NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ANTÓNIO SALVADOR BOUZA SERRANO DOS SANTOS COSTA

Bsc in Electrical and Computer Engineering

CONTROL OF A SHUTTLE DRONE FOR COOPERATIVE CAPTURE WITH AIRFLOW DISTURBANCE COMPENSATION

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
September, 2024

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING

CONTROL OF A SHUTTLE DRONE FOR COOPERATIVE CAPTURE WITH AIRFLOW DISTURBANCE COMPENSATION

ANTÓNIO SALVADOR BOUZA SERRANO DOS SANTOS COSTA

Bsc in Electrical and Computer Engineering

Adviser: Bruno João Nogueira Guerreiro
Assistant Professor, NOVA University Lisbon

Examination Committee

Chair: Rodolfo Alexandre Duarte Oliveira
Associate Professor, NOVA University Lisbon

Rapporteur: Fernando José Almeida Vieira do Coito
Associate Professor, NOVA University Lisbon

Member: Bruno João Nogueira Guerreiro
Assistant Professor, NOVA University Lisbon

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
September, 2024

Control of a shuttle drone for cooperative capture with airflow disturbance compensation

Acknowledgements

The first expression of my gratitude goes to my advisor, Prof Bruno Guerreiro, for his exemplary help and support through this project, displaying regularly good guidance, patience, and suggestions. I also chose this project because I was told Prof. Bruno was a good advisor, which proved true.

Secondly, I'm proud to have been a student at this faculty, NOVA FCT, and, therefore, I also want to thank this institution for these formative years during which I acquired valuable tools for my professional and personal life. It was at this faculty that I met my friends Carolina Luís, Guilherme Ribeiro, Francisco Almirante, João Oliveira, João Henriques, João André, Duarte Alves and Matilde Sacavém who along with my friend Vasco Graça were crucial for me in achieving my academic goals and have a positive experience during these years.

Also, I want to thank Vodafone and my team for providing work hours to conclude this thesis, which was admirable and very helpful. Special thanks to Rubén Sousa, Cristiano Pereira, Matilde Feliciano and Tiago Fernandes for the great support and professional advice.

Finally, for the example, the amazing support and motivation my family gives me, I want to leave a final and the best note I can for them. For my grandparents, uncles, close cousins, and 2 brothers, especially my parents and my friend Felisbela Fortes, who excel in all these factors and to whom I dedicate my best success.

This work acknowledges the support of FCT projects CAPTURE (DOI:10.54499/PTDC/EEIAUT/1732/2020) and CTS (DOI:10.54499/UIDP/00066/2020).

Abstract

In this thesis, the complex task of autonomously controlling a shuttle quadrotor to capture another one cooperatively is enhanced by considering the aerodynamic disturbances caused by environmental factors and proximity flight between the two drones.

To achieve this goal, this thesis integrates modelling techniques for both drones, accounting for aerodynamic effects such as rotor and frame drag, along with real-time control nonlinear algorithms to mitigate these influences. Furthermore, a downwash model is studied and included in the shuttle drone, which is allied with a comprehensive compensation method for the target drone. This method dynamically adjusts the drone's control inputs and coordinates with the target drone, improving stability and precision during capture manoeuvres.

Simulation results demonstrate the effectiveness of the proposed controlling strategies, showcasing improved performance in trajectory tracking and disturbance mitigation in various scenarios in solo and proximity flight between the drones. The trajectory tracking performance of a single quadcopter under different conditions is analyzed, gradually enhancing the controller's robustness to understand the impact of aerodynamic phenomena. The proximity flight between two quadcopters is also portrayed, showing the effect of the downwash on the target drone's trajectory and its controller's ability to compensate for this airflow.

Therefore, this work and its findings provide valuable insights into the challenges and solutions for autonomous control of drones in close proximity, considering aerodynamic disturbances. The modelling and control techniques presented can contribute to developing more precise and stable drone capture systems, allowing for more efficient and safer operations where these manoeuvres are performed.

Keywords: Shuttle Drone, Cooperative Capture, Airflow Disturbance Compensation, Aerodynamic Modelling

Resumo

Nesta tese, a complexa tarefa de controlar autonomamente um quadrotor de transporte para capturar outro cooperativamente é aprimorada ao considerar as perturbações aerodinâmicas de fatores ambientais e voo de proximidade entre os dois drones.

Tendo em conta este objetivo, esta tese integra técnicas de modelação para ambos os drones, onde efeitos aerodinâmicos como rotor e arrasto da fuselagem são contabilizados e mitigados através de algoritmos não lineares de controlo em tempo real. Além disso, um modelo de downwash é estudado e incluído no drone de transporte, que é aliado a um método de compensação abrangente para o drone alvo. Este método ajusta dinamicamente as entradas de controlo do drone e as coordenadas com o drone alvo, melhorando a estabilidade e a precisão durante as manobras de captura.

Os resultados da simulação demonstram a eficácia das estratégias de controlo propostas, mostrando uma melhor execução no rastreamento de trajetória e na mitigação de distúrbios em vários cenários de voo solo e de proximidade entre entre dois drones. O desempenho de rastreamento de trajetória de um único quadricóptero sob diferentes condições é analisado, aumentando gradualmente a robustez do controlador para entender o impacto dos fenómenos aerodinâmicos. O voo de proximidade entre dois quadricópteros também é retratado, mostrando o efeito do downwash na trajetória do drone alvo e a capacidade do seu controlador em compensar este fluxo de ar.

Portanto, este trabalho e suas descobertas fornecem informações importantes sobre os desafios e soluções para o controlo autónomo de drones em estreita proximidade, considerando distúrbios aerodinâmicos. As técnicas de modelação e controlo apresentadas podem contribuir para o desenvolvimento de sistemas de captura de drones mais precisos e estáveis, permitindo operações mais eficientes e seguras onde estas manobras são executadas.

Palavras-chave: Drone de transporte, Captura cooperativa, Compensação de perturbações de fluxos de ar, Modelação aerodinâmica

Contents

List of Figures

List of Tables

Glossary

Blade Element Theory — A technique employed in aerodynamics to determine the performance of propellers or rotors by dividing the blade into smaller components, each of which is regarded as a 2D airfoil. It estimates thrust and torque by accounting for fluctuations in aerodynamic forces throughout the blade span. *(p. 13)*

Computational Fluid Dynamics — A subfield of fluid mechanics that analyzes and solves problems with fluid flow using data structures and numerical analysis. Computational fluid dynamics (CFD) applications include engineering design, weather modelling, and aerodynamics. *(p. 32)*

ESP8266 — A low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability, often used in IoT and drone projects for wireless communication *(p. 15)*

Marvelmind Indoor Navigation System — A high-precision indoor positioning system that employs ultrasonic beacons to provide real-time location data for robotics, drones, and other objects in indoor environments. It is frequently used for navigation and positioning in robotics, automation, and virtual reality applications, providing centimeter-level accuracy. *(p. 15)*

MATLAB A numerical computing environment and high-level programming language developed by MathWorks. It is frequently employed in engineering applications for algorithm development, data analysis, and simulation. *(p. 14)*

Momentum Theory A principle of aerodynamics that uses momentum conservation to explain how propellers or rotors produce thrust. It models the rotor as an actuator disk, assuming steady, inviscid, and incompressible flow, relating thrust to the induced velocity of air through the rotor *(p. 13)*

QGroundControl An open-source ground control station software for drones, providing a user interface for flight control, mission planning, and vehicle setup *(p. 15)*

Acronyms

2D Two-dimensional *(p. 31)*
3D Three-dimensional *(p. 7)*

ANNs Artificial Neural Networks *(p. 10)*

DOF Degrees of Freedom *(p. 16)*

GPS Global Positioning System *(p. 15)*

KFs Kalman Filters *(p. 6)*

LMPC Linear Model Predictive Control *(p. 9)*
LQG Linear-Quadratic Gaussian *(p. 6)*
LQR Linear-Quadratic Regulator *(p. 6)*
LTI linear time-invariant *(p. 5)*

MPC Model Predictive Control *(p. 9)*

NMPC Nonlinear Model Predictive Control *(p. 9)*

PID Proportional-Integral-Derivative *(p. 6)*

RMSE Root Mean Square Error *(p. 39)*
ROS Robot Operating System *(p. 14)*

SMC Sliding Mode Control *(p. 9)*
SO(3) Special Orthogonal group in three-dimensional space *(p. 24)*

UAVs Unmanned Aerial Vehicles *(p. 1)*

VTOL Vertical Take-off and Landing *(p. 11)*

Wi-Fi Wireless Fidelity *(p. 15)*

Symbols

A_d	Rotor disk area *(p. 29)*
$a_{fd} = \frac{f_{fd}}{m}$	Frame drag acceleration *(p. 26)*
$a_{rd} = \frac{f_{rd}}{m}$	Rotor drag acceleration *(p. 26)*
$A = diag(A_{xx}, A_{yy}, A_{zz})$	Drone's projected area *(p. 28)*
C_d	Frame drag coefficient *(p. 28)*
$C = \begin{bmatrix} x_C & y_C & z_C \end{bmatrix}^T$	C frame *(p. 18)*
$C_{rd} = diag(C_{rd,xx}, C_{rd,yy}, C_{rd,zz})$	Rotor drag coefficient *(p. 26)*
e_ω	Angular velocity error *(p. 23)*
e_p	Position error *(p. 23)*
e_R	Rotation error *(p. 23)*
e_v	Velocity error *(p. 23)*
f_a	Aerodynamic drag force *(p. 21)*
f_w	Weight force *(p. 21)*
f_{des}	Desired force vector *(p. 23)*
f_{fd}	Frame drag force *(p. 26)*
f_{rd}	Rotor drag force *(p. 26)*
$I = diag(I_{xx}, I_{yy}, I_{zz})$	Inertia matrix *(p. 22)*
k, h	Downwash shaping parameters *(p. 29)*
K_ω	Angular rates gain matrix *(p. 25)*
K_p	Controller gain for position *(p. 23)*
K_R	Orientation gain matrix *(p. 25)*
K_v	Controller gain for velocity *(p. 23)*
K_{ip}	Controller gain for position error's integral *(p. 23)*

ϕ	Roll angle *(p. 19)*
ψ	Yaw angle *(p. 19)*
R	Rotation matrix *(p. 19)*
r	Rotor radius *(p. 29)*
r_d	Radial distance between two points in a 3D space *(p. 29)*
ρ	Air density *(p. 29)*
$\hat{\omega}$	Skew simetric matrix of ω *(p. 22)*
T	Thrust force *(p. 21)*
θ	Pitch angle *(p. 19)*
u_1	Thrust control input *(p. 22)*
$u_\tau = \begin{bmatrix} u_2 & u_3 & u_4 \end{bmatrix}^T$	Torque vector control input *(p. 23)*
$v = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T$	Drone's linear velocity in the world frame *(p. 21)*
$v_{air} = \begin{bmatrix} v_{air,x} & v_{air,y} & v_{air,z} \end{bmatrix}^T$	Drone's air relative velocity in the world frame *(p. 26)*
$v_{Bair} = R^T v_{air}$	Drone's air relative velocity in the body frame *(p. 26)*
$V_{d_1}(z)$	Shuttle drone's downwash vertical velocity *(p. 29)*
$v_{dw_1} = \begin{bmatrix} 0 & 0 & -V_{d_1} \end{bmatrix}^T$	Shuttle drone's downwash velocity in the world frame *(p. 29)*
V_{i_1}	Drone's induced speed *(p. 29)*
$v_w = \begin{bmatrix} v_{w,x} & v_{w,y} & v_{w,z} \end{bmatrix}^T$	Wind velocity in the world frame *(p. 27)*
$\omega = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T$	Angular velocity *(p. 21)*
x_B	X-axis unit vector in the body frame *(p. 18)*
$x_W = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$	X-axis unit vector in the world frame *(p. 18)*
y_B	Y-axis unit vector in the body frame *(p. 18)*
$y_W = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$	Y-axis unit vector in the world frame *(p. 18)*
z_B	Z-axis unit vector in the body frame *(p. 18)*
z_r	Rotor height *(p. 29)*
$z_W = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$	Z-axis unit vector in the world frame *(p. 18)*

1 Introduction

Over the past century, aerospace systems have gained prominence and undergone significant evolution: from the simple act of propelling objects into the air to manoeuvring them manually and, more recently, instructing them to operate autonomously without needing a human pilot onboard. During this time, Unmanned Aerial Vehicles (UAVs), also commonly called drones, have seen widespread adoption [1] due to their efficiency in fields such as search and rescue manoeuvres [2], commercial applications such as package transport and delivery [3], agriculture [4], among others.

Moreover, amid these soaring achievements, new areas of study have targeted the power of operating with multiple drones that operate cooperatively, lightening the burden of the shared task, shared load or shared resource [5]. For example, search-and-rescue operations can become much easier since drones can work together to locate missing persons, assess disaster-affected areas, and deliver essential supplies. A swarm of search-and-rescue drones can cover large regions quickly, providing real-time data to first responders. These drones can communicate with each other to optimise search patterns, share information, and coordinate efforts. Even in arts, more shows opt for a team of drones with LED-lit acrobatics as more sustainable in pollution and the environment than fireworks as it was in the Olympics in Tokyo 2020 [6]. However, for this cooperation to yield results, drones must behave differently. Their movements are no longer solitary; they synchronise, communicate, and adapt. Challenges emerge — collision avoidance, communication protocols, and dynamic coordination. This thesis explores the challenges of executing a cooperative capture manoeuvre between two quadrotors, focusing on effectively controlling a shuttle drone while compensating for existing airflow disturbances. Whether these disturbances derive from environmental factors or the presence of other drones, addressing them demands innovative solutions.

Therefore, we delve into the intricacies of cooperative drone capture, exploring techniques and algorithms that mitigate airflow disturbances. By understanding the dynamics of quadrotor drones and their interactions, we aim to enhance the reliability and safety of this complex and meticulous operation.

1.1 Context

This thesis was created as part of the CAPTURE project [7] to investigate the various obstacles surrounding UAVs' ability to interact with their environment and carry out more daring manoeuvres like capturing and moving other vehicles or items. This project considers two scenarios: the first scenario involves the shuttle drone assisting the other aircraft during launch or capture by giving information about their intended course or carefully coordinating its movement to facilitate manoeuvring. This makes it possible to design specialized vehicles for a particular purpose, like long-lasting fixed-wing UAVs, while shuttle drones handle these vehicles' vertical takeoff and landing. In a second scenario, the shuttle drone may or may not engage in passive or active cooperation with the other aircraft it is attempting to launch or capture. Because of this, the shuttle UAV could function as a security tool to remove drones or other items from restricted areas where they might actively try to evade detection.

Many fascinating scientific and technological issues need to be resolved in both approaches, such as planning cooperative and optimal trajectories for a group of heterogeneous vehicles, cooperative, hybrid, and implementing distributed control strategies for critical rendezvous manoeuvres, cooperative and distributed estimation of the motion of the shuttle drones, other vehicles, and the surrounding environment, and handling of strategies based on differential game theory for estimation, control, and planning for the non-cooperative scenario. Due to their ability to manoeuvre in this way, both UAVs can work together to complete various specialized tasks. For example, in package delivery, the fixed-wing drone's long reach and high speed allow it to deliver the package to its destination, while the shuttle UAV aids in landing in more confined spaces. Additionally, by optimising the use of available space and infrastructure, VTOL drones in airports can assist aircraft with take-off and landing, thereby transforming the layout of airports.

1.2 Problem statement and proposed solution

This operation, exemplified in Figure 1.1, requires a high-precision flight from the two quadrotors to ensure its effectiveness and safety, resulting in many challenges, as both drones are continuously influenced by environmental factors such as wind gusts and air turbulence throughout the entire process, and, in the final stage when trying to capture the other drone, additional stabilization challenges will arise due to changes in air dynamics caused by the proximity of the two drones. Therefore, this thesis proposes a scientifically-backed solution for modelling and controlling the shuttle drone to face this challenge, devising three main objectives:

- Develop a model and controller capable of autonomously managing a quadrotor during basic flight manoeuvres;
- Create model and control strategies for proximity flight between two quadrotors;

- Validate proposed strategies in simulation with various scenarios of trajectory following and proximity flights.

First, a nonlinear controller is designed and implemented following an existing example.

Then, for the situation where a drone flies above or beneath another one, new modelling and control algorithms must be developed and implemented to address the aerodynamic interactions in these flight scenarios.

Lastly, simulations are run using a T-Drone M690B quadrotor (displayed in Figure 1.2) to test the algorithm's ability to track trajectories and show how aerodynamic forces affect flights between two drones that are close together.

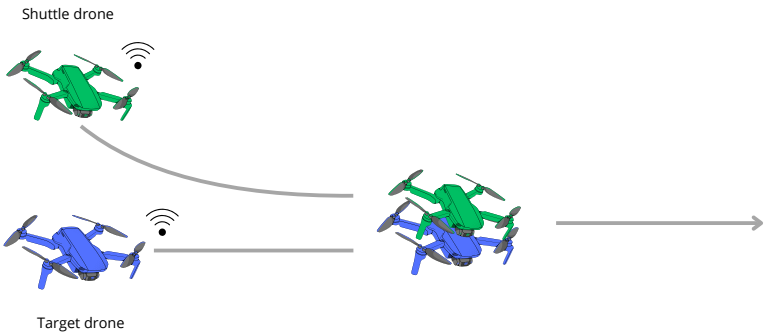


Figure 1.1: Cooperative capture manoeuvre between two quadrotors

Figure 1.2: T-Drone M690B model

1.3 Outline

The remainder of this document is organized as follows:

- Chapter 2 reviews the work already done in this area, specifically addressing some relevant drone modelling and control techniques and aerodynamic considerations.

- In Chapter 3, the model of the quadcopters is described alongside a nonlinear controller responsible for guiding the drones to follow given reference trajectories.
- Chapter 4 introduces new algorithms for modelling and controlling drones, which are more capable of addressing new aerodynamic conditions and scenarios involving proximity flight between two quadcopters.
- In Chapter 5, the results of simulations to test the developed controller performance in various scenarios are presented.
- Finally, in Chapter 6, some concluding observations are drawn and future work suggestions are made.

2

State of the Art

As previously mentioned, the CAPTURE project's scope, where this dissertation is inserted, covers different topics about this vehicle, from its flight motion and control to its simulation and testing. Therefore, this section is divided into various groups. In the first two, 2.1 presents some research regarding conventional control techniques behind controlling a single quadrotor autonomously, and 2.2 follows with devising strategies for cooperative manoeuvres between quadrotors. After that, in 2.3, a brief review of the existing work in aerodynamics for these drones is also provided. This understanding is crucial for safe and effective operations, including when the shuttle and target drones operate in close quarters. Finally, in 2.4, relevant and recent studies about flight simulation and testing are also addressed. The CAPTURE project's objective of conducting practical flight simulations and tests of a drone aligns with the need to validate theoretical concepts in actual flight scenarios.

2.1 Quadrotor control techniques for non-cooperative flight

When controlling a single drone in a non-cooperative flight scenario such as a capture manoeuvre, it is essential to consider both the vehicles and the environmental constraints. This is because UAVs vary in shape, size, weight, and purpose, requiring tailored approaches that account for their interaction with the environment. Consequently, it is important to note that these procedures often consider multiple control methods to achieve the most suitable solution to the desired requirements.

Furthermore, it is possible to associate a technique with one of the two significant categories of control: linear and nonlinear control. These two types derive from the different groups of systems that need to be controlled, as linear and nonlinear ones, where the main distinction is the principle or rule they follow. For instance, linear systems that abide by the superposition principle and, in some cases, have parameters that remain constant over time are known as linear time-invariant (LTI) systems. Graphically, the output of these systems is directly proportional to the input, which is why they are termed "linear" systems. On the other hand, in nonlinear systems, the output does not have

a proportional relationship to the input and produces more complex responses. This principle results in more intricate and varied behaviour in these systems, making them more common as they cover a much broader range of real-world situations.

2.1.1 Linear control

Linear control methods have the least computational power burden and are much easier to implement. These controllers are less complex and, therefore, faster to design and converge to a desired outcome (when that result is within the control capabilities). However, due to their limited robustness, traditional linear control methods are inadequate for managing underactuated vehicles. These vehicles have fewer actuators than the degrees of freedom to be controlled. External forces can cause acceleration changes in any direction, making it challenging to stabilize the drone when it is operated away from its local equilibrium point or during agile manoeuvres. Thus, in these situations, linear controllers are usually used only for attitude stability control. To overcome this limitation, linearization must be applied to the system by providing relative equilibrium conditions around a steady-state operating point.

One of the most common linear techniques is the **Linear-Quadratic Regulator (LQR)**, a type of feedback controller. It determines the optimal control law that minimizes a quadratic cost function, keeping the system's state close to the desired trajectory while reducing control effort. Moreover, it can handle perturbations and time-varying models, such as wind gusts and sudden changes in load, by optimizing its feedback gain k. This method is regularly accompanied by estimators, such as the Kalman Filters (KFs), which estimate the state vector continuously, forming a Linear-Quadratic Gaussian (LQG) controller.

In [8], an LQR controller is used to help stabilize a quadrotor in a hovering state by calculating the K value through real experimentations. The study in [9] illustrates the application of Linear Quadratic Gaussian (LQG) control to manage the flight of a quadrotor. The approach employs a Linear Quadratic Regulator (LQR) for optimal control, ensuring the quadrotor follows a predefined spline path accurately and avoids obstacles. Additionally, the system integrates Inertial Measurement Unit (IMU) data to maintain stable orientation estimates. Figure 2.1 shows a schematic of an LQG controller used in a quadcopter.

Another regularly chosen method for linear controllers is **Proportional-Integral-Derivative (PID)** control because of its high reliability and versatility and the low computational effort it needs to operate and implement. This technique is already widely used in process industries and other control applications, being also effective for quadcopter hover stabilization. As Figure 2.2 demonstrates, a standard PID controller used in quadrotors fundamentally works by calculating the difference between the values of the desired setpoint and the output and formulating the error sig. Then, through the mathematical applications of the proportional, integral, and derivative terms, it is possible to obtain

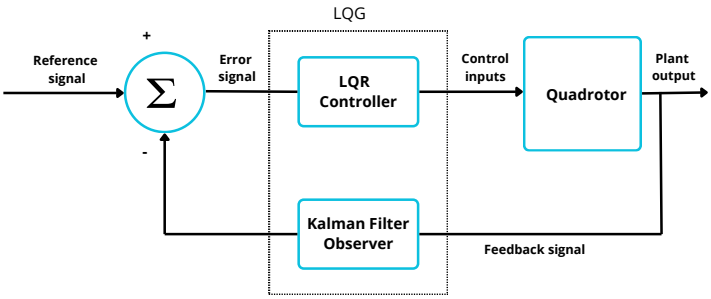


Figure 2.1: Block diagram of an LQR controller with Kalman Filters applied to a quadrotor (adapted from [10])

a more desirable response with the newly calculated setpoints. Study [11] investigates the utilization of PID regulators for quadcopter adjustment. It features PID control's reliability and flexibility in maintaining stable flight. Additionally, the research addresses streamlining PID boundaries utilizing meta-heuristic calculations to upgrade execution, exhibiting the adequacy of PID regulators in different control applications, including quadcopter hovering stabilization.

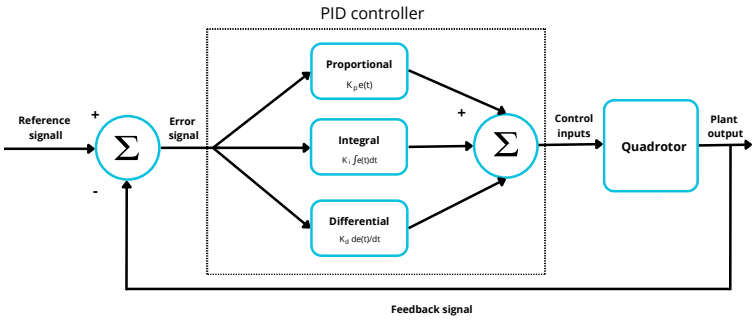


Figure 2.2: Block diagram of a PID controller applied to a quadrotor (adapted from [10])

it is important to notice that, although the growing technology has enabled the creation of more robust linear control strategies, few algorithms have proven to be as efficient as well-designed nonlinear solutions when controlling a UAV in a real Three-dimensional (3D) environment, mainly when dealing in the best possible way with its disturbances. In [10], explanatory and comparative work about various linear and nonlinear control methods applied to quadcopters can be seen.

2.1.2 Nonlinear control

Nonlinear techniques are much more robust and versatile solutions in modelling and controlling quadcopters than linear ones since they capture many complex and nonlinear behaviours present in these drone's flight, from its non-linear motion and physics

to typical uncertainties such as turbulence, varying wind conditions and aerodynamic interactions. However, this leads to some costs that engineers must carefully consider when approaching solutions for controlling and modelling drones in non-cooperative flight scenarios. Nonlinear systems, including unpredictable dynamics, chaos, and limit cycles, are tricky to model since the equations used can have varying coefficients. Furthermore, solving nonlinear equations and real-time control of these systems may often lead to increased computational demands and further financial costs for the project. Other essential tasks, like adjusting the controllers and parameters to get the desired outcomes and finding workable empirical solutions with no closed-form analytical ones, can take much time and slow down the whole process [10]. Some algorithms consistently show promising results when implemented in controllers to manage the flight of individual drones in this situation.

Backstepping Control is one of the most common overall techniques used for controlling UAVs, particularly for quadrotors, as it fits well with their cascade control structure due to their layered dynamics (position, attitude, and motor thrust), allowing for a systematic design process that shows promising results in maintaining their trajectory and external disturbances. This method operates in a recursive algorithm, dividing the controller into smaller subsystems that work as steps, gradually stabilizing each one of them. This is done by inserting intermediate control laws, referred to as "virtual controls" or "virtual variables", to describe some state variables of the system that is being controlled. Thus, this offers some advantages, as these controllers provide a straightforward, step-by-step, capable method for controller design that handles uncertainties and disturbances effectively. However, this comes with the cost of computational demands, mainly because of the abrupt growth of complexity when performing each differentiation continuously in the control laws. In [12], a more complex task is handled in which backstepping laws integrate a controller capable of effectively tracking the trajectory and handling the disturbances caused by a payload swing.

Another convenient method is using **Adaptive Control Algorithms**. They are beneficial for controlling systems with unknown or fluctuating characteristics, such as shuttle drones for capturing manoeuvres. Here, the controls lessen the impact of a range of erratic behaviours, including shifts in payload weight, aerodynamic changes from the surroundings, and closeness to other drones. Conventional controllers may not function as intended in these situations because they are usually made to manage fixed conditions. An example of these capabilities being applied can be seen in [13], where this technique is applied to a trajectory control of a quadrotor, successfully achieving stability and showing the convergence of the boundedness of the adaption parameters and tracking errors. Also, a comparison between this method and a non-adaptive version of it is made, showing that adding an adaptive algorithm provided better tracking performance. Moreover, a relevant approach is made in [14], where an adaptive tracking and perching scheme is proposed for quadrotors in dynamic scenarios, mainly focusing on the challenge of landing on moving platforms. This adaptive technique dynamically modifies the drone's flight route

depending on real-time perception, allowing it to retain a stable relative state concerning the changing platform. This approach works best when the platform's motion is erratic or other influences, such as wind gusts, may create variations in the drone's path. Promoting this adaptability is vital in practical applications, such as when the drone must land on a moving vehicle or be captured by one without requiring modifications or cooperation from the car.

Sliding Mode Control (SMC) is a very robust and simple technique with decades of functional use in various fields like robotics, control, and other scientific and industrial sectors. Its working principles consist of two different phases: firstly, the reaching phase, where the system is stabilized by the controller into a plane trajectory, reaching the sliding surface; and, secondly, the discontinuous control signal is applied to the system to instruct it to slide along its normal behaviour, as it is possible to see in Figure 2.3, also altering the dynamics of the system being controlled. While the system goes through this phase, the closed-loop response becomes independent from the exact reference model because it depends only on the chosen surface where it slides. Thus, this comes with advantages such as better disturbance rejection and lower sensitivity to parameter variations, which are very relevant in drone control [15]. In [16], an adaptive version of this method is used to better track and control a quadcopter and a PID controller for the sliding surface. A new flight controller is suggested in [17] that combines a nonlinear disturbance estimator with terminal sliding mode control to lower chattering. This controller takes into account both significant model uncertainties and outside disturbances. Experiments verify the controller's efficiency by simulating partial rotor failure and abrupt load shifts.

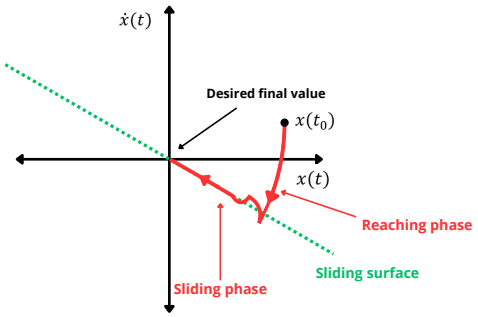


Figure 2.3: Slide Mode Control concept (adapted from [18])

In addition to Linear Model Predictive Control (LMPC), **Nonlinear Model Predictive Control (NMPC)** is a variation of model predictive control (Model Predictive Control (MPC)). This contemporary control approach considers several restrictions during the

control process, including roll angles and torque limits. It also handles MIMO (Multiple Input Multiple Output) systems and their interactions with effectiveness. As shown in Figure 2.4 [19], the NMPC controller uses an internal model of the dynamic system that is being controlled, the current plant measurements, and the desired targets and limits to predict changes in the dependent variables - which are often control objectives and other process limits. By doing this, the system can drive these variables to the desired values while respecting the constraints on dependent and independent variables. This preview capability allows the UAV to adopt the desired behaviour with enough time not to do it abruptly but rather in a smoother manoeuvre. The downside of this robustness is that powerful computational process capabilities and memory are required to find and store the solutions calculated by the algorithm. The paper [20] compares the two types of MPC for tracking paths in quadcopters. It finds that NMPC nonlinear system models are better at ignoring disturbances and responding swiftly to step changes. However, mitigating the excessive processing time remains challenging when operating in real-time. To get around this problem, the Newton generalized minimal residual (Newton/GMRES) method is successfully used on a quadcopter controlled by a high-level NMPC controller in [21].

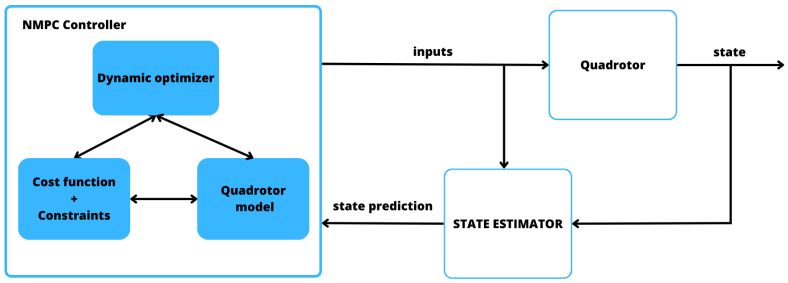


Figure 2.4: Basic NMPC controller loop applied for a quadrotor (adapted from [19])

Artificial Neural Networks (ANNs) offer a promising approach to nonlinear control for quadrotors. These networks are designed with a complex mathematical model biologically based on the human brain and nervous system and try to replicate their learning capacity. This is done by connecting artificial neurons, or perceptions, that work similarly to biological ones and forming a mathematical function that receives a signal from one or more inputs and sums them to produce an output.

As illustrated in Figure 2.5 [22], a neural network consists of three main layers: the input layer, the hidden layer, and the output layer. The input layer receives sensor data from the quadrotor, such as position, velocity, orientation, and angular rates. The hidden layer processes these inputs to learn complex patterns and relationships that are not easily captured by traditional control methods. This layer helps understand the dynamics of

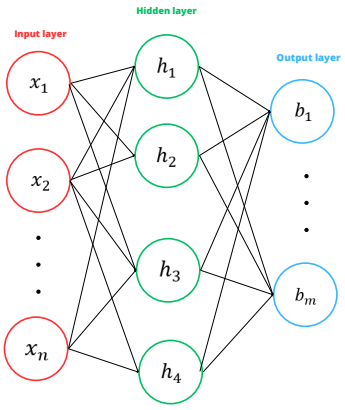


Figure 2.5: A Neural Network (adapted from [22])

the quadrotor and the effects of disturbances. Finally, the output layer generates control signals that adjust the quadrotor's motors to achieve the desired flight behaviour.

The neural network's training process involves processing examples with known "results" from known "inputs," creating probability-weighted associations between the two stored in the network's data structure. Training also involves continuously evaluating the difference between the predicted and target outputs, defining an error on which the network bases its learning rule, and using it to adjust the weighted associations. Through repeated adjustments, the network approximates its output to the desired one.

Therefore, this control method is suitable for a highly nonlinear vehicle like a quadrotor. It can effectively minimize the impact of changes in plant parameters and manage environmental disturbances by adjusting for uncertainties, making it more capable than traditional controllers that use fixed gain values. However, implementing this method typically requires complex and substantial computational resources, which is a significant limitation. Examples of this method applied to quadrotor control in [23], where successful real-time simulations were performed in a confined space. These simulations showed promising results in closed-loop stability and robustness against unknown uncertainties and in creating a real-time model to handle uncertainties encountered during flight in that specific environment.

In more recent years, **Hybrid Control** for UAVs has been increasingly researched and developed. With it, controllers can perform different interactions with the environment due to the method's characteristics, where it is possible to manage a problem or situation with different operational stages containing continuous, logical, and discrete states. In this way, usually, this option is applied to hybrid systems with distinctive behaviour, such as a hybrid UAV - a vehicle that can change between Vertical Take-off and Landing (VTOL) and fixed-wing flight mode - and, consequently, needs to be modelled for better control effectiveness as a hybrid system. In the context of this thesis, there's a need to handle the distinct phases present in our quadcopter's trajectory. Accordingly, at some

point, this shuttle drone must be able to follow another quadrotor, catch, carry, and release it. These can cause abrupt changes in direction, velocity, attitude, weight, and aerodynamics, which cause the need for a robust controller to effectively and safely achieve this trajectory. After capturing another vehicle, the controller implemented on the shuttle drone must be able to handle the abrupt change in weight while carrying it, maintaining stability as best as possible. In work [24], the problems with hybrid control schemes for underactuated quadrotors are discussed, along with the effect of dynamic factors that are not modelled. This offers valuable perspectives on stable trajectory monitoring and stabilisation. Valuable work can also be seen in [25], which exposes a relevant project where a quadrotor is set to exchange a package with another one, needing, as in this project, to carry a payload and coordinate its flight with another drone. To do that, a Hybrid Model Predictive Controller-based system controls the trajectory and the gripping mechanism.

2.2 Quadrotor control strategies for cooperative manoeuvres

Cooperative manoeuvres require coordinated control strategies that ensure stability and precise flight. Challenges include real-time coordination in dynamic environments, security concerns, and energy management [26].

One of the forms of devising these strategies is using distributed control architectures where each quadcopter makes autonomous decisions based on local coordination and information with neighbouring quadcopters. With this, it is possible to design scalable and robust control of quadrotor swarms [27]. Furthermore, a group of drones can converge to a standard state or decision by performing local interactions and reaching a consensus on a specific task, such as formation flying, flocking, or rendezvous. Therefore, these algorithms are called consensus-based algorithms. The work done in [28] shows an example of this strategy for trajectory planning for multiple UAVs in dynamic and uncertain environments.

Behaviour-based architectures decompose complex cooperative behaviours into more straightforward, modular behaviours that can be combined. Consequently, more flexible programming frameworks can be defined, potentially serving as an excellent response to a meticulous manoeuvre such as an autonomous capture of another UAV. One fundamental behaviour in this context is leader-following, when the shuttle quadrotor can follow that UAV's trajectory. For instance, in [29], a method is implemented for n follower quadrotors to be able to steer another one, the leader, while dealing with environmental disturbances in a real-time three-dimensional trajectory. Each follower can plan its converging trajectory individually, reducing the need to communicate with other vehicles, and the leader can adjust his motion to his followers so that each one of them can steer the leader. Testing experiments have also successfully proved this approach, obtaining more credibility.

2.3 Aerodynamic modelling

To achieve precise and safe operations, such as capturing another quadrotor, meticulous modelling of both drones becomes essential. This modelling process must account for motion, physics, and aerodynamics, particularly as aggressive manoeuvres and airflow interactions between drones add further challenges.

One of the most common aerodynamic factors implemented when modelling quadrotors is their frame drag. In [30] and [31], the importance of this drag is highlighted, as neglecting it can lead to more considerable trajectory tracking errors and power consumption, especially during agile manoeuvres or at higher speeds.

When studying the interaction between the quadcopter and the surrounding air, observing how the last one affects the drone's rotors is essential. For that, the work done in [32] explores this effect in various flight conditions, such as the ground effect (proximity to the horizontal surfaces) and near other quadcopters. Nevertheless, it only evaluates the impact of quadcopters in the same horizontal plane. Moreover, some of these authors in [33] determine "safe regions" around each vehicle in a team setting by empirically quantifying the effect of the downwash produced by the rotors of one drone on another flying below the first.

On another perspective, an interesting approach is taken in [30] where a propeller model is designed combining essential concepts such as Blade Element Theory with quadrotor classical dynamics to predict the side force, pitching moment and additional thrust generated by each propeller. By adding the shielding effect (not all rotors are exposed to the wind in the same way), the work produces a satisfactory solution to the problem of the wind effect. In [34], to have a vehicle flying underneath another, Momentum Theory is applied to model the downwash the quadrotor produces. Here, the designed controller is incorporated with a recursive Bayes filter to estimate the downwash flow field's parameters. When performing the respective simulations, despite the results showing that a quadrotor can hover below the first successfully, the propeller diameter of the quadcopter is never disclosed; therefore, this outcome is somewhat inconclusive.

In [35], it is also possible to calculate the rotor drag, where the author proves that a quadcopter subjected to a linear version of this effect is differentially flat in its position and heading. With this property, feed-forward control terms are calculated directly from the reference trajectory and used in a cascaded, nonlinear feedback control law that lets quadrotors fly accurately and quickly on paths that they do not know about beforehand. Therefore, the respective results show that the tracking position error is reduced significantly, especially in agile manoeuvres.

2.4 Flight simulation and testing

Also, to conduct our studies, it is necessary to know how to perform flight tests with quadrotors, studying how to assemble a controlled space where real experimental try-outs

can be performed. In this way, this section aims to expose relevant work about quadcopter flight simulation and testing procedures.

The literature reveals that MATLAB is widely used for simulating algorithms and conducting research, given its simplicity and low computational requirements compared to other methods. However, other, more realistic environments can be used to devise more credible simulations. For example, in [36], Robot Operating System (ROS) - an open-source framework that allows the writing of robot software (further information regarding this software can be found in [37]) - is used along with its interface to the simulation environment Gazebo and a firmware PX4 autopilot system that controls the vehicle. The first tool offers an open-source virtual environment capable of simulating robot behaviour by providing a robust physics engine with multiple options that can accurately simulate real-life conditions for indoor and outdoor scenarios. More detailed information about this simulator can be seen in [38]. On the other hand, PX4 is a flight control system for low-cost UAVs, and, in this case, it connects to the ROS through MAVROS protocol, delivering finer real-time testing samples through its integrated simulator.

Further information regarding PX4 and its vast available capabilities can be explored at [39]. A connection between Simulink and PX4 is also established to test the developed controller. Figure 2.6 depicts the architecture of this system.

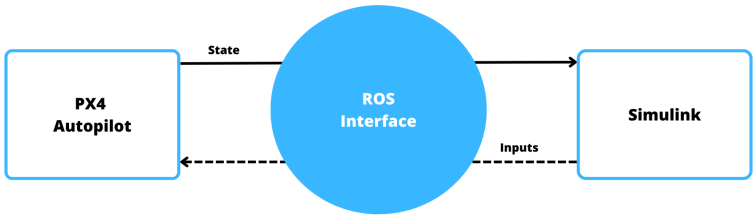


Figure 2.6: Virtual simulations system's architecture

Two major categories of tests can be considered for the project where this thesis is inserted: one to check a single quadrotor's introductory flight, performing other slow-speed and low-distance manoeuvres indoors, such as take-off, hovering, and landing, and, in an outdoors environment, the second controller responsible for drone aggressive manoeuvring will be put to the test. Each type of test will require separate hardware and software systems.

In [40], a valuable method to develop a safe environment to perform essential indoor flights with one and multiple drones is suggested. The UAV simulation flight is done in a controlled environment: a testbed, a sort of arena with all the sensors and guard measures

to provide the best conditions for UAV experimental flights. The drone's trajectory is followed with good precision and low latency using the Marvelmind Indoor Navigation System, which communicates through a modem to the QGroundControl software. The trajectory data is transmitted to a ROS topic and sent to the PX4 mini autopilot via a Wireless Fidelity (Wi-Fi) module (ESP8266), ensuring low latency in the overall system. Figure 2.7 illustrates the high-level architecture of this system.

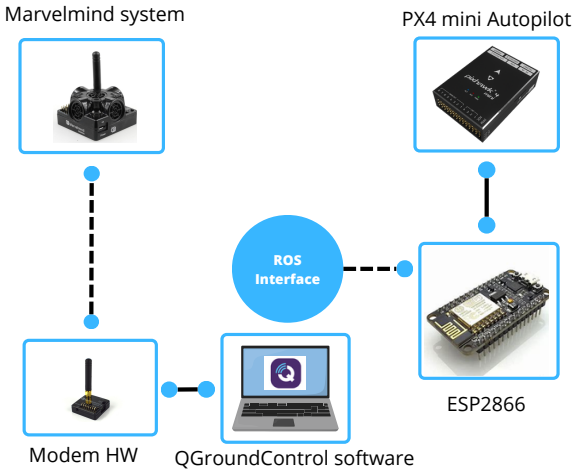


Figure 2.7: System architecture for indoor drone navigation and control (the dashed lines represent a Wi-Fi connection)

On the other hand, outdoor quadrotor testing presents more complex challenges compared to the previous scenarios. One of the biggest problems of these types of operations is the unpredictable effects of environmental factors, such as wind, rain, temperature changes, etc., which can significantly affect the performance, battery life, navigation, and control of the quadrotor. Also, suppose a UAV loses control or crashes. In that case, these tests can pose safety risks to people and property, ensuring that these procedures are conducted in a safe environment, away from people and objects that could be damaged, and comply with the location's regulatory restrictions. Moreover, another central issue of outdoor quadrotor testing is its navigation and positioning, which often rely on the Global Positioning System (GPS) system. This method can be problematic because the signals used by this system to communicate with the drone can be weak or unreliable in specific environments, such as urban areas with tall buildings or dense forests, disturbing the testing procedure's effectiveness.

The research paper [41] includes a section on outdoor testing. The authors describe how the MBZIRC 2020 challenge was conducted outdoors, focusing on the autonomous capturing of agile flying objects using UAVs. They provide a detailed description of the outdoor test arena and highlight the difficulties of performing the challenge in an outdoor environment, such as dealing with wind and other weather conditions and ensuring the safety of participants and bystanders. The paper also includes visual materials, such as videos and images, to illustrate the outdoor testing during the challenge.

3

Basic drone modelling and control

As previously mentioned, one of the objectives of this thesis is to design and implement a controller capable of following predefined, basic trajectories with precision. The algorithm for this controller will serve as the "foundation" for a second and more robust one, which will later be compared for a performance study.

Therefore, this chapter presents, first, some background theory about quadcopter movement and dynamics in 3.1. Then, the techniques behind modelling and controlling the shuttle quadrotor are devised in 3.2 and 3.3, respectively. Table 3.1 displays the constant values used throughout this chapter.

3.1 Drone motion

To effectively control a quadcopter, it is vital to understand its motion capabilities and limitations in a three-dimensional environment. This understanding, combined with the project's goals, will allow for creating an accurate mathematical model.

Accordingly, this vehicle has six Degrees of Freedom (DOF), including translational movements along the X, Y, and Z axes and rotational motion, commonly known as attitude, which can be described by Euler angles for pitch, roll and yaw, as depicted in Figure 3.1.

To perform this movement, this type of vehicle uses its four control inputs, representing the speed of each rotor, and by having a lower number of control inputs than degrees of freedom from where it can move, the quadrotor can be characterized as an underactuated system. For instance, while the drone can move along the z-axis without needing to variate any other state, the same doesn't happen for the x and y-axis, as it needs to change its attitude.

The motion in these directions can be achieved by having the following propeller speed variation [43]:

- Changing the speed of all propellers simultaneously will generate vertical z motion;
- Varying speed 2 and 4 rotors inversely will make a roll rotation;
- Varying the speed 1 and 3 rotors inversely will produce a pitch rotation;

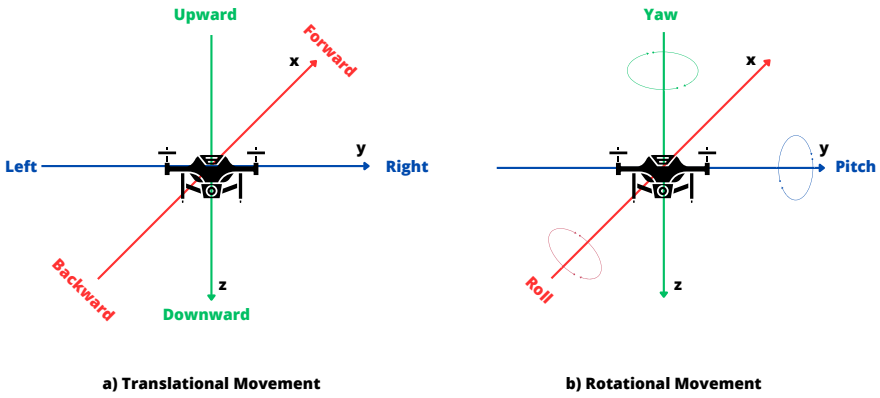


Figure 3.1: Quadrotor's 6 degrees of freedom (adapted from [42])

- The difference in the counter-torque between each pair of rotors will generate a yaw rotation.

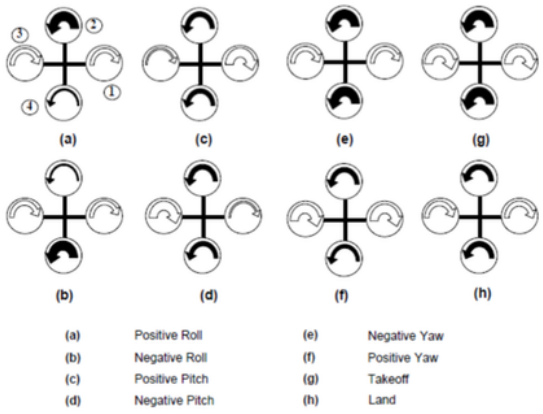


Figure 3.2: Quadrotor operating principle, the relationship between propellers speeds and vehicle movements (adapted from [43])

This proves that quadrotor motion is a complex theme. By adding disturbances of real-world environments, which are usually difficult to integrate and model, the problem of devising feasible control solutions for quadcopters becomes more complex. Therefore, in the following sections, some modelling and control techniques to tackle this problem will be presented and analyzed to see their impact.

3.2 Drone modeling

3.2.1 Reference frames

To represent the complete UAV motion in a 3D environment, one must consider two references: one for the environment where the drone is flying, the world frame W, and another for the body of the vehicle, the body frame B. The **world frame (or inertial frame)** is fixed in space and remains stationary. It is an external reference for the quadcopter's position and orientation, or angular position. The origin is typically at a fixed point on the Earth's surface (e.g., a GPS location), and the orientation of the three axes usually aligns with the cardinal directions. There are two main right-handed coordinate system variants: East, North, Up (ENU) and North, East, Down (NED) coordinates. We choose the latter, which is illustrated in Figure 3.3, and its three axes are:

- $x_W = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$: Points north (toward the geographic north pole).
- $y_W = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$: Points east (perpendicular to the X-axis).
- $z_W = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$: Points down (toward the Earth's centre).

We chose this coordinate system because, in aerospace applications, NED coordinates are frequently more naturally compatible with the aircraft's navigation and control systems. The z-axis pointing downward makes altitude and vertical speed easier to comprehend because positive numbers denote a descent, which aligns with many aviation traditions.

The **body frame** is directly tied to the quadcopter itself, being the vehicle's centre of mass, the origin (centre) of the frame. It moves and rotates with the drone, referencing its linear and angular velocities. Therefore, the three axes of this frame are:

- x_B: Points forward (along the drone's nose).
- y_B: Points to the right (perpendicular to the X-axis).
- z_B: Points downward.

A third intermediate frame $C = \begin{bmatrix} x_C & y_C & z_C \end{bmatrix}^T$ considering just the reference yaw rotation angle, ψ_{ref}, is also used to help facilitate control design, as it bridges the computation gaps between the two main frames.

3.2.2 Rotation representation

Representing the rotational dynamics of the quadcopter is essential for accurate modelling. This motion accounts for half of the drone's degrees of freedom and is also highly nonlinear due to aerodynamic effects and the constant need to balance torques and forces.

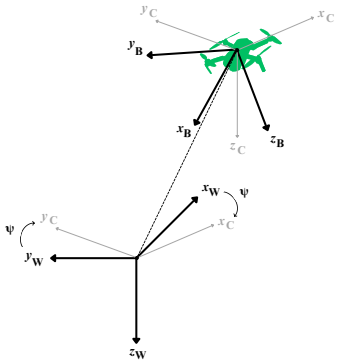


Figure 3.3: Reference frames

The two most common methods for representing rotation angles are Euler angles and quaternions. The chosen technique for this thesis primarily considers how the quadrotor is being used and the available computational resources.

3.2.2.1 Rotation Matrix and Euler Angles

In this first method, the orientation of the drone can be defined as transformations between the world and body frames where a rotation matrix, R, is used to describe the orientation of the body frame on the world frame, enabling the conversion of the coordinates from one frame to another as

$$\begin{bmatrix} x_B \\ y_B \\ z_B \end{bmatrix} = R \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix}. \tag{3.1}$$

To define this matrix, Euler angles (ϕ, θ, ψ), introduced by Leonhard Euler [44], are commonly used since they provide an intuitive way to describe rotations around a rigid body's fixed axes. We consider rotation given by Z-Y-X [45], which is defined as a sequence of these three elementary rotations: a roll rotation (ϕ) about the x-axis, a pitch rotation (θ) about the y-axis, and yaw (ψ) about the z-axis, as defined in Figure 3.1.

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \tag{3.2}$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{3.3}$$

$$R_z(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.4}$$

Therefore, the rotation matrix is determined by calculating the matrix product of the three, accounting for the rotation order as

$$R_{zyx} = R_z(\psi)R_y(\theta)R_x(\phi). \tag{3.5}$$

Finally, the complete rotation can be obtained as

$$R = \begin{bmatrix} c(\theta)c(\psi) & s(\phi)s(\theta)c(\psi) - c(\phi)s(\psi) & c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi) \\ c(\theta)s(\psi) & s(\phi)s(\theta)s(\psi) + c(\phi)c(\psi) & c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi) \\ -s(\theta) & s(\phi)c(\theta) & c(\phi)c(\theta) \end{bmatrix} \tag{3.6}$$

where, for notational simplicity, $c = \cos$ and $s = \sin$.

3.2.2.2 Quaternions

While Euler angles are widely used, other representations for rotations exist. William Rowan Hamilton introduced one notable alternative: quaternions [46]. Quaternions offer several advantages over Euler angles, including greater mathematical complexity and the ability to avoid the issue of gimbal lock, also discussed in [44]. This singularity can occur with Euler angles: two angles become dependent on each other, leading to a loss of information, which can cause problems in control algorithms.

A quaternion is a four-dimensional number that extends the concept of complex numbers, typically represented as

$$q = a + bi + cj + dk \tag{3.7}$$

where a is the real part; b, c, and d are the imaginary parts; and i, j, and k are the quaternion units (similar to the imaginary unit i in complex numbers, being $i^2 = j^2 = k^2 = ijk = -1$). A parameterization of the quaternions with a Z-X-Y sequence of Euler angles can be obtained as

$$q_{313}(\phi, \theta, \psi) = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} \cos\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} - \sin\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2} \\ \cos\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2} + \sin\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2} \\ \sin\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} + \cos\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2} \\ \sin\frac{\phi}{2}\sin\frac{\psi}{2}\sin\frac{\phi}{2} + \cos\frac{\psi}{2}\cos\frac{\phi}{2}\cos\frac{\theta}{2} \end{bmatrix}. \tag{3.8}$$

However, Euler angles are often preferred for quadcopter applications due to their more straightforward interpretation of roll, pitch, and yaw while demanding less computational complexity. More detailed information about this topic can be found in [46].

3.2.3 System's dynamics

To effectively model the dynamics of this vehicle, we also need to determine and study the forces that act on it, influencing its motion and stability. We consider the following:

- **Thrust (T)**: Thrust is a perpendicular force to the drone's plane that results from the propellers' rotation. It enables the drone to ascend, descend, or hover in the air. The collective thrust (T_R) controls the drone's vertical motion.
- **Aerodynamic Drag (f_a)**: Aerodynamic drag is the force that the air exerts on the drone's body and propellers as it moves through the air. It opposes the direction of motion and can affect the drone's speed and stability.
- **Drone's Weight (f_w)**: The quadrotor's weight, including its frame, motors, battery, and payload, directly affects its ability to maintain altitude. This force is aligned with the z_W axis but in the opposite direction. Therefore, balancing the drone's weight and thrust is pivotal for stable flight, as when thrust exceeds weight, the drone ascends, and when weight exceeds thrust, it descends.

In addition to these primary forces, wind is also considered in the drone's flight dynamics. External wind forces can push the drone off its intended path by affecting both position and orientation. In Figure 3.4, we illustrate how these forces act on the quadrotor's body as it moves horizontally at a speed of v in a wind-free environment. Each force is represented by its corresponding vector (with no requirement for coherence in dimensions).

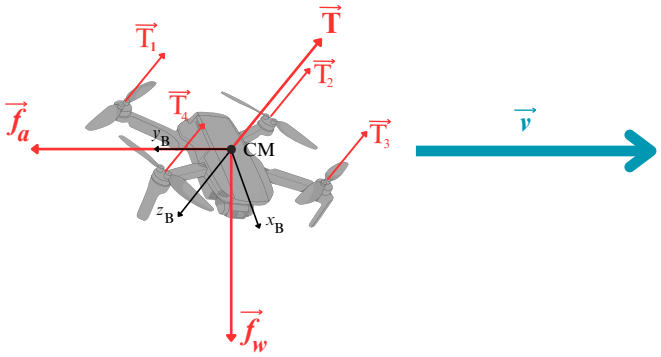


Figure 3.4: Illustration of the forces acting on a quadrotor during horizontal motion at a speed of v, neglecting the wind.

Now, to describe the system's model state, we introduce a state vector x:

$$x = \begin{bmatrix} p^T & \text{vec}(R)^T & v^T & \omega^T \end{bmatrix}^T$$

In this representation, $p = \begin{bmatrix} x & y & z \end{bmatrix}^T$ denotes the drone's position in the world frame, $v = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T$ represents its linear velocity, $\omega = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T$ describes the

angular velocity of the body frame B relative to the world frame W, and R is the rotation matrix formulated in 3.6. The term vec(R) refers to the vectorization of matrix R into a column vector.

These state variables are foundational for modelling the drone's dynamics. By deriving each variable, we obtain the following dynamical equations, which govern the drone's motion:

First, the equation describing the rate of change of position is given by

$$\dot{p} = v \tag{3.9}$$

being $\dot{p}$ equal to the linear velocity, v. Then, the rate of change of linear velocity is represented by

$$\dot{v} = -gz_W + \frac{u_1}{m} z_B \tag{3.10}$$

being $-gz_W$ the gravitational force acting on the drone, where g is the acceleration due to gravity, and z_W denotes the upward direction in the world frame. The term $\frac{u_1}{m} z_B$ refers to the mass-normalized collective thrust, u_1, generated by the rotors, which counteracts gravity and controls the drone's vertical motion. The rate of change of the angular velocity vector relative to the body frame is described by

$$\dot{\omega} = I^{-1}(-\hat{\omega} I \omega + u_r) \tag{3.11}$$

Here, $I = diag(I_{xx}, I_{yy}, I_{zz})$ is an inertia matrix with diagonal elements that characterize the mass distribution within the UAV and its influence on rotational behaviour. The moment of inertia values are adapted to our drone following the work in [47]. Moreover, the term $-\hat{\omega} I \omega$ represents the torque due to angular velocity, and u_r represents the input torques acting on the drone. Finally, the equation describing the rate of change of the rotation matrix is

$$\dot{R} = R\hat{\omega}. \tag{3.12}$$

The skew-symmetric matrix $\hat{\omega}$, derived from the angular velocity vector ω, is essential for transforming angular velocities between frames. It is defined as

$$\hat{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{3.13}$$

where each element of the skew-symmetric matrix corresponds to the cross product of the angular velocity vector with the unit vectors along the axes of rotation. This matrix facilitates the transformation of angular velocities between the body-fixed frame and the world frame, enabling accurate modelling of the drone's rotational dynamics.

3.3 Drone control

For the drone to hover and move from one point to another while maintaining stability, a controller based on the one developed in [48] was implemented. As most controllers try to approximate a system to a reference value, this one will do it by trying to guide the controlled state variables to zero, minimizing the error between the reference value and the state's current one. This means the system follows the path even closer as the error approaches the value zero. Also, a quadrotor can rotate around any axis in the horizontal plane of its body. This means that reducing the snap or 4th derivative of independent splines in three dimensions will reduce the sudden changes in control inputs needed to stay on the path. In this case, we need to compute the position and velocity errors, e_p and e_v, that can be defined as

$$e_p = p - p_{ref} \tag{3.14}$$

$$e_v = v - v_{ref}. \tag{3.15}$$

These values will constantly be used by the controller to calculate the desired force vector f_{des}, which will try to follow as

$$f_{des} = -K_p e_p - K_v e_v - K_{ip} e_{ip} + mgz_W + ma_{ref} \tag{3.16}$$

where K_p, K_v and K_{ip} are the controller gains for position, velocity, and the integral of the position's error, respectively, and are defined as positive definite diagonal matrices. Also, the terms mgz_W and ma_{ref} represent the gravitational force and the inertial force due to acceleration reference, respectively. These terms ensure the control input accounts for the gravitational and inertial effects on the quadrotor's motion.

Then, we can obtain the first control input, the thrust one, u_1, by calculating the scalar projection of f_{des} onto z_B, such as

$$u_1 = f_{des} \cdot z_B. \tag{3.17}$$

This first control input directly affects the UAV's vertical motion by adjusting the thrust force. The other three control inputs, $u_\tau = \begin{bmatrix} u_2 & u_3 & u_4 \end{bmatrix}^T$, which relate to the body's attitude, are obtained by computing the remaining errors, rotation and angular velocity, e_R and e_ω respectively. To do that, we need to calculate the desired rotation matrix, $R_{des} = \begin{bmatrix} x_{B_{des}}, y_{B,des}, z_{B,des} \end{bmatrix}$, operating on its coordinates. First, taking in account that $\left\| f_{des} \right\| \neq 0$), we are able to obtain $z_{B,des}$ as

$$z_{B,des} = \frac{f_{des}}{\left\| f_{des} \right\|}. \tag{3.18}$$

Now, we will also need to use the intermediate coordinate frame $C = \begin{bmatrix} x_C & y_C & z_C \end{bmatrix}$ to obtain $x_{C_{des}}$ as

$$x_{C,des} = \begin{bmatrix} \cos\psi_{ref} & \sin\psi_{ref} & 0 \end{bmatrix}^T. \tag{3.19}$$

With this, we can obtain another component of the desired rotation matrix, the $y_{B,des}$ value, by performing the orthogonal between $z_{B,des}$ and $x_{C,des}$, supposing $\left\| z_{B,des} \times x_{C,des} \right\| \neq 0$, as

$$y_{B,des} = \frac{z_{B,des} \times x_{C,des}}{\left\| z_{B,des} \times x_{C,des} \right\|}. \tag{3.20}$$

Finally, we can compute the vector, $x_{B_{des}}$, as

$$x_{B_{des}} = y_{B,des} \times z_{B,des}. \tag{3.21}$$

Also, we can obtain the rotation error, which quantifies the discrepancy between the desired and actual orientations of the quadrotor, enabling the controller to adjust the attitude accordingly. This error is written as

$$e_R = \frac{1}{2}(R_{des}^T R - R^T R_{des})^\vee \tag{3.22}$$

where $\vee$, vee map, is the inverse calculation of the skew matrix, transforming its elements from Special Orthogonal group in three-dimensional space (SO(3)) to $\mathbb{R}^3$.

Now, to define the desired angular velocity ω_{des}, we need to work with the formula of the derivative of the acceleration $\dot{a}_{ref}$, which is

$$m\dot{a}_{ref} = \dot{u}_1 z_{B,des} + R_{des}\omega_{des} \times u_1 z_{B,des} \tag{3.23}$$

and project it along z_B, knowing that $\dot{u}_1 = z_B \cdot m\dot{a}$, by creating the vector $h_{w,des}$ as

$$h_{w,des} = (R_{des}\omega_{des}) \times z_{B,des} = \frac{m}{u_1}(\dot{a}_{ref} - (z_{B,des} \cdot \dot{a}_{ref}) z_{B,des}) \tag{3.24}$$

which is the projection of $\frac{m}{u_1}\dot{a}$ on the $x_B - y_B$ plane. From here, using the definition of $h_{w,des}$ and the already known values of $y_{B,des}$ and $x_{B,des}$, it's possible to express $\omega_{\text{x,des}}$ and $\omega_{\text{y,des}}$ as

$$\omega_{x,des} = -h_{w,des} \cdot y_{B,des} \tag{3.25}$$

$$\omega_{y,des} = h_{w,des} \cdot x_{B,des} \tag{3.26}$$

and, therefore, calculate $\omega_{z,des}$ by resolving the following system, which relates the derivatives of the Euler angles and the angular velocity, written as

$$R_{des} \begin{bmatrix} \omega_{x,des} \\ \omega_{y,des} \\ \omega_{B_{z,des}} \end{bmatrix} = \begin{bmatrix} x_{C,des} & y_{B,des} & z_{B,des} \end{bmatrix} \begin{bmatrix} \phi_{des} \\ \theta_{des} \\ \psi_{des} \end{bmatrix}. \tag{3.27}$$

Ultimately, we can establish the desired angular velocity, which signifies the target rotation rates around the body-fixed axes. These rates are fundamental for attaining the intended orientation of the quadrotor. Therefore, this velocity is defined as

$$\omega_{des} = \begin{bmatrix} \omega_{x,des} & \omega_{y,des} & \omega_{z,des} \end{bmatrix}^T. \tag{3.28}$$

After this, we can also measure the deviation between the actual and desired angular velocities, which guides the adjustment of control inputs to stabilize the quadrotor's motion by calculating the error

$$e_\omega = \omega - \omega_{des} \tag{3.29}$$

and, with it, obtain the remaining control inputs u_τ as

$$u_\tau = -K_R e_R - K_\omega e_\omega \tag{3.30}$$

being K_R and K_ω positive definite diagonal matrices of the rotation and angular velocity gains, respectively.

Parameter	**Unit**	**Value**
mass	kg	3.55
g	m/s	9.81
I_{xx}	$kg.m^2$	2×10^{-2}
I_{yy}	$kg.m^2$	2×10^{-2}
I_{zz}	$kg.m^2$	3×10^{-2}

Table 3.1: Constant values for basic drone modelling and controlling

4

Aerodynamics modelling and control

Different aerodynamic parameters affect the quadcopter's motion and depend on the drone's physical attributes, flying environment, and flight trajectory. This chapter is structured in two main parts: one for describing modelling and controlling strategies for aerodynamic interactions found in the case of a single drone flying (4.1) and the other for two drones flying close to each other (4.2). After these, a description of the procedures taken to obtain some aerodynamic coefficients is also given (4.3).

4.1 Single drone

As shown in Chapter 3, we consider the drone subjected to an aerodynamic force, f_a, that opposes the drone's motion. This force accounts for the drag the quadrotor's rotors and fuselage produce. Therefore, we can formulate the aerodynamic force as

$$f_a = f_{rd} + f_{fd} \tag{4.1}$$

where f_{rd} and f_{fd} are the rotor drag and frame drag forces, respectively. Moreover, following Newton's second law, we can obtain the equivalent accelerations for these effects as $a_{rd} = \frac{f_{rd}}{m}$ and $a_{fd} = \frac{f_{fd}}{m}$.

4.1.1 Rotor drag

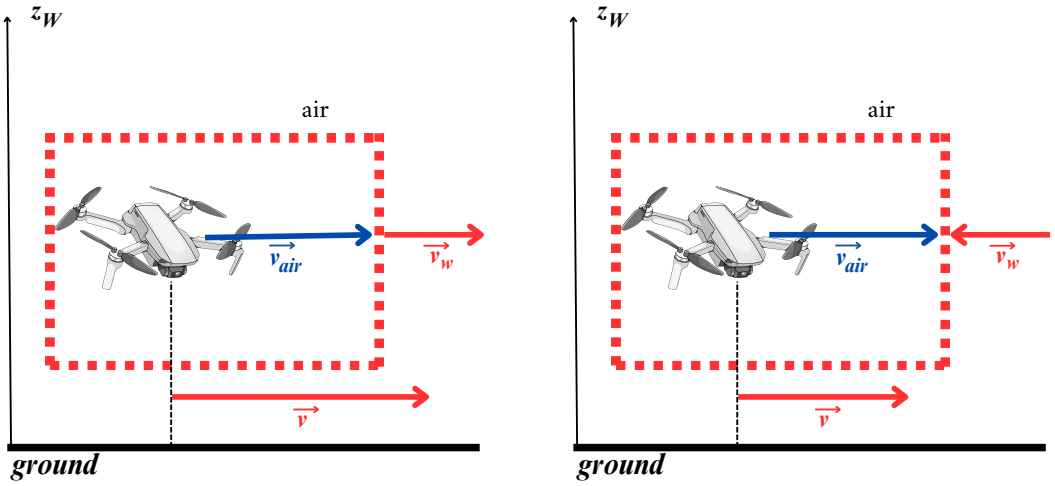
Following the work developed in [35], we consider the rotor drag effect for this model, inferring in the same manner that it is deferentially flat in its position and heading. This acceleration term can be calculated as

$$a_{rd} = -RC_{rd}v_{Bair} \tag{4.2}$$

where R is the rotation matrix from the body frame to the inertial frame, $C_{rd} = diag(C_{rd,xx}, C_{rd,yy}, C_{rd,zz})$ is a diagonal matrix containing rotor drag coefficients, and $v_{Bair} = R^T v_{air}$ is the drone air relative velocity defined in the body's frame. Therefore, following the idea in [49], the relative air velocity, $v_{air} = \begin{bmatrix} v_{air,x} & v_{air,y} & v_{air,z} \end{bmatrix}^T$, is calculated as

$$v_{air} = v - v_w \tag{4.3}$$

where $v_w = \begin{bmatrix} v_{w,x} & v_{w,y} & v_{w,z} \end{bmatrix}^T$ is the wind velocity felt on the drone's body, and v is the drone's speed in the world frame. As mentioned in [49], if the drone is travelling with a headwind (Figure 4.1b), the wind vector will have non-positive values in the x and y components. This means the wind is acting against the drone's forward motion, increasing its airspeed for the same components. In the opposite situation, where the drone travels with a tailwind (Figure 4.1a), these components will have non-negative values that reduce the drone's air relative velocity.



(a) Drone flying with tailwind

(b) Drone flying with headwind

Figure 4.1: Drone air relative speed (adapted from [49])

After this, some new changes are applied to the rate of change of the linear velocity, which is influenced by various forces acting on the drone, including gravitational force, collective thrust, and now rotor drag, and can be formulated as

$$\dot{v} = -g z_W + \frac{u_1}{m} z_B + a_{rd}. \tag{4.4}$$

Furthermore, the authors in [35] also present a new formulation for the angular velocity dynamics. Still, as these effects are typically less significant than those of the linear velocity dynamics, this effort is left for future work in the scope of this thesis. Instead, we use Equation (3.11) and only consider the effects on the velocity of the aircraft's frame.

4.1.2 Frame drag

While moving through the air, the quadrotor's frame experiences resistance due to its shape and interaction with the surrounding airflow. This resistance, quantified by the

drag coefficient (C_d), affects the drone's ability to manoeuvre efficiently, especially in windy conditions. It alters the drone's flight dynamics, requiring more significant control inputs to achieve desired manoeuvres.

To model this effect, we take into account the work developed in [31] where this drag acceleration is defined as

$$a_{fd} = -\frac{1}{2m}\rho C_d A v_{Bair} \odot v_{Bair} \odot sgn(v_{Bair}) \tag{4.5}$$

being C_d the drag coefficient, v_{air} the relative air speed to the drone's body, $A = diag(A_{xx}, A_{yy}, A_{zz})$ the platform's projected area in the corresponding plane and $\odot$ the element-wise multiplication between two matrices. It's important to note that this reference area remains constant regardless of the vehicle's orientation because, in [31], it has been chosen to calculate the frame drag directly in the body frame. The drag coefficient is assumed to be the same when the drone is flying in any direction along a body axis because of the symmetry of the quadrotor body frame in the xy plane and for simplicity.

This way, we could, once again, redefine a new dynamical equation for the drone's velocity (already including the rotor drag acceleration), inserting the frame drag acceleration as

$$\dot{v} = -gz_W + \frac{u_1}{m}z_B + a_{rd} + a_{fd}. \tag{4.6}$$

4.1.3 Rotor drag and frame drag compensation

To predict and compensate for the effects of the aircraft's propeller drag, we use the feed-forward control terms suggested in [35], which implies that the states $[p, v, R, w]$ and the thrust inputs $[u_1, u_t]$ can be written as algebraic functions of four specific flat outputs and also a finite number of their derivatives. For this purpose, we change the desired force defined before in 3.3 by subtracting the rotor drag force when calculating the desired force of the vehicle's body, such as

$$f_{des} = -K_p e_p - K_{ip} e_{ip} - K_v e_v + m(gz_W + a_{ref}) - f_{rd}. \tag{4.7}$$

Also, in [35], a quality comparison study of other controllers that consider the rotor drag effect is done. It shows that the proposed controller, which used the differential flatness property, was better at tracking the trajectory than the others.

To compensate for the drone's body drag, we subtract the force f_{fd} to the drone's desired force f_{des} in the controller, as

$$f_{des} = -K_p e_p - K_{ip} e_{ip} - K_v e_v + m(gz_W + a_{ref}) - f_{rd} - f_{fd}. \tag{4.8}$$

Having established the aerodynamic influences on a single drone, we now examine the complex interactions that arise when two drones operate in close proximity.

4.2 Two interacting drones

This section analyses a solution for modelling and compensating for the aerodynamic interactions between two quadrotors flying in close proximity.

Firstly, to better distinguish the mathematical formulas featured in each of the drones' algorithms, we will assume that the aircraft flying above is Drone 1 and the other is Drone 2. Furthermore, we assume the two drones are identical in shape and size, utilizing the same physical components and materials. Consequently, the parameters A, m, C_d, and C_{rd} are identical for both drones.

In this way, following the work done in [34], we start by calculating vehicle 1's induced speed, V_{i_1}, using momentum theory, obtaining

$$V_{i_1} = \sqrt{\frac{T}{2\rho A_d}} \tag{4.9}$$

where ρ is the air density, A_d is the rotor disk area and T is the total thrust produced by the drone. As a result, it's possible to compute an estimate of the vertical velocity $V_{d_1}(z)$ at any point z below the rotor height z_r as

$$V_{d_1}(z) = V_{i_1} + V_{i_1} \tanh\left(-k\frac{z_r - z}{h}\right). \tag{4.10}$$

The parameters k, h shape and control the rate at which the area of the streamtube below the rotor contracts to its steady-state value. Moreover, momentum theory assumes that V_{d_1} is uniform over the xy plane for a given z position. However, as the boundary of the downwash contracts due to acceleration, the radial condition for the vertical velocity reaching zero changes to

$$r_d < \frac{r}{\sqrt{1 + \tanh\left(-k\frac{z_r - z}{h}\right)}} \tag{4.11}$$

where r_d is the radial distance from the rotor centre and r is the rotor radius. By using this, it is possible to determine the vertical velocity at any given point in the flowfield as

$$v_{dw_1} = \begin{cases} V_{d_1}, & \text{if } r_d < \frac{r}{\sqrt{1+\tanh\left(-k\frac{z_r - z}{h}\right)}} \\ 0, & \text{otherwise.} \end{cases} \tag{4.12}$$

Moreover, it is essential to note that this model only applies to separations exceeding 1-2 rotor radii, as the downwash is more likely to stabilize in vertical flow at this distance from the rotor.

After calculating the velocity of downwash in the z-axis from the drone flying above, we reformulate the downwash velocity to a three-dimensional vector as $v_{dw_1} = \begin{bmatrix} 0 & 0 & -V_{d_1} \end{bmatrix}^T$. The term $-V_{d_1}$ is justified because the downwash velocity vector, v_{dw_1}, has a z-component

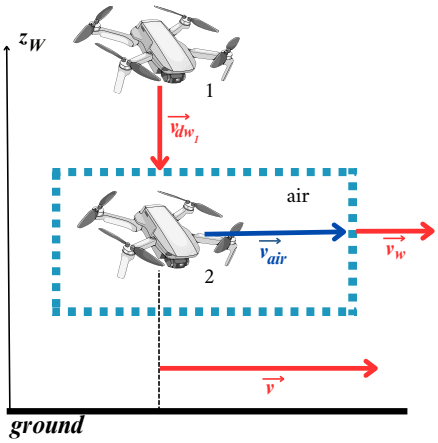


Figure 4.2: Target drone's air relative velocity under downwash from the drone above

that is always non-positive due to being opposite to the shuttle drone's z-axis direction, as illustrated in Figure 4.2.

After this, we incorporate this velocity in the air relative velocity of the drone, considering that this airflow impacts the drone below similarly to a wind gust. The airspeed of the lower in the world and body frames drone is, therefore, given by

$$v_{air_2} = v_2 - v_w - v_{dw_1} \tag{4.13}$$

$$v_{Bair_2} = R_2^T v_{air}. \tag{4.14}$$

Then, we can determine the rotor drag and the frame drag accelerations from the second drone, which can be formulated again in the same way as in the individual drone flight, obtaining

$$a_{rd_2} = -R_2 C_{rd} v_{Bair_2} \tag{4.15}$$

$$a_{fd_2} = -\frac{1}{2m} C_d A v_{Bair_2} \odot v_{Bair_2} \odot sign(v_{Bair_2}). \tag{4.16}$$

being R_2 and v_{air_2} the target drone's rotation matrix and air relative speed, respectively. Then, in the same way as in (4.6), we calculate the this drone dynamical speed as

$$\dot{v_2} = -g z_{W_2} + \frac{u_{1_2}}{m} z_{B_2} + a_{rd_2} + a_{fd_2} \tag{4.17}$$

where a_{rd_2} and a_{fd_2} account for this drone's rotor and frame drags, respectively.

Figures 4.3 and 4.4 depict the downwash velocity as a function of distance from the quadrotor's rotors centre graphically in Two-dimensional (2D) and in 3D. The drone is hovering at $z = 2.5\,\text{m}$ in a windless environment.

Assuming the incoming flow is orthogonal to the drone below, the airflow spread and loss of intensity can be seen in both radial and axial distances to the centre of the rotors.

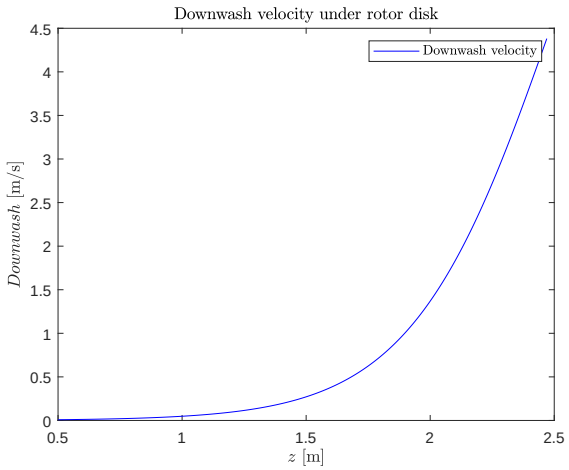


Figure 4.3: 2D plot of downwash velocity under a rotor disk in the function of the z distance to its centre

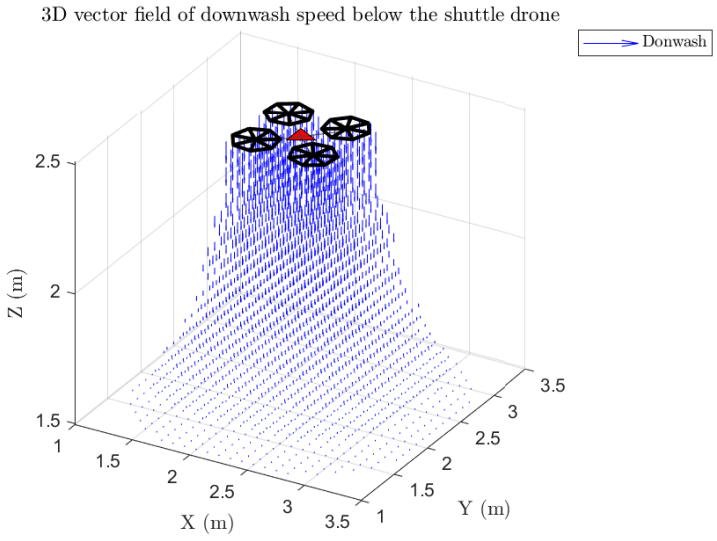


Figure 4.4: Flow beneath a quadrotor using (4.10) as the downwash model

For the drone to consider the effect of downwash, we calculate the air relative velocity of the drone incorporating the downwash velocity of the drone above as in (4.13). After that, as in (4.8) we obtain the target drone desired force, f_{des_2}, by adding the effects corresponding forces as

$$f_{des_2} = -k_{p_2} e_{p_2} - k_{ip_2} e_{ip_2} - k_{v_2} e_{v_2} + m(g z_{W_2} + a_{ref_2}) - f_{rd_2} - f_{fd_2}. \tag{4.18}$$

4.3 Aerodynamic parameters determination

This section outlines the process for determining the aerodynamic parameters C_{rd}, C_d, h, and k. Table 4.1 lists the values used in this chapter.

The frame drag coefficient is obtained following the work [50], which considers the drone's aerodynamic characteristics to be the same as a flat plate.

For the rotor drag coefficient, we assume the drone to be symmetric also along the Z-axis, setting $C_{rd,xx} = C_{rd,yy} = C_{rd,zz}$. This simplification is necessary due to a lack of experimental data for each coefficient. We adopted the value from a study [51] that tested a similar rotor blade using Computational Fluid Dynamics simulations and experimental methods. This study, along with another that analyzed quadrotor downwash effects [52], informed our adjustments for parameters h and k based on rotor similarity.

Table 4.1: Constant values for drone modelling and controlling two interacting drones

Parameter	Unit	Value
h	N/A	1.15
C_d	N/A	1.1800
k	N/A	1.75
$C_{rd,xx}$	N/A	0.1800
$C_{rd,yy}$	N/A	0.1800
$C_{rd,zz}$	N/A	0.1800
A_{xx}	m	0.5750
A_{yy}	m	0.5750
A_{zz}	m	0.3800
r	m	0.2300
A_d	m^2	0.1662
ρ	kg/m^3	1.2250

5

Simulation results and discussion

The effectiveness of the developed solution can be observed in this section through simulation results regarding various flight situations involving one or two drones, which are displayed and consequently analysed. Therefore, the overall setup of this process is briefly described, and the implementation of the model and control schematics is explained in Section 5.1. Then, the performed simulations are disposed of in two groups: one regarding trajectory following a performance by a single quadcopter in Section 5.2 and another portraying the proximity flight between two quadcopters in Section 5.3.

5.1 Environment setup

MATLAB development environment was used to build and depict these virtual simulations. Furthermore, Simulink's feature in MATLAB was used to compute the quadcopter's controller and dynamic model. These implementations are displayed below in Figures 5.1 and 5.2. The controller receives reference values for location, velocity, angular velocity, and rotation matrix from the drone's sensors and actual data for each. The controller then computes the derivative values of these parameters, which it uses to generate the control inputs $[u_1, u_\tau]$. The drone's model then receives these most recent values and applies them to replicate the quadrotor's actual behaviour.

To obtain maximum stability and smoothness through the desired trajectories, also having a quick rising time, it is necessary to adjust the respective gain values K_p, K_v, K_R, K_ω, as operating on K_p, K_v gains contributes to better position tracking and K_R, K_ω to less oscillation. A gain for the position error's integral, K_{ip}, is also used to mitigate this error. Multiple simulations were conducted, empirically obtaining the following values

$$\underbrace{\begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 6 \end{bmatrix}}_{K_p} \quad \underbrace{\begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 5 \end{bmatrix}}_{K_v} \quad \underbrace{\begin{bmatrix} 15 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 15 \end{bmatrix}}_{K_R} \quad \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{K_w} \quad \underbrace{\begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}}_{K_{ip}}.$$

Table 5.1: Control gains

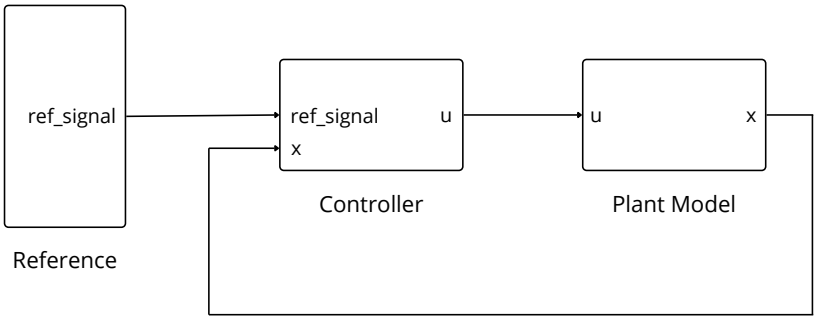


Figure 5.1: Block diagram of quadcopter control loop

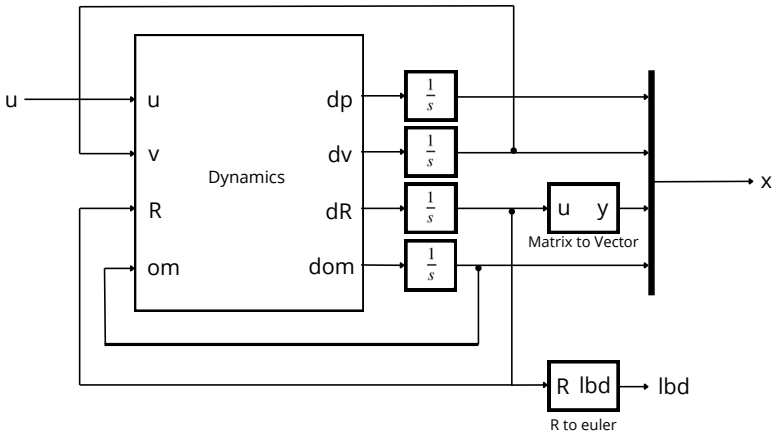


Figure 5.2: Block diagram of quadcopter dynamics

Moreover, we evaluate the effectiveness of both position and rotation tracking performance using a a step signal in the yaw angle reference from 0 to 90 degrees at t = 2 seconds and a reference step position at t = 4 seconds. The respective results are displayed in figures 5.3 and 5.4, respectively.

Analysing these figures, it is possible to conclude that the tracking response of this controller presents fast rising and settling times and lack of overshoot, which are good attributes and, therefore, more complex trajectories can be tested. For this case, we will use a circular trajectory with vertical motion, defined as

$$\begin{cases} x(t) & = 5\cos(0.2t) \\ y(t) & = 5\sin(0.2t) \\ z(t) & = 0.1t + 3 \\ & 0 \leq t \leq 60. \end{cases}$$

Also, the effects of rotor drag and frame drag are not considered. Figures 5.5, 5.6, and 5.9

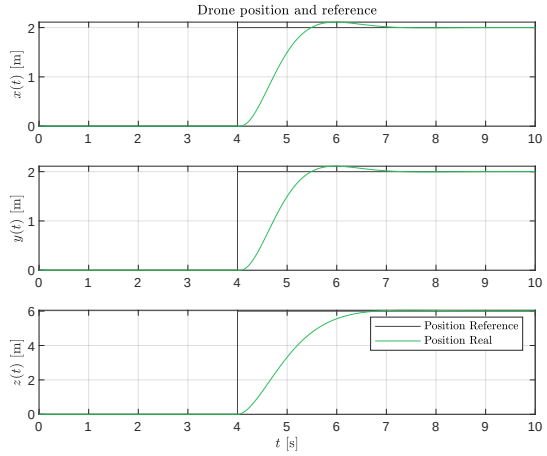


Figure 5.3: Step response for the positions in X, Y, and Z

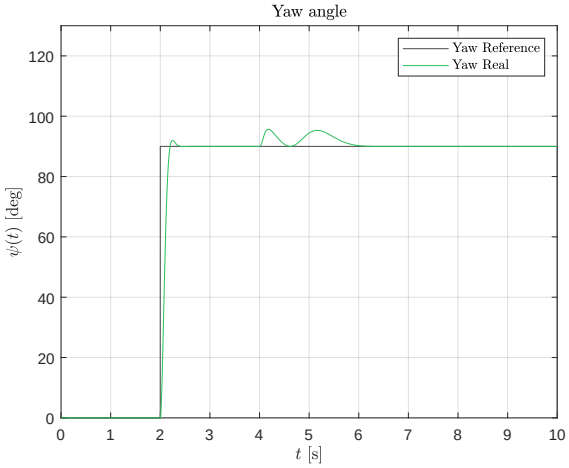


Figure 5.4: Step response for Yaw angle

illustrate the controller's positioning and speed performance in the x, y, and z axes. On the other hand, Figures 5.7 and 5.8 display the control inputs and the drone's attitude, respectively.

These simulations demonstrate that the quadrotor has good trajectory and velocity tracking throughout the flight path. Furthermore, the drone's very reasonable, steady flying qualities reinforce its capacity to precisely carry out intricate manoeuvring operations. Therefore, we are ready to test its response to more realistic scenarios in the next sections.

5.2 Trajectory following by one drone

To comprehensively evaluate the single quadrotor's performance under various conditions, we present a detailed analysis of its trajectories in distinct circumstances. We begin by assessing the fundamental controller in its most basic form without incorporating compensations for aerodynamic effects. Subsequently, we gradually enhance its robustness

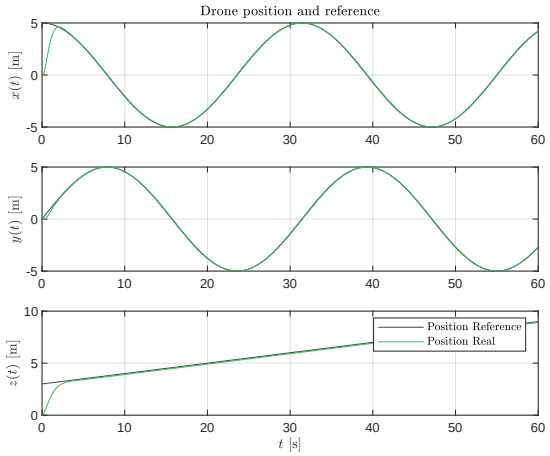


Figure 5.5: Position overtime in X, Y, and Z

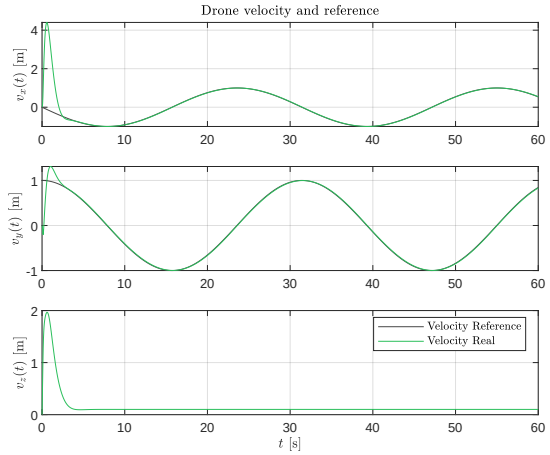


Figure 5.6: Velocity overtime in X, Y, and Z

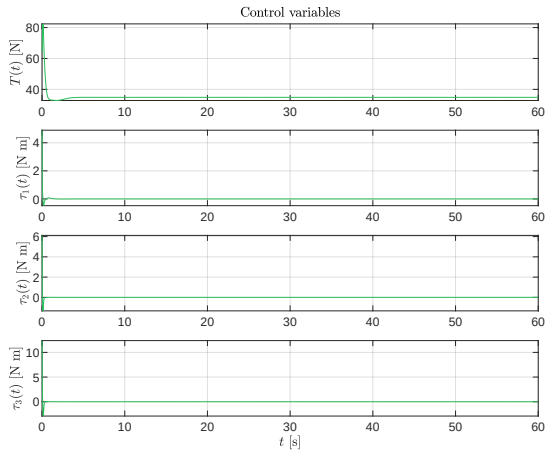


Figure 5.7: Control inputs overtime

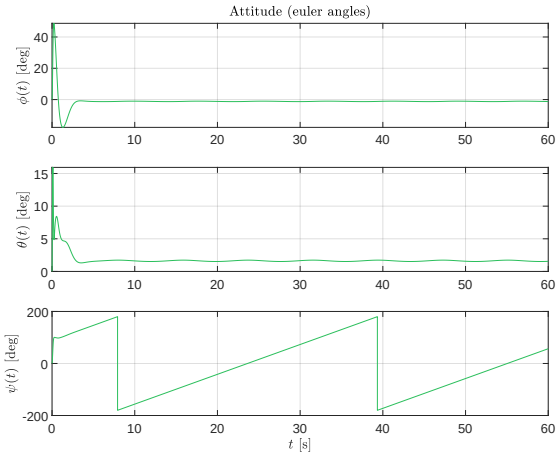


Figure 5.8: Euler angles overtime

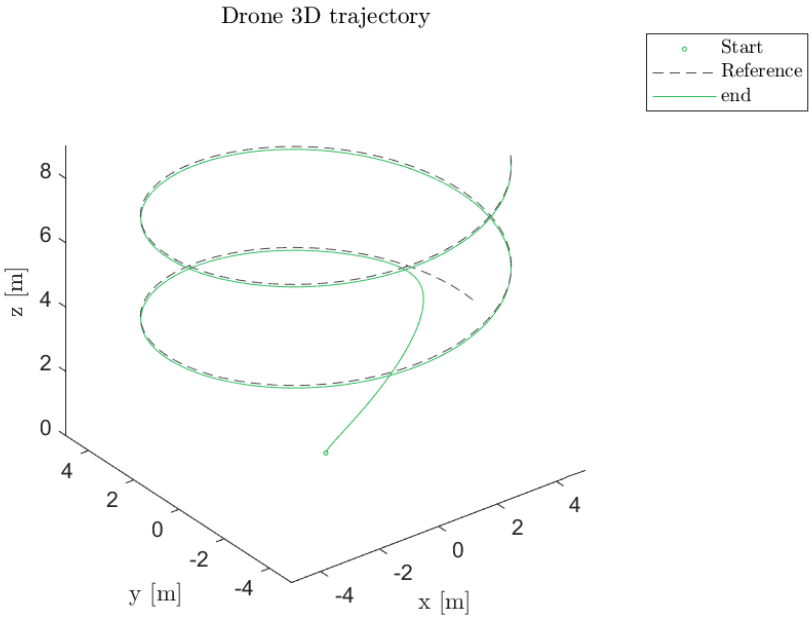


Figure 5.9: 3D plot of the drone's trajectory

to understand better the impact of the aerodynamic phenomena discussed in Section 4.2.

To effectively investigate these instances, four main flight-type scenarios have been designed to serve as the base conditions for the simulations. From here, only the drone's speed, trajectory, and wind speed vary to test the controller's response:

1. **Scenario A: Baseline Scenario**
   - **Description:** Without any dynamic response to velocity, acceleration, or jerk, the quadrotor functions under the effects of frame and rotor drag.
   - **Objective:** Evaluate the quadrotor's performance without any adjustment when aerodynamic effects are present.
2. **Scenario B: Feed Forward Scenario**
   - **Description:** The quadrotor experiences the same aerodynamic effects, but feed-forward terms are considered.
   - **Objective:** Analyse the basic controller's capacity to mitigate these effects.
3. **Scenario C: Rotor Drag Compensation Scenario**
   - **Description:**
     The quadrotor's response is examined while adjusting the rotor drag effect.
   - **Objective:** Study how the quadrotor responds to the specific problem of rotor drag.
4. **Scenario D: Rotor and Frame Drag Compensation Scenario**
   - **Description:** The controller is configured to compensate for rotor and frame drag effects under the same aerodynamic conditions.
   - **Objective:** Analyse the combined influence of the rotor and frame drag compensation on the quadrotor's performance.

The trajectory reference for the drone to follow is a circular one defined below as

$$\begin{cases} x(t) & = 5\cos(\omega_{ref} t) \\ y(t) & = 5\sin(\omega_{ref} t) \\ z(t) & = 3 \\ & 0 \leq t \leq 60 \end{cases}$$

where ω_{ref} is the reference angular velocity. The drone starts at the position $p_0 = \begin{bmatrix} 2 & 3 & 0 \end{bmatrix}^T$.

Table 5.2 presents the drone response to the four scenarios by measuring the norm of its position root minimum square error. The error Root Mean Square Error (RMSE) can be defined as

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} \|p_i - p_{\text{des}_i}\|^2}{N}} \tag{5.1}$$

where N is the number of samples, p is the ith sample of the actual position, and p_{des} is the ith sample of the desired position. Therefore, we present two sets of simulations. For the first one, we focus only on studying the rotor and frame drag aerodynamic effects on the drone, meaning the wind is not introduced to the experiment. The second one is to investigate the drone's trajectory tracking under the influence of different wind disturbances.

Table 5.2: Position RMSE at different drone speeds

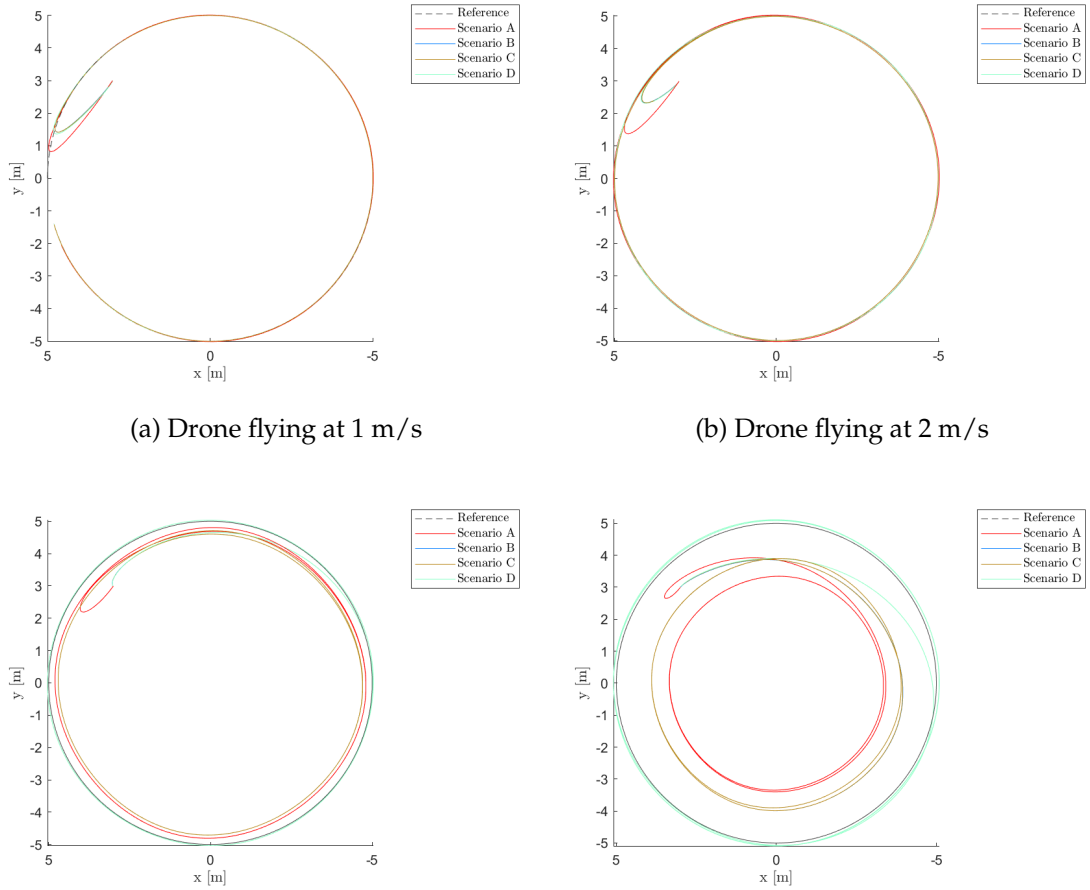
Drone's Speed (m/s)	RMSE A (m)	RMSE B (m)	RMSE C (m)	RMSE D (m)
1	0.944	0.602	0.597	0.586
2	1.610	0.624	0.605	0.588
5	3.797	0.893	0.785	0.596
10	5.868	1.402	1.264	0.651

Additionally, we evaluate the impact of wind disturbance on quadrotor motion by incorporating a constant wind speed into its model. Although it does not perfectly replicate real-world wind conditions, it provides a simplified representation of wind effects suitable for studying fundamental drone dynamics. Therefore, with the drone travelling at a speed of 5 m/s, we consider an increasing wind speed for each scenario and analyze its impact again, measuring the root mean square error for each speed at each scenario as illustrated in Table 5.3.

Table 5.3: Position RMSE for different wind speeds at X coordinate

Wind Speed (m/s)	RMSE A	RMSE B	RMSE C	RMSE D
(0,0,0)	3.797	0.893	0.785	0.596
(-2,0,0)	3.825	1.002	0.884	0.600
(-5,0,0)	4.013	1.524	1.388	0.611
(-10,0,0)	4.959	3.334	3.204	0.632
(-15,0,0)	7.381	6.461	6.389	0.653

Tables 5.2 and 5.3, and Figures 5.10 to 5.14 effectively demonstrate the impact of these aerodynamic conditions on the quadrotor's flight on a trajectory tracking level. On the one hand, as the controller's compensation capabilities are incorporated, the decreasing RMSE position values underscore the controller's effectiveness in mitigating the detrimental effects of aerodynamic drag. In both tables, the non-existing compensation leads to the most significant difference for the first scenario, with at least some compensatory factors present. On the other hand, a more substantial increase in drone and wind speeds leads



(a) Drone flying at 1 m/s

(b) Drone flying at 2 m/s

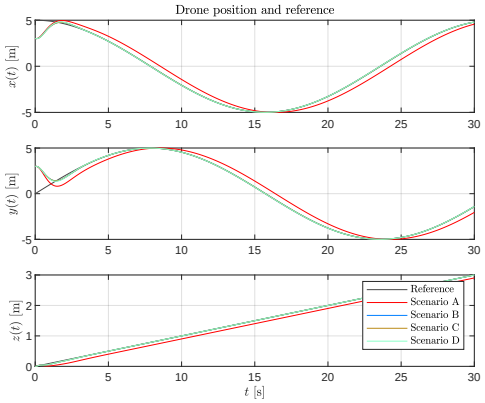
(c) Drone flying at 5 m/s

(d) Drone flying at 10 m/s

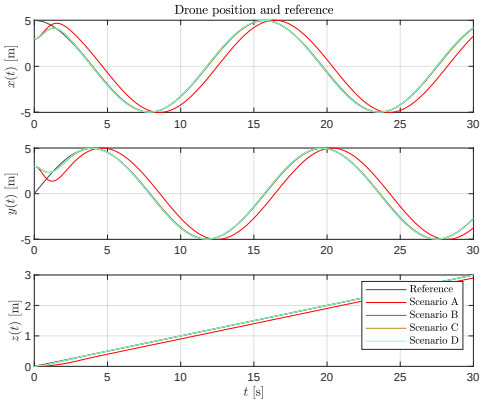
Figure 5.10: Trajectories of the drone flying at different speeds with various compensation scenarios

to a more considerable increase in RMSE values. It's also noticeable, in both tables, that the frame drag effect has the most interference on the drone's trajectory tracking because, when the first compensated, there is a significant decrease in the position error. Concretely, there is approximately half error reduction for the first table, and almost a ten times reduction in the second one in the most extreme aerodynamic scenarios. This proves the importance of correctly modelling this effect, which is regularly considered in the literature, especially in [30] and [31].

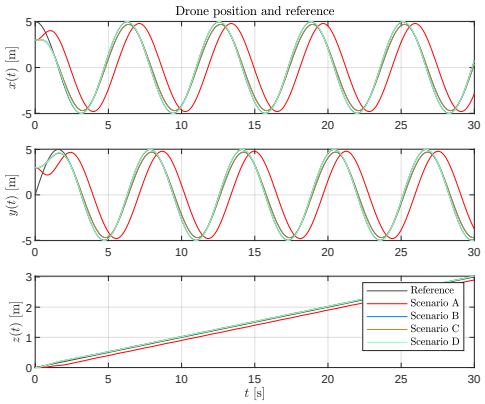
Furthermore, Figures 5.13 and 5.14 help us understand the behaviour of the quadrotor's state variables under different wind conditions and compensation scenarios. The first shows the position, velocity, Euler angles, and control inputs of the quadrotor flying at 5 m/s in four scenarios with a wind speed of (-5,0,0) m/s. Here, the position plots demonstrate the quadrotor's stable trajectory, with some considerable deviations from the intended path when not considering all the compensation for the aerodynamic factors and the opposite when the controller compensates for them (the error reduces more than 60 % from the scenario A to D), showcasing the effectiveness of the compensation strategies



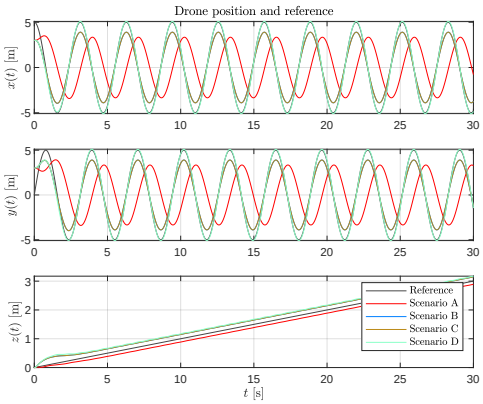
(a) Drone flying at 1 m/s



(b) Drone flying at 2 m/s



(c) Drone flying at 5 m/s

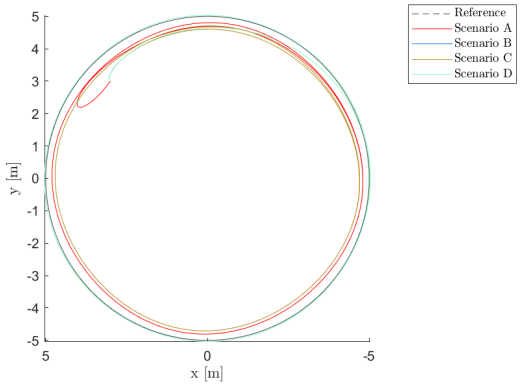


(d) Drone flying at 10 m/s

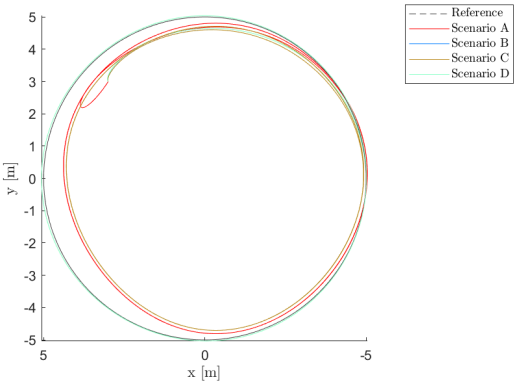
Figure 5.11: Plot for the coordinates of the quadrotor flying at 10 m/s in all four scenarios

in mitigating the impact of the moderate wind disturbance. The velocity components remain relatively constant, reflecting effective control input adjustments to maintain the desired speed. The roll and pitch angles stay within acceptable limits at this wind speed, especially after considering the feed-forward terms in the control compensation (reducing the oscillation significantly for both angles), demonstrating effective stabilization. Therefore, by considering the yaw angle behaviour in the four scenarios, not considering the feed-forward terms also leads to more substantial deviations in the drone's attitude. Finally, the control inputs exhibit a stable pattern, indicating that the control system effectively compensates for minor disturbances.

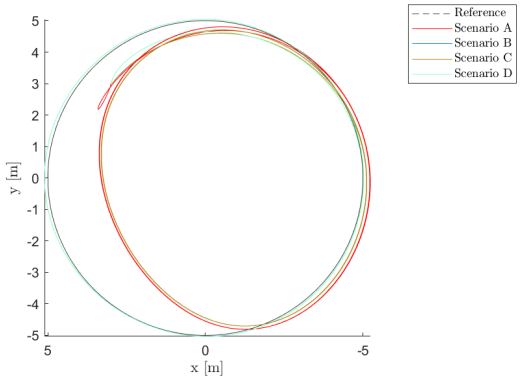
In Figure 5.14, on the other hand, the position plots show large oscillations, especially in the X direction. This means that the trajectory tracking accuracy is lower because the wind disturbance on the x-axis is more potent. The velocity and air relative velocity plots also show increased fluctuations in the X direction, suggesting that the quadrotor struggles to maintain its target speed. The roll and pitch angles display significantly more pronounced oscillations, with more than double the range of oscillation angles in all scenarios. This reflects the quadrotor's challenge in maintaining its orientation against



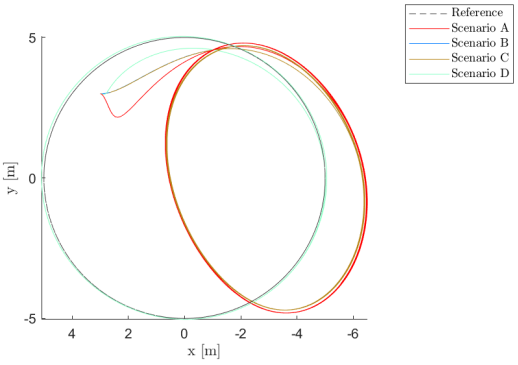
(a) Wind speed at [0, 0, 0] m/s



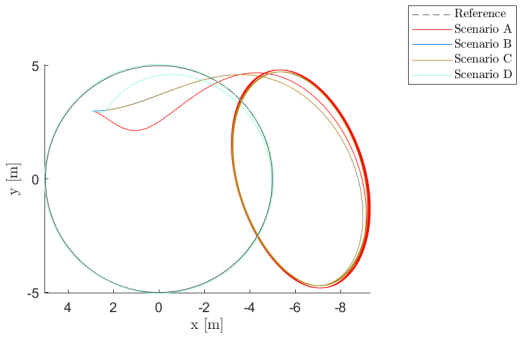
(b) Wind speed at [-2, 0, 0] m/s



(c) Wind speed at [-5, 0, 0] m/s

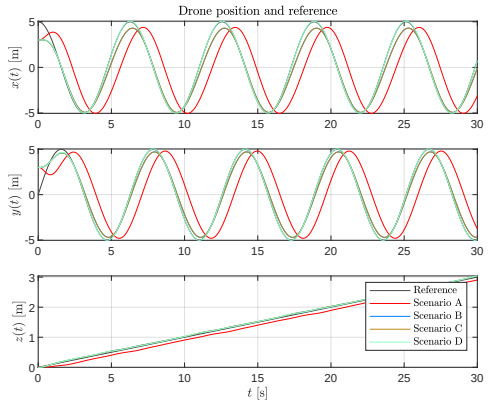


(d) Wind speed at [-10, 0, 0] m/s

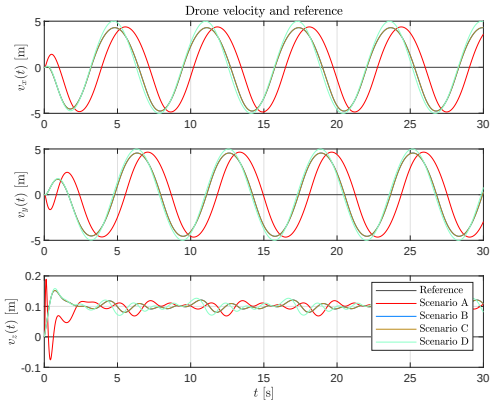


(e) Wind speed at [-15, 0, 0] m/s

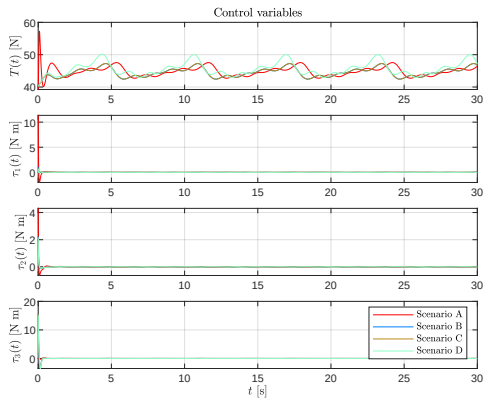
Figure 5.12: Drone upper-view trajectories under different wind conditions in all four scenarios



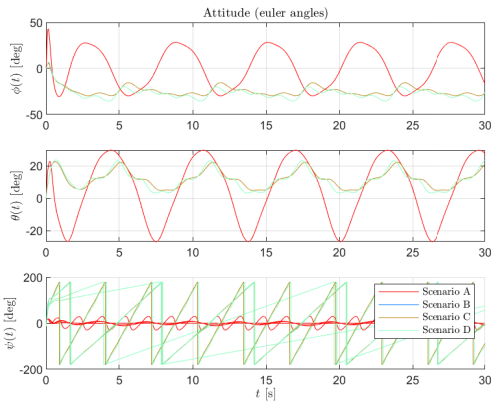
(a) Position overtime in X, Y, and Z



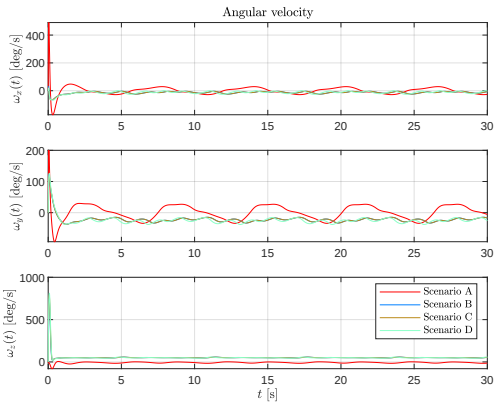
(b) Velocity overtime in X, Y, and Z



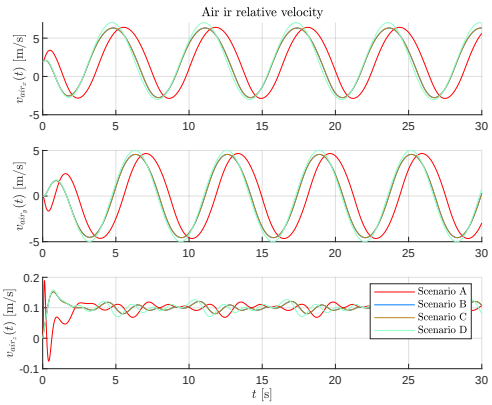
(c) Control inputs overtime



(d) Euler angles overtime

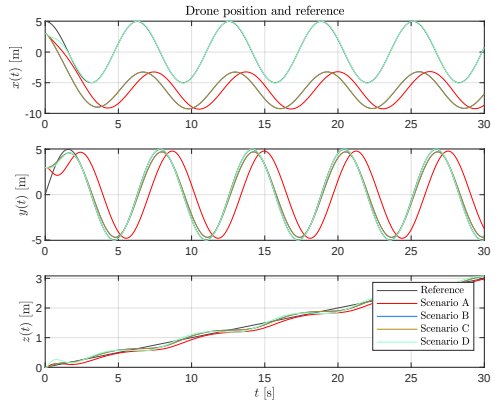


(e) Angular velocity overtime

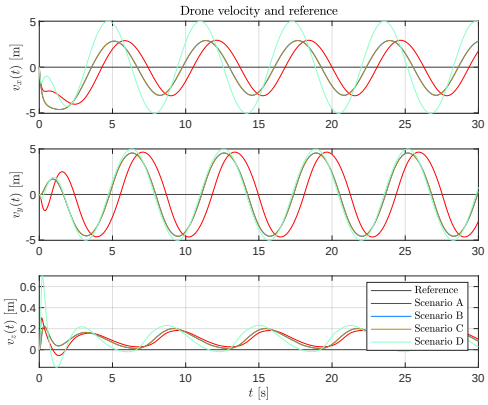


(f) Air relative velocity

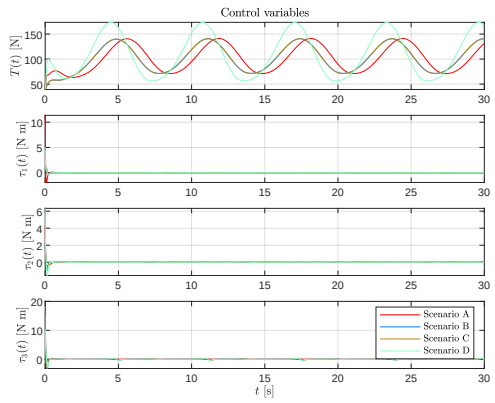
Figure 5.13: Plots for the state variables of the quadrotor flying 5 m/s in all four scenarios with a wind speed of (-5,0,0) m/s



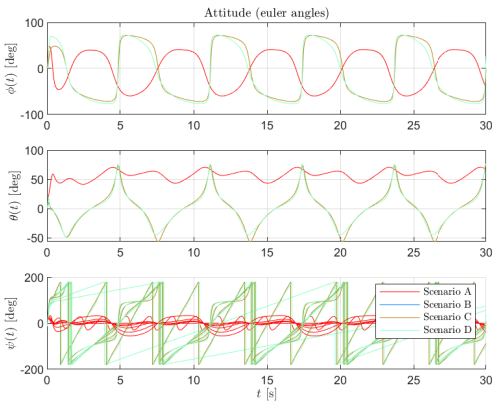
(a) Position overtime in X, Y, and Z



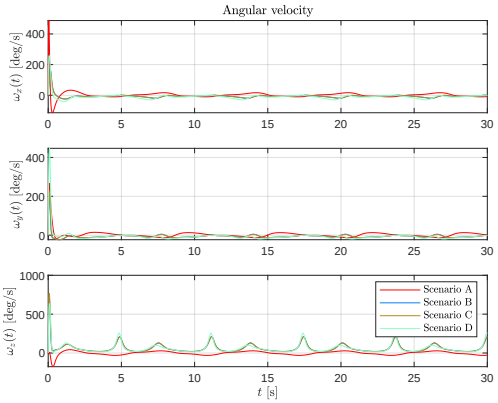
(b) Velocity overtime in X, Y, and Z



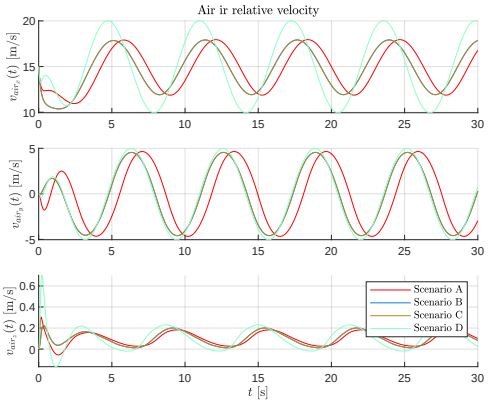
(c) Control inputs overtime



(d) Euler angles overtime



(e) Angular velocity overtime



(f) Air relative velocity over time

Figure 5.14: Plots for the state variables of the quadrotor flying 5 m/s in all four scenarios with a wind speed of (-15,0,0) m/s

stronger wind forces. Therefore, the control inputs exhibit more significant variability (more than twice the earlier wind speed in all the scenarios), highlighting the system's increased demand for stability and trajectory-tracking adjustments.

Altogether, these pictures show how the quadrotor reacts to different wind conditions and how well the suggested ways of adjusting for wind affect keeping the flight stable and following the path during cooperative capture manoeuvres. The results indicate that while the quadrotor can preserve stability and trajectory under lower wind conditions, increased wind speeds necessitate more sophisticated control strategies to ensure robust performance. Moreover, neglecting the control feed-forward terms leads to the most significant deviations from the reference signal in both position and attitude.

With the satisfying results obtained for the trajectory tracking of one drone in an aggressive manoeuvre subject to relevant aerodynamic factors, it's possible to experiment with a capture trajectory where two drones operate simultaneously.

5.3 Two drones flying in close proximity

In this second set of simulations, we consider two drones: the shuttle quadrotor and the target quadrotor. The target drone flies beneath first and is subject to its downwash. Figure 5.15 displays the control loop diagram designed for this part. Both the model and the controller of the drone below receive the position and generated thrust from the drone above ("p_1" and "T_1"), with the controller also having a flag signal for when to use the compensation feature for the downwash.

The initial positions of these drones are as follows: the shuttle drone starts at position $p_{o1} = [0, 3, 7 + h]$, where h represents the nearby distance limit between the aircraft along the z-axis, while Drone 2 begins at position $p_{o2} = [0, 0, 2]$.

The flight behaviour unfolds as follows:

1. **Shuttle quadrotor**: gradually descends above the target drone, approaching it in a controlled manner, where its trajectory is defined as

$$\begin{aligned} x(t) &= t \\ y(t) &= 3 \\ z(t) &= 7\exp(-0.3t) + 2.00 + h. \end{aligned}$$

2. **Target quadrotor**: follows a linear reference trajectory defined as

$$\begin{aligned} x(t) &= t \\ y(t) &= 3 \\ z(t) &= 2 \end{aligned}$$

During the interaction, the shuttle quadrotor remains in a windless environment above the other quadrotor for 60 seconds. However, they operate in close proximity for a specific interval of approximately 40 seconds.

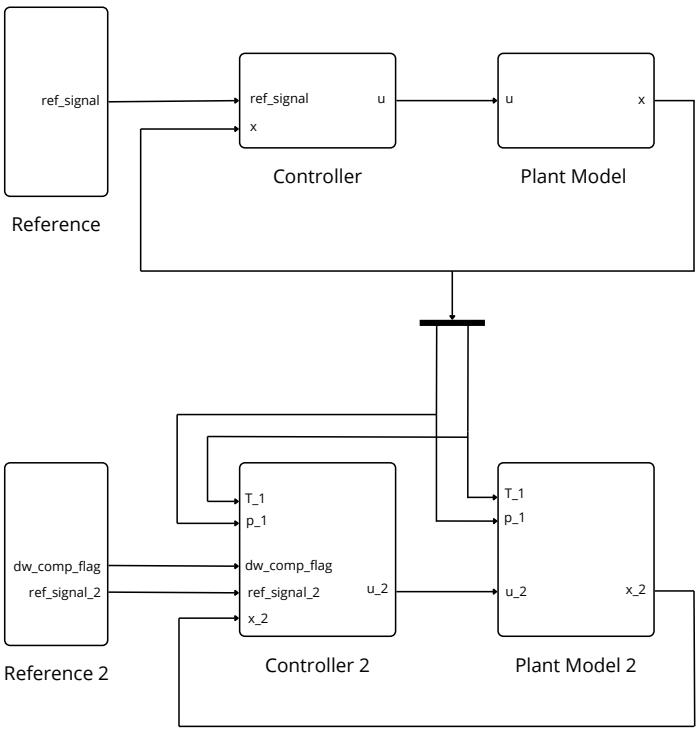


Figure 5.15: Cooperative capture manoeuvre controller's loop design

Therefore, we test the target drone response in two scenarios:

1. The drone is under the effect of the position error's integral but doesn't use compensatory measures for the downwash of the drone above;

2. The drone adjusts for the upwash force coming from above and also considers the position error's integral in its controller.

These two scenarios are devised to understand this integral's capability to mitigate the downwash's effects and, consequently, the true importance of the downwash compensation described in Chapter 4.

The simulation results for the first scenario are shown in Figures 5.17 to 5.22. They show the plots of the shuttle ("Shuttle Drone"), target ("Target Drone") quadrotors with and without downwash compensation, and the target drone with this effect compensation ("Target Drone Complete").

Table 5.4 displays, for different minimal distances between the two drones, the target drone's position root mean error, RMSE, and maximum thrust generated using (complete) or not using (basic) compensation for the downwash coming from the drone above.

The capture manoeuvre's simulation results display the effectiveness of the developed control strategies in counteracting the aerodynamic disturbances. In particular, these simulations enable the two drones to precisely cooperate in the capture, even in difficult downwash situations.

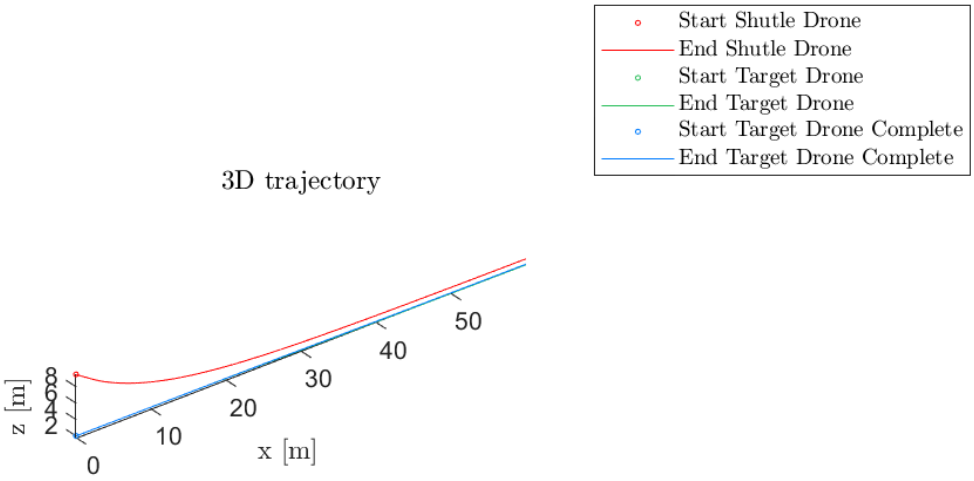


Figure 5.16: 3D plot of the capture manoeuvre

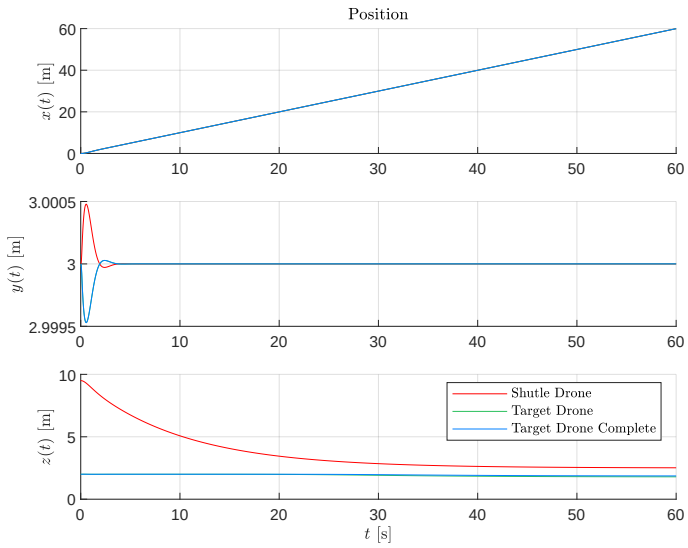


Figure 5.17: Position comparison in the capture manoeuvre

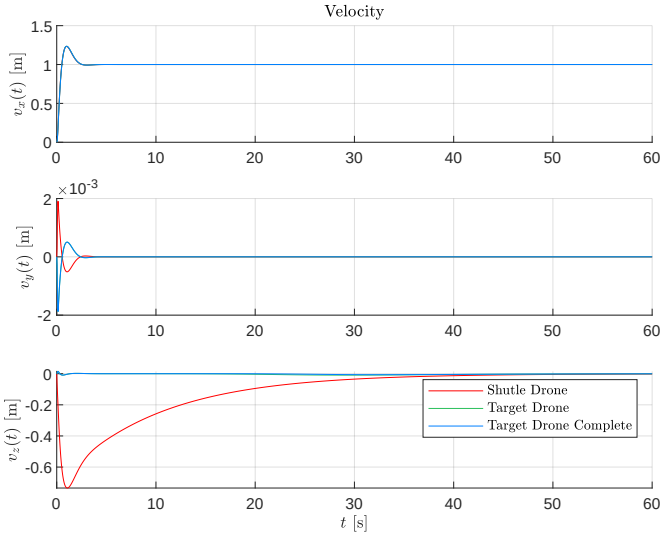


Figure 5.18: Velocity comparison in capture manoeuvre

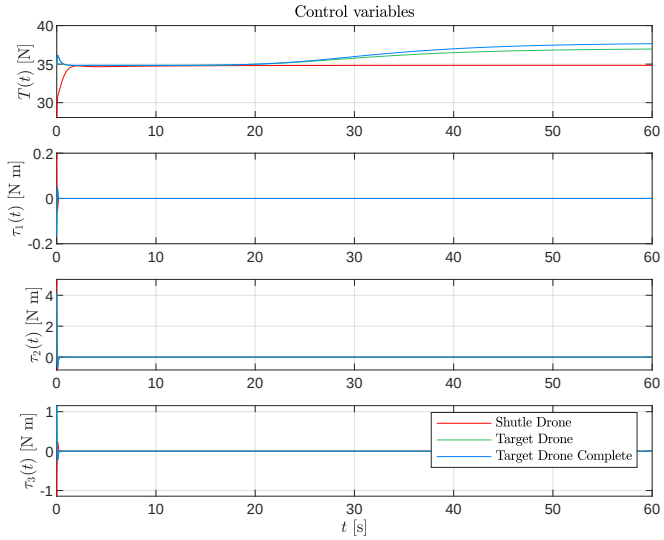


Figure 5.19: Control inputs comparison in capture manoeuvre

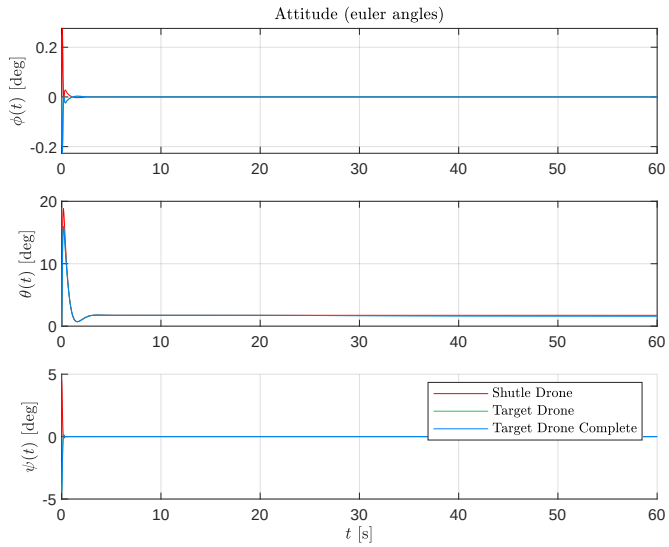


Figure 5.20: Euler angles comparison in capture manoeuvre

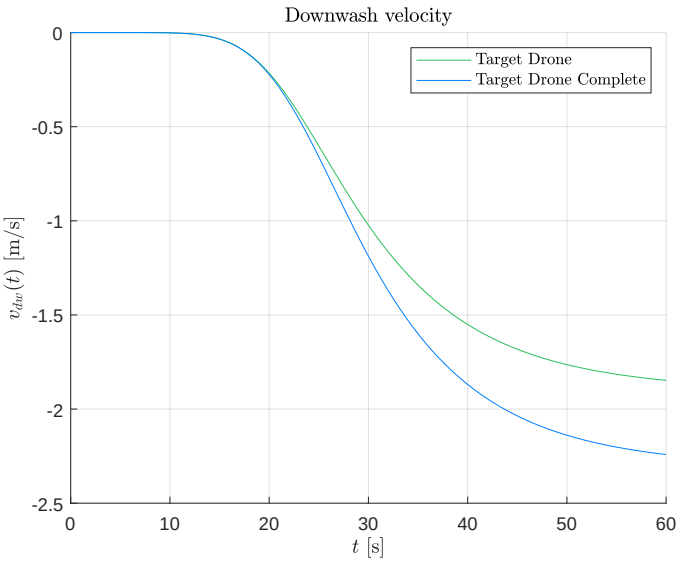


Figure 5.21: Comparison of the downwash felt by the target drone in the capture manoeuvre

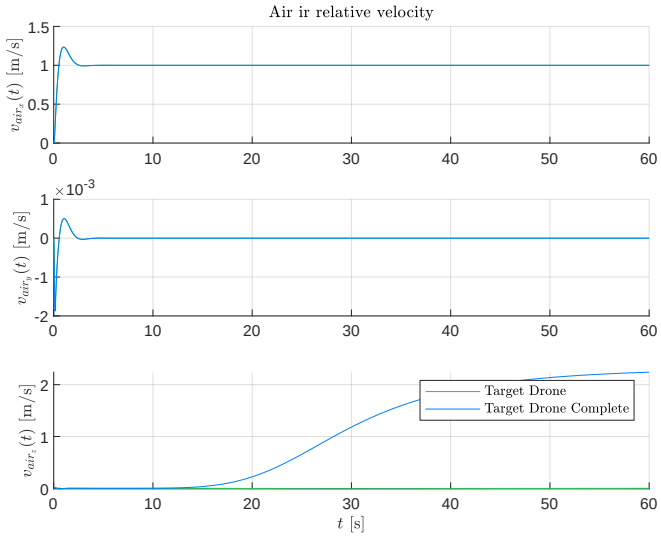


Figure 5.22: Comparison of the air relative velocity of the target drone in the capture manoeuvre

Table 5.4: Performance Metrics for Different Minimal Distances Between Drones

Min Distance (m)	RMSE (basic/complete) (m)	Max Thrust (basic/complete) (N)
0.50	0.13/0.08	36.96/37.65
0.70	0.08/0.05	36.16/36.51
1.00	0.05/0.04	36.16/36.16
2.00	0.04/0.04	36.16/36.16

As the axial distance between the drones decreases, the target drone is more affected by the downwash, as seen in Figure 5.21. This causes deviations in this drone's trajectory, with more significance in the z-axis. To compensate, the target drone's controller incorporates this downwash velocity in the drone's air relative velocity, as seen in Figure 5.22, and inputs more thrust to oppose it. This increase is displayed in Figure 5.19, where it is also possible to observe that this increase in thrust depends on the distance to the upper drone - more closeness between them represents a more significant increase in the thrust produced by the target drone.

The target drone also experiences speed changes when affected by the downwash, mainly a decrease in the z-velocity component as the drone above gets closer, as seen in Figure 5.18. Figure 5.20 demonstrates the effects of the shuttle drone's downwash on the attitude of the lower drone, especially the higher pitch angle that the uncompensated drone has than the compensated drone, which could indicate a struggle to maintain altitude or position due to downwash effects. Also, the roll angle plots show that the shuttle drone and target drone (no compensation) exhibit slight positive roll angles (up to around 1.6°), suggesting minor lateral adjustments or corrections. However, when compensating for the downwash effect, the target drone slightly approximates (around 0.04°) to a zero roll angle, indicating some stable lateral control and mitigation of downwash effects.

On the other hand, Table 5.4 shows how the downwash compensation feature improves the target drone's performance. For example, at a minimum distance of 0.5 m between the drones, the RMSE in z-trajectory is reduced from 0.13 m to 0.08 m using downwash compensation. The maximum thrust the target drone requires increases slightly from 36.96N to 37.65N.

Furthermore, all the variances exposed and explained earlier behave less intensely for higher minimal distances between the two drones. For instance, the lower drone trajectory deviance error continuously decreases with or without total controller compensation. The lower drone's maximum thrust experiences the same phenomenon until a distance of 1 m between the drones, where the downwash effect is minimal and, therefore, doesn't need almost any increase in thrust to compensate for the downward force.

Overall, the simulation results validate the proposed control system's ability to effectively mitigate the impact of the shuttle drone's downwash on the target drone during the cooperative capture manoeuvre. The downwash compensation enables more precise trajectory tracking and stable flight of the target drone close to the shuttle drone. However, as shown by the slight difference between the position RMSE values of the drone using the basic and the complete controller, only using the position error's integration already proves to be a satisfactory measure for this case, especially up to 1 m of the minimal distance between the shuttle and target drones, where the difference in the position RMSE values between the controllers is below 1 cm. Furthermore, further testing is required not only to ensure the resilience of our technology against increasingly unpredictable and dynamic environments, such as varying wind conditions, but also to identify and address potential discrepancies between simulated and actual flying behaviour, which may arise due to factors like sensor inaccuracies or unmodelled physical effects.

6

Conclusions

This thesis successfully achieves its primary objective by developing an algorithm to control a quadrotor-type shuttle drone during aggressive manoeuvres. It also enables the target drone to manage aerodynamic interactions effectively during a cooperative capture manoeuvre. The findings provide a solid foundation for further research in cooperative drone capture and control. The simulation results align well with the proposed strategies, although they lack some precision due to the absence of experimental tests. This objective was accomplished by going through four different phases described in this thesis.

In fact, the first step to start creating algorithms to implement on controllers able to complete the complex tasks talked about earlier was to conduct an extensive and objective investigation aimed at obtaining explanations for the different obstacles and difficulties, but also solutions for them. As a result, Chapter 2 talks about related work that looks at control strategies for cooperative and non-cooperative flight, the aerodynamics of solo flight and close flight between a drone and a surface or another drone, and ways to simulate and test drones of the same size.

The second phase involved understanding the operation of quadrotor controllers by designing one, based on an existing and tested nonlinear controller. This controller is a solid foundation for managing aggressive quadrotor manoeuvres, focusing on generating optimal trajectories that minimize snap (the fourth derivative of position) while ensuring the quadrotor can follow these trajectories accurately. Chapter 3 provides a detailed description of its design, and, in Chapter 5, we used MATLAB to implement the controller and simulate it using our drone, yielding encouraging outcomes where the drone presents good reference tracking for simple step signals and under a more robust trajectory.

The third phase, as outlined in Chapter 4, was to enhance the controller's adaptability and robustness by incorporating supplementary aerodynamic considerations that are essential for a capture manoeuvre, as the nonlinear controller we used as inspiration to develop our own was intended to operate a quadrotor in an indoor environment. By integrating the effects of rotor drag and frame drag into the controller, the drone could navigate more realistically in various aerodynamic conditions and flight speeds while still showing acceptance of position tracking. Based on the findings in Chapter 5, it was

concluded that frame drag has the most significant effect on the aerodynamic factors studied in this thesis, leading to more substantial position changes during simulations. This conclusion is also coherent with the studies present in the related work. Unfortunately, these results are not entirely precise, as the rotor drag and frame drag coefficients were taken from related work and were not experimentally determined.

The fourth phase, a scenario where the aerodynamic interaction between two quadrotors flying one above the other, is addressed in the last part of Chapter 4 and effectively implemented in Chapter 5. In this way, a downwash model is considered in the dynamic section of the Target Drone's controller loop, and downwash compensation mechanisms are inserted into its control section. The results shown in Chapter 5 are in line with the proposed strategy. This is because the target drone usually acts when the downward force of the airflow from the drone above it hits it. For example, it starts to veer off course and oscillate as the shuttle drone gets closer.

Moreover, comparing the sole use of the integral gain's effect with including the downwash compensation on the first drone, the latter compensation method presents better position tracking. However, the difference is still minimal (around 5 centimetres) even at the closest minimal distance simulated (0.5 m), where the target drone suffers the most from the downwash above. However, as in the earlier phase, all the downwash coefficients were not experimentally determined, so the impact this airflow causes on this drone may not be exact.

Finally, the developed work of this thesis ends in Chapter 5, which was methodically constructed to conduct a set of numeric simulations to evaluate, in each of the last three phases of this work, the developed controller's performance over a wide range of scenarios where its capability to compensate for the aerodynamic phenomena earlier mentioned is tested extensively. The results show the controller's competence, reactivity, and flexibility in various environmental circumstances, flight speeds, and complex manoeuvres previously studied in this dissertation.

6.1 Future work

For future work, some research areas could be pursued to develop this project further.

In fact, the first step could be making the developed quadrotor model more stable by adding the forward flight condition that would work better in a capture manoeuvre. This would provide more precise values for the induced airflow speed that the propellers cause and their effects on wind gusts and other airflow. For that reason, having more realistic wind disturbance models would improve the accuracy of the operation since, as proved through the simulations earlier, wind can interfere significantly with the trajectory tracking of the drone, which is even more troubling considering the need for high accuracy in this operation.

On the other hand, using more realistic simulation environments would significantly increase the credibility of the developed controller, setting it on a better course to later

perform real-world flight experiments. An excellent example of this simulation technique would be similar to the one presented in Section 2.6 since it provides better real-time testing samples and more environmentally complex scenarios.

Therefore, it would also be interesting to integrate the T-Drone M690B quadrotor model and later use it to perform real-world flight experiments. This would enable us to validate better the aerodynamic effects studied during this thesis and fulfil one of the project's goals. For instance, an anemometer attached to the quadrotor during an actual flight test could measure the drone's downwash velocity during various flight manoeuvres.

On the other hand, analysing the drone's response to wind gusts and flying under various speeds could enhance the determination of the different aerodynamic coefficients and parameters earlier studied, such as the rotor drag coefficient. This would further increase this drone's model robustness and accuracy, easing future research where this quadrotor might be used.

Bibliography

[1] M. R. Polaris. *Commercial UAV (Unmanned Aerial Vehicle) Market Share, Size, Trends Industry and Analysis Report*. Accessed online: 2022-07-15. 2021. URL: https://www.polarismarketresearch.com/industry-analysis/commercial-uav-market (cit. on p. 1).

[2] M.-T. O. Hoang et al. "Drone Swarms to Support Search and Rescue Operations: Opportunities and Challenges". In: *Cultural Robotics: Social Robots and Their Emergent Cultural Ecologies*. Ed. by B. J. Dunstan et al. Cham: Springer International Publishing, 2023, pp. 163–176. ISBN: 978-3-031-28138-9. DOI: 10.1007/978-3-031-28138-9_11. URL: https://doi.org/10.1007/978-3-031-28138-9_11 (cit. on p. 1).

[3] B. Rabta, C. Wankmüller, and G. Reiner. "A drone fleet model for last-mile distribution in disaster relief operations". In: *International Journal of Disaster Risk Reduction* 28 (2018-06), pp. 107–112. ISSN: 22124209. DOI: 10.1016/j.ijdrr.2018.02.020 (cit. on p. 1).

[4] W. H. Maes and K. Steppe. "Perspectives for Remote Sensing with Unmanned Aerial Vehicles in Precision Agriculture". In: *Trends in Plant Science* 24 (2 2019-02), pp. 152–164. ISSN: 13601385. DOI: 10.1016/j.tplants.2018.11.007 (cit. on p. 1).

[5] R. S. Jamisola. "Editorial: Control of cooperative drones and their applications". In: *Frontiers in Robotics and AI* 9 (2022-09). ISSN: 22969144. DOI: 10.3389/frobt.2022.1014510 (cit. on p. 1).

[6] S. McDill and A. Richardson. *Could display drones snuff out the firework?* (Acessed: 2022-02-16). 2021. URL: https://www.reuters.com/business/sustainable-business/could-display-drones-snuff-out-firework-2021-07-28/ (cit. on p. 1).

[7] *Project CAPTURE – Shuttle drone for launch and capture*. (Acessed: 2022-02-16). URL: http://capture.isr.tecnico.ulisboa.pt/pt/ (cit. on p. 2).

[8] O. Dhewa, A. Dharmawan, and T. Priyambodo. "Model of Linear Quadratic Regulator (LQR) Control Method in Hovering State of Quadrotor". In: *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)* 9 (2017-09), pp. 135–143 (cit. on p. 6).

[9] O. Bouaiss et al. "Optimal Control of Quadrotor with a Novel Madgwick/Extended Kalman Observer to Track a Spline Trajectory for Obstacle Avoidance". In: *Iranian Journal of Science and Technology, Transactions of Electrical Engineering* 47 (2022), pp. 269–283. URL: https://api.semanticscholar.org/CorpusID:252889758 (cit. on p. 6).

[10] A. Zulu and S. John. "A Review of Control Algorithms for Autonomous Quadrotors". In: *Open Journal of Applied Sciences* 04 (2014-01), pp. 547–556. DOI: 10.4236/ojapps.2014.414053 (cit. on pp. 7, 8).

[11] A. Sheta et al. "Optimization of PID Controller to Stabilize Quadcopter Movements Using Meta-Heuristic Search Algorithms". In: *Applied Sciences* 11.14 (2021). ISSN: 2076-3417. DOI: 10.3390/app11146492. URL: https://www.mdpi.com/2076-3417/11/14/6492 (cit. on p. 7).

[12] K. Klausen, T. Fossen, and T. Johansen. "Nonlinear control of a multirotor UAV with suspended load". In: *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015* (2015-07), pp. 176–184. DOI: 10.1109/ICUAS.2015.7152289 (cit. on p. 8).

[13] R. Perez-Alcocer and J. Moreno-Valenzuela. "Adaptive Control for Quadrotor Trajectory Tracking with Accurate Parametrization". In: *IEEE Access* 7 (2019), pp. 53236–53247. ISSN: 21693536. DOI: 10.1109/ACCESS.2019.2912608 (cit. on p. 8).

[14] Y. Gao et al. "Adaptive Tracking and Perching for Quadrotor in Dynamic Scenarios". In: *IEEE Transactions on Robotics* 40 (2024), pp. 499–519. DOI: 10.1109/TRO.2023.3335670 (cit. on p. 8).

[15] C. Kunusch, P. Puleston, and M. Mayosky. "Fundamentals of Sliding-Mode Control Design". In: *Sliding-Mode Control of PEM Fuel Cells*. London: Springer London, 2012, pp. 35–71. ISBN: 978-1-4471-2431-3. DOI: 10.1007/978-1-4471-2431-3_3. URL: https://doi.org/10.1007/978-1-4471-2431-3_3 (cit. on p. 9).

[16] H. L. N. N. Thanh and S. K. Hong. "Quadcopter robust adaptive second order sliding mode control based on PID sliding surface". In: *IEEE Access* 6 (2018), pp. 66850–66860. ISSN: 21693536. DOI: 10.1109/ACCESS.2018.2877795 (cit. on p. 9).

[17] G. Yu et al. "Aggressive maneuvers for a quadrotor-slung-load system through fast trajectory generation and tracking". In: *Autonomous Robots* 46 (2022-04). DOI: 10.1007/s10514-022-10035-y (cit. on p. 9).

[18] K. Holkar and L. M. Waghmare. "Sliding Mode Control with Predictive PID Sliding Surface for Improved Performance". In: *International Journal of Computer Applications* 78 (2013), pp. 1–5. URL: https://api.semanticscholar.org/CorpusID:8703685 (cit. on p. 9).

[19] R. Findeisen and F. Allgöwer. "An Introduction to Nonlinear Model Predictive Control". In: (2002). URL: https://api.semanticscholar.org/CorpusID:59790631 (cit. on p. 10).

[20] M. Kamel, M. Burri, and R. Siegwart. "Linear vs Nonlinear MPC for Trajectory Tracking Applied to Rotary Wing Micro Aerial Vehicles". In: *IFAC-PapersOnLine* 50.1 (2017). 20th IFAC World Congress, pp. 3463–3469. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2017.08.849. URL: https://www.sciencedirect.com/science/article/pii/S2405896317313083 (cit. on p. 10).

[21] H. Mohammadi Daniali. "Fast Nonlinear Model Predictive Control of Quadrotors: Design and Experiments". PhD thesis. University of Waterloo, 2020-01 (cit. on p. 10).

[22] G. Lu, D. Xu, and Y. Meng. "Dynamic Evolution Analysis of Desertification Images Based on BP Neural Network". In: *Computational Intelligence and Neuroscience* 2022 (2022-03). DOI: 10.1155/2022/5645535 (cit. on pp. 10, 11).

[23] J. Paredes et al. "Development, implementation, and experimental outdoor evaluation of quadcopter controllers for computationally limited embedded systems". In: *Annual Reviews in Control* 52 (2021), pp. 372–389. ISSN: 1367-5788. DOI: https://doi.org/10.1016/j.arcontrol.2021.06.001. URL: https://www.sciencedirect.com/science/article/pii/S1367578821000420 (cit. on p. 11).

[24] G. E. M. Abro et al. "Review of Hybrid Control Designs for Underactuated Quadrotor with Unmodelled Dynamic Factors". In: *Emerging Technologies in Computing*. Ed. by M. H. Miraz et al. Cham: Springer International Publishing, 2020, pp. 62–85. ISBN: 978-3-030-60036-5 (cit. on p. 12).

[25] B. Neves and B. Guerreiro. "Flight control of hybrid drones towards enabling parcel relay manoeuvres". In: *2021 International Young Engineers Forum (YEF-ECE)*. 2021, pp. 13–19. DOI: 10.1109/YEF-ECE52297.2021.9505161 (cit. on p. 12).

[26] M. Abdelkader et al. "Aerial Swarms: Recent Applications and Challenges". In: *Current Robotics Reports* 2 (2021-09), pp. 1–12. DOI: 10.1007/s43154-021-00063-4 (cit. on p. 12).

[27] J. Kusyk et al. "Artificial intelligence and game theory controlled autonomous UAV swarms". In: *Evolutionary Intelligence* 14.4 (2021-12), pp. 1775–1792. DOI: 10.1007/s12065-020-00456-y. URL: https://doi.org/10.1007/s12065-020-00456-y (cit. on p. 12).

[28] Y. Zhang et al. "Decentralized cooperative trajectory planning for multiple UAVs in dynamic and uncertain environments". In: *2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*. 2015, pp. 377–382. DOI: 10.1109/IntelCIS.2015.7397248 (cit. on p. 12).

[29] P. O. Pereira et al. "Leader following trajectory planning: A trailer-like approach". In: *Automatica* 75 (2017-01), pp. 77–87. ISSN: 00051098. DOI: 10.1016/j.automatica.2016.09.001 (cit. on p. 12).

[30] N. Tran, E. Bulka, and M. Nahon. "Quadrotor control in a wind field". In: 2015-07, pp. 320–328. DOI: 10.1109/ICUAS.2015.7152306 (cit. on pp. 13, 40).

[31] D. Di Bacco, M. Giurato, and M. Lovera. "Accurate positioning of multirotor UAVs for civil infrastructure monitoring". In: *Proceedings of the 7th European Conference for Aeronautics and Space Sciences*. 2017-07. DOI: 10.13009/EUCASS2017-472 (cit. on pp. 13, 28, 40).

[32] C. Powers et al. "Influence of Aerodynamics and Proximity Effects in Quadrotor Flight". In: 2013-01, pp. 289–302. ISBN: 978-3-319-00064-0. DOI: 10.1007/978-3-319-00065-7_21 (cit. on p. 13).

[33] J. A. Preiss et al. "Downwash-aware trajectory planning for large quadrotor teams". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 250–257. DOI: 10.1109/IROS.2017.8202165 (cit. on p. 13).

[34] N. Sydney, B. Smyth, and D. Paley. "Dynamic control of autonomous Quadrotor flight in an estimated wind field". In: 2013-12, pp. 3609–3616. ISBN: 978-1-4673-5717-3. DOI: 10.1109/CDC.2013.6760438 (cit. on pp. 13, 29).

[35] M. Faessler, A. Franchi, and D. Scaramuzza. "Differential Flatness of Quadrotor Dynamics Subject to Rotor Drag for Accurate Tracking of High-Speed Trajectories". In: *IEEE Robotics and Automation Letters* PP (2017-11), pp. 1–1. DOI: 10.1109/LRA.2017.2776353 (cit. on pp. 13, 26–28).

[36] B. Neves and B. Guerreiro. "Flight control of hybrid drones towards enabling parcel relay manoeuvres". In: 2021-07, pp. 13–19. DOI: 10.1109/YEF-ECE52297.2021.9505161 (cit. on p. 14).

[37] *Robot Operating System (ROS)*. (Accessed: 2022-02-16). URL: https://www.ros.org/ (cit. on p. 14).

[38] *Gazebo - Robot simulation made easy*. (Accessed: 2022-02-14). URL: https://gazebosim.org/home (cit. on p. 14).

[39] *PX4 User Guide*. (Accessed: 2022-02-16). URL: https://docs.px4.io/master/en/ (cit. on p. 14).

[40] H. Cabrita and B. Guerreiro. "NOVA.DroneArena: design and control of a low-cost drone testbed". In: *2021 International Young Engineers Forum (YEF-ECE)*. 2021, pp. 20–25. DOI: 10.1109/YEF-ECE52297.2021.9505090 (cit. on p. 14).

[41] M. Vrba et al. "Autonomous capture of agile flying objects using UAVs: The MBZIRC 2020 challenge". In: *Robotics and Autonomous Systems* 149 (2021-12), p. 103970. DOI: 10.1016/j.robot.2021.103970 (cit. on p. 15).

[42] G. Strimel, S. Bartholomew, and E. Kim. "Engaging Children in Engineering Design Through the World of Quadcopters". In: *Children's Technology and Engineering Journal* 21 (2017-05), pp. 7–11 (cit. on p. 17).

[43] A. Zul Azfar and D. Hazry. "Simple GUI design for monitoring of a remotely operated quadrotor unmanned aerial vehicle (UAV)". In: *2011 IEEE 7th International Colloquium on Signal Processing and its Applications*. 2011, pp. 23–27. DOI: 10.1109/CSPA.2011.5759836 (cit. on pp. 16, 17).

[44] J. Diebel. "Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors". In: *Matrix* 58 (2006-01), pp. 1–35 (cit. on pp. 19, 20).

[45] M. D. T. S. Company. *Euler Angles, Quaternions, and Transformation Matrices for Space Shuttle Analysis*. Tech. rep. 19770019231. Design Note No. 1.4-8-020. Houston, TX: NASA, 1977. URL: https://ntrs.nasa.gov/api/citations/19770019231/downloads/19770019231.pdf (cit. on p. 19).

[46] W. R. Hamilton. "II. On quaternions; or on a new system of imaginaries in algebra". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 25.163 (1844), pp. 10–13. DOI: 10.1080/14786444408644923. eprint: https://doi.org/10.1080/14786444408644923. URL: https://doi.org/10.1080/14786444408644923 (cit. on p. 20).

[47] J.-M. Kai et al. "Nonlinear feedback control of Quadrotors exploiting First-Order Drag Effects". In: *IFAC-PapersOnLine* 50.1 (2017). 20th IFAC World Congress, pp. 8189–8195. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2017.08.1267. URL: https://www.sciencedirect.com/science/article/pii/S2405896317317822 (cit. on p. 22).

[48] D. Mellinger and V. Kumar. "Minimum snap trajectory generation and control for quadrotors". In: *2011 IEEE International Conference on Robotics and Automation*. 2011, pp. 2520–2525. DOI: 10.1109/ICRA.2011.5980409 (cit. on p. 23).

[49] N. G. R. Center. *Relative Velocity - Ground Reference*. (Accessed: 12-08-2024). URL: %5Curl%7Bhttps://www.grc.nasa.gov/WWW/k-12/airplane/move.html%7D (cit. on pp. 26, 27).

[50] R. Nangia. "Aerodynamics, Aeronautics and Flight Mechanics — Second edition, B.W. McCormick, John Wiley;amp; Sons, Baffins Lane, Chichester, West Sussex P019 1UD. 1995. 652pp. Illustrated. £22.50". In: *The Aeronautical Journal* 100.991 (1996), pp. 36–36. DOI: 10.1017/S000192400002724X (cit. on p. 32).

[51] Y. Lei and J. Wang. "Aerodynamic Performance of Quadrotor UAV with Non-Planar Rotors". In: *Applied Sciences* 9 (2019-07), p. 2779. DOI: 10.3390/app9142779 (cit. on p. 32).

[52] Q. Guo et al. "Numerical Simulation and Validation of Downwash Airflow During Dual-Aircraft Collaborative Operations of Quad-Rotor Agricultural UAVs". In: *SSRN Electronic Journal* (2024). DOI: 10.2139/ssrn.4801394. URL: https://ssrn.com/abstract=4801394 (cit. on p. 32).