_t$ and $\Delta \mathbf{v}_t$), and components aimed at controlling forward velocity ($v_x t$) and mitigating extreme oscillations in the controller output ($c_{vx} \times \min(vx_t, v_{\max})$). Additionally, the reward function addresses tilt-angle variations during the transition with a variable coefficient ($f(i_t)$) and introduces a penalty term ($c_{da} \left\|\Delta \mathbf{a}_t\right\|$) to reduce output oscillations. The overarching strategy encourages controlled forward movement while maintaining flight stability, optimizing the TRUAV's performance during its transition phase.

3.1.5 Training

At the training phase, the TRUAV controller uses the PPO algorithm within a simulation environment developed based on the open-source project framework known as Flightmare (SONG et al., 2021). The agent interacts with the simulation environment to train the attitude controller using reinforcement learning algorithms. To simulate a fully loaded situation, the maximum thrust output of each motor is limited to 30% of the takeoff weight of the vehicle. The training objective is to complete the transition process within 4 seconds, equivalent to 200 simulation steps with a time step of 0.02 seconds. During this time, the TRUAV's attitude stability is maintained while the rotor tilt angle transitions from 0° to 90°. The training task involves attaining attitude control to complete the transition from copter to airplane mode, beginning from random initial states, as outlined in Table 3.1.

Table 3.1 – Initial State Random Ranges and Expectation Values

Parameter	**Range**
Initial Euler angle	[±30°, ±30°, ±30°]
Expected Euler angle	[0.0°, 0.0°, 0.0°]
Initial speed	[±1.0, ±1.0, ±1.0] m/s
Expected speed	[10.0, 0.0, 0.0] m/s
Initial position	[±1.0, 5.0 ± 1.0] m
Expected position	[0.0, 5.0] m

Subsequently, the well-trained neural network controller undergoes testing under various conditions, such as rotor tilt angle rate, rotor axis lengths, and takeoff weights, to assess its robustness and adaptability.

3.1.6 Results and Conclusions

The study assessed the PPO algorithm's performance in controlling TRUAV during the transition process. They refined the reward function to minimize controller output jitter and reduce attitude angle deviations at the end of transitions. Three simulations were conducted to test the adaptability of the RL controller in different conditions, including changes in takeoff weight, diagonal wheelbase, and rotor tilt rates. Furthermore, they compared their method with the Advantage Actor-Critic (A2C) and standard PPO algorithms to evaluate its performance.

The researchers found that the PPO algorithm effectively converged after 1500 iterations, ensuring TRUAV stability during transitions and reaching maximum rewards after 5000 iterations. They also demonstrated that the RL controller could adapt to various conditions without requiring parameter adjustments, such as changes in takeoff weight, diagonal wheelbase, and rotor tilt rates. Additionally, they emphasized how the enhanced reward function resulted in lower controller output variance and reduced attitude angle deviations.

Regarding performance, their method outperformed the A2C algorithm in attitude control during transitions and demonstrated more efficient energy consumption compared to the standard PPO.

As their conclusions, the researchers summarized that refining the reward function and employing reinforcement learning significantly improved attitude control during TRUAV transitions. They underscored the adaptability of the RL controller and its potential to enhance flight control. Nevertheless, they acknowledged the need for further research to address real-world complexities and uncertainties.

3.2 Deep Learning Model-Agnostic Controller for VTOL Class UAS (2022) (HOLMES et al., 2022)

The authors' primary objective in this paper is to demonstrate the performance and robustness of their model-agnostic controller against perturbations. Real flight test data is used to validate the controller's resilience in disturbances. The paper highlights the contributions, including developing an end-to-end automated system for generating model-agnostic controllers and evaluating the controller's model-agnostic property. The research showcases the potential for deep reinforcement learning in Unmanned Aerial Systems (UAS) control, offering a significant advancement in flight control, which traditionally required extensive manual tuning and controller retraining.

3.2.1 Aircraft Layout

The paper used two different VTOL (Vertical Takeoff and Landing) Unmanned Aerial Systems (UASs), the Eris UAS (Figure 3.4) and the Zephyrus UAS (Figure 3.5). These two aircraft have substantial differences in their physical properties and dynamics, with Zephyrus being heavier than Eris, and their wing airfoil and aspect ratio being fundamentally different. The paper aims to evaluate the controller's ability to adapt to these significant variations in physical and dynamic characteristics without retraining.

Figure 3.4 – Eris UAS. Source: (HOLMES et al., 2022).

3.2.2 Algorithm Framework

The framework employed in this work is centered around PPO. The authors chose PPO due to its computational efficiency, robustness to hyperparameters, trust region-based optimization, and ability to work well in on-policy reinforcement learning settings. According to them, these features make PPO suitable for their work and contribute to its success in training controllers for their specific tasks.

Figure 3.5 – Zephyrus UAS. Source: (HOLMES et al., 2022).

3.2.3 Network Architecture

The deep neural network controller at the core of this model-agnostic system acts as an open-loop system and functions by processing its state and command observations through a series of nonlinear layers to generate the final output for the control surfaces.

Regarding the network architecture, the authors specify a design that comprises three layers: an input layer with 12 neurons, two fully connected hidden layers, each consisting of 256 neurons, and an output layer with 4 neurons. They acknowledge the impact of architectural choices on computational complexity and the model's ability to generalize. They use the hyperbolic tangent (tanh) activation function to introduce nonlinearity, which is especially chosen for handling negative values within the network. While larger neural networks often employ the ReLU activation to mitigate vanishing gradient issues, their architecture, with only two hidden layers, doesn't require such extensive mitigation considerations.

In terms of inputs and pre-processing, the authors detail the raw observations provided to the controller, including information related to attitude, rates, velocity, and velocity errors. These inputs are pre-processed by normalizing them to the range [-1, 1] based on maximum expected values determined by termination condition thresholds. They emphasize that this normalization enhances the neural network's learning speed and convergence. Additionally, to reduce oscillations caused by non-zero velocity commands when near waypoints, the authors apply a smoothing function to inertial velocity commands, further optimizing the system.

The simulation environment in this study involves the VTOL agent and external factors influencing the agent's state and actions. The state vector of the aircraft comprises twelve variables represented as:

$$\mathbf{s} = (\phi, \theta, \psi, P, Q, R, N, E, D, V_N, V_E, V_D)^\top \in \mathbb{R}^{12}$$

Where ϕ, θ, and ψ denote roll, pitch, and yaw angles; P, Q, and R represent roll, pitch, and yaw rates; N, E, and D correspond to the north, east, and down position coordinates; and V_N, V_E, and V_D are the components of inertial velocity in the world frame. Additionally, $V_{N_{err}}$,

$V_{N_{err}}$, and $V_{N_{err}}$ indicate the differences between the commanded inertial velocity and the actual inertial velocity.

When it comes to outputs and post-processing, the authors explain that the controller provides four thrust values, each within the range (-1, 1), corresponding to the four quad motors. To enhance the description of these normalized motor thrusts, the authors transform these values to the range (0, 1) using a simple isomorphism: $f(x) = x + \frac{1}{2}$. This transformation completes the post-processing phase, preparing the controller for training and deployment.

3.2.4 Reward Function

The authors employed a reward engineering strategy to guide the training of their controllers. In this strategy, they mathematically defined how the actions taken by the agent in the environment should be rewarded or penalized. Specifically, for their application, the controller's actions needed to be rewarded when accurately tracking the commanded state and penalized for undesirable behavior such as oscillation and risky maneuvers.

The reward system was framed primarily in terms of costs rather than rewards. In this context, costs were represented as negative rewards. To do this, they defined a cost function (C) for a given aircraft state, controller output, and guidance commands, denoted as:

$$C = \mathbf{w} \cdot \mathbf{c}$$

Where $\mathbf{w}$ is a vector of weight components:

$$\mathbf{w} = (w_{\text{att}}, w_{\text{att rates}}, w_{\text{Verr}}, w_{\text{ctrl}}, w_{\text{term}})$$

The terms w_{att}, $w_{\text{att rates}}$, w_{Verr}, w_{ctrl}, and w_{term} represent weight components that are used to determine the relative importance of the cost components in the reward function. These weights specify how much each cost component contributes to the overall cost calculation. w_{att} influences the importance of the attitude cost, encouraging the aircraft to maintain a level position. In contrast, $w_{\text{att rates}}$ governs the significance of attitude rates cost, discouraging violent rotations. w_{Verr} balances the priority between stability and accurate tracking through the velocity error cost. w_{ctrl} guides the agent to predict motor values closer to trim values to maintain stability, and w_{term} severely penalizes actions leading to termination, promoting safer and longer-lasting behavior. These weights shape the agent's learning process and determine its behavior in response to different cost components.

The vector $\mathbf{c}$ is the vector of cost components:

$$\mathbf{c} = (c_{\text{att}}, c_{\text{att rates}}, c_{\text{Verr}}, c_{\text{ctrl}}, c_{\text{term}})$$

The terms c_{att}, $c_{\text{att rates}}$, c_{Verr}, c_{ctrl}, and c_{term} describe how the measured error maps to a cost value, and they play a critical role in determining the cost assigned to the agent's actions.

c_{att} represents the attitude cost, punishing non-zero attitude to encourage maintaining a level aircraft position. $c_{\text{att rates}}$ pertains to the attitude rates cost, penalizing non-zero attitude rates to discourage violent rotations. c_{Verr} is related to the velocity error cost, penalizing deviations from the commanded velocity and influencing the balance between stability and tracking accuracy. c_{ctrl} deals with the controller cost, penalizing deviations from motor trim values and encouraging stability and safe behavior. Finally, c_{term} is the termination cost, severely punishing actions that lead to the aircraft transitioning into a terminating state, particularly as the episode progresses. These cost components collectively determine how the agent's actions are rewarded or punished, ultimately guiding its learning process and behavior in the environment.

To represent how these cost components map to cost values, the authors used the squared Euclidean norm of their values, which is effective because it penalizes deviations regardless of their sign. The weights controlled the relative importance of each cost component.

In a post-processing step, the cost components were normalized to ensure they all fell within the $[0, 1]$ range. This normalization made it easier to interpret and tune the weights, as they now directly represented the contribution of each cost component to the total cost (C). The weights were initially assigned arbitrarily and adjusted iteratively based on the agent's performance in training runs. The final weight values for each cost component are shown in Table 3.2.

Table 3.2 – Cost Component Weights

Cost Component	Weight
c_{att}	0.25
$c_{attrates}$	0.1
c_{Verr}	0.25
c_{ctrl}	0.1
c_{term}	0.1

This reward engineering strategy played a vital role in training the controller to achieve the desired behavior in response to guidance commands while maintaining stability and safety throughout the flight.

3.2.5 Training

At the training stage, PPO is used to optimize the controller policy, and each simulation episode has a maximum time limit of 60 seconds of simulated real-time. Termination condition thresholds, linked to state variables, are established to identify excessive instability and trigger early episode termination during training, except for N, E, D, and ψ. These termination conditions prevent unnecessary evaluation of the neural network model and simulation environment when the aircraft's state has diverged beyond potential recovery. Table xx outlines the conditions that lead to early termination, where $V_{N_{err}}$, $V_{N_{err}}$, and $V_{N_{err}}$ indicate the differences between the commanded inertial velocity and the actual inertial velocity.

Table 3.3 – Termination Conditions

Variable	**Condition**
ϕ: roll angle	$\lvert\phi\rvert > 40$ deg
θ: pitch angle	$\lvert\theta\rvert > 40$ deg
P: roll rate	$\lvert P\rvert > 120$ (deg/s)
Q: pitch rate	$\lvert Q\rvert > 120$ (deg/s)
R: yaw rate	$\lvert R\rvert > 120$ (deg/s)
V_N: north position coordinate	$\lvert V_N\rvert > 3.048$ m/s
V_E: east position coordinate	$\lvert V_E\rvert > 3.048$ m/s
V_D: down position coordinate	$\lvert V_D\rvert > 3.048$ m/s
$V_{N_{\text{err}}}$ north commanded inertial velocity's error	$\lvert V_{N_{\text{err}}}\rvert > 3.048$ m/s
$V_{E_{\text{err}}}$ east commanded inertial velocity's error	$\lvert V_{E_{\text{err}}}\rvert > 3.048$ m/s
$V_{D_{\text{err}}}$ down commanded inertial velocity's error	$\lvert V_{D_{\text{err}}}\rvert > 3.048$ m/s

3.2.6 Results and Conclusions

The authors' results indicate the successful development of highly adaptable, model-agnostic VTOL controllers with strong performance and robustness across different aircraft and environmental conditions.

The training process converged after 3500 epochs and took approximately 11 hours. The controller exhibits strong correlations between training epochs and its ability to survive longer and obtain higher rewards. The test suite evaluates the controller's performance under various conditions, and the results show its adaptability and robustness to different scenarios, including wind perturbations and off-trim initial conditions. The controller's behavior on two different aircraft demonstrates its model-agnostic nature.

As a conclusion, the authors emphasize the key contribution of their work, which is the development of model-agnostic VTOL controllers capable of adapting to different aircraft without retraining, offering cost and labor-saving benefits. The study also highlights the robustness of the controllers through rigorous analysis and evaluation in simulation. Future work is suggested to explore other neural network architectures, extend the controller to mode-free guidance and control algorithms, and conduct real flight tests.

4 Materials and methods

This research investigates and compares control strategies for eVTOL aircraft using Reinforcement Learning. The primary focus is on enhancing control strategies through the application of Reinforcement Learning, with the PPO algorithm serving as the base approach due to its state-of-the-art status. In addition to PPO, this study includes a comparison with the TD3 algorithm to assess their effectiveness. Moreover, this study assesses the robustness of control strategies in the face of failures to explore resilience and adaptability in adverse scenarios

4.1 UAV Systems

This section provides a brief overview of the UAVs' physical designs used in this work.

Quadrotor-X; The Quadrotor-X is an aircraft with four control inputs, one to each rotor. The "X" configuration, as depicted in Fig. 4.1, has the rotors mounted in an "X" formation relative to what is considered the front of the aircraft.

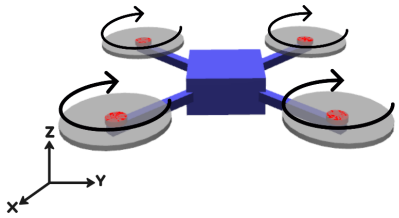


Figure 4.1 – Quadrotor-X.

Quadrotor-Plus: The Quadrotor-Plus is an aircraft with four control inputs, one to each rotor, as represented in Fig. 4.2. The "+" configuration are rotated 45° along the z-axis if compared to the "X" configuration.

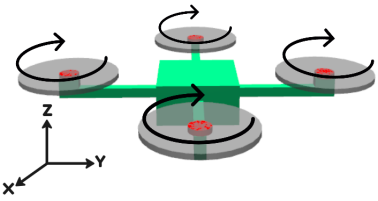


Figure 4.2 – Quadrotor-Plus.

Tiltrotor-Plus: The Tiltrotor-Plus shown in Figure 4.3, is an aircraft design featuring a "+" configuration, offering four additional control inputs compared to the Quadrotor-Plus. These additional inputs enable the manipulation of tilt-enabled rotors. In total, the Tiltrotor-Plus

utilizes eight control inputs, with four dedicated to rotor control and four for activating the tilting movement of each rotor. Consequently, the Tiltrotor-Plus aircraft leverages both rotor force magnitude and tilting servos to achieve the desired flight state.

(a) Neutrally positioned rotors (b) Tilted rotors

Figure 4.3 – Tiltrotor-Plus.

Tiltrotor-Parallel: The Tiltrotor-Parallel also have eight control inputs, four for the rotors and four for the tilting rotors movement, as in the Tiltrotor-Plus configuration. However, the tilting direction differs between them. While in the Tiltrotor-Plus the tiltrotors direction are rotated 90° around the z-axis, in the Tiltrotor-Parallel design, the tilting angles are aligned, and all of then can only rotate around the y-axis as illustrated in 4.4b. It also utilizes not only the rotor force magnitude, but also the tilting servos to achieve the desired state during flight, respecting the achievable tilting directions.

(a) Neutrally positioned rotors (b) Tilted rotors

Figure 4.4 – Tiltrotor-Parallel.

4.2 RL-Based Approach for UAV Control

The Reinforcement Learning approach for UAV control involves training an agent to complete a given task by learning the optimal behavior through repeated trial-and-error interactions with the environment without human involvement.

An agent is made of a policy and a learning algorithm. The agent receives a reward for each action when interacting with the environment. Its goal is to find an optimal policy that maximizes the cumulative reward received during task completion.

The policy component is a mapping that selects actions based on the observations received from the environment, typically a function approximator with adjustable parameters

such as a deep neural network. On the other hand, the learning algorithm is responsible for continuously updating the policy parameters by evaluating how successful the action chosen is concerning completing the goal task.

4.2.1 Policy Optimization Methods

In RL, when optimizing a policy, the choice of function approximator is crucial. A function approximator method views reinforcement learning as a numerical optimization problem that aims to optimize the expected reward concerning the policy's parameters. There are various methods for optimizing a policy, and in this work, PPO (SCHULMAN et al., 2017) and TD3 (FUJIMOTO; HOOF; MEGER, 2018) methods will be utilized.

PPO employs a stochastic policy, enabling automatic execution of the exploration phase. It entails collecting a mini-batch of experiences while interacting with the environment and using it to update the policy. Since the policy is updated, a newer batch is collected with the updated policy, characterizing it as an on-policy algorithm.

Different from PPO, TD3 uses a deterministic policy, which implies that the exploration is handled manually, in contrast to a stochastic policy, where the exploration is done automatically. Additionally, it is an off-policy algorithm since the function used to collect experiences differs from the function updated for learning. TD3 policy gradient optimization method is designed to operate on potentially large continuous state and action spaces with a deterministic policy, that is, the policy function (π) directly outputs an action, as opposed to a stochastic policy, where the output is a probability distribution over actions. Also, TD3 is designed to be less sensitive to hyper-parameters tuning by minimizing the overestimation bias maintaining a pair of critics $Q1$ and $Q2$ (hence the name "twin") along with a single actor. In this process, TD3 uses the smaller of the two critic values and updates the policy less frequently than the critic networks for each time step.

To systematically evaluate the effectiveness of different reinforcement learning algorithms and network configurations for UAV control, this study conducts experiments using both PPO and TD3 methods. The goal is to compare the performance of PPO and TD3 algorithms while increasing the geometry complexity of aircraft. For each type of UAV aircraft, namely Quadrotor-X, Quadrotor-Plus, Tiltrotor-Plus, and Tiltrotor-Parallel, experiments are conducted with PPO and TD3 configurations as depicted in Table 4.1.

This comparative analysis is expected to shed light on the interplay between algorithm choice, geometry complexity, and overall performance in enhancing control strategies for eVTOL aircraft.

	Algorithm	Aircraft Type
Policy Optimization Config. 1	PPO	Quadrotor-X Quadrotor-Plus Tiltrotor-Plus Tiltrotor-Parallel
Policy Optimization Config. 2	TD3	Quadrotor-X Quadrotor-Plus Tiltrotor-Plus Tiltrotor-Parallel

Table 4.1 – Combinations of policy optimization algorithms and aircraft geometries

4.2.2 Policy Architecture

In the field of Reinforcement Learning, a policy architecture refers to the design and structure of a neural network for modeling the policy function since the neural network can capture complex patterns and make predictions based on input data. The policy architecture typically consists of several layers of interconnected neurons, each performing specific computations on the input data. The neural network's input layer receives the state or observation from the environment as input. It could be raw sensor data or a pre-processed representation of the environment state, and the output layer produces the action probabilities or action values based on the processed information from the hidden layers.

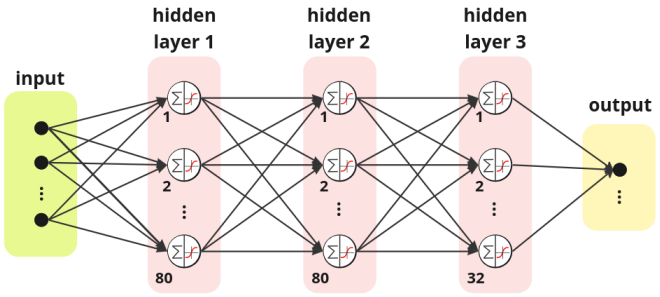
In this work, the input layer receives data input corresponding to the position and velocity errors in the UAV system, and the output layer outputs the rotors' inputs. For the PPO algorithm, the output layer outputs a probability distribution over the action space, while in TD3, the output layer directly estimates the action values for each possible action.

Policy training is done using the actor-critic method. The actor proposes a set of possible actions given a state, while the critic evaluates actions taken by the actor based on the given policy.

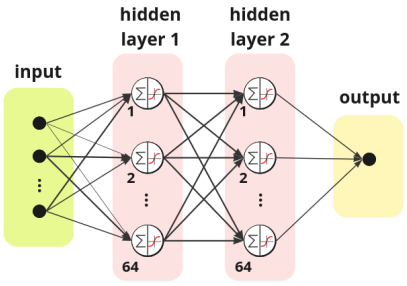
The architectures for the actor and critic neural networks are shown in Fig. 4.5, with Fig. 4.5a representing the actor neural network and Fig. 4.5b illustrating the critic neural network. The number of inputs and outputs varies depending on the aircraft type and complexity. For example, refer specifically to Fig. 4.6, which provides a detailed actor-critic architecture for the tiltrotor aircraft. Both the actor and critic networks employ the sigmoid activation function, and the Mean Squared Error (MSE) serves as the loss function for both networks. These neural network architectures were selected based on previous literature (DESHPANDE et al., 2020) and were not optimized in terms of the number of layers, neurons, or activation functions.

4.2.3 Observation Space

The Observation Space represents the set of all possible observations that the UAV agent can perceive from the environment and serves as input to the UAV agent's decision-making

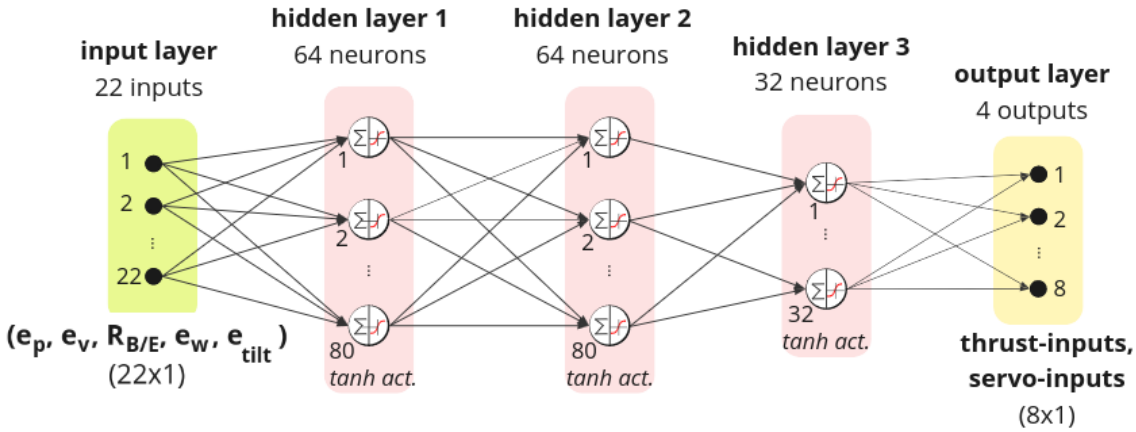


(a) Actor neural network architecture with three fully connected hidden layers with 80, 80 and 32 neurons respectively

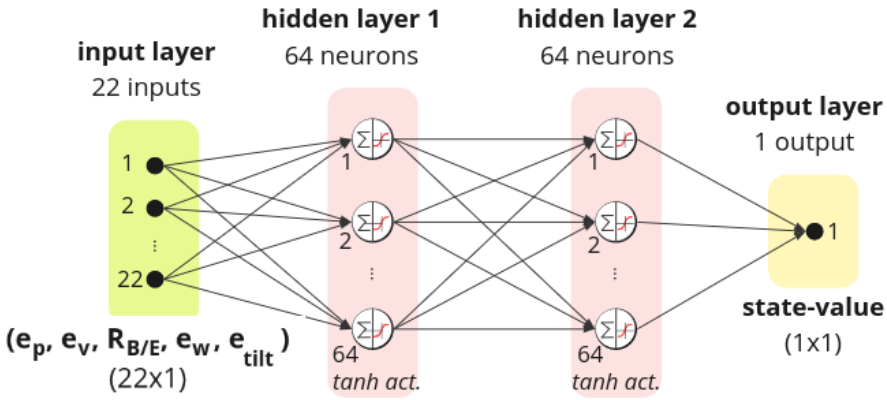


(b) Critic neural network architecture with two fully connected hidden layers with 64 neurons each

Figure 4.5 – Actor (a) and Critic (b) neural networks to be used as actor and critic configurations.



(a) Tiltrotor actor network



(b) Tiltrotor critic network

Figure 4.6 – Actor (a) and critic (b) neural networks for the Tiltrotor-Parallel UAV.

process, allowing it to estimate or infer the underlying system state based on the available information of its sensors. In this work, the observations include the information on the errors that the UAV can sense at each time step relative to the system's desired state. It is given by $Observation = (e_p, e_v, R_{B/E})$, where $e_p \in R^3$ is the error in desired position, $e_v \in R^3$ is the error in desired velocity, $e_\Omega \in R^3$ is the error in body rates, $R_{B/E}$ is the flattened 3×3 rotational

matrix ($R_{B/E}$ converted to a 9-dimensional vector by flattening), and $e_{tilt} \in R^4$ is the error in rotors tilt angles.

The Quadcopters' observation space does not include the tilt angles errors and, thus, are 18 dimensional. The Tiltrotors systems are 22 dimensional because they include errors in tiltrotor angles, represented by e_{tilt}.

4.2.4 Action Space

The action space represents the set of all possible actions that the UAV agent can take in the environment. It defines the choices or decisions it can make at each time step to interact with the environment and influence its future state. In this work, the dimension of the action space of the systems under consideration is equal to the number of their active actuators. The action space for the Quadcopters is $a \in R^4$ corresponding to their four motors. In the case of Tiltrotors systems, the action space is $a \in R^8$ given that they have four additional servos for tilting.

4.2.5 Reward Function

The Reward Function is the mathematical function that quantifies the quality of the UAV agent's actions within the environment, providing a measure of desirability to guide the UAV agent's decision-making and learning toward achieving the desired waypoint. In this work, the reward r_t earned by the UAVs during each time step is:

$$r_t = \beta - \alpha_a \|a\|_2 - \sum_{k \in \{p,v,\omega\}} a_k \|e_k\|_2 - \sum_{j \in \{\phi,\theta\}} a_j \|e_j\|_2 - \alpha_{tilt} \|e_{tilt}\|_2, \tag{4.1}$$

where $\beta > 0$ is a bonus value earned by the agent for staying alive in the simulation and $\alpha_{\{.\}} \geq 0$ represent a weight for terms in the reward function. The second term represents the penalty for the robot's actions and the two last terms are the summation of penalties imposed due to errors in the state of the robot, including the error in position (e_p), the error in velocity (e_v), the error in body rates (e_ω) of the system, the error in roll (e_ϕ) and the error in pitch (e_θ) angles.

4.3 Simulation Setup for UAV Training

This section details information about the experimental configuration and parameters setting for training the UAVs.

4.3.1 Physics Simulation Environment

The training of the UAVs using Reinforcement Learning uses the MuJoCo (Multi-Joint dynamics with Contact) Physics Engine (TODOROV; EREZ; TASSA, 2012) and OpenAi Gym (BROCKMAN et al., 2016). MuJoCo is used for simulating the UAV's dynamics, providing

the environment where they can learn to control its actions, and the OpenAi Gym provides the interface for developing and evaluating the Reinforcement Learning algorithms.

The implementation code, which includes functions for initializing an environment, resetting its state, taking actions, and obtaining observations, rewards, and termination signals, was available in previous work (DESHPANDE et al., 2020). The UAVs' physical parameters are detailed in Table 4.2. These parameters were chosen rather than optimized in the reference work (DESHPANDE et al., 2020). The weighting coefficients used in the reward function are $\alpha_p = 1.0$, $\alpha_v = 0.05$, $\alpha_\omega = 0.25$, $(\alpha_\phi, \alpha_\theta) = (0.1, 0.1)$, $\beta = 5.0$, $\alpha_{tilt} = 0.5$, and $\alpha_a = 0.25$.

Table 4.2 – UAV physics model parameters

Parameter	**Value**
Timestep	0.01 sec
Motor lag	0.05 sec
Mass m	1.5 kg
Arm length l	0.13 m
Tilt angle range $[\theta_{\min}, \theta_{\max}]$	$[-\frac{\pi}{2}, +\frac{\pi}{2}]$ rad
Motor thrust range $[F_{\min}, F_{\max}]$	[0, 15.0] N

4.3.2 Simulation Setup for training using RL Algorithms

The UAV's training aims to achieve the desired position set as the coordinates (0m, 0m, 3m) in the global coordinate system, that is, 3 meters in height.

When starting a new episode, the UAV's initial position is sampled uniformly from a cube spanning 2m around the goal coordinates, and the initial linear and angular velocities are sampled from a uniform distribution in the range of [-1, 1]. The termination of each episode is established if the episode length exceeds the maximum number of time-steps set in the simulation or if the UAV flies beyond a cube of 3m around the target position. For the UAV training using PPO, the actions are sampled from the policy's outputs, with a constant diagonal covariance matrix with elements of equal magnitude σ^2. PPO training parameters are: i) training iterations 2 $\times$ 10^6; ii) α 5e-5; iii) γ 0.99; iv) Max. episode length 1000; v) clipping parameter for PPO 0.05; vi) λ 0.95; vii) epochs per PPO update - 10; viii) σ^2 1.0. The weights are randomly initialized, and both input and output data are normalized. Additionally, mini-batch normalization is applied before the hidden layers.

In UAV training using TD3, on the other hand, the actions are directly chosen from the policy's outputs, given that TD3 is a deterministic policy. TD3 training parameters are: i) training iterations = 2 $\times$ $= 10^6$; ii) $\alpha = 5e-5$; iii) $\gamma = 0.99$; iv) Max. episode length 1000; v) policy

delay=2; vi) $\tau = 0.005$

4.3.3 Reward Function Parameters

The detailed values used in the reward equation 4.1 for the reference training setup are: i) Bonus for reaching the goal = 15.0; ii) Max. reward for velocity towards goal $= 2.0$; iii) Constant Position $= 5.0$; iv) Constant orientation $= 0.02$; v) Constant linear velocity $= 0.01$; vi) Constant angular velocity $= 0.001$; vii) Constant action $= 0.0025$; ix) reward for staying alive $= 5.0$; x) reward scaling coefficient $= 1.0$; xi) Constant tilt position $= 1.0$. TD3 is also trained with an increased parameter "constant orientation" value, changing its value from 0.02 to 0.5.

4.4 Failure Analysis and Robustness Testing

In this section, an analysis of failures and robustness testing procedures for developing and evaluating eVTOL control strategies is undertaken. The focus is on understanding how the control strategies perform under adverse conditions and assessing their robustness in real-world scenarios. The robustness of UAV control policy trained using the PPO algorithm is assessed in the face of rotor failures.

The failure analysis encompasses simulated scenarios where rotors of the eVTOL system experience malfunctions or disruptions. Specifically, failures such as uncontrollable rotors for specific fractions of time, single rotor failures, and combinations of multiple rotor failures are simulated. These failure scenarios are designed to mimic real-life contingencies and test the ability of the control strategies to adapt and maintain stability.

To systematically evaluate the robustness to failures, focus is placed on two UAV configurations: the lower complexity Quadrotor-X and the higher complexity Tiltrotor-Parallel. These configurations represent distinct geometries, which may influence the response to failure scenarios. The experimental configurations are compiled in Tables 4.3 and 4.4 for Quadrotor-X and Tiltrotor-Parallel, respectively.

This systematic analysis is expected to shed light on the performance in the presence of failures.

Rotor Failure Configuration	Thrust-Rotor	Fraction of time
X-1.a	1 thrust-rotor	10%
X-1.b	1 thrust-rotor	30%
X-1.c	1 thrust-rotor	50%
X-2.a	2 adjacent thrust-rotors	10%
X-2.b	2 adjacent thrust-rotors	30%
X-2.c	2 adjacent thrust-rotors	50%
X-3.a	2 opposite thrust-rotors	10%
X-3.b	2 opposite thrust-rotors	30%
X-3.c	2 opposite thrust-rotors	50%

Table 4.3 – Combinations of thrust-rotor failure conditions for Quadrotor-X UAV

Rotor Failure Configuration	Thrust-Rotor	Fraction of time
T-1.a	1 thrust-rotor	10%
T-1.b	1 thrust-rotor	30%
T-1.c	1 thrust-rotor	50%
T-2.a	2 adjacent thrust-rotors	10%
T-2.b	2 adjacent thrust-rotors	30%
T-2.c	2 adjacent thrust-rotors	50%
T-3.a	2 opposite thrust-rotors	10%
T-3.b	2 opposite thrust-rotors	30%
T-3.c	2 opposite thrust-rotors	50%
Tilt-1.b	1 thrust-rotor + tiltrotor	30%
Tilt-2.b	2 adjacent thrust-rotors + tiltrotors	30%
Tilt-3.b	2 opposite thrust-rotors + tiltrotors	30%

Table 4.4 – Combinations of thrust-rotors and tiltrotors failure conditions for Tiltrotor-Parallel UAV

5 Results and Discussion

This chapter presents the results of the investigation into learning-based approaches for controlling UAVs equipped with fixed-rotors and tiltrotors. The primary focus is on evaluating the performance of the PPO method and comparing it with the TD3 algorithm across UAVs with varying design complexities. Furthermore, failure tests are conducted for the Quadrotor-X and Tiltrotor-Parallel UAVs configurations, representing the lowest and highest complexity geometries, respectively, trained with PPO. These tests simulate scenarios such as single rotor failures and combinations of multiple rotor failures.

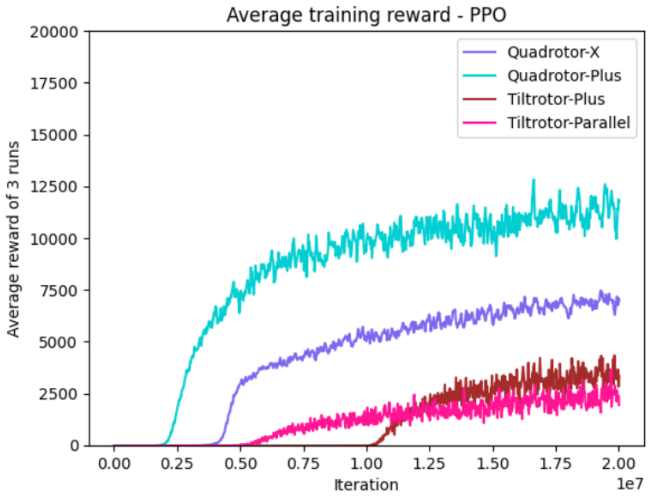
The results presented offer valuable insights into the effectiveness of different control strategies under various conditions and also shed light on their adaptability and robustness under failure scenarios.

5.1 Robustness of Policy Optimization Algorithms as Aircraft Geometry Complexity Increases

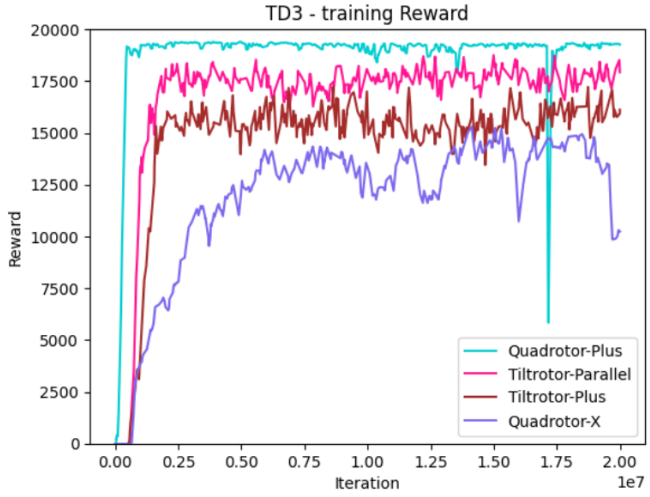
5.1.1 Algorithm Performance Results

Reward Values Progression during the Training Stage: For PPO, the model is trained three times with different initial seeds to avoid convergence to sub-optimal solutions due to the stochastic nature of the training process and the potential for getting stuck in local optima. The average training reward is depicted in Figure 5.1(a). Figure 5.1(b) displays the training reward using the TD3 algorithm and the reference reward parameters. Conversely, figure 5.1(c) illustrates the training using TD3 with the *Constant orientation* increased from 0.02 to 0.5. The training was conducted on a GeForce RTX 4060 Ti GPU with 8GB of memory, and it took approximately 3 days per training session.

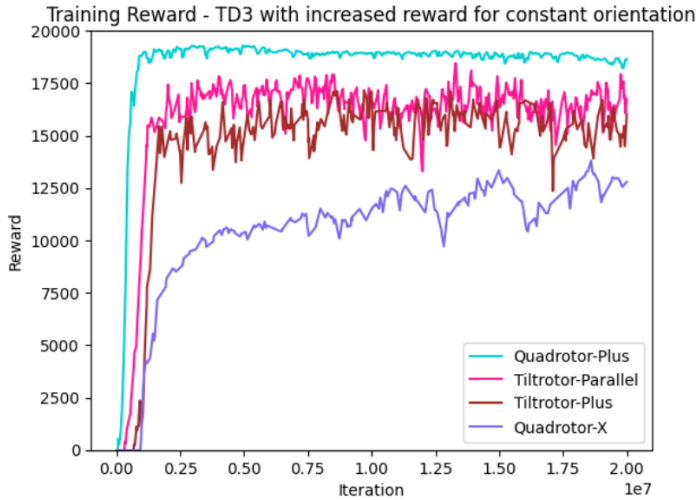
Performance of Trained UAVs: To demonstrate the performance of the trained UAVs in achieving a goal position (0m, 0m, 3m), the agent was initiated from a position sampled uniformly from a cube spanning 2m around the goal coordinates. Figure 5.2 illustrates the linear and angular displacement of the UAVs trained using the PPO algorithm. Conversely, Figures 5.3 and 5.4 depict the linear and angular displacement of the UAVs trained using the TD3 algorithm, where Figure 5.3 presents the UAV trained using TD3, and Figure 5.4 displays the UAV trained using the TD3 algorithm with the *Constant orientation* parameter increased from 0.02 to 0.5.



(a) Reward Values Progression during Training using PPO.



(b) Reward Values during Training using TD3



(c) Reward Values during Training using TD3 with increased reward for constant orientation

Figure 5.1 – Comparison of Reward Values Progression during the Training Stage.

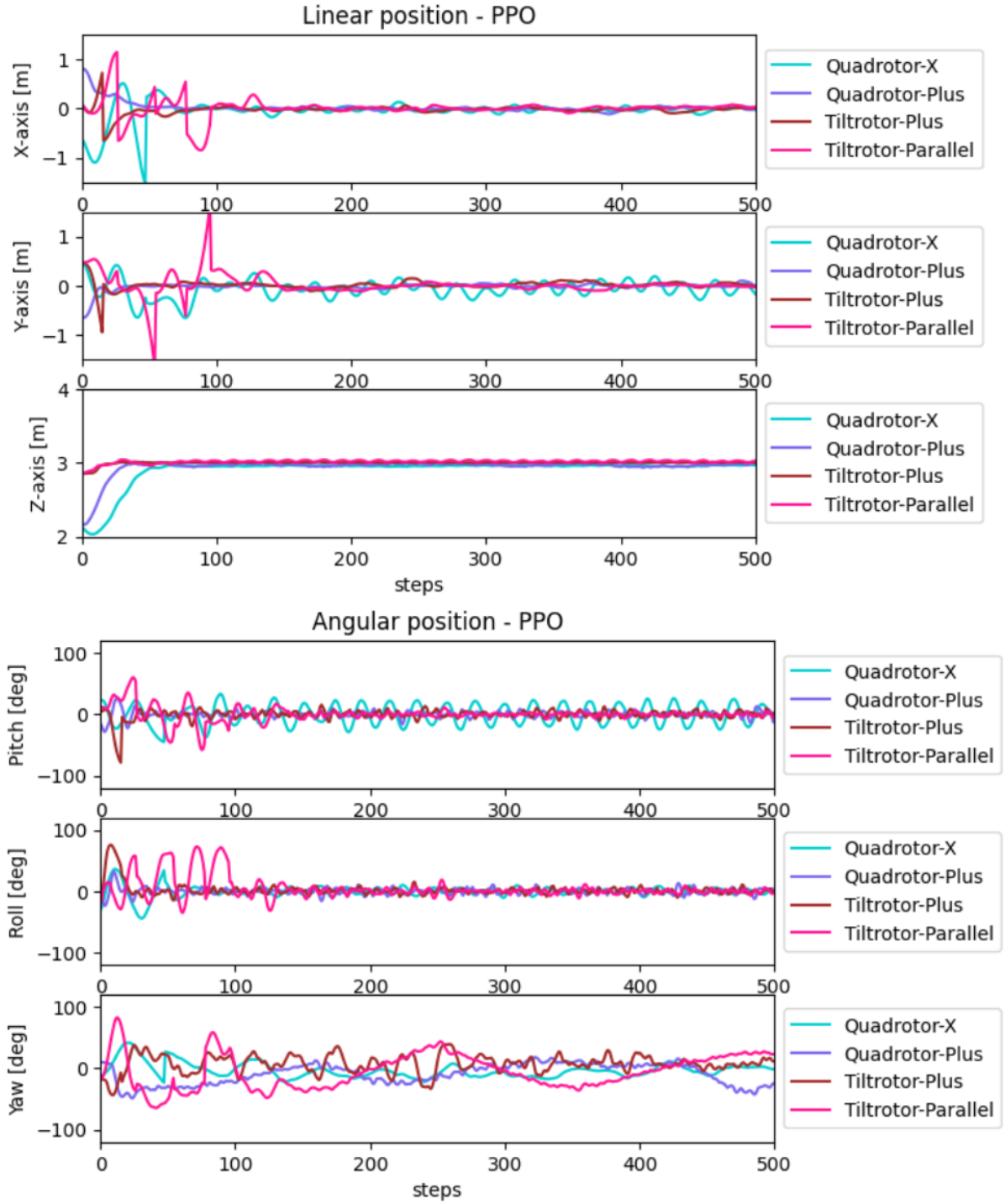


Figure 5.2 – UAVs' linear and angular displacement reaching the goal position, trained using PPO.

5.1.2 Discussion on Algorithms Performances and Behaviors

During the training process, the proposed TD3 algorithms (Figure 5.1(b) and Figure 5.1(c)) achieved higher reward values than PPO (5.1(a)), for all UAVs systems. A possible reason for those higher reward values is that TD3 uses deterministic policies, meaning it directly outputs the best action given to a state. This can lead to more precise actions and faster convergence towards optimal policies compared to stochastic policies used in PPO. Additionally, TD3 main-

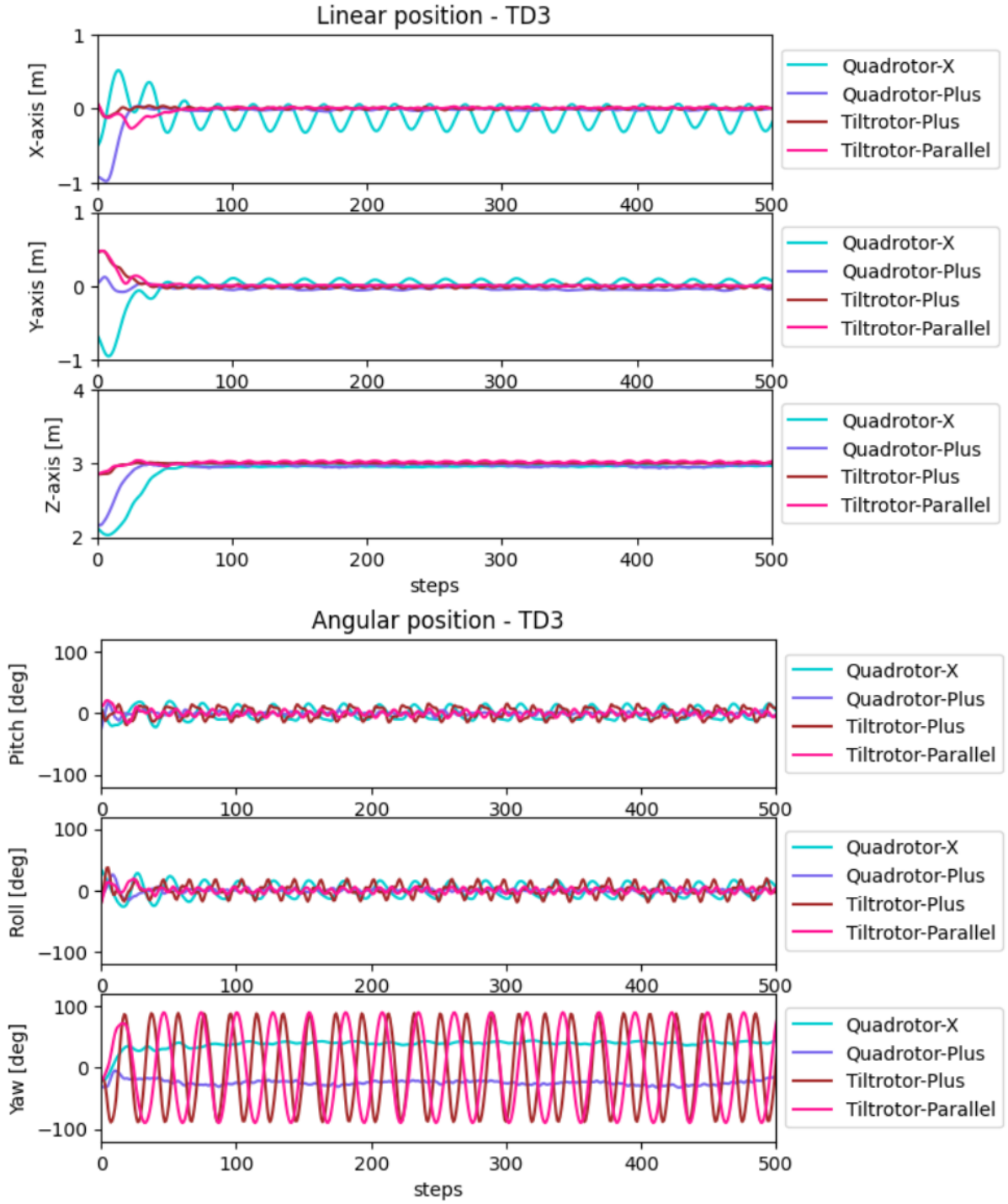


Figure 5.3 – UAVs' linear and angular displacement reaching the goal position, trained using TD3.

tains twin value functions to reduce overestimation bias, further improving the accuracy of value estimates. These factors can contribute to TD3's ability to achieve higher reward values.

Once trained, all the UAVs are able to reach the desired position (0, 0, 3m), as demonstrated by Figures 5.2, 5.3, and 5.4.

For the PPO displacement results, a comparison of algorithm behavior across geometries

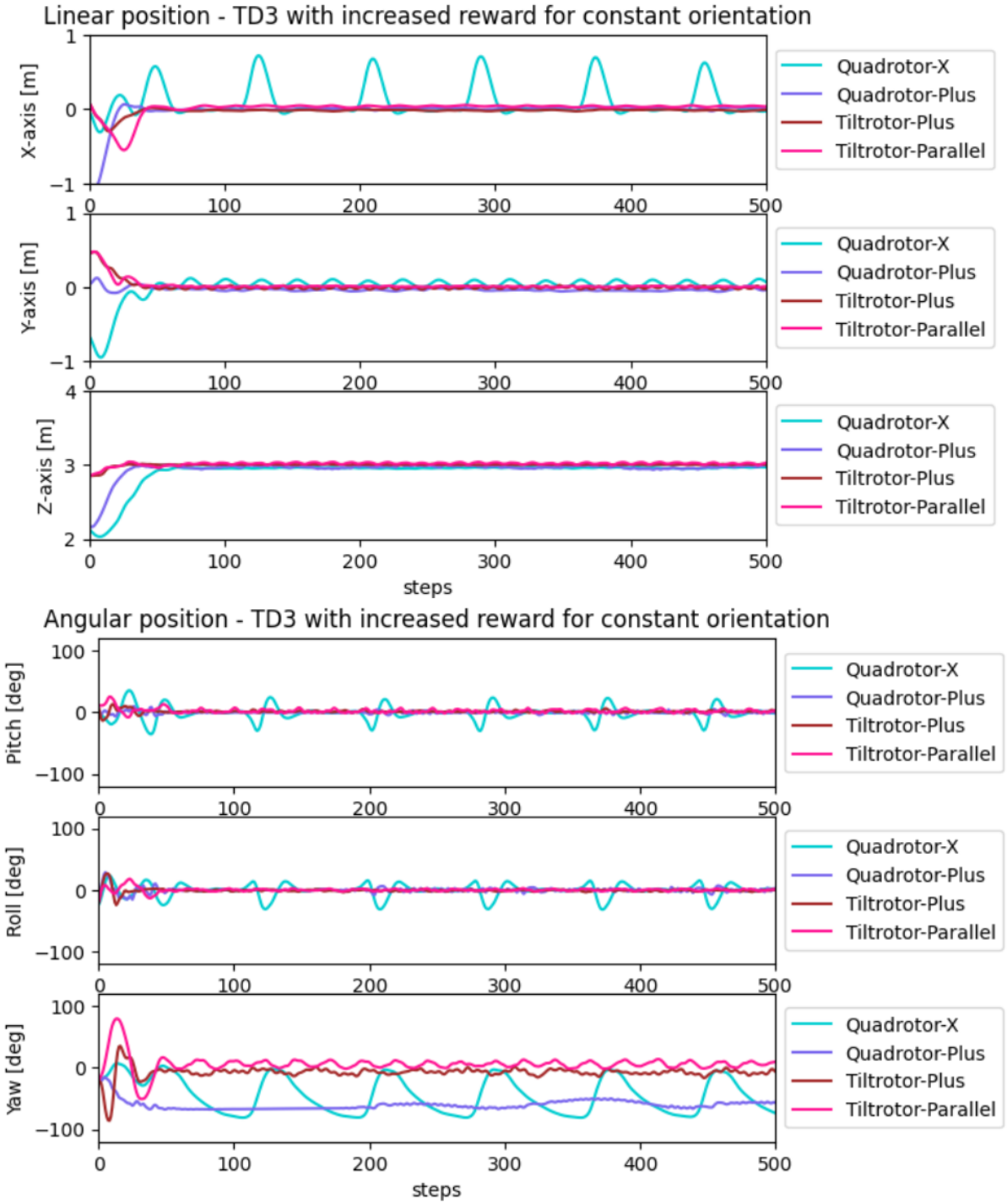


Figure 5.4 – UAVs' linear and angular displacement reaching the goal position, trained using TD3 with increased reward for UAV constant orientation.

reveals that the Tiltrotor-Plus exhibited the fastest convergence to stability, even outperforming the similar Quadrotor-X. This difference suggests that the misalignment of the Quadrotor-X's rotors axis with the Cartesian axis influences control, making it more challenging. Conversely, the Tiltrotor-Parallel required the longest time to reach stability, which aligns with expectations due to its higher geometric complexity and DOFs. Analysis of angular displacement indicates a residual small back-and-forth rotating movement for all geometries. Specifically, for the Yaw axis,

Tiltrotor-Parallel exhibited longer oscillation periods compared to Quadrotors, which oscillated more rapidly. This Quadrotors' behavior could be attributed to their lower number of DOFs, necessitating faster movement to maintain stability. In general the angular displacement are more prevalent in simpler designs.

A comparison of the displacements of the UAVs trained using PPO (Fig. 5.2) and those trained using TD3 (Fig. 5.3), under the same reward parameters, reveals that the TD3 UAVs exhibit faster convergence in linear displacement, albeit with some instability observed in the X-axis for Quadrotor-X. Regarding angular displacement, however, UAVs equipped with tiltrotors do not stabilize in the yaw direction. Instead, they continue rotating around the z-axis. This behavior stems from the reward parameters, which prioritize increasing reward through rotation rather than stabilization.

The increase in the reward value for constant orientation from 0.02 to 0.5, a change determined empirically, addressed the instability problem, as demonstrated in Figure 5.4. This adjustment improved the balance of rewards for maintaining orientation, resulting in enhanced stability. However, it is observed that this modification led to a slight deterioration in the stability of Quadrotor-X. This outcome can be attributed to the limited degrees of freedom of Quadrotor-X, making it more challenging to maintain constant orientation under the revised reward scheme.

5.2 Robustness Analysis under Failure Conditions

The experiments focus on inducing failures in UAVs to evaluate their performance under challenging conditions. Specifically, the UAVs, which were trained without exposure to failures, were subjected to induced rotor failures to evaluate their ability to reach the goal position across diverse failure scenarios and assess their generalization abilities. Each experiment begins with the agents initialized from positions uniformly sampled within a 2-meter radius around the goal coordinates and is replicated 10 times for robust statistical analysis. Failures are induced for varying durations, targeting different thrust-rotors and tiltrotors to ensure a comprehensive assessment of their performance and generalization capabilities.

5.2.1 Failures Scenarios Results

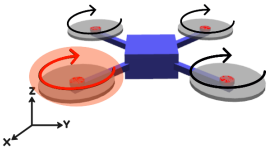
Failure affecting a single rotor: Failures were induced in individual rotors of both UAV configurations and evaluating their responses. Figures 5.5a and 5.5b illustrate the configuration of the UAVs with one rotor under failure for Quadrotor-X and Tiltrotor-Parallel, respectively. Figures 5.5c, 5.5e, and 5.5g depict the displacement trajectories of the Quadrotor-UAV when subjected to rotor failures occurring 10%, 30%, and 50% of the time, respectively, and figures 5.5d, 5.5f, and 5.5h depict the displacement trajectories of the Tiltrotor-Parallel UAVs when subjected to the same failure configurations.

Failure affecting two adjacent rotors: Failures were also induced simultaneously in two rotors for both UAV configurations, and their responses were evaluated. Figures 5.6a and 5.6b illustrate the configuration of the UAVs with two adjacent rotors failing for Quadrotor-X and Tiltrotor-parallel, respectively. Figures 5.6c, 5.6e, and 5.6g depict the displacement trajectories of the Quadrotor-UAV when subjected to rotor failures occurring 10%, 30%, and 50% of the time, respectively. Similarly, figures 5.6d, 5.6f, and 5.6h depict the displacement trajectories of the Tiltrotor-Parallel UAVs when subjected to the same failure configurations.

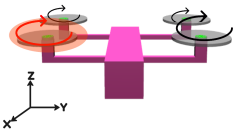
Failure affecting two opposite rotors: Eventually, failures were induced simultaneously in two opposite rotors for both UAV configurations, and their responses were examined. Figures 5.7a and 5.7b illustrate the configuration of the UAVs with two opposite rotors failing for Quadrotor-X and Tiltrotor-parallel, respectively. Figures 5.7c, 5.7e, and 5.7g show the displacement trajectories of the Quadrotor-UAV when subjected to rotor failures occurring 10%, 30%, and 50% of the time, respectively. Similarly, figures 5.7d, 5.7f, and 5.7h display the displacement trajectories of the Tiltrotor-Parallel UAVs when subjected to the same failure configurations.

Failure affecting tiltrotors: Multiple configurations of Tiltrotor-UAV failures were induced to evaluate their responses. Figure 5.8 illustrates the configuration of the Tiltrotor-UAV with one tiltrotor under failure 5.8a, with two adjacent tiltrotors under failure 5.8c, and with two opposite tiltrotors under failure 5.8e. Figure 5.8b, 5.8d and 5.8fdepicts the displacement trajectories of the Tiltrotor-Parallel when subjected to these tiltrotor failures, occurring 30% of the time.

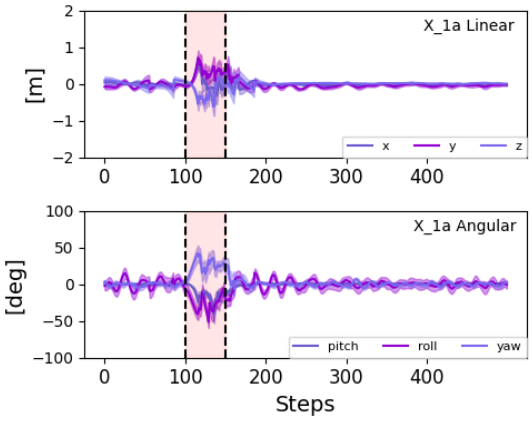
UAV Rotors and Tiltrotors behavior under failure scenarios: The behavior of rotors during UAV failure scenarios was investigated for both UAV configurations when subjected to rotor failures occurring 30% of the time. Figures 5.9a, 5.9c, and 5.9e illustrate the power required from the rotors of the Quadrotor-X when failures occur in one rotor, two adjacent rotors, and two opposite rotors, respectively. Similarly, Figures 5.9b, 5.9d, and 5.9f depict the power required from the rotors of the Tiltrotor-Parallel when failures occur in one rotor, two adjacent rotors, and two opposite rotors, respectively. The behavior of tiltrotors was also evaluated for the Tiltrotor-UAV configuration when subjected to tiltrotor failures occurring 30% of the time. Figure 5.10 shows the power required from the tiltrotors when failures occurs.



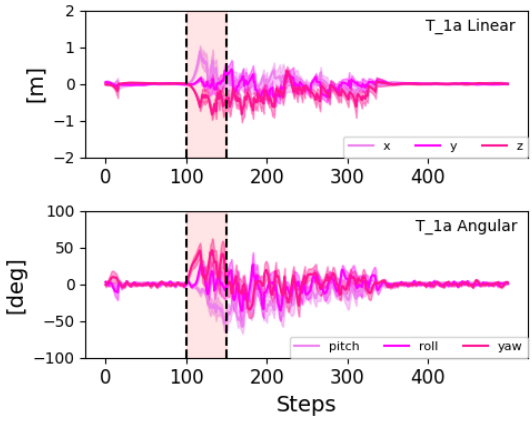
(a) Quadrotor-X with 1 rotor under failure



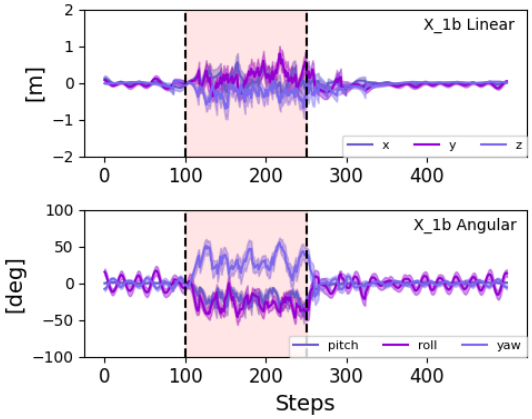
(b) Tiltrotor-parallel with 1 rotor under failure



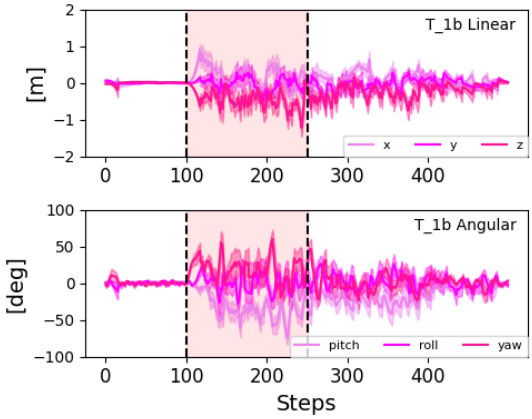
(c) Quadrotor under failure 10% of the time



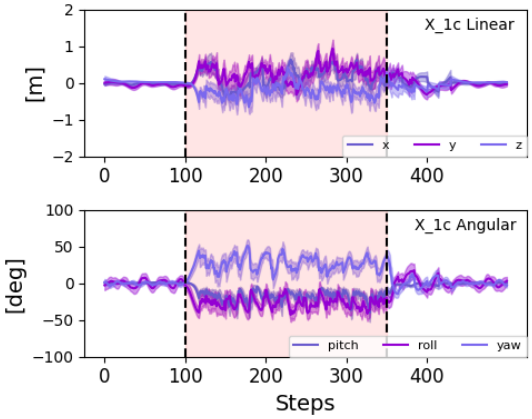
(d) Tiltrotor under failure 10% of the time



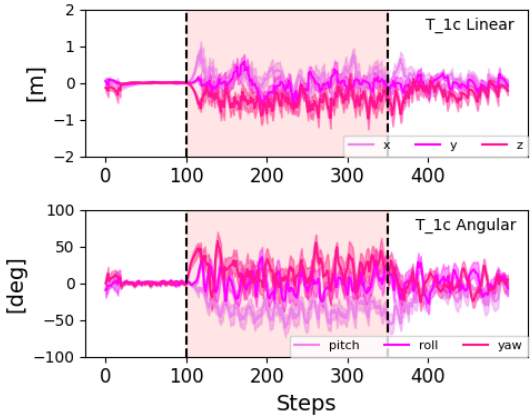
(e) Quadrotor under failure 30% of the time



(f) Tiltrotor under failure 30% of the time

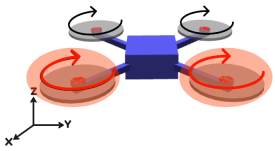


(g) Quadrotor under failure 50% of the time

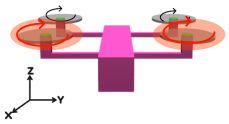


(h) Tiltrotor under failure 50% of the time

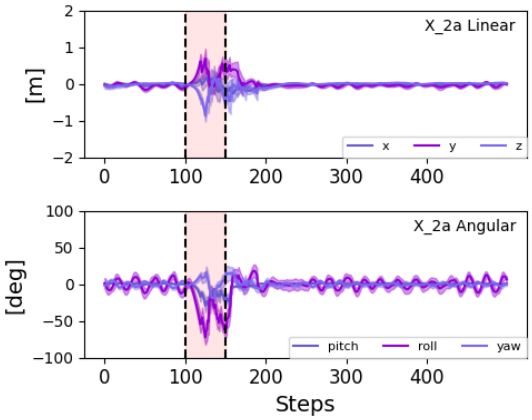
Figure 5.5 – Comparison of various scenarios with 1 rotor under failure for Quadrotor-X and Tiltrotor-Parallel



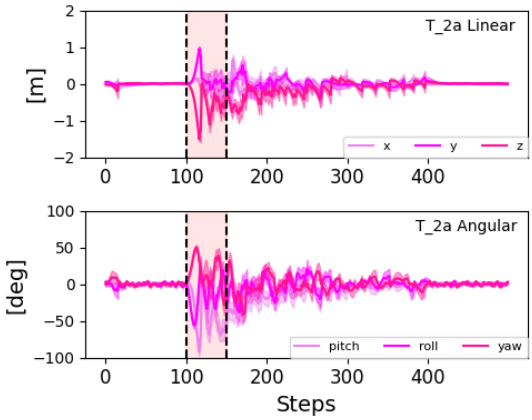
(a) Quadrotor-X with 2 adjacent rotors under failure



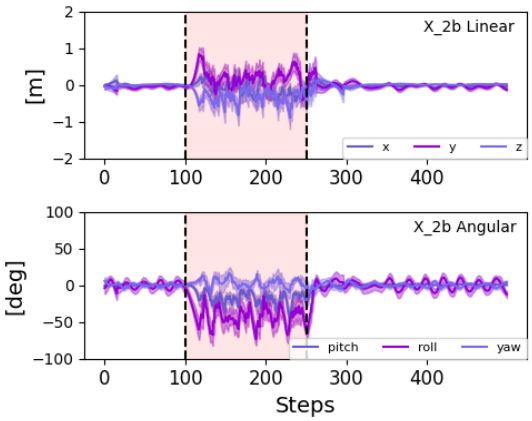
(b) Tiltrotor-Parallel with 2 adjacent rotors under failure



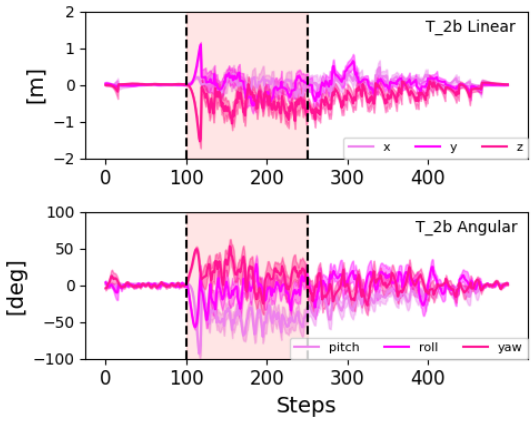
(c) Quadrotor under failure 10% of the time



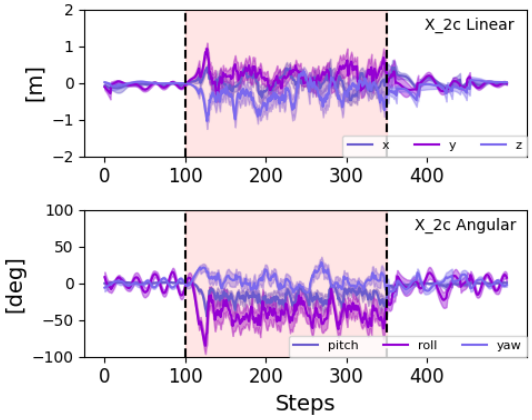
(d) Tiltrotor under failure 10% of the time



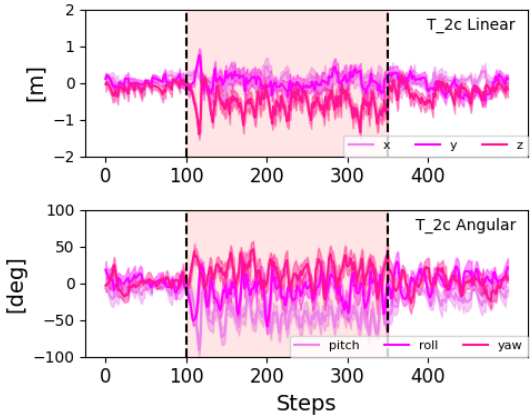
(e) Quadrotor under failure 30% of the time



(f) Tiltrotor under failure 30% of the time

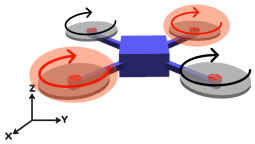


(g) Quadrotor under failure 50% of the time

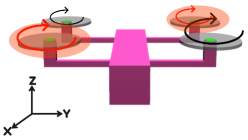


(h) Tiltrotor under failure 50% of the time

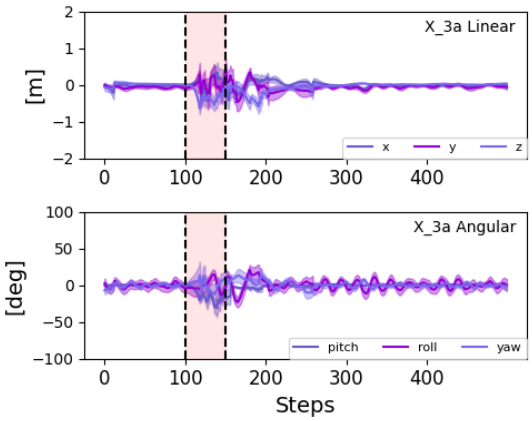
Figure 5.6 – Comparison of various scenarios with 2 adjacent rotors under failure for Quadrotor-X and Tiltrotor-Parallel



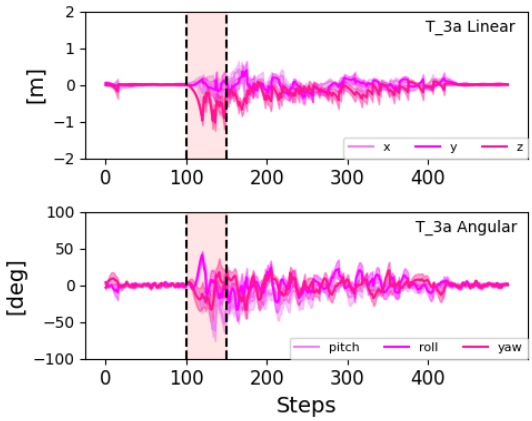
(a) Quadrotor-X with 2 opposite rotors under failure



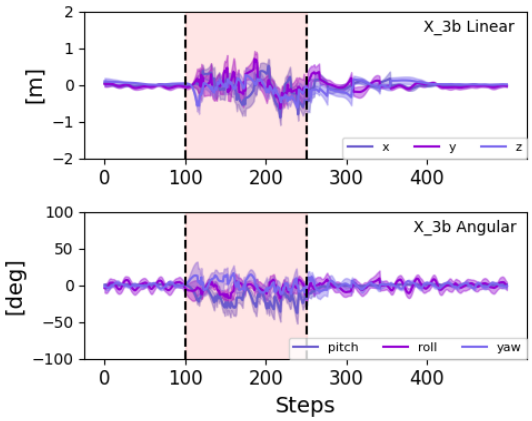
(b) Tiltrotor-Parallel with 2 opposite rotors under failure



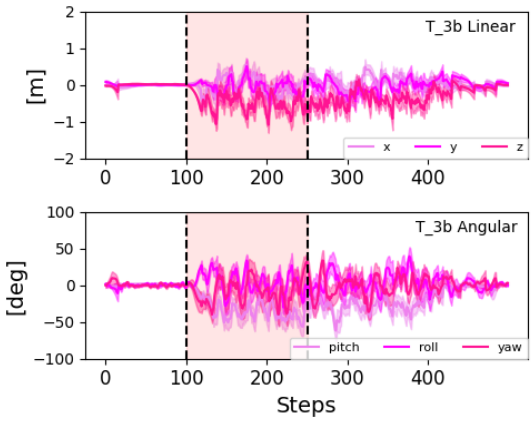
(c) Quadrotor under failure 10% of the time



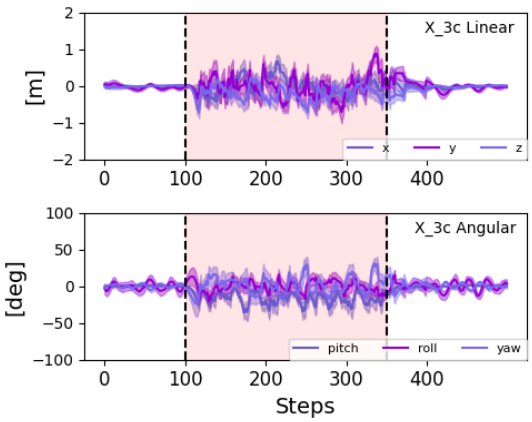
(d) Tiltrotor under failure 10% of the time



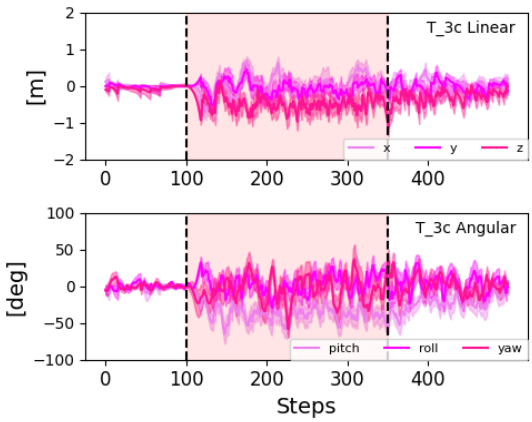
(e) Quadrotor under failure 30% of the time



(f) Tiltrotor under failure 30% of the time

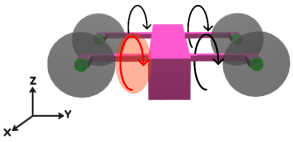


(g) Quadrotor under failure 50% of the time

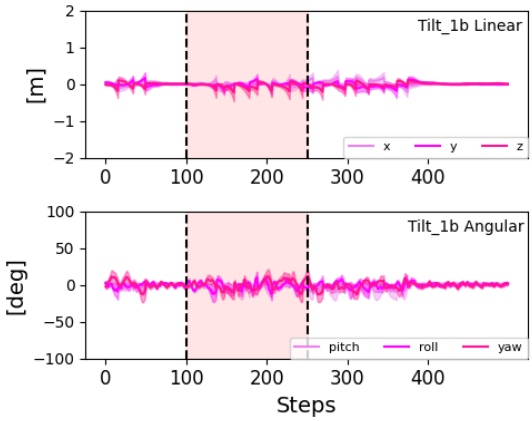


(h) Tiltrotor under failure 50% of the time

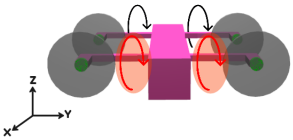
Figure 5.7 – Comparison of various scenarios with 2 opposite rotors under failure for Quadrotor-X and Tiltrotor-Parallel



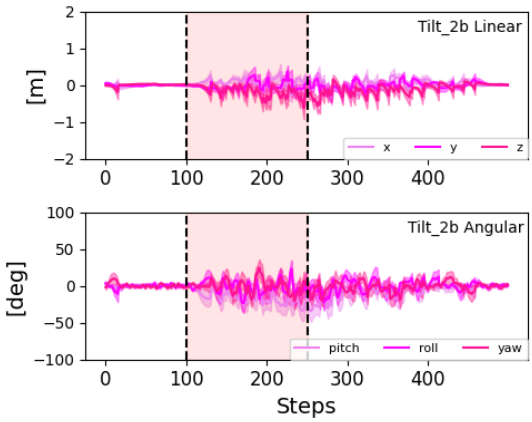
(a) Tiltrotor-Parallel with 1 tiltrotor under failure



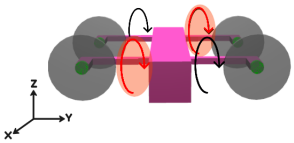
(b) Tiltrotor-UAV displacement with 1 tiltrotor under failure 30% of the time



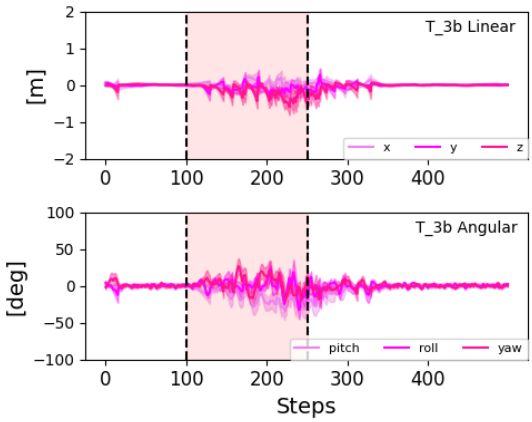
(c) Tiltrotor-Parallel with 2 adjacent tiltrotors under failure



(d) Tiltrotor-UAV displacement with 2 adjacent tiltrotors under failure 30% of the time



(e) Tiltrotor-Parallel with 2 opposite tiltrotors under failure

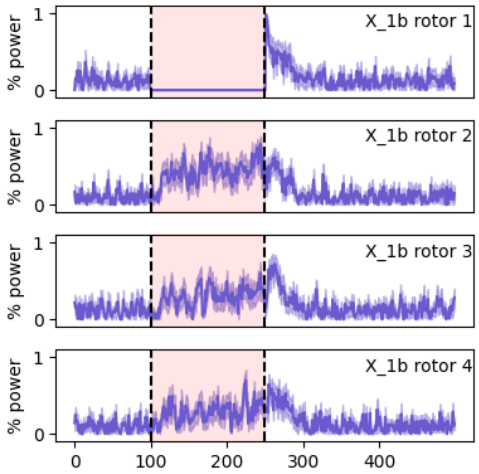


(f) Tiltrotor-UAV displacement with 2 opposite tiltrotors under failure 30% of the time

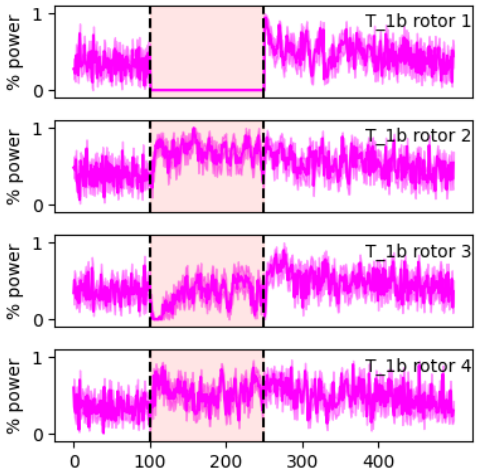
Figure 5.8 – Tiltrotors failures configurations for 1 tiltrotor (a), 2 adjacent tiltrotors (c) and 2 opposite tiltrotors (e), and correspondent UAV's displacement curves (b), (d) and (f)

5.2.2 Discussion on Robustness and Recovery

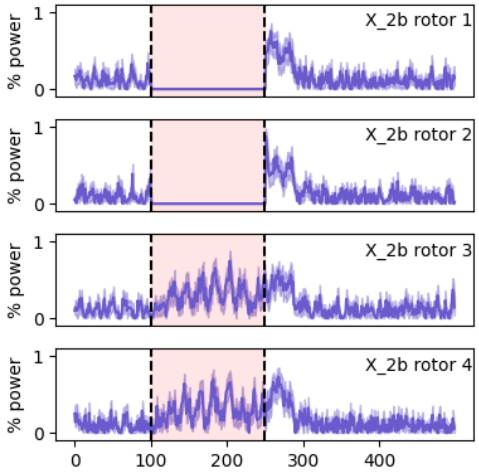
This section analyzes the robustness and recovery capabilities of UAV configurations under the failure scenarios.



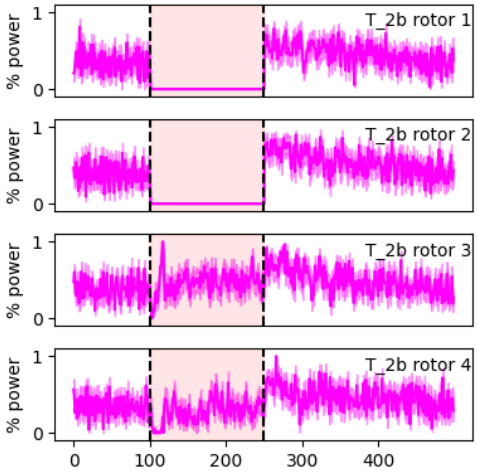
(a) Quadrotor-X with failure in 1 rotor



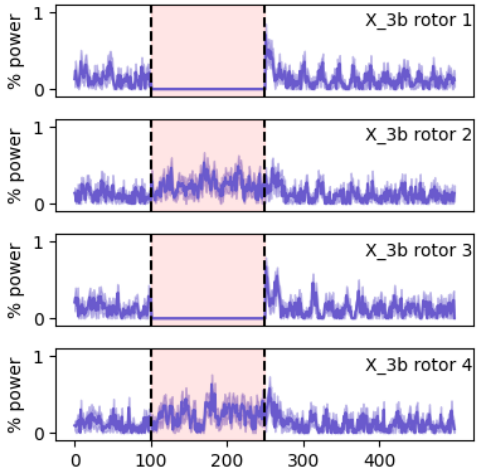
(b) Tiltrotor-UAV with failure in 1 rotor



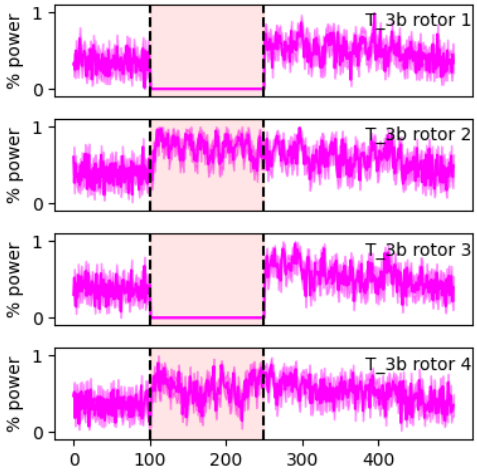
(c) Quadrotor-X with failure in 2 adjacent rotors



(d) Tiltrotor-UAV with failure in 2 adjacent rotors

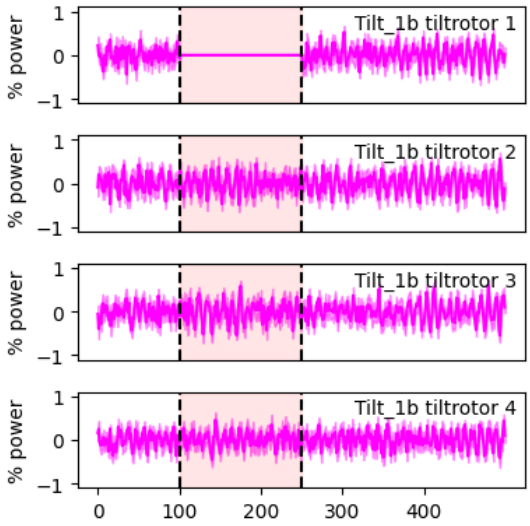


(e) Quadrotor-X with failure in 2 opposite rotors

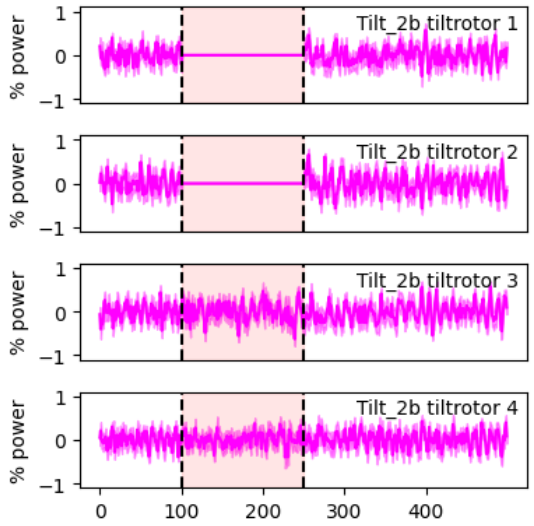


(f) Tiltrotor-UAV with failure in 2 opposite rotors

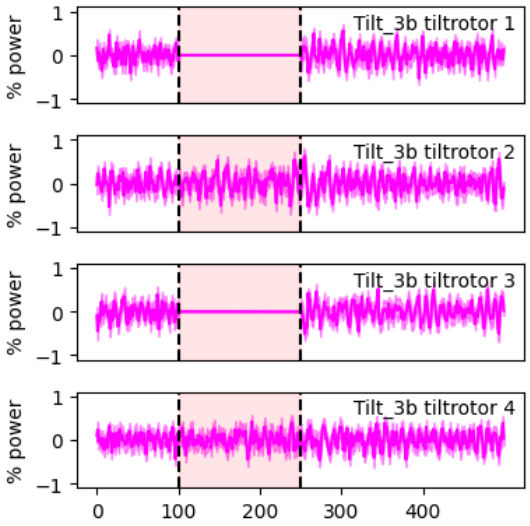
Figure 5.9 – Comparison of how different failure scenarios influence the power required by the remaining active rotors



(a) Tiltrotor-UAV with failure in 1 tiltrotor



(b) Tiltrotor-UAV with failure in 2 adjacent tiltrotors



(c) Tiltrotor-UAV with failure in 2 opposite tiltrotors

Figure 5.10 – Comparison of how different failure scenarios influence the power required by the remaining active tiltrotors

Notably, the Tiltrotor-Parallel exhibits greater sensitivity to failures, as evidenced by the prolonged time required to stabilize compared to the Quadrotor-X configuration. Specifically, when testing the failure in one thrust-rotor for 10% of the time, both UAVs were able to restabilize, but the tiltrotor took up to four times as many steps to recover stability. Further exacerbating the failure conditions to 30% and 50% of the time, neither became irreversibly uncontrollable, but the Quadrotor-X maintained its characteristic of faster recovery. This suggests that the tiltrotor, with more degrees of freedom and higher geometric complexity, requires a more robust failure treatment.

In scenarios where two rotors fail, either two adjacent or two opposite rotors, the Quadrotor-X showed a worse recovery scenario for opposite thrust-rotor failure than for ad-

jacent thrust-rotor failure, but it still presents a faster recovery than the Tiltrotor in the same configuration.

The results underscore the adaptability of the control policies, with the Quadrotor-X showcasing more robust responses compared to the Tiltrotor-Parallel. Interestingly, the profile of the thrust-rotors' power requirement reveals that if one rotor fails (Figures 5.9b and 5.9a) or two of them (Figures 5.9c, 5.9e, 5.9d, and 5.9f), the remaining rotors change their behavior to compensate for the inactive thrust-rotors, and immediately after the thrust-rotors are re-established, the reactivated rotors are highly required in all configurations to re-establish stability.

When failures are induced in one tiltrotor (Figure 5.8a), two adjacent tiltrotors (Figure 5.8c), or two opposite tiltrotors (Figure 5.8e), the instability caused by the tiltrotor failure is less impactful than the thrust-rotor failures. This behavior is expected given that, once the thrust-rotor fails, the tilt rotor couplet in their axis will also be ineffective. When tiltrotor failures are induced, it becomes uncontrollable, but its orientation is kept constant, so the thrust-rotor still has an effectiveness. Although the impact of tiltrotor failure is less prominent, its behavior is similar to thrust-rotor failure concerning the time to re-stabilization. None of them remain unstable after tiltrotor failure ceased. The tiltrotors' power requirement also increased in the remaining active tiltrotors when failure occurs, showing a similar result to thrust-rotor failures.

6 Conclusion and Future Directions

The experiments conducted in this work demonstrated the effectiveness of both PPO and TD3 algorithms in learning control policies for UAVs. The performance of the algorithms varied with the complexity of the UAV models, with more pronounced differences as complexity increased. As future directions, it would be valuable to explore the performance of other reinforcement learning algorithms, such as Soft Actor-Critic (SAC) or Proximal Value Optimization (PVO), for UAV control tasks and to consider constraints to drive the learning process.

Secondly, the analysis of failure scenarios highlights the nuanced responses of UAV control policies trained using PPO. While Tiltrotor-Parallel demonstrates heightened sensitivity to failures, Quadrotor-X exhibits greater resilience, evidencing the influence of geometry complexity on the robustness to failures. Notably, multiple rotor failures required more time for recovery of stability. These findings underscore the importance of robustness testing in UAV control policy development, especially in real-world applications where failures can happen unexpectedly. Furthermore, they emphasize the need for tailored control strategies to mitigate the impact of failures, particularly in UAVs with diverse geometries and complexities.

Furthermore, our robustness analysis under failure conditions revealed important insights into the adaptability of PPO-controlled UAVs. Despite facing simulated rotor failures, PPO demonstrated the ability to generalize control strategies for both the simplest (Quadrotor-X) and most complex (Tiltrotor-Parallel) UAV configurations. Our results indicate that the simplest UAV designs exhibit faster recovery times under failure scenarios, emphasizing the importance of considering geometric complexity in eVTOL design and control strategies.

In response to the research questions posed:

i. Can eVTOL geometry be adapted to simulated environments considering critical factors like degree of freedom, joint configuration, and other key characteristics? Our findings suggest that adapting eVTOL geometries for simulation is feasible, with varying degrees of adaptation required based on geometric complexity and operational demands.

ii. Can the flight of eVTOLs be controlled to meet distinct operational demands using RL-based control? The use of RL-based control, particularly with algorithms like PPO and TD3, has shown promise in meeting distinct operational demands by learning adaptive control policies.

iii. How do these control policies respond to potential challenging scenarios associated with flight, such as rotor failures? Our analysis indicates that while control policies exhibit varying responses to challenging scenarios like rotor failures, they generally demonstrate adaptive resilience, particularly influenced by geometric design complexities.

In summary, this research addresses a critical gap in the application of RL for eVTOL control under fault conditions. The findings of this study promise to contribute to the development of safer and more reliable eVTOLs, expanding their applications, including urban air mobility. Our combination of adapting eVTOL geometries for simulation and implementing RL algorithms in practical applications makes this research both innovative and practically significant.

Bibliography

AHMED, S. S.; FOUNTAS, G.; EKER, U.; STILL, S. E.; ANASTASOPOULOS, P. C. An exploratory empirical analysis of willingness to hire and pay for flying taxis and shared flying car services. *Journal of Air Transport Management*, v. 90, p. 101963, 2021. ISSN 0969-6997. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0969699720305469>.

AHMED, S. S.; HULME, K. F.; FOUNTAS, G.; EKER, U.; BENEDYK, I. V.; STILL, S. E.; ANASTASOPOULOS, P. C. The flying car—challenges and strategies toward future adoption. *Frontiers in Built Environment*, v. 6, 2020. ISSN 2297-3362. Disponível em: <https://www.frontiersin.org/articles/10.3389/fbuil.2020.00106>.

ALLENSPACH, M.; DUCARD, G. J. J. Nonlinear model predictive control and guidance for a propeller-tilting hybrid unmanned air vehicle. *Automatica*, v. 132, p. 109790, 2021. ISSN 0005-1098. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0005109821003101>.

ANGLADE, A.; KAI, J.-M.; HAMEL, T.; SAMSON, C. Automatic control of convertible fixed-wing drones with vectorized thrust. abr. 2019. Disponível em: <https://hal.science/hal-02111045>.

ARULKUMARAN, K.; DEISENROTH, M. P.; BRUNDAGE, M.; BHARATH, A. A. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, v. 34, n. 6, p. 26–38, 2017.

AZAR, A. T.; KOUBAA, A.; MOHAMED, N. A.; IBRAHIM, H. A.; IBRAHIM, Z. F.; KAZIM, M.; AMMAR, A.; BENJDIRA, B.; KHAMIS, A. M.; HAMEED, I. A.; CASALINO, G. Drone deep reinforcement learning: A review. *Electronics*, v. 10, n. 9, 2021. ISSN 2079-9292. Disponível em: <https://www.mdpi.com/2079-9292/10/9/999>.

BAUERSFELD, L.; DUCARD, G. Fused-pid control for tilt-rotor vtol aircraft. p. 703–708, 2020.

BAUERSFELD, L.; SPANNAGL, L.; DUCARD, G. J. J.; ONDER, C. H. Mpc flight control for a tilt-rotor vtol aircraft. *IEEE Transactions on Aerospace and Electronic Systems*, v. 57, n. 4, p. 2395–2409, 2021.

BROCKMAN, G.; CHEUNG, V.; PETTERSSON, L.; SCHNEIDER, J.; SCHULMAN, J.; TANG, J.; ZAREMBA, W. *OpenAI Gym*. 2016.

CETINSOY, E. Design and control of a gas-electric hybrid quad tilt-rotor uav with morphing wing. p. 82–91, 2015.

CHIAPPINELLI, R.; COHEN, M.; DOFF-SOTTA, M.; NAHON, M.; FORBES, J. R.; APKARIAN, J. Modeling and control of a passively-coupled tilt-rotor vertical takeoff and landing aircraft. p. 4141–4147, 2019.

Developmental Reinforcement Learning of Control Policy of a Quadcopter UAV With Thrust Vectoring Rotors, v. 2 de *Dynamic Systems and Control Conference*, (Dynamic Systems and Control Conference, v. 2). V002T36A011 p.

eHang. *EHang 216-S*. 2023. Accessed: 2023-10-22. Disponível em: <https://www.ehang.com/ehangaav/>.

EKER, U.; FOUNTAS, G.; AHMED, S. S.; ANASTASOPOULOS, P. C. Survey data on public perceptions towards flying cars and flying taxi services. *Data in Brief*, v. 41, p. 107981, 2022. ISSN 2352-3409. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2352340922001925>.

FUJIMOTO, S.; HOOF, H. van; MEGER, D. Addressing function approximation error in actor-critic methods. *CoRR*, abs/1802.09477, 2018. Disponível em: <http://arxiv.org/abs/1802.09477>.

GOTA, S. *Fuel Economy of Passenger Cars in the Global South: A case of two steps forward, one step back as fuel economy improvements are reduced by increasing car power and weight*. 2023. Accessed: 2023-08-02. Disponível em: <https://sustmob.org/GFEI/PDF/FuelEconomyofPassengerCars_GlobalSouth.pdf>.

HEGDE, N. T.; GEORGE, V. I.; NAYAK, C. G. Modelling and transition flight control of vertical take-off and landing unmanned tri-tilting rotor aerial vehicle. p. 590–594, 2019.

HERNANDEZ-GARCIA, R. G.; RODRIGUEZ-CORTES, H. Transition flight control of a cyclic tiltrotor uav based on the gain-scheduling strategy. p. 951–956, 2015.

HOLMES, G.; CHOWDHURY, M.; MCKINNIS, A.; KESHMIRI, S. Deep learning model-agnostic controller for vtol class uas. In: *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. [S.l.: s.n.], 2022. p. 1520–1529.

Joby Aviation. *Joby S4*. 2023. Accessed: 2023-10-22. Disponível em: <https://www.jobyaviation.com/>.

LIU, Z.; THEILLIOL, D.; YANG, L.; HE, Y.; HAN, J. Observer-based linear parameter varying control design with unmeasurable varying parameters under sensor faults for quad-tilt rotor unmanned aerial vehicle. *Aerospace Science and Technology*, v. 92, p. 696–713, 2019. ISSN 1270-9638. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1270963818321722>.

MOFOLASAYO, A. Potential policy issues with flying car technology. *Transportation Research Procedia*, v. 48, 10 2020.

NAKAMURA, Y.; ARAKAWA, A.; WATANABE, K.; NAGAI, I. Transitional flight simulations for a tilted quadrotor with a fixed-wing. p. 1829–1836, 2018.

PAN, G.; ALOUINI, M.-S. Flying car transportation system: Advances, techniques, and challenges. *IEEE Access*, Institute of Electrical and Electronics Engineers, 2021. ISSN 2169-3536.

POSTORINO, M. N.; SARNE, G. M. L. Reinventing mobility paradigms: Flying car scenarios and challenges for urban mobility. *Sustainability*, v. 12, n. 9, 2020. ISSN 2071-1050. Disponível em: <https://www.mdpi.com/2071-1050/12/9/3581>.

SCHULMAN, J.; WOLSKI, F.; DHARIWAL, P.; RADFORD, A.; KLIMOV, O. *Proximal Policy Optimization Algorithms*. 2017.

SHEN, D.; LU, Q.; HU, M.; KONG, Z. Mathematical modeling and control of the quad tilt-rotor uav. p. 1220–1225, 2018.

SONG, Y.; NAJI, S.; KAUFMANN, E.; LOQUERCIO, A.; SCARAMUZZA, D. *Flightmare: A Flexible Quadrotor Simulator*. 2021.

SUTTON, R. S.; BARTO, A. G. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018. Disponível em: <http://incompleteideas.net/book/the-book-2nd.html>.

The Vertical Flight Society. *EHang 216-S*. 2023. Accessed: 2023-10-22. Disponível em: <https://evtol.news/ehang-216>.

The Vertical Flight Society. *Joby Aviation S4 2.0*. 2023. Accessed: 2023-10-22. Disponível em: <https://evtol.news/joby-s4>.

The Vertical Flight Society. *Volocopter VoloRegion*. 2023. Accessed: 2023-10-22. Disponível em: <https://evtol.news/volocopter-voloconnect>.

TODOROV, E.; EREZ, T.; TASSA, Y. Mujoco: A physics engine for model-based control. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. [S.l.: s.n.], 2012. p. 5026–5033.

United Nations. *Sustainable transport, sustainable development. Interagency report for second Global Sustainable Transport Conference*. 2021. Accessed: 2023-08-02. Disponível em: <https://sdgs.un.org/sites/default/files/2021-10/Transportation%20Report%202021_FullReport_Digital.pdf>.

United Nations Department of Economic and Social Affairs. *Population Division (2018). The World's Cities in 2018 — Data Booklet*. 2018. Accessed: 2023-08-02. Disponível em: <https://www.un.org/development/desa/pd/sites/www.un.org.development.desa.pd/files/files/documents/2020/Jan/un_2018_worldcities_databooklet.pdf>.

Volocopter. *VoloRegion*. 2023. Accessed: 2023-10-22. Disponível em: <https://www.volocopter.com/en/solutions/voloregion>.

World Resources Institute. *Climate Watch Historical GHG Emissions*. 2023. Accessed: 2023-08-02. Disponível em: <https://www.climatewatchdata.org/ghg-emissions>.

YANG, R.; DU, C.; ZHENG, Y.; GAO, H.; WU, Y.; FANG, T. Ppo-based attitude controller design for a tilt rotor uav in transition process. *Drones*, v. 7, n. 8, 2023. ISSN 2504-446X. Disponível em: <https://www.mdpi.com/2504-446X/7/8/499>.

YU, L.; ZHANG, D.; ZHANG, J. Transition flight modeling and control of a novel tilt tri-rotor uav. p. 983–988, 2017.