UNIVERSIDADE FEDERAL DE CAMPINA GRANDE

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

DOCTORATE DISSERTATION

Fault Detection and Diagnosis System Considering Aerodynamic Effects for Unmanned Aerial Vehicles

Sarah Pontes Madruga

Campina Grande - PB

March 2023

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

DOCTORAL DISSERTATION

Fault Detection and Diagnosis System Considering Aerodynamic Effects for Unmanned Aerial Vehicles

Sarah Pontes Madruga

Doctoral Dissertation submitted to the Coordenação do Curso de Pós-Graduação em Engenharia Elétrica from Universidade Federal de Campina Grande, as part of the requirements for obtaining the degree of D.Sc. in Electrical Engineering.

Concentration area: Information Processing

Antonio Marcus Nogueira Lima, Dr.
Supervisor
Tiago Pereira Nascimento, Dr.
Supervisor

Campina Grande - PB
March 2023

M183f Madruga, Sarah Pontes.

Fault detection and diagnosis system considering aerodynamic effects for unmanned aerial vehicles / Sarah Pontes Madruga. – Campina Grande, 2023.

124 f. : il. color.

Tese (Doutorado em Engenharia Elétrica) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, 2023.

"Orientação: Prof. Dr. Antonio Marcus Nogueira Lima, Prof. Dr. Tiago Pereira Nascimento".

Referências.

1. Processamento da Informação. 2. Quadricóptero. 3. VANT. 4. Batimento das Pás das Hélices. 5. Rede Neural. 6. Detecção de Diagnóstico de Falhas. I. Lima, Antonio Marcus Nogueira. II. Nascimento, Tiago Pereira. III. Título.

CDU 004.451.7:004.7(043)

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECÁRIA SEVERINA SUELI DA SILVA OLIVEIRA CRB-15/225

Sarah Pontes Madruga

Fault Detection and Diagnosis System Considering Aerodynamic Effects for Unmanned Aerial Vehicles

Doctoral Dissertation submitted to the Coordenação do Curso de Pós-Graduação em Engenharia Elétrica from Universidade Federal de Campina Grande, as part of the requirements for obtaining the degree of Doctor in Electrical Engineering, with specialization in Information Processing.

Approved on: March 31st 2023

JOSE SERGIO DA ROCHA NETO:11007729449 Assinado de forma digital por JOSE SERGIO DA ROCHA NETO:11007729449 Dados: 2023.04.28 13:29:06 -03'00'

D.Sc. José Sérgio da Rocha Neto, *Evaluating commission member*

Saulo O. D. Luiz

D.Sc. Saulo Oliveira Dornellas Luiz, *Evaluating commission member*

[illegible]

Dr. André Gustavo Scolari Conceição, *Evaluating commission member*

Paulo Lilles

Dr. Paulo Lilles Jorge Drews Júnior, *Evaluating commission member*

Campina Grande - PB

March 2023

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
POS-GRADUACAO EM ENGENHARIA ELETRICA
Rua Aprigio Veloso, 882, - Bairro Universitario, Campina Grande/PB, CEP 58429-900

REGISTRO DE PRESENÇA E ASSINATURAS

ATA DA DEFESA PARA CONCESSÃO DO GRAU DE DOUTOR EM CIÊNCIAS, NO DOMÍNIO DA ENGENHARIA ELÉTRICA, REALIZADA EM 31 DE MARÇO DE 2023 (Nº360)

Filipe Emmanuel Porfírio Correia
Secretário

2.1. Segue a presente Ata de Defesa de Tese de Doutorado do candidato SARAH PONTES MADRUGA, assinada eletronicamente pela Comissão Examinadora acima identificada.

2.2. No caso de examinadores externos que não possuam credenciamento de usuário externo ativo no SEI, para igual assinatura eletrônica, os examinadores internos signatários certificam que os examinadores externos acima identificados participaram da defesa da tese e tomaram conhecimento do teor deste documento.

Resumo

O controle de atitude em quadricópteros usualmente não considera a possibilidade de perda de effetividade no atuator, tornando-se ineficiente para operar em certas condições. os algoritmos de Detecção e Diagnóstico de Falhas (FDD) disponíveis na literatura não levam em consideração a presença de efeitos aerodinâmicos como batimento das pás das hélices (*blade flapping*) e a força de arrasto induzida (*induced drag*), que prejudicam o desempenho de voo. A atenuação na velocidade de rotação dos motores causada por estes efeitos pode inclusive ser confundida por uma falha de perda de efetividade. Assim, é oportuno ter um FDD que é capaz de considerar esses efeitos.

Sistemas de controle de quadricópteros como um todo tendem a negligenciar a presenca desses efeitos aerodinâmicos. Esta tese propõe, primeiramente, que estes efeitos podem ser compensados na alocação de controle, usando uma *function-fitting neural network* como uma ferramenta que substitui uma matriz de alocação clássica. O treino da rede é feito *offline*, economizando poder computacional. O sistema alvo é um drone PARROT Mambo. Este quadricóptero é particularmente susceptível aos efeitos aerodinâmicos de interesse desta tese, dado o seu tamanho reduzido. Neste sentido, quando comparam-se os comandos enviados pelo controlador, que devem ser seguidos pelos atuadores, com os conjugados mecânicos gerados por eles, o uso da rede neural proposta torna possível diminuir o erro de alocação. Ou seja, os atuadores executam o esforço de controle apropriado pedido pelo controlador, enquanto a matriz de alocação clássica não consegue atingir o mesmo desempenho. Além disso, o comportamento de malha fechada também foi melhorado com o uso da nova alocação de controle (NNCA), bem como a qualidade dos sinais de torque e empuxo, nos quais perceberam-se comportamentos mais suaves e menos ruidosos.

A fim de manter a coerência com a inclusão da modelagem dos efeitos aerodinâmicos no sistema por meio da NNCA, estas equações também foram incluídas da formulação do FDD. O novo sistema de detecção de falhas é baseado em um EKF de ordem reduzida tolerante a efeitos aerodinâmicos (ROEKF-TAE), capaz de detectar falhas de perda de efetividade (LOE) em mais de um atuador simultaneamente. Com este objetivo, introduziu-se na formulação do estimador de falhas um modelo do quadricóptero que inclui as equações dos efeitos aerodinâmicos pertinentes. A NNCA previamente desenvolvida também foi utilizada em todas as simulações computacionais e experimentos de voo do FDD. Os testes de voo para esta nova abordagem de FDD foram realizados utilizando o mesmo mini drone PARROT mambo. Compararam-se os resultados do ROEKF-TAE proposto com os do EKF tradicional e com os de uma solução do estado da arte que utiliza um filtro de Kalman adaptativo. O ROEKF-TAE foi capaz de distinguir melhor entre os efeitos da falha de fato e aqueles que era interferência do *blade flapping*. Ainda, ele tem melhor precisão ao diferenciar os atuadores defeituosos dos saudáveis.

Palavras-chave: quadricóptero, VANT, batimento das pás das hélices, rede neural, detecção e diagnóstico de falhas

Abstract

Attitude control in quadrotors usually does not consider the possibility of actuator loss-of-effectiveness, becoming then inefficient at certain operating conditions. Fault-Detection and Diagnosis (FDD) algorithms available in the literature do not take into consideration the presence of aerodynamic effects such as blade flapping and induced drag, which hinders flight performance. The attenuation these effects cause on actuator speed could even be mistaken for a loss-of-effectiveness fault, so an FDD system capable of identifying them could be opportune.

Typical quadrotor control systems as a whole tend to neglect the presence of such aerodynamic effects. As a primary effort, this thesis proposes that these effects can be compensated in the control allocation step, using a function-fitting neural network as a tool to replace the classic allocation matrix, without using the aerodynamic inflow equations directly. The network training is performed offline, saving on computational power. The target system is a PARROT Mambo drone. This specific quadrotor is particularly susceptible to the aerodynamic effects of interest to this thesis, given its small size. In this sense, when comparing the commands sent by the controller that must be achieved by the actuators, to the mechanical torques generated by them, the usage of the proposed neural network control allocation (NNCA) makes it possible to achieve a close match, meaning the actuators execute the appropriate control effort demanded by the controller, while the classic allocation matrix cannot perform the same way. Furthermore, the closed-loop performance was also improved with the use of the new control allocation, as well as the quality of the thrust and torque signals, in which a much less noisy behavior was perceived.

In order to be coherent with the inclusion of the aerodynamic effects modeling into the system through the control allocation, these equations are also included in the FDD formulation. The new fault detection scheme is based on a Reduced Order EKF Tolerant to Aerodynamic Effects (ROEKF-TAE) capable of detecting loss-of-effectiveness (LOE) faults in more than one actuator at the same time. With this goal, a more complete model of the quadrotor system including the pertinent aerodynamic effects equations is introduced in the filter's development. Moreover, the previously developed NNCA was applied in all the new FDD scheme simulations and experiments. The experimental flights for the new FDD approach were performed using the same PARROT Mambo micro drone. They were compared to those of a traditional EKF, as well as to those of a state-of-the-art adaptive Kalman Filter available in the literature. The ROEKF-TAE is able to better distinguish between the effects of the actual fault and those of the blade-flapping disturbance. Also, it has better accuracy in identifying which actuators are truly defective and which are not.

Keywords: quadrotor, UAV, blade flapping, control allocation, neural network, fault detection and diagnosis

Acknowledgments

First and foremost I would like to thank my husband. It is probably even an understatement to say that this dissertation would not have been finished without him by my side, supporting me endlessly while facing his own struggles in this shared profession of ours. But since this is all I can put into words, it will have to do. Anywhere you wanna go, baby.

Thank you to my family, my parents, brother, and in-laws, for supporting and helping me in this long and hard professional journey I chose. My mum, for always showing me what it really means to be a woman. My dad, for his impeccable example of generosity and diligence. I've come to learn that a PhD dissertation is written by a lot of invisible hands beside mine.

I must also thank my supervisors, Prof. Antonio Marcus and Prof. Tiago Nascimento, for their patience with my mistakes, for their always immediate availability and advice. Thank you as well to Prof. Saulo Dornellas, for always being available to contribute to this research, and to other colleagues and researchers that helped me in this endeavor, from UFPB, UFCG and TUM-FSD.

My introverted self is very lucky to say that I have been blessed with so many friends in these past few years that I couldn't possibly list them all by name. My armadillo friends, thank you for absolutely always being there to hear my crises, my complaints, my successes, and my infodumpings. Thank you to my ineffable friends (you know who you are), who were so important to me in the last and most difficult stretch of this journey, taking my mind off work when I needed it, and welcoming me into such an amazing environment. Nath, Milla and Cris, thank you so much for always cheering me on as if it were yourselves. Not many people in life do that with the sheer amount of honesty that you do.

More than half of this research was conducted during a genuine global scale pandemic, and I would not be here were it not for the support of all these amazing people who stood by me in these challenging times.

Finally, this study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and the UFCG Post-graduate Program of Electrical Engineering (PPgEE).

"You don't need sleep. I don't need sleep. Evil never sleeps, and Virtue is ever-vigilant."
"Evil in general, maybe. This specific part of it has got into the habit of getting its head down occasionally."

— Terry Pratchett, Neil Gaiman. *Good Omens: The Nice and Accurate Prophecies of Agnes Nutter, Witch*

Contents

List of abbreviations, acronyms and symbols

α^i	Loss-of-effectiveness fault factor
α_T	Thrust coefficient
$\beta_{lf,i}$	Longitudinal flapping
γ_{aero}	Lock number
ν_k	Uncorrelated zero-mean white Gaussian noise
$\mathbf{\Omega}(t)$	Angular velocity vector in the body-frame
$\mathbf{\Psi}(t)$	Angular position vector
$\mathbf{\Sigma}_{k+1}$	Covariance of $\mathbf{e}_{k+1}$
$\mathbf{A}_k$	Jacobian of the predicted state with respect to the previous state
$\mathbf{B_m}$	Mixing matrix, allocation matrix or control effectiveness matrix
$\mathbf{e}_x$	Unit vector in the x-axis
$\mathbf{e}_y$	Unit vector in the y-axis
$\mathbf{e}_z$	Unit vector in the z-axis
$\mathbf{e}_{k+1}$	EKF innovation sequence
$\mathbf{F_d}$	Disturbance caused by drag forces

$\mathbf{J_x}$	Moment of inertia in x-axis
$\mathbf{J_y}$	Moment of inertia in y-axis
$\mathbf{J_z}$	Moment of inertia in z-axis
$\mathbf{J}$	Matrix of moments of inertia
$\mathbf{J}_{ss,i}$	Sideslip rotation matrix
$\mathbf{M}$	Vector of torques
$\mathbf{Q}_k$	Process covariance matrix
$\mathbf{R_x}$	x-axis rotation matrix
$\mathbf{R_y}$	y-axis rotation matrix
$\mathbf{R_z}$	z-axis rotation matrix
$\mathbf{R_{il}}$	Rotation matrix from body to inertial frame coordinates
$\mathbf{r}(t)$	Linear position vector
$\mathbf{r}^{B}_{prop}$	Propeller position with respect to the center of mass
$\mathbf{R}_k$	Measurement covariance matrix
$\mathbf{S}(\mathbf{\Psi})$	Coupling matrix
$\mathbf{T}^B$	Total thrust force acting over the quadrotor's center of mass
$\mathbf{T}^{B,i}$	Individual thrust force produced by each rotor
$\mathbf{v}(t)$	Linear velocity vector in the inertial frame
$\mathbf{v}^{B}_{ra,i}$	Relative speed at propeller i
$\mathbf{w}_k$	Uncorrelated zero-mean white Gaussian noise
μ_i	Planar components
ω_i	Angular velocity of each actuator (motor + propeller set) i

$\overline{\alpha}^i$	Lower bound for the fault factor that keeps the quadrotor controllable
ϕ	Angle of rotation around the x-axis *roll*
ψ	Angle of rotation around the z-axis *yaw*
ρ	Air density
τ	Vector of total torques acting on the quadrotor
τ_ϕ	Torque around the *roll* axis (x)
τ_ψ	Torque around the *yaw* axis (z)
τ_θ	Torque around the *pitch* axis (y)
b	Neural Network bias
K	Kalman gain
P	State covariance
u	Control inputs, actuator commands
W	Neural Network weights
θ	Angle of rotation around the y-axis *pitch*
θ_{b0}	Blade root angle
θ_{b1}	Blade twist angle
$\widetilde{F}$	Model of the virtual function F which describes the transformation of the actuator commands
$\widetilde{F}^*$	Function which performs the inverse operation of $\widetilde{F}$
a_0	Slope of the lift curve per radian
$a_{1,s,i}$	Longitudinal *blade flapping* angle
$b_{1,s,i}$	Lateral *blade flapping* angle

c	Chord of the blade
C_o	Controllability matrix
E	Expectation function
g	Gravitational acceleration
I	Identity matrix
I_b	Moment of inertia of the blade about the hinge
j_i	Sideslip azimuth retalive to x-axis
Kd	Derivative gain
Ki	Integral gain
Kp	Proportional gain
l_i	Non-dimensionalized normal inflow
m	Mass of the vehicle
O	Observability matrix
$P(t)$	Angular velocity in the x-axis
$Q(t)$	Angular velocity in the y-axis
Q_i	Propeller-drag-induced torque
R	Rotor radius
$R(t)$	Angular velocity in the z-axis
t_r	Rise time
t_s	Settling time
t_{Fi}	The instant in time when the fault occurs
T_{x_i}	Thrust force component in the x-axis for each actuator i

T_{y_i}	Thrust force component in the y-axis for each actuator i
T_{z_i}	Thrust force component in the z-axis for each actuator i
$U(t)$	Linear velocity in the x-axis
$V(t)$	Linear velocity in the y-axis
$W(t)$	Linear velocity in the z-axis
$x(t)$	Position in the x-axis
$y(t)$	Position in the y-axis
$z(t)$	Position in the z-axis
w_k^α	Uncorrelated zero-mean white Gaussian noise
λ	Eigenvalues
EKF	Extended Kalman Filter
FDD/FDI	Fault Detection and Isolation/Identification
LOE/LIE	Loss-of-effectiveness/Loss-in-effectiveness
MMCA	Mixing Matrix Control Allocation
MSE	Mean Squared Error
PD	Proportional Derivative
PID	Proportional Integral Derivative
ROEKF-TAE	Reduced Order Extended Kalman Filter Tolerant to Aerodynamic Effects
UAV	Unmanned Aerial Vehicles

List of Figures

List of Tables

Chapter 1

Introduction

In this introductory chapter, we discuss the current scenario that harbors the motivations for this research. We present our hypothesis based on the state-of-the-art literature that is investigated in-depth in Chapter 2, as well as the research goals, main contributions, published results and the document structure.

1.1 Motivation

The usage of Multi-Rotor Unmanned Aerial Vehicles (UAV) has had substantial growth in a series of applications in the last fifteen years. Such drones are powered by multiple rotors (motor and propeller) to generate lift and control torques, which likely operate in considerable speed conditions, making aerodynamic effects more relevant [Sun et al. 2018]. As a consequence, the research of modeling and control techniques for this type of vehicle, especially the quadrotors, has also grown considerably [Nascimento & Saska 2019].

A quadrotor is a classically unstable and under-actuated nonlinear dynamical system [Bouktir, Haddad & Chettibi 2008], [Hussein & Nemah 2015], making the study of suitable algorithms to improve stability and reduce the influence of disturbances indispensable. Furthermore, since a minimum of four fixed-pitch actuators is required to achieve complete attitude control, the quadrotor UAV types suffer most from rotor failures, because there is no actuator redundancy. Hence, the failure of rotors severely threatens flight safety.

In this regard, we can highlight that many approaches have been proposed as Fault Detection and Diagnostics (FDD) solutions, but only with simulation results, as in [Ma, Xu &

Yang 2021], [Rot, Hasan & Manoonpong 2020], [Zhao, Liu & Lewis 2021]. Works which explore the aircraft behavior through real robot experiments, i.e. flight tests, are still a minority in the literature. Especially if we search for flight tests with maneuvers other than hover, where the influence of aerodynamic effects is significant. In other words, there is a shortage of existing literature regarding the application of FDD techniques in real systems operating under realistic conditions [Nascimento & Saska 2019], that is, outside hover behavior.

The presence of aerodynamic effects such as blade flapping and induced drag generates restrictions that lead to actuator saturation [Alves 2019]. These effects are especially relevant, because they induce forces in the xy-plane of the rotor, precisely the under-actuated directions of the flight dynamics which cannot be easily managed by high gain control [Mahony, Kumar & Corke 2012]. An FDD system that does not take into consideration aerodynamic effects might be subject to mistake a loss-of-effectiveness actuator fault for the attenuation of actuator speed caused by these disturbances [Okada et al. 2021].

In the context of considering the presence of aerodynamic effects in the system, the use of control allocation offers the advantage of a modular design where the motion control algorithm can be designed without detailed knowledge about the actuators (here, meaning the motor and propeller set). Several issues such as input saturation, rate constraints, actuator fault tolerance, power efficiency etc., could be managed within the control allocation algorithm [Johansen & Fossen 2013]. What we know as control laws specify which total control effect to produce but not how the actuators should produce it; hence, the need for control allocation arises [Härkegård 2003]. This research will contribute to managing aerodynamic effects in a quadrotor system by considering them in the control allocation step.

Conventional multirotor UAVs have rotors in parallel directions to collectively counteract gravity and optimize flight time. This structure induces an under-actuated dynamics due to the coupling between the horizontal translational and rotational dynamics [Rashad et al. 2020], as aforementioned. As a result, the control of a quadrotor tends to present a high sensibility when subjected to an input constraint [Alves 2019]. Therefore, control allocation is necessary for setting suitable commands for each rotor, so that it generates the desired control effect, that is, the forces and torques, while presenting an acceptable performance and respecting the constraints imposed by the minimal and maximal angular velocities and hence the maximum and minimum forces and torques [Alves 2019].

In the classic quadcopter model used in most applications, the control allocation is often performed through the control effectiveness matrix, usually named mixing matrix or allocation matrix. It provides an invertible matricial relation between mechanical conjugates and motor rotation speeds. This matrix results from a model simplification that neglects the aerodynamic effects that influence thrust forces and torque behavior, especially regarding small aircraft or aggressive maneuvering [Huang et al. 2009]. As a matter of fact, this negligence leaves the quadcopter subject to saturation, and the torque and thrust commands sent by the controller are not properly generated by the actuators, because the motors cannot achieve the necessary rotation speed.

While it is true that there are several results regarding UAV control in the literature, this research did not find works approaching the problem of the error inserted by the disregard of aerodynamic effects in the control allocation. Even the robust controllers available in the literature mentioned in Chapter 2 still use the classic mixing matrix, forcibly inserting an error in the rotation speeds established at the motors and hence in the control performance. In fact, even with many proposed techniques for quadrotor UAV control, there is still a need for contributions regarding the compensation of small disturbances that affect the system overall [Nascimento & Saska 2019]. Though one must make it clear to the reader that the focus of this thesis is on the control allocation step and fault detection and diagnosis in the presence of aerodynamic effects. The main contribution of this work does not rely on the controller algorithm itself.

For the sake of coherence, the aerodynamic effects must also be included within the FDD system formulation. The proposed fault detection scheme is based on an Extended Kalman Filter state estimator, however with reduced order, called Reduced Order Extended Kalman Filter Tolerant to Aerodynamic Effects (ROEKF-TAE), focused on detecting, estimating and locating detecting loss-of-effectiveness (LOE) faults in more than one actuator at the same time.

We reiterate once more that the majority of solutions available in the literature for quadrotor UAVs do not take into account aerodynamic disturbances such as blade-flapping. If an actuator fault occurs, for instance, during high-speed flights, such as the cruising phase of a delivery drone, air inflow with respect to rotors and the airframe brings significant aerodynamic disturbances [Sun et al. 2018], which can greatly deteriorate the performance of the

FDD and of the system overall [Madruga et al. 2022]. Also, in the case of a system with multiple actuators, for instance, a multi-rotor aircraft, not all results available in the literature allow the identification of faults in multiple actuators simultaneously.

Figure 1.1 is a simplified diagram of the quadrotor control system, highlighting the points where this research's contribution lies. Focusing on the actuator commands and command generation, the modifications we made in the control allocation block ensure that the Actuator Velocities will generate the correct Thrust and Torques. Meanwhile, the Fault Detection and Diagnosis System detects that the command generation is disturbed by loss-of-effectiveness faults.

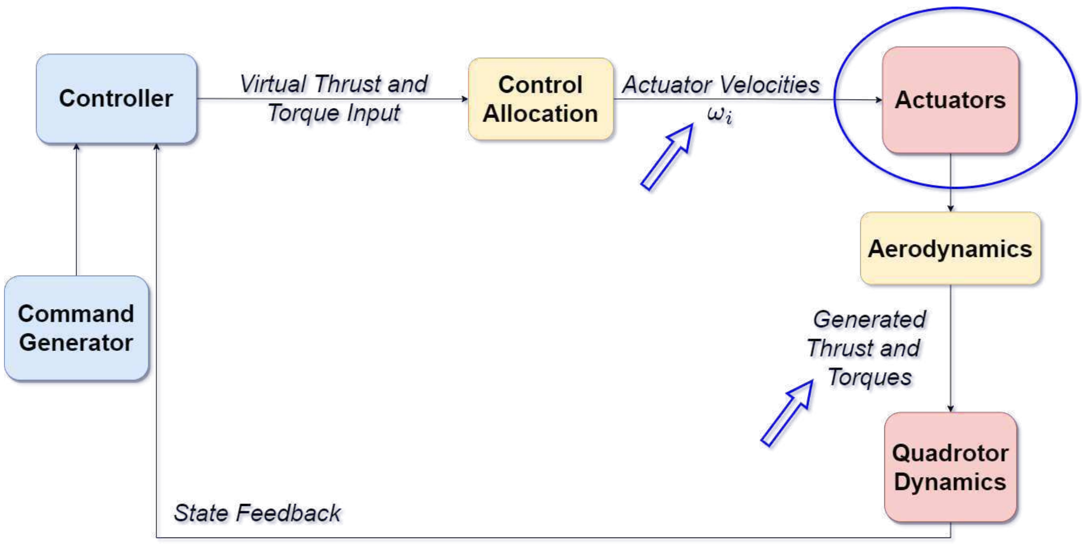

Figure 1.1: Research focus.

1.2 Methodology and Objectives

As a solution for the research problem defined in the previous Section, the first goal of this thesis is to propose a new control allocation method to take into account the aerodynamic effects, most notably those regarding blade flapping and inflow effects as described in [Riether 2016], based on a function fitting neural network to replace the classic mixing matrix. This method allows us to improve the controller performance without the need for directly using aerodynamic effects equations in the control algorithm or the allocation problem. Furthermore, network training is performed offline, and the in-flight control allocation depends only on the virtual control inputs and some state-space variables, which will be detailed further in this thesis.

Further, in the second part of this research, we finally deal with the problem of fault detection and diagnosis in the presence of aerodynamic effects. Similarly as stated above, when dealing with FDD, the methods available in the state-of-the-art literature still neglect said effects, either in the modeling or in the experiments [Zhong et al. 2018], [Fourlas & Karras 2021]. The lack of a proper representation of such effects leads to errors in motor fault estimation. Thus, an estimation method will be proposed in this thesis to consider aerodynamic effects when detecting faults in UAV actuators. The fault detection and diagnosis (FDD) scheme is based on a Reduced Order Extended Kalman Filter Tolerant to Aerodynamic Effects, capable of detecting loss-of-effectiveness faults in more than one actuator at the same time. This FDD will then be combined with the Neural Network Control Allocation in the final system.

The main platform used in this research was the PARROT Mambo mini-drone. MATLAB/Simulink provides a hardware support package for this equipment, with a simulation environment and Bluetooth connection for flight tests. The quadrotor is represented in Figure 3.1.

As will be detailed further in Chapter 4 and 5, the mini-drone is equipped with a downward-facing camera for positioning using optical-flow odometry, an ultrasonic sensor, a barometric sensor, and an inertial measurement unit (IMU). All these sensors are used for stabilization purposes. Its dimensions are 180 mm $\times$ 180 mm $\times$ 40 mm, weighting 70 g. Power is provided by a GiFi Power 3.7 V, 600 mAh, 2.2 Wh, 15c Li-Po battery, enabling a flight time of 7-10 minutes, depending on the nature of the task to be performed and the accessories (i.e. the load) implemented.

We can then summarize the main objectives of this research and the respective Chapters where they are dealt with:

- Characterize the effects of the blade flapping on the thrust force rotation (Chapter 3).
- Develop a new control allocation approach that is able to compensate for aerodynamic effects such as blade flapping and induced drag (Chapter 4).
- This new control allocation will be based on a function-fitting neural network with

offline training so as not to be computationally demanding during flight.

- Improve the flight performance with the proposed technique (Chapter 4).
- Develop an actuator fault detection and diagnosis system that is able to distinguish small faults from the aforementioned aerodynamic disturbances (Chapter 5).
- The FDD must be able to detect faults in more than one actuator simultaneously (Chapter 5).

Our results have shown that neglecting the presence of aerodynamic effects in the quadrotor system hinders flight performance, especially, but not limited to, in the case of small aircraft and aggressive maneuvers. The proposed neural network is successfully applied in the control allocation. Afterward, the new FDD system is developed, and we show through comparison that considering the aerodynamic effects improves the detection quality.

Each final contribution of this work is enumerated in Chapter 6, as well as the two scientific papers that resulted from the contributions in Chapters 3, 4 e 5, following the list of objectives above.

1.3 Dissertation Structure

Previously this Chapter has presented the motivations that drove this research, the main objectives of this work and the methodology used throughout the thesis. The leading contributions have also been introduced. Now, we list the structure in which this document is organized so that it may facilitate comprehension.

In Chapter 2, we discuss the related works and the state-of-the-art literature, comparing our contributions to what has already been published and studied in the literature in recent years.

Chapter 3 presents the basic concepts for understanding the quadrotor mathematical model, as well as the state variables and observability and controllability analysis. This Chapter also discusses the aerodynamic effects equations and how they modify the classic model, along with one of the theoretical contributions of this work, regarding an analysis of the thrust force rotation around the rotor plane when flying under blade flapping effects.

Chapter 4 deals with the neural network control allocation contribution. It starts by approaching the basic control structure that is used in the system during simulations and real flight. The control allocation concept is discussed, and the results of simulation and flight experiments are shown and analyzed.

In Chapter 5, we present the details about the fault detection and diagnosis system, capable of detecting, estimating and locating actuator faults in more than one rotor at the same time while taking into consideration the aerodynamic effects. The results of the simulation and flight experiments of this contribution are discussed in the end.

Finally, Chapter 6 concludes this dissertation, recapitulating the whole discussion and listing the contributions, as well as suggesting possible future works that might derive from this research.

Chapter 2

Related Works

In this Chapter, related works in the form of scientific papers, theses, or dissertations regarding the relevant themes for this research are analyzed, in an effort to compare contributions and evaluate where the current state-of-the-art lands.

Given the lack of works approaching the subjects in the ways proposed in this thesis, that is, modification of the control allocation step to take into account aerodynamic effects by means of a neural network, followed by a fault detection and diagnosis scheme that also considers said effects, and also performing real flight experiments with trajectories other than hover to validate the results, this bibliographic review is to be divided into the four parts that build the main focus of this research.

First, the discussion will concentrate on analyzing the works that address aerodynamic effects in its scope, be it as a modeling effort (which is the majority of cases mentioned here), or in the control. Next, works that study control allocation within the UAV context will be presented, especially those that mention the compensation of aerodynamic effects. Some applications of neural networks in UAV control systems were also collected here, as well as a brief revision of available fault detection and diagnosis system in the literature.

Applications for quadrotor UAVs is a theme with quite literally thousands of available references, and one needs to select those more relevant adjacent to one's research. Even if some of the references cited here may have more than one decade from the date of publication, the selection criteria laid in the neighboring research themes or pertinent contributions. This review has also given priority to those works with strong results backed by real flight experiments and/or the inclusion of aerodynamic effects modeling in a step of the system.

Most of the available relevant applications fall within the control algorithm spectrum, so some works regarding the control are to be discussed here, since many times they are the closest possible comparison to what is being proposed in this research.

2.1 Aerodynamic Effects

Dynamic models with minimum complexity for quadrotors are well documented in the literature [Cai et al. 2017]. These models generally use six-degree-of-freedom (6-DOF) rigid-body equations and highly simplified rotor dynamics as a function of the square of motor speed. There are several aerodynamic effects that can modify the classic model, but specifically from the rotors' point of view, the most significant are blade-flapping and induced drag [Riether 2016]. In this bibliographic review, works dealing with other types of aerodynamic effects will also be mentioned, given the gap in the literature for taking into account aerodynamic effects in general when dealing with quadrotors.

The treatment of quadcopter dynamics around steady-state conditions has often ignored rotorcraft aerodynamic effects due to its complex physical modeling [Kantue & Pedro 2018]. In certain circumstances similar to hover flight, these effects could be neglected. Nevertheless, they can prove to be very relevant during agile maneuvers [Huang et al. 2009] or in the case of small aircraft [Davis & Pounds 2017], [Ning, Wlezien & Hu 2017]. As modeling efforts, one can cite, for example, [Davis & Pounds 2017] which sought to understand and exploit the aerodynamic effects acting on a small-scale quadrotor to regulate relative position, attitude, and velocity without extrinsic sensors. Also, [Ning, Wlezien & Hu 2017] have investigated the aerodynamics and aeroacoustics of propellers for small unmanned aerial vehicles, while [Powers et al. 2013] demonstrated that the dynamic response and performance of a micro UAV are greatly influenced by its aerodynamics, and the introduction of these effects into controllers is also discussed. They suggest that the controller performance can be improved when considering aerodynamic effects in the model.

The works of [Pounds et al. 2004] [Pounds 2007], [Pounds, Mahony & Corke 2010], [Pounds, Mahony & Corke 2006] have included some aerodynamic effects modeling in the rotor dynamics equations in order to enhance model accuracy. The authors adopted a generalized flapping dynamics structure, based on helicopter theories, to model the rotor flapping phenomenon. [Bristeau et al. 2009] studied the aerodynamic role of the propellers, while [Hoffmann et al. 2007] and [Huang et al. 2009] modeled aerodynamic effects associated with blade flapping, airflow distribution, and thrust generation, to derive a model that can cover extreme maneuvers.

In [Cai et al. 2017], the authors introduced a frequency-domain modeling methodology that can be applied to various multi-rotor aerial vehicles. [Niemiec & Gandhi 2016] created a dynamic simulation for a 2 kg quadcopter, where several inflow models were used to analyze the behavior of the aircraft in trim, hover, and forward flight. [Kantue & Pedro 2018], [Kantue & Pedro 2018] used a Radial Basis Functions Neural Network (RBF-NN) for model identification of an unmanned quadcopter in accelerated flight to estimate rotor dynamics parameters from an unknown flapping dynamics model.

The authors in [Omari et al. 2013] present the design and evaluation of a nonlinear control scheme for a six-rotor helicopter that takes the first-order drag effects into account explicitly, and a hierarchical nonlinear controller is designed to actively compensate for the nonlinear effects caused by these drag forces. Note that, even though a dynamic model including the blade flapping and induced drag forces is included in the controller algorithm in this paper, the corresponding classic mixing matrix for a six-rotor helicopter is still employed by the authors. Even if the controller takes into account the drag forces, the control allocation completely neglects their existence, jeopardizing some of the controller's performance because the conversion of controller commands to the actuators' rotation speeds will be inaccurate.

This thesis does not aim to modify the controller algorithm in order to consider blade-flapping effects in the system. The approach proposed by this research performs modifications in the control allocation algorithm, that is, in the step that transforms the control input into actuator commands.

In [Chebbi et al. 2020], polynomial regression is used to obtain the control allocation of a coaxial multirotor. However, the problem the authors are trying to solve in this work differs from the one we attack in our paper. They try to estimate a model that will solve the interference between the propellers in a pair of coaxial motors. Once more, even though they estimate a thorough model of the actuators' nonlinear effects, they neglect them in the main control allocation step when converting the commands to the actuators.

The works presented above mainly focus on modeling several types of aerodynamic effects that impact multirotor aircraft systems. Literature is still lacking in regards to techniques to actually take these effects into account in the quadrotor system, especially when it comes to the control allocation, which even papers proposing robust controllers seem to neglect, maintaining the use of the classic allocation matrix, resulting in an inevitable accuracy error when sending commands to the actuators.

2.2 Control Allocation

Control allocation is the step in a control system where the control inputs, i.e. the commands sent by the controller, are converted into actuator commands so that the plant can behave accordingly [Johansen & Fossen 2013]. A control allocation algorithm is designed in order to map the vector of commanded virtual input forces and moments required by the controller into individual forces and moments generated by the actuators such that they amount to the commanded virtual input. In other words, the control allocation coordinates the different actuators such that they together produce the desired control efforts.

The fundamentals of control allocation theory are presented in [Härkegård 2003], [Johansen & Fossen 2013] and [Alves 2019]. They will also be detailed in Chapter 4. Most specifically, we find a least squares control allocation combined with backstepping techniques applied to aerospace control [Härkegård 2003]; a survey regarding different control allocation algorithms and applications to aerospace, maritime, automotive, and other application areas is presented in [Johansen & Fossen 2013], while the applications of several methods of control allocation to an experiment with a differential-drive mobile robot and simulations of a quadrotor UAV and a Remotely Operated Underwater Vehicle are presented in [Alves 2019].

Most recently, the interest in control allocation applied to multirotor vehicles has increased, given the rise in the usage of this type of aircraft. [Kirchengast, Steinberger & Horn 2018] showed the comparison of different control allocation techniques, as well as [Alves 2019], but this time they are applied to a laboratory setup that emulates the rotational dynamics of a quadrotor, though not to a real quadcopter. [Monteiro, Lizarralde & Liu Hsu 2016] and [Monteiro, Lizarralde & Liu Hsu 2016] also analyzed control allocation algorithms to countermeasure saturation in the actuators, however, they generalized the classic direct control allocation (DCA), which is optimization-based and computationally demanding, to suggest new methods, called partial control allocation (PCA) [Monteiro, Lizarralde & Liu Hsu 2016] and sub-optimal partial control allocation (S-PCA) [Monteiro, Lizarralde & Liu Hsu 2016].

In [Doman & Oppenheimer 2002], it is stated that forces and moments produced by the vehicle's aerodynamic control surface are often nonlinear functions and that this limits most control allocation algorithms applications, since they are based on the assumption that the control variable rates are linear functions. The authors then presented a mixed optimization control allocation scheme that minimizes the norm of the error and the deviation of the control input from a preferred value, to attack the inaccuracies introduced by these linear assumptions.

Since then, other authors have tried to address the problem of nonlinear input dynamics in control allocation, as [Johansen 2004], who developed an optimizing control allocation algorithm in the form of a dynamic update law, for a general class of nonlinear systems, using a control-Lyapunov approach. [Stolk 2017] proposed an incremental control allocation technique that calculates the input increments based on the control demand increment, instead of the total inputs to achieve the total control demand. [Waters et al. 2013] aimed to test the linearity assumption in the control allocation matrix by measuring the aerodynamic performance of several control allocation algorithms in a wind tunnel experiment for a blended wing body aircraft model, and the effect of different variables on nonlinearities in the control moment curves was also investigated.

Many works state that aerodynamic effects, such as wind disturbances, can lead to saturation [Smeur, Höppener & Wagter 2017], [Alves 2019]. Since these effects are nonlinear, [Waters et al. 2013] showed that some linear control allocation algorithms achieved only 50% of the requested moment in a wind tunnel. [Stolk 2017] used control allocation to achieve minimum drag, but the authors linearized the nonlinear control input dynamics in order to use linear methods. In addition, [Smeur, Höppener & Wagter 2017] and [Smeur, Höppener & Wagter 2017] dealt with the problem of actuator saturation for controlled flying vehicles. The authors integrated the Weighted Least Squares optimization-based control allocation algorithm into the controller algorithm, which allows for prioritization among roll, pitch, yaw, and thrust.

The conventional application area for control allocation algorithms is over-actuated mechanical systems, however, the principles of control allocation are general [Johansen & Fossen 2013] and can be applied to under-actuated systems subject to saturation and nonlinear effects, such as quadrotors [Smeur, Höppener & Wagter 2017], [Alves 2019], as seen in many of mentioned works above. [Oppenheimer, Doman & Sigthorsson 2010] and [Doman, Oppenheimer & Sigthorsson 2010] described a control allocation method to control five degrees of freedom using two physical actuators that drive flapping wings.

The main technique most available control allocation algorithms have in common is solving a constrained optimization problem [Ducard & Hua 2011], and this process is usually performed online [Härkegård 2004], requiring computational resources from embedded systems [Johansen & Fossen 2013]. Even though their computational complexity is mostly within the capabilities of today's computer technology, some works highlight the advantages of requiring a low computational power online [Schneider et al. 2012], such as the method we propose in this thesis. [Ducard & Hua 2011] presented an approach based on a weighted pseudo-inverse matrix method capable of exploiting a much larger domain of virtual control inputs for a hexacopter. The proposed control allocation algorithm is designed with explicit laws for fast operation and low computational load, suitable for a small microcontroller with limited floating-point operation capability. Finally, [Schneider et al. 2012] formulated the control allocation problem for a hexacopter as a parametric program and solved it for an explicit solution, which is stored in lookup tables. The authors emphasize that their control allocator requires very low computational power and immediately provides an optimal solution to the torque commands issued by the flight controller.

The control allocation theory establishes that it can be used to treat actuator saturation problems or other disturbances which may affect their optimal functioning [Johansen & Fossen 2013]. However, as of the moment of this research, to the best of the author's knowledge, there are no works proposing to specifically compensate aerodynamic effects that affect the actuators (blade-flapping and induced drag) inside the control allocation, replacing the traditional control effectiveness matrix and thus mitigating the allocation error.

2.3 Neural Networks Applications in Quadrotors

Neural networks can have several applications in a quadrotor UAV system. Mostly they are used for building a specific system model or identifying disturbance behavior through data analysis so that an inaccurate model is improved. The applications range from the controller algorithm to the sensing system, fault detection, and beyond.

Even though the focus of this thesis does not rely on the controller algorithm, which will be left unchanged here as a PD-PID cascade, the need to cite related works that deal with the controller rises, with the purpose of showing the adjacent literature contributions. However, we remind the reader that this thesis is proposing to use a neural network as a control allocation method to convert control inputs into actuator commands, without modifying the controller itself.

Neural networks have been receiving wide attention in the control field, being proposed especially for the identification and control of nonlinear dynamic plants [Wang, Zhang & Han 2016]. A common application of neural networks to controllers is as an adaptive element to controllers, as stated [Cao et al. 2019], [Jiang, Pourpanah & Hao 2020], and [Wang, Zhang & Han 2016]. Likewise, [Ansari, Bajodah & Hamayun 2019] and [Ansari & Bajodah 2019] employed a baseline controller with a dynamic inversion in the inner loop for attitude tracking and body rates stabilization, with the addition of a Radial Basis Function Neural Network to estimate the unknown attitude dynamics of the quadrotor.

In [Dierks & Jagannathan 2010] and [Dierks & Jagannathan 2008], a new nonlinear controller for a quadrotor is proposed using neural networks and output feedback. In these, the authors acknowledge that certain assumptions about UAV dynamics are not always practical, so the NN is introduced to learn the complete dynamics of the UAV online, including blade flapping. Nonetheless, the results of [Dierks & Jagannathan 2010] and [Dierks & Jagannathan 2008] are still shown only in simulation, with no real flight experiment. Besides, using a neural network to modify the whole control algorithm may be considered a solution of much more difficult implementation than the one in this thesis, which only alters the control allocation step for any type of quadrotor controller.

According to [Achermann et al. 2019], small UAVs are particularly susceptible to wind drag because of their relatively low mass and their operation at low altitudes where the wind environment can be more complex. The authors there proposed an approach for predicting high-resolution wind fields using a deep convolutional neural network to generate 3D wind estimates. In a similar topic, [Shi et al. 2019] presented a deep-learning-based robust nonlinear controller called Neural-Lander that improves the controller performance of a quadcopter during landing, since near-ground trajectory control is difficult for multi-rotor drones, due to the aerodynamic effects caused by interactions between multi-rotor airflow and the environment. [Verberne & Moncayo 2019] described the design of a controller architecture for wind disturbance rejection in quadrotor UAVs, followed by the application of adaptive artificial neural networks to correct inversion errors caused by wind disturbance.

Finally, [Reddinger & Gandhi 2017] trained a predictive neural network to estimate power as a function of the redundant control settings for a range of flight speeds for a compound helicopter. This neural network is used as a surrogate model for a gradient-based optimization to allocate the redundant control settings with a minimized power requirement.

Note that even though neural networks are used to identify aerodynamic models, or even in the controller itself, the application of a neural network to perform the control allocation of a quadrotor control system is still uncommon in the literature, especially considering aerodynamic effects, which is an application our research did not find in the most relevant works in this area of study.

2.4 Fault Detection and Diagnosis in Quadrotors

The FDD methods consist of two primary steps: the detection is responsible for identifying the presence of a problem, whereas the diagnosis incorporates the isolation (locating where it occurred) and identification/estimation (type and/or magnitude/intensity of the fault) of a fault.

Although many approaches can be found in the literature [Veras et al. 2019; Karlina & Indriawati 2020; Alex, Daniel & Jayanand 2016; Zhong et al. 2018], such as signal analysis, model-based, and data-driven [Okada et al. 2021], research shows that not all techniques will necessarily be capable of performing all steps involving FDD. It is not only possible, but common, that within a proposed technique only the fault detection is achievable, but not the isolation and estimation, while other approaches can perform detection, isolation, and identification [Avram, Zhang & Muse 2017; Zhao, Liu & Lewis 2021].

In the case of a system with multiple actuators, such as a multi-rotor aircraft, it is also worth noting that not all results available in the literature allow the identification of faults in multiple actuators at the same time. Evidently, faults arise the possibility of critical problems in the system if they are poorly diagnosed or if they remain undetected. Therefore, there is a necessity for FDD methods that can correctly estimate the magnitude of a fault with robustness to the natural system disturbances, e.g. the aerodynamic effects on a quadrotor [Sun et al. 2018], [Okada et al. 2021].

For instance, recently [Ma, Xu & Yang 2021] suggested a data-driven reinforcement learning approach for a fault detection scheme working with a fault-tolerant controller. This scheme is capable of detecting, estimating, and locating faults. However, it falls into another problem from the literature as of this moment: the lack of experimental flight results to solidify the method, since the results are tested only with a simulation example of a quadrotor. [Zhang et al. 2021] and [Iannace, Ciaburro & Trematerra 2019] proposed data-driven and neural network-based approaches, respectively, on the diagnosis of blade damage on UAVs. [Zhang et al. 2021] perform time-domain frequency estimation, where the verification of the results is made offline, after the quadcopter flight. [Iannace, Ciaburro & Trematerra 2019] analyze the acoustics of the noise emitted by the UAV to detect unbalanced blades, but the diagnosis is performed off-board and is proposed as part of a pre-flight checklist.

In the case of inaccurate process models, data-driven methods can emerge as an alternative, since they rely on available information on the system's behavior and historical process data [Okada et al. 2021]. Still, a large amount of information would be required depending on the complexity of the system, leading to significant data processing time [Okada et al. 2021] and computation power, which can be a limiting factor in small aircraft for civilian use.

One should also mention [Rudin, Ducard & Siegwart 2020], where the authors propose a robust Fault-Tolerant Control with an alarm FDD system, but the modeling and flight tests are applied to a fixed-wing aircraft. Furthermore, this FDD is not capable of estimating the magnitude of the fault. Another application to a fixed-wing aircraft was presented by [Freeman et al. 2013], where a model-based residual generation is combined with data-driven anomaly detection, however, this work deals with other types of fault pertinent to the type of aircraft they are studying, being outside of the scope of this thesis, which deals with loss-of effectiveness actuator faults in quadrotors.

There is some research being conducted directly on DC motors which are employed on multi-rotor aircraft, as in [Veras et al. 2019] and [Karlina & Indriawati 2020]. [Veras et al. 2019] perform an analysis of sound signals from the motor for fault detection based on density of maxima, and bench test experiments are executed directly on a single DC motor, while [Karlina & Indriawati 2020] use a linear state observer for fault tolerant control of a DC motor and test their approach in simulation models. Alex et al. [Alex, Daniel & Jayanand 2016] applied a reduced order EKF for state estimation of a brushless DC motor, but this paper does not apply the technique to fault detection, nor it considers aerodynamic effects or real flight experiments.

However, research suggests that, for the case of multiplicative faults, as in the scope of this thesis, the use of parameter estimation techniques stands out [Okada et al. 2021]. It is also highlighted in [Okada et al. 2021] that Kalman Filters provide a low false alarm rate, a short delay in detecting the fault, robustness with respect to model uncertainties, isolation of simultaneous faults, and the possibility of fault estimation.

For example, [Zhong et al. 2018] developed an Adaptive Augmented Three-Stage Kalman Filter for the detection of LOE actuator faults in the presence of external disturbances. The technique is applied to the linearized system, and once more the results are tested only in a simulation environment with a quadcopter in hover state. [Zhang et al. 2021], [Avram, Zhang & Muse 2017] and [Lee et al. 2020] use nonlinear adaptive estimators to identify and isolate single actuator LOE faults, and [Baldini et al. 2020] use a custom observer to estimate actuator faults.

As aforementioned, only few previous works in the literature have performed real flight experiments for the application of fault detection and diagnosis in quadrotors, especially with maneuvers or trajectories beyond hover state. [Avram, Zhang & Muse 2017] use nonlinear adaptive estimators to identify and isolate single actuator LOE faults. This approach was indeed applied to a flight test consisting of a circular maneuver, however, the proposed modeling does not allow the detection of faults in multiple actuators concurrently. Furthermore, the consideration of aerodynamic effects is neglected, contrary to the proposal of this thesis. Similar remarks can be made about the work of Lee et al. [Lee et al. 2020], with the addition that here the experiments were executed in hover state on a hexacopter, which has rotor redundancy, contrary to a quadcopter. [Baldini et al. 2020] use a custom observer to estimate actuator faults. The flight experiment involved a square-like trajectory, again only for a single-actuator fault. The fault estimation was also very affected by noise and other disturbances which were not taken into account, such as the airflow around the vehicle, according to the authors.

In [Sun et al. 2021] and [Nan et al. 2022], real-flight experiments on a faulty quadrotor are performed. These works, however, are on a different scope than the one proposed here. The first work [Sun et al. 2021] approaches the problem of onboard vision-based state estimation with the complete loss of one rotor, which yields high-speed yaw rotation. The second [Nan et al. 2022] develops a nonlinear MPC for fault-tolerant control of quadrotors. Both works deal with the complete loss of an actuator, which is on a different scope from the LOE faults here analyzed. Most importantly, none of these contain a fault-detection and diagnosis system. Hence the FDD system to be presented by this thesis can be a way of improving some of the state-of-the-art results found in the literature at the moment.

The work of [Harshavarthini et al. 2020] uses a robust residual nonlinear observer for a fault alarm-based hybrid controller, but without fault estimation and with results tested only on simulation models. [Xu et al. 2018] propose the use of set-theoretic unknown input observers for robust fault detection and isolation that can be applied to detect LOE faults in multiple actuators. Their results are however only tested in simulation, without aerodynamic effects and with a linearized system around hover conditions.

In [Madruga et al. 2022], this research has shown the relevance of the inclusion of aerodynamic effects in the system and, to the best of our knowledge, there has been no development yet of an FDD system that explicitly takes into account blade-flapping aerodynamic effects, especially with real-flight experiments and the capability of isolating faults in more than one actuator. In [Guo, Jiang & Zhang 2018], one sees the development of an extended state observer for the fault-tolerant control of a quadrotor subjected to wind gusts, which are an external perturbation different from blade-flapping aerodynamic effects. Besides that, the fault occurs in only a single actuator, and the results are only demonstrated in simulation. Furthermore, there is no fault isolation (localization) in this system.

Certainly, it is possible to conclude that a major contribution to the current state-of-the-art is to design and validate FDD and fault-tolerant control methods reliable under the disturbance of significant aerodynamic effects for real-time implementations [Sun 2020]. Table 2.1 summarizes the main results found in our literature research for fault detection and diagnosis in the relevant context for this work, highlighting their differences and contributions when compared to this research. Notice that a contribution for quadrotors that ticks all the boxes is being proposed in this thesis.

Reference	**Method**	**Nonlinear**	**Real Flight Experiment**	**Target Platform**	**Aerodynamic Effects**	**Simultaneous Actuator Faults**	**Type of Fault**
[Guo, Jiang & Zhang 2018]	Extended state observer	✓	×	quadrotor	✓	×	LOE
[Rudin, Ducard & Siegwart 2020]	Alarm (no diagnosis) + robust fault tolerant control	×	✓	fixed-wing	×	✓	LOE
[Veras et al. 2019]	Density of maxima + sound	-	×	motor	×	×	-
[Karlina & Indriawati 2020]	Linear state observer	×	×	motor	×	×	-
[Xulin & Yuying 2018]	Linear state observer	×	×	quadrotor	×	×	LOE
[Xu et al. 2018]	Set-theoretic unknown input observer	×	×	quadrotor	×	×	LOE
[Freeman et al. 2013]	Data-driven anomaly detection	✓	✓	fixed-wing	×	✓	-
[Ma, Xu & Yang 2021]	Reinforcement learning	✓	×	quadrotor	×	×	-
[Harshavarthini et al. 2020]	Robust residual linear observer	×	×	quadrotor	×	**?**	-
[Avram, Zhang & Muse 2017]	Nonlinear adaptive estimators	✓	✓	quadrotor	×	×	LOE
[Wang et al. 2020]	Deep learning	✓	✓	fixed-wing	×	✓	-
[Park et al. 2020]	Data-driven + comparison of statistical analysis techniques	×	bancada	quadrotor	×	×	Float
[Zhang et al. 2021]	Time-domain frequency estimator	✓	bancada	blade damage	×	×	LOE (blade damage)
[Moghadam & Caliskan 2015]	Two-stage Kalman Filter	×	×	quadrotor	×	✓	LOE + sensor
[Aishwarya & Jayanand 2016]	Extended Kalman Filter	✓	×	motor	×	×	-
[Schijndel, Sun & Visser 2021]	Sensors + Kalman Filter	×	✓	quadrotor	×	✓	LOE
[Alex, Daniel & Jayanand 2016]	Reduced order Extended Kalman Filter	✓	×	motor	×	-	-
[Zhong et al. 2018]	Adaptive Augmented 3-stage Kalman Filter	×	×	quadrotor	×	✓	LOE
[Wang & Puig 2016]	Zonotopic Extended Kalman Filter (no fault-isolation)	✓	×	quadrotor	×	✓	-
[Madruga et al. 2023]	**ROEKF-TEA (contribution of this research)**	✓	✓	**quadrotor**	✓	✓	**LOE**

Table 2.1: Recollection of the main results found in the literature that are pertinent to this thesis. The characterization of the types of faults mentioned in the last column will be detailed in Chapter 5.

2.5 Final Considerations

In this Chapter, the foundations of the current state-of-the-art have been set, by briefly analyzing some works related to the object of study, helping to build a knowledge base for this research. We sought to establish the relevance of the study of aerodynamic effects, as well as their consideration into quadrotor systems. In here, in the control allocation by means of a neural network and in a fault detection and diagnosis scheme by introducing said effects modeling into the filter's formulation.

Many of the mentioned alternatives deal with the study of modeling techniques for aerodynamic effects, but few still apply them to UAV systems. This thesis differs where it considers these effects in the control allocation without the need of using their equations directly, because of the function fitting neural network, and without the need of modifying the controller algorithm. It also does not demand high computational capacity from the system like traditional control allocation methods.

The next contribution relies on a fault detection and diagnosis system for loss-of-effectiveness actuator faults, capable of taking into account aerodynamic effects and detecting faults in more than one actuator simultaneously. It has been made clear that a major contribution to the current literature is the design and validation of FDD methods reliable under the disturbances, and the encountered methods were summarized in a descriptive Table at the end.

Evidently, it is not possible to completely investigate all the available literature on a specific topic, especially one as vast as quadrotor UAV systems. Therefore, a selection of what is most relevant to the research had to be made in order to move forward.

Chapter 3

Mathematical Modeling for a Quadrotor

This chapter presents the mathematical modeling of a quadrotor UAV, highlighting the main theoretical concepts regarding the quadrotor movement and its nomenclature in Section 3.1. In Section 3.2, we identify the state variables for this system, as well as the matrix which performs the conversion between the inertial and local coordinate frames. With these concepts, Section 3.3 shows the kinematic and dynamic equations for a quadrotor aircraft.

In Section 3.4, we evaluate the main aerodynamic effects which affect the quadrotor and how they affect the aforementioned physical model. This research then contributes mathematically with a modeling of the thrust force rotation around the z-axis in the local coordinate frame in Section 3.4.1. Finally, in Section 3.5, the classic quadrotor model is linearized via Taylor Series, in order to evaluate its behavior around one or more equilibrium points.

3.1 Basic Concepts

The traditional quadrotor model assumes that the aircraft is built as a rigid structure, usually in form of *X*, with four arms, as shown in Fig. 3.1, where the axis convention used in this thesis is presented. Note also that opposite motors rotate in the same sense, which is opposite to the other pair. This balances the system torques and eliminates the need for a back rotor, as in a traditional helicopter [Bresciani 2008].

We use the Tait-Bryan convention for the angles, where the movement of rotation around the x-axis is called *roll* and is represented here by the angle ϕ. The *roll* motion induces a linear displacement of the aircraft to the sides.

The rotation around the y-axis is called *pitch*, represented by the angle θ, and causes a linear displacement in the forward or backward direction.

The rotation around the z-axis is called *yaw*, represented by the angle ψ. The torques that induce each one of these motions are here called τ_ϕ, τ_θ, τ_ψ, respectively.

The force that enables the drone to move upwards is the $thrust\ T$. Ideally, it would only have one vertical component in the z-axis. Aerodynamic effects however rotate the thrust around the z-axis, as will be shown further in this Chapter.

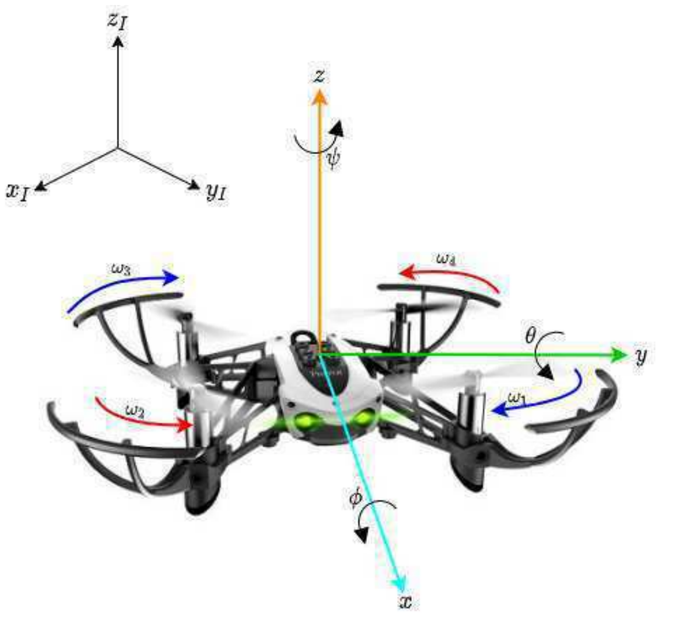

Figure 3.1: In the front, we show the axis configuration in the local frame, as well as the rotation sense for each rotor and the roll (ϕ), pitch (θ), yaw (ψ) notations used in the quadcopter model. In the back, there is the axis configuration in the inertial frame.

Table 3.1 shows the influence of the angular velocities of the actuators in the resulting motion of the aircraft. Here, ω_i, $i = 1, 2, 3, 4$, is the angular velocity of each rotor. An upward arrow means an increase in the velocity of a specific motor, while a downward arrow corresponds to a decrease in velocity. Note that the arrows can be inverted for a change of direction of motion.

Type of motion	ω_1	ω_2	ω_3	ω_4
Thrust (T_z)	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$
Roll (ϕ)	$\downarrow$	$\uparrow$	$\uparrow$	$\downarrow$
Pitch (θ)	$\downarrow$	$\downarrow$	$\uparrow$	$\uparrow$
Yaw (ψ)	$\downarrow$	$\uparrow$	$\downarrow$	$\uparrow$

Table 3.1: Rotor speed influence on quadcopter motion.

3.2 State Variables

The system state variables can be defined as:

- The linear position vector $\mathbf{r}(t) = \begin{bmatrix} x(t) & y(t) & z(t) \end{bmatrix}^T \in \mathbb{R}^3$, in [m];
- The angular position vector $\boldsymbol{\Psi}(t) = \begin{bmatrix} \phi(t) & \theta(t) & \psi(t) \end{bmatrix}^T \in \mathbb{R}^3$, in [rad];
- The linear velocity vector $\mathbf{v}(t) = \begin{bmatrix} v_x(t) & v_y(t) & v_z(t) \end{bmatrix}^T \in \mathbb{R}^3$, in [m/s];
- The angular velocity vector in the body-frame $\boldsymbol{\Omega}(t) = \begin{bmatrix} P(t) & Q(t) & R(t) \end{bmatrix}^T \in \mathbb{R}^3$, in [rad/s].

Note that the linear position $\mathbf{r}$, the angular position $\boldsymbol{\Psi}$, and the velocity of the center of mass $\mathbf{v}$ are represented in the inertial frame, while the vehicle's angular velocity $\boldsymbol{\Omega}$ is represented in the local frame.

The angles which constitute the $\boldsymbol{\Psi}(t)$ vector are those of the Tait-Bryan convention, as aforementioned, which are here notated as *roll*, *pitch*, e *yaw*. The positive orientation of these variables in a generic coordinate frame is indicated in Figure 3.2. Throughout this thesis, a counter-clockwise rotation will be considered positive, while a clockwise rotation will be negative.

For a quadcopter, one needs to deal with the difference in the values of the state variables when taken in the inertial or body-frame, which is part of the localization problem when dealing with mobile robotics. In Figure 3.3, the characteristics of this problem are presented, for simplicity's sake, for the bi-dimensional case, whose properties may be extended for the three-dimensional movement. The inertial and body-frame for the case of the quadrotor used in this thesis are shown in Figure 3.1.

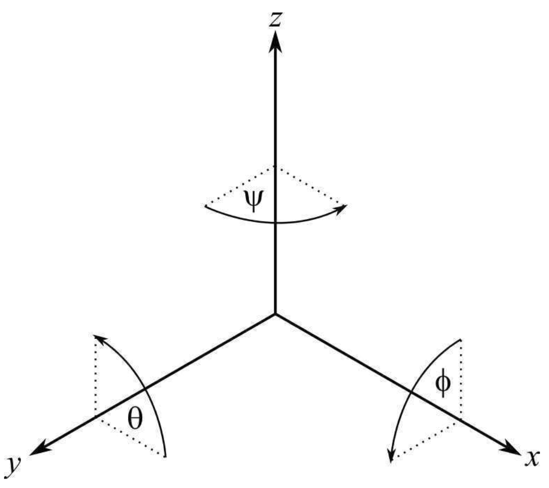

Figure 3.2: Coordinate frame with the positive orientation for ϕ, θ, and ψ.

In Figure 3.3 one can see a body (represented in black) moving through the plane, and its respective coordinates for the beginning and end of the movement, for the inertial-frame (in blue) and body-frame (in red). Generally, one wants to know the position of the object in the inertial frame, hence it is necessary to establish what is called a rotation matrix that is able to convert coordinates from one frame to the other. Such a process is only necessary for position variables and their derivatives, since the resulting displacement will be the same in both frames, even if the coordinates throughout the movement are different, as seen in Figure 3.3.

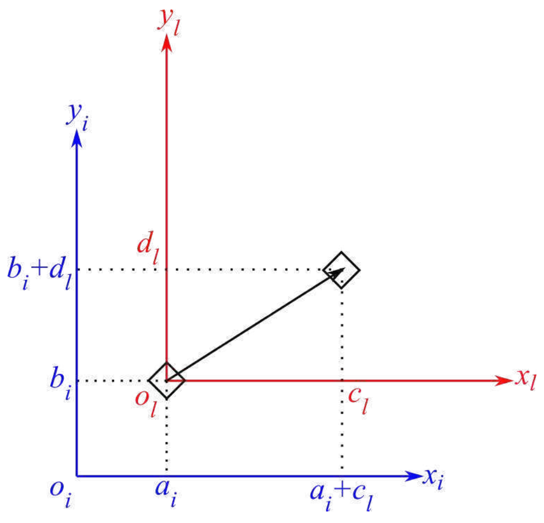

Figure 3.3: Diagram showing the difference between local and inertial coordinates of a body dislocating in the xy-plane.

In this manner, the conversion of vectors from the body frame to the inertial frame can be derived from the rotation matrix $\mathbf{R_{il}} \in \mathbb{R}^{3\times3}$, shown in (3.3).

The matrix $\mathbf{R_{il}}$ is obtained by means of $\mathbf{z}$, $\mathbf{y}$ e $\mathbf{x}$ rotations, respectively, as shown in (3.1). Notice that the inverse matrix of a rotation matrix equals its transpose, that is $\mathbf{R_{il}}^{-1} = \mathbf{R_{il}}^{T}$.

$$\mathbf{R_{il}} = \mathbf{R_z}\ \mathbf{R_y}\ \mathbf{R_x} \tag{3.1}$$

$$\mathbf{R_{il}} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{3.2}$$

$$\mathbf{R_{il}} = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{3.3}$$

It is also necessary to use a coupling matrix, which expresses the relation between the angular velocity vectors in both coordinate frames. This matrix is shown in (3.4).

$$\mathbf{S}(\mathbf{\Psi}) = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \tag{3.4}$$

3.3 Kinematic and Dynamic Equations

A quadrotor's physical behavior can be described by the following equations [Sabatino 2015]:

$$\dot{\mathbf{r}} = \mathbf{v} \tag{3.5}$$

$$\dot{\mathbf{v}} = \frac{1}{m}\mathbf{R_{il}}\mathbf{T}^{B} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \frac{1}{m}\mathbf{R_{il}}\mathbf{F_d}, \tag{3.6}$$

$$\dot{\mathbf{\Psi}} = \mathbf{S}(\mathbf{\Psi})\mathbf{\Omega} \tag{3.7}$$

$$\dot{\mathbf{\Omega}} = \mathbf{J}^{-1}(-\mathbf{\Omega} \times \mathbf{J}\mathbf{\Omega} + \mathrm{M}) \tag{3.8}$$

- The gravitational acceleration g, in [m/s^2].
- The total mass of the vehicle m, in [kg].
- The total thrust force $\mathbf{T}^B$ acting over the center of mass, in [N].
- The matrix of moments of inertia $\mathbf{J} = \begin{bmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{bmatrix}$, each in [kg.m^2].
- The vector of torques in the quadrotor body $\mathrm{M} = \begin{bmatrix} \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^T$, in [N.m].
- The disturbance caused by drag forces, in [N], detailed in (3.9), where α_T is a constant called thrust coefficient.

$$\mathbf{F_d} = \alpha_T \left(\omega_1 + \omega_2 + \omega_3 + \omega_4\right) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.9}$$

Once more we reiterate that the linear position vector $\mathbf{r}$ and the linear velocity of the center of mass $\mathbf{v}$ are given in the inertial frame, while the angular velocity vector $\mathbf{\Omega}$ is given in the body frame.

In the quadrotor model, the vector of torques M and total thrust T are the system inputs, through which the angular and linear velocities are modified, allowing thus the attitude $\mathbf{\Psi}$ control and subsequently the position $\mathbf{r}$ control. However, in most quadrotor control architectures, the control commands are given as rotational velocity commands to each actuator (i.e. the motor). This means it is necessary to establish a conversion between these two sets of values.

This transformation is performed in a step called Control Allocation and, in the classic quadrotor model, it is given by:

$$\begin{bmatrix} T \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \mathbf{B_m} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}, \tag{3.10}$$

where $\mathbf{B_m}$ is an invertible matrix, called mixing matrix, allocation matrix, or control effectiveness matrix, and will be detailed further in Section 3.4.

The actuator velocities ω_i would then be determined by:

$$\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} = \mathbf{B_m}^{-1} \begin{bmatrix} T \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix}, \tag{3.11}$$

Still, the expressions shown in (3.10) and (3.11) come from a simplification of the quadrotor model which neglects important aerodynamic effects that influence flight behavior. As will be shown further, disregarding them hinders flight performance and may cause actuator saturation. In the next Section, the main aerodynamic effects that affect the relation between mechanical conjugates (torques $\mathbf{M}$ and thrust T) and actuator velocities will be addressed.

3.4 Aerodynamic Effects

There are several aerodynamic and gyroscopic effects that can modify the force model introduced in equations (3.5) to (3.8), the most important of them being the ones called *blade flapping* and *induced drag*, which are of significant importance in understanding the natural stability of quadrotors. These effects are particularly relevant because they induce forces in the x-y rotor plane of the quadrotor, precisely the under-actuated directions in the dynamics that cannot be easily dominated by high gain control [Mahony, Kumar & Corke 2012].

While disturbances caused by drag forces are already included in the basic model in (3.6) as an additive term, the inclusion of blade flapping equations is not so straightforward. This phenomenon is caused by the flapping effect generated in the blades of the propellers by the horizontal translation of the rotor in the air. During the translation of the rotor, one blade (called the *advancing blade*) will move in the direction of translation, while the other (the *retreating blade*) will move opposite to the direction of motion. The net velocity of the advancing blade will increase, and that of the retreating blade will decrease with respect to the air. This creates a disparity of lift between the advancing side of the rotor and the retreating side of the rotor (see Fig. 3.4) [Pounds, Mahony & Corke 2010].

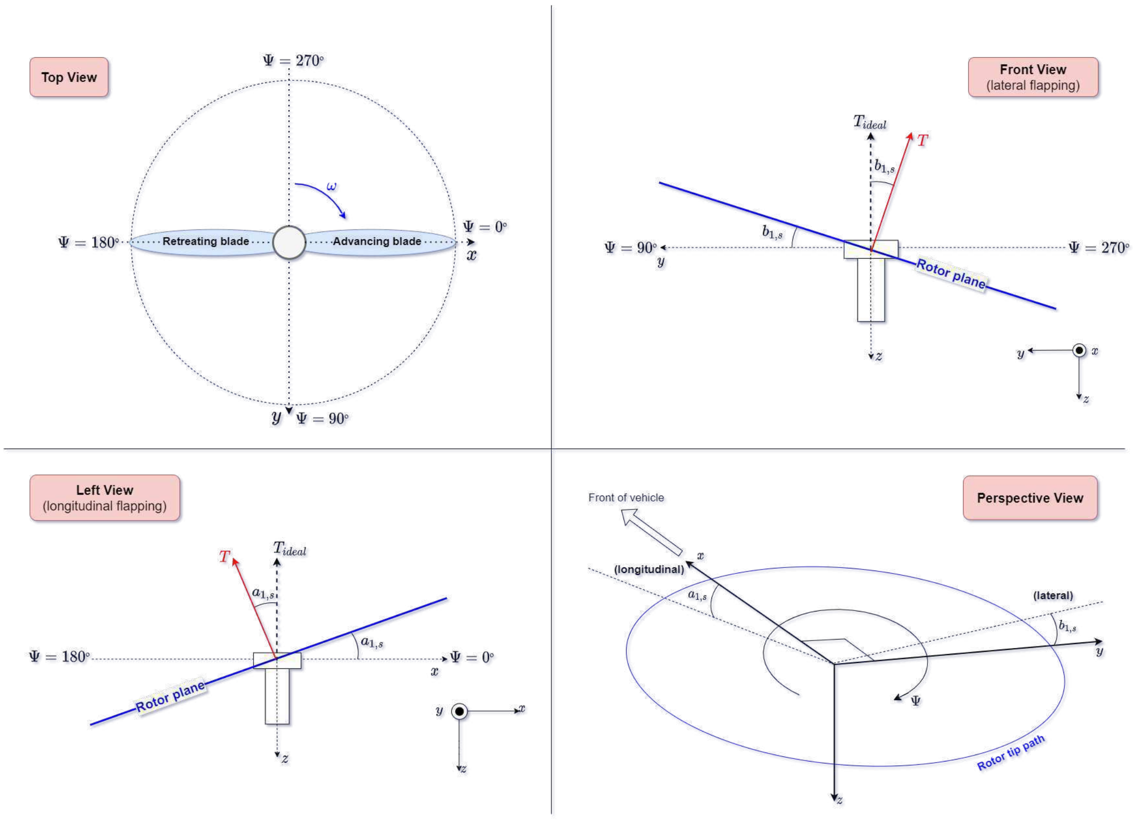

Figure 3.4: Perspective views of a rotor during flight. Longitudinal and lateral blade flapping angles are $a_{1,s}$ and $b_{1,s}$, respectively, and Ψ here is the blade rotation azimuth angle.

According to [Pounds, Mahony & Corke 2010], a rotor blade balances the lift imbalance (aerodynamic drag force), centripetal force, and blade weight force by allowing free rotation around a hinge or flexure, and this is known as rotor or blade flapping. The advancing blade will rise, the retreating blade will fall, as in Fig. 3.4, and therefore the rotor plane will tilt in an effort to meet a new equilibrium. Increasing translational velocity leads to increasing rotor tilt. When the rotor plane tilts, the thrust force generated by the blades is correspondingly tilted, resulting in a horizontal component that opposes the direction of translation.

Allowing the rotor blade to bend is very important in the mechanical design of the quadcopter. A rotor that is too rigid can cause transmission of these aerodynamic forces directly through to the rotor hub and may result in a mechanical failure of the motor mounting or the airframe itself [Mahony, Kumar & Corke 2012]. [Leishman 2006] also states that if the rotor disc is perfectly rigid, the helicopter will flip, as was the case in early experiments.

As aforementioned, rotor flapping causes the rotor plane to tilt and hence the thrust force will also be tilted and will have a horizontal component. The total thrust $\mathbf{T^B}$ that appears in (3.6) results from the sum of the thrusts $\mathbf{T^{B,i}}$ produced by each of the four propellers $i = 1, 2, 3, 4$ [Riether 2016]:

$$\mathbf{T^B} = \sum_{i=1}^{4} \mathbf{T^{B,i}}, \tag{3.12}$$

$$\mathbf{T^{B,i}} = T^i \begin{bmatrix} -\cos(b_{1,s,i})\sin(a_{1,s,i}) \\ \sin(b_{1,s,i}) \\ -\cos(a_{1,s,i})\cos(b_{1,s,i}) \end{bmatrix}^B, \tag{3.13}$$

with $T_i = \alpha_T \omega_i^2$, where α_T is called the thrust coefficient, $b_{1,s,i}$ and $a_{1,s,i}$ are shown in Figure 3.4 and detailed in Table 3.2, as described by [Riether 2016].

In this Table, $\mathbf{r}^B_{prop,i}$ refers to the propeller i's position with respect to the center of mass, θ_{b0} is the equivalent blade pitch at the rotor axis, and θ_{b1} is the washout of a linear twist blade [Pounds, Mahony & Corke 2010].

The constant $\gamma_{aero} = \frac{\rho a_0 c R}{I_b}$ is the non-dimensional Lock Number [Hoffmann et al. 2007], [Leishman 2006], which can be viewed as a **measure of the ratio of aerodynamic forces to inertial forces**. Here, I_b is the moment of inertia of the blade about the hinge, c is the chord of the blade, $R = \frac{d}{2}$ is the rotor radius, ρ is the density of air, and a_0 represents the slope of the lift curve per radian [Hoffmann et al. 2007], [Pounds, Mahony & Corke 2010]. For a typical helicopter rotor, the value of the Lock Number varies from 5 to 10 [Leishman 2006]. In the case of our mini-drone, it is 0.6051, as detailed in Table 3.3. The values

Relative air speed at propeller i: $\mathbf{v}^B_{ra,i}$	$\Omega \times \mathbf{r}^B_{prop,i} + \mathbf{R}^{-1}_{li}\mathbf{v}$
Planar components: μ_i	$\sqrt{{v^B_{x,ra,i}}^2 + {v^B_{y,ra,i}}^2} / \left\|\omega_i \frac{d}{2}\right\|$
Non-dimensionalized normal inflow: l_i	$\left\|v^B_{z,ra,i}\right\| / \left\|\omega_i \frac{d}{2}\right\|$
Sideslip azimuth relative to x-axis x: j_i	$\arctan \frac{v^B_{x,ra,i}}{v^B_{y,ra,i}}$
Sideslip rotation matrix: $\mathbf{J}_{ss,i}$	$\begin{bmatrix} \cos(j_i) & -\sin(j_i) \\ \sin(j_i) & \cos(j_i) \end{bmatrix}$
Longitudinal flapping: $\beta_{lf,i}$	$\mathbf{J}^T_{ss,i} \begin{bmatrix} ((\frac{8}{3}\theta_{b0} + 2\theta_{b1}) - 2l_i)/(1/\mu_i - \mu_i/2) \\ 0 \end{bmatrix}$
$a_{1,s,i}$	$\beta_{x,lf,i} - 16Q/(\gamma_{aero}\|\omega_i\|)$
$b_{1,s,i}$	$\beta_{y,lf,i} - 16P/(\gamma_{aero}\|\omega_i\|)$

Table 3.2: Aerodynamic effects on total thrust $\mathbf{T}^B_i$.

shown in Table 3.3 correspond to the PARROT Mambo mini-drone and were provided by the MATLAB Support Package [Inc. 2021].

Propeller i's positions: $\mathbf{r}^B_{prop,i}$	$\begin{bmatrix} \pm 0.0441 & \pm 0.0441 & -0.0159 \end{bmatrix}^T$
Rotor radius: $d/2$	0.0330
Lock number: γ_{aero}	0.6051
Blade root angle: θ_{b0}	0.2548
Blade twist angle: θ_{b1}	-0.1361

Table 3.3: Constants shown in Table 3.2.

Finally, the vector of torques $\boldsymbol{\tau} = \begin{bmatrix} \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^T$ acting on the body-frame of the drone can be calculated as:

$$\tau = \sum_{i=1}^{4} \left(\mathbf{r}^B_{prop,i} \times \mathbf{T}^{\mathbf{B,i}} + Q_i \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^B \right), \tag{3.14}$$

where $Q_i = \alpha_Q \omega_i |\omega_i|$ is the propeller-drag-induced torque [Riether 2016], and α_Q is called the torque coefficient. Note that, in (3.14), $\mathbf{T}_i^B$ is as in (3.13) and is dependent on aerodynamic effects, since all the terms multiplied by the Lock Number γ_{aero} come from aerodynamics [Leishman 2006], as is the case with $b_{1,s,i}$ and $a_{1,s,i}$. Hence, the vector of torques $\boldsymbol{\tau}$ will likewise be influenced by the aerodynamic effects. Section 3.4.1 details the thrust rotation caused by aerodynamic effects thoroughly, which also provides an alternative manner to visualize the thrust equation.

These aerodynamic effects are neglected in most models, assuming $a_{1,s,i} = b_{1,s,i} = 0$ and ignoring the ratio of aerodynamic forces to inertial forces. Therefore, in those cases, the thrust vectors are supposed to be fully aligned with the body-frame z-axis and do not have horizontal components caused by blade flapping. The angles $a_{1,s,i}$ and $b_{1,s,i}$ cannot be zero in realistic conditions, as explained above, after all the rotor bend is essential to the quadrotor design. Since increasing translational velocities increase rotor tilt, aggressive maneuvering is specifically subject to the consequences of ignoring rotor flapping, as well as small aircraft [Achermann et al. 2019], [Huang et al. 2009].

One of the contributions of this thesis is the compensation of the described aerodynamic effects using control allocation, without the need to include them in the control algorithm itself in this work, as will be shown in Chapter 4.

As addressed in Chapter 2, most works currently available in literature neglect these effects, especially in the control allocation itself. This is the case because, if one makes this approximation, an analytical simplification of the mathematical expressions involved will result in an invertible matricial relation, which is very convenient in terms of system modeling. However, this negligence leaves the quadrotor subject to saturation, and the torque and thrust commands generated by any controller will not be correctly achieved by the actuators.

Most specifically, overlooking these effects results in $\mathbf{T^{B,i}} \approx \begin{bmatrix} 0 & 0 & T_i \end{bmatrix}^T$ and

$$\mathbf{T}^B \approx \begin{bmatrix} 0 & 0 & T \end{bmatrix}^T, \tag{3.15}$$

where $T \approx \sum T_i$. With these assumptions, the plant input is then simplified from six dimensions $\begin{bmatrix} \mathbf{T}_c^B & \tau_{\phi_c} & \tau_{\theta_c} & \tau_{\psi_c} \end{bmatrix}^T$ to four dimensions $\begin{bmatrix} T_c & \tau_{\phi_c} & \tau_{\theta_c} & \tau_{\psi_c} \end{bmatrix}^T$. Hence the above mentioned equations (3.13) and (3.14) can be simplified as:

$$\begin{bmatrix} T \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \approx \alpha_T \begin{bmatrix} 1 & 1 & 1 & 1 \\ r_{y,1} & r_{y,2} & r_{y,3} & r_{y,4} \\ r_{x,1} & r_{x,2} & r_{x,3} & r_{x,4} \\ \frac{\alpha_Q}{\alpha_T} & -\frac{\alpha_Q}{\alpha_T} & \frac{\alpha_Q}{\alpha_T} & -\frac{\alpha_Q}{\alpha_T} \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}. \tag{3.16}$$

Note that (3.16) shows the aforementioned approximate matricial relation between the motor's velocities squared and the plant input $\mathbf{u} = \begin{bmatrix} T & \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^T$, given that the matrix is non-singular and that the velocities ω_i do not saturate (which we cannot assume in this case [Alves 2019]).

Definition 1 (Mixing Matrix Control Allocation (MMCA))**.** *The control allocation performed by means of a matrix that provides an invertible relation between squared motor speeds and the plant input* $\begin{bmatrix} T_c & \tau_{\phi_c} & \tau_{\theta_c} & \tau_{\psi_c} \end{bmatrix}^T$*, as in (3.16), provided that the matrix is non-singular and the angular velocities* ω_i *are not saturated.*

If we write (3.16) as

$$\begin{bmatrix} T_c \\ \tau_{\phi_c} \\ \tau_{\theta_c} \\ \tau_{\psi_c} \end{bmatrix} = \mathbf{B_m} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}, \tag{3.17}$$

the angular velocities would then be determined as

$$\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} = \mathbf{B_m}^{-1} \begin{bmatrix} T_c \\ \tau_{\phi_c} \\ \tau_{\theta_c} \\ \tau_{\psi_c} \end{bmatrix}, \tag{3.18}$$

where $\mathbf{B_m}$ *is the aforementioned control effectiveness matrix, mixing matrix, or allocation matrix.*

In (3.14), the analytical inversion to obtain the actuator's velocities (ω_i) from the commanded thrust and torques (T and $\tau_{\phi,\theta,\psi}$) would be much more complex than in (3.18). That is the reason why this simplification disregarding aerodynamic effects is prevailing in control models found in the state-of-the-art literature.

Additionally, expanding (3.14), we obtain the following:

$$\begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \sum_{i=1}^{4} \left(\begin{bmatrix} T_{z_i} r_{y_i} - T_{y_i} r_{z_i} \\ T_{x_i} r_{z_i} - T_{z_i} r_{x_i} \\ T_{y_i} r_{x_i} - T_{x_i} r_{y_i} \end{bmatrix} + Q_i \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right)^B, \tag{3.19}$$

which gives us, for each actuator i:

$$\tau_\phi = \sum_{i=1}^{4} (T_{z_i} r_{y_i} - T_{y_i} r_{z_i}), \tag{3.20}$$

$$\tau_\theta = \sum_{i=1}^{4} (T_{x_i} r_{z_i} - T_{z_i} r_{x_i}), \tag{3.21}$$

$$\tau_\psi = \sum_{i=1}^{4} (T_{y_i} r_{x_i} - T_{x_i} r_{y_i} + Q_i). \tag{3.22}$$

In (3.20), (3.21) and (3.22), we make explicit the influence of the thrust components on x and y-axes on the torques that perform the roll, pitch and yaw movements, τ_ϕ, τ_θ and τ_ψ, respectively. These components are neglected when using the traditional Mixing Matrix Control Allocation, which considers that the thrust only has the z-axis component. We state once more that, due to aerodynamic effects such as blade flapping, however, thrust and moments are not necessarily orthogonal to the body axis [Hoffmann et al. 2007], so these x-axis and y-axis components do exist. Moreover, in Chapter 4, we show through real robot experiments that these components T_x and T_y are not of negligible order when compared to the order of the system torques they have an influence on.

3.4.1 Thrust Rotation

In this Section, we present a vectorial analysis of the thrust vector rotation when subjected to *blade flapping*, this being one of the contributions of this thesis. In addition, this approach also provides an alternative way of visualizing the thrust equation with the aerodynamic effects as seen in (3.13).

Assumption 1. *The thrust $\mathbf{T}_i^B$ produced by each of the four propellers $i = 1, 2, 3, 4$ is equal to the vector $\alpha_T \omega_i^2 \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T$ rotated around the Y and X axes, respectively, by the longitudinal and lateral flapping angles $a_{1,s,i}$ and $b_{1,s,i}$, as shown below.*

$$\mathbf{T}_i^B = \mathbf{R_y}(a_{1,s,i})\mathbf{R_x}(b_{1,s,i})\alpha_T \omega_i^2 \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \tag{3.23}$$

Where

$$\mathbf{R_y}(a_{1,s})\mathbf{R_x}(b_{1,s}) = \begin{bmatrix} \cos(a_{1,s}) & 0 & \sin(a_{1,s}) \\ 0 & 1 & 0 \\ -\sin(a_{1,s}) & 0 & \cos(a_{1,s}) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(b_{1,s}) & -\sin(b_{1,s}) \\ 0 & \sin(b_{1,s}) & \cos(b_{1,s}) \end{bmatrix}, \tag{3.24}$$

$$\mathbf{R_y}(a_{1,s})\mathbf{R_x}(b_{1,s}) = \begin{bmatrix} \cos(a_{1,s}) & \sin(a_{1,s})\sin(b_{1,s}) & \sin(a_{1,s})\cos(b_{1,s}) \\ 0 & \cos(b_{1,s}) & -\sin(b_{1,s}) \\ -\sin(a_{1,s}) & \cos(a_{1,s})\sin(b_{1,s}) & \cos(a_{1,s})\cos(b_{1,s}) \end{bmatrix}, \tag{3.25}$$

$$\mathbf{R_y}(a_{1,s})\mathbf{R_x}(b_{1,s}) \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} \sin(a_{1,s})\cos(b_{1,s}) \\ \sin(b_{1,s}) \\ -\cos(a_{1,s})\cos(b_{1,s}) \end{bmatrix}. \tag{3.26}$$

Note that, with this development, we can actually rewrite (3.13), shown in [Riether 2016], using (3.26), as a function of the rotation of the flapping angles, as in (3.23) shown in **Proposition 1**.

Lemma 1. *The longitudinal and lateral flapping angles $a_{1,s,i}$ and $b_{1,s,i}$ are*

$$a_{1,s,i} = \frac{\mathbf{e}_x \mathbf{v}_{ra,i}}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|} \left(\frac{|\omega_i r|\Theta - 2|\mathbf{P}_z \mathbf{v}_{ra,i}|}{\frac{|\omega_i r|^2}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|} - \frac{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|}{2}} \right) - \frac{16q}{\gamma_{aero}|\omega_i|} \tag{3.27}$$

and

$$b_{1,s,i} = -\frac{\mathbf{e}_y \mathbf{v}_{ra,i}}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|}\left(\frac{|\omega_i r|\Theta - 2|\mathbf{P}_z \mathbf{v}_{ra,i}|}{\frac{|\omega_i r|^2}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|} - \frac{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|}{2}}\right) - \frac{16p}{\gamma_{aero}|\omega_i|}, \tag{3.28}$$

where $\mathbf{e}_z = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$, $\mathbf{e}_y = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$ *and* $\mathbf{e}_x = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \in \mathbb{R}^3$; $\mathbf{P}_{xy}$ *and* $\mathbf{P}_z$ *represent the projections on the* xy*-plane and* z*-axis respectively; and* $\Theta = \frac{8}{3}\theta_{b0} + 2\theta_{b1}$.

Proof. Now, analyzing the other terms on Table 3.2, let us start with the sideslip rotation matrix $\mathbf{J}_{ss,i}$. It can be interpreted as the rotation of j_i, defined in Table 3.2 and in Fig. 3.5, along the z axis, as in (3.29) below:

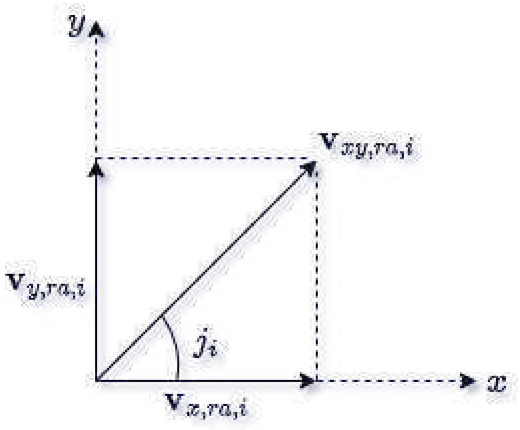

Figure 3.5: $\mathbf{v}_{ra,i}$ projection on the xy-plane.

$$\mathbf{J}_{ss,i} = \mathbf{R}_z(j_i), \tag{3.29}$$

$$\mathbf{J}^T_{ss,i} = \mathbf{R}^T_z(j_i) = \begin{bmatrix} \cos(j_i) & \sin(j_i) & 0 \\ -\sin(j_i) & \cos(j_i) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.30}$$

$$\mathbf{J}^T_{ss,i} = \mathbf{R}_z(-j_i) = \begin{bmatrix} \cos(-j_i) & -\sin(-j_i) & 0 \\ \sin(-j_i) & \cos(-j_i) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.31}$$

It is important to remember that (3.30) and (3.31) are equivalent because the inverse of a rotation matrix is equal to its transpose. Therefore, we can rewrite the longitudinal flapping $\beta_{lf,i}$, also on Table 3.2, as:

$$\beta_{lf,i} = \mathbf{R}_z(-j_i) \begin{bmatrix} \frac{(\frac{8}{3}\theta_{b0}+2\theta_{b1})-2l_i}{1/\mu_i-\mu_i/2} \\ 0 \\ 0 \end{bmatrix}. \tag{3.32}$$

Now for the planar components μ_i and the non-dimensionalized normal inflow l_i, from Table 3.2, we know that

$$\mu_i = \frac{\sqrt{{\mathbf{v}^B_{x,ra,i}}^2 + {\mathbf{v}^B_{y,ra,i}}^2}}{\left|\omega_i \frac{d}{2}\right|}, \tag{3.33}$$

therefore,

$$\mu_i = \frac{|\mathbf{v}^B_{xy,ra,i}|}{\left|\omega_i \frac{d}{2}\right|}. \tag{3.34}$$

Also,

$$l_i = \frac{|\mathbf{v}^B_{z,ra,i}|}{\left|\omega_i \frac{d}{2}\right|}. \tag{3.35}$$

Hence, if $\mathbf{P}_{xy}$ and $\mathbf{P}_z$ represent the projections on the xy-plane and z-axis respectively, we can write

$$\mathbf{v}_{ra,i} = \mathbf{v}_{xy,ra,i} + \mathbf{v}_{z,ra,i} = \mathbf{P}_{xy}\mathbf{v}_{ra,i} + \mathbf{P}_z\mathbf{v}_{ra,i}, \tag{3.36}$$

and, from (3.34) and (3.35),

$$|\mathbf{v}_{ra,i}| = \sqrt{|\mathbf{v}_{xy,ra,i}|^2 + |\mathbf{v}_{z,ra,i}|^2}, \tag{3.37}$$

$$|\mathbf{v}_{ra,i}| = \sqrt{\mu_i|\omega_i r||\mathbf{v}_{xy,ra,i}| + l_i|\omega_i r||\mathbf{v}_{z,ra,i}|}, \tag{3.38}$$

where $r = \frac{d}{2}$.

Now, taking $\cos(j_i))$ and $\sin(j_i))$ as in Fig. 3.5:

$$\cos(j_i) = \frac{\mathbf{v}_{xy,ra,i} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}}{|\mathbf{v}_{xy,ra,i}| \left\| \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \right\|} = \frac{\mathbf{v}_{x,ra,i}}{|\mathbf{v}_{xy,ra,i}|}, \tag{3.39}$$

$$\sin(j_i) = \frac{\mathbf{v}_{y,ra,i}}{|\mathbf{v}_{xy,ra,i}|}, \tag{3.40}$$

we can rewrite the rotation matrix $R_z(-j_i)$, shown in (3.31), as

$$\mathbf{R}_z(-j_i) = \frac{1}{|\mathbf{v}_{xy,ra,i}|} \begin{bmatrix} \mathbf{v}_{x,ra,i} & \mathbf{v}_{y,ra,i} & 0 \\ -\mathbf{v}_{y,ra,i} & \mathbf{v}_{x,ra,i} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.41}$$

If we consider $\mathbf{e}_z = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$, $\mathbf{e}_y = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$ and $\mathbf{e}_x = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \in \mathbb{R}^3$, we have:

$$\mathbf{R}_z(-j_i) = \frac{1}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|} \begin{bmatrix} \mathbf{e}_x \cdot \mathbf{v}_{ra,i} & \mathbf{e}_y \cdot \mathbf{v}_{ra,i} & 0 \\ -\mathbf{e}_y \cdot \mathbf{v}_{ra,i} & \mathbf{e}_x \cdot \mathbf{v}_{ra,i} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.42}$$

Using $\mathbf{P}_{xy}$, $\mathbf{P}_z$ and (3.34) and (3.35), we have

$$\mu_i = \frac{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|}{\omega_i r}, \tag{3.43}$$

$$l_i = \frac{|\mathbf{P}_z\mathbf{v}_{ra,i}|}{|\omega_i r|}, \tag{3.44}$$

and $a_{1,s,i}$ and $b_{1,s,i}$ can be expressed as:

$$\begin{bmatrix} a_{1,s,i} \\ b_{1,s,i} \\ 0 \end{bmatrix} = \beta_{lf,i} - \begin{bmatrix} 16q/\gamma_{aero}|\omega_i| \\ 16p/\gamma_{aero}|\omega_i| \\ 0 \end{bmatrix}, \tag{3.45}$$

$$\begin{aligned} \begin{bmatrix} a_{1,s,i} \\ b_{1,s,i} \\ 0 \end{bmatrix} = \frac{1}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|} \begin{bmatrix} \mathbf{e}_x \cdot \mathbf{v}_{ra,i} & \mathbf{e}_y \cdot \mathbf{v}_{ra,i} & 0 \\ -\mathbf{e}_y \cdot \mathbf{v}_{ra,i} & \mathbf{e}_x \cdot \mathbf{v}_{ra,i} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \\ \begin{bmatrix} \frac{(\frac{8}{3}\theta_{b0}+2\theta_{b1})-2l_i}{1/\mu_i-\mu_i/2} \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 16q/\gamma_{aero}|\omega_i| \\ 16p/\gamma_{aero}|\omega_i| \\ 0 \end{bmatrix} \end{aligned} \tag{3.46}$$

□

Corollary 1. *Finally, from **Assumption 1** and **Lemma 1**, it is possible to write an expression for $\mathbf{T}_i^B$ as:*

$$\begin{aligned} \mathbf{T}_i^B = R_y & \left(\frac{\mathbf{e}_x \mathbf{v}_{ra,i}}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|} \left(\frac{|\omega_i r|\Theta - 2|\mathbf{P}_z \mathbf{v}_{ra,i}|}{\frac{|\omega_i r|^2}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|} - \frac{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|}{2}} \right) - \frac{16q}{\gamma_{aero}|\omega_i|} \right) \cdot \\ R_x & \left(-\frac{\mathbf{e}_y \mathbf{v}_{ra,i}}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|} \left(\frac{|\omega_i r|\Theta - 2|\mathbf{P}_z \mathbf{v}_{ra,i}|}{\frac{|\omega_i r|^2}{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|} - \frac{|\mathbf{P}_{xy}\mathbf{v}_{ra,i}|}{2}} \right) - \frac{16p}{\gamma_{aero}|\omega_i|} \right) \cdot \\ & \alpha_T \omega_i^2 \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}, \end{aligned} \tag{3.47}$$

where $\Theta = \frac{8}{3}\theta_{b0} + 2\theta_{b1}$.

Therefore, the thrust $\mathbf{T}_i^B$ produced by the propeller i is equal to the thrust $T_i = \alpha_T \omega_i^2 \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T$ rotated around the y and x axes, because of the longitudinal and lateral flapping angles. Such rotations not only depend on the rotational and translational states $\mathbf{\Omega}$ and $\mathbf{v}$ but also depend on the angular velocities of the propellers ω_i. Thus, both the intensity and the rotation of the thrust $\mathbf{T}_i^B$ are functions of ω_i.

3.5 Classic Model Linearization and Analysis

In this Section, we will continue our analysis of the presented quadrotor model, expanding all the state equations and then performing a Taylor Series linearization in order to evaluate the behavior around one or more equilibrium points, as well as observability and controllability.

Recalling the mathematical model equations presented in Section 3.3, this time considering the three components of the thrust force caused by aerodynamic effects as in (3.13):

$$\dot{\mathbf{r}} = \mathbf{v} \tag{3.48}$$

$$\dot{\mathbf{v}} = \frac{1}{m}\mathbf{R_{il}}\begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \frac{1}{m}\mathbf{R_{il}}\mathbf{F_d} \tag{3.49}$$

$$\dot{\mathbf{\Psi}} = \mathbf{S}(\mathbf{\Psi})\mathbf{\Omega} \tag{3.50}$$

$$\dot{\mathbf{\Omega}} = \mathbf{J}^{-1}(-\mathbf{\Omega} \times \mathbf{J}\mathbf{\Omega} + \mathbf{M}) \tag{3.51}$$

Expanding the equations (3.48) to (3.51), we obtain (3.52) to (3.55), respectively. Note that some state variables are converted from inertial to local frame and vice-versa, in the following equations.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \mathbf{R_{il}} \begin{bmatrix} U \\ V \\ W \end{bmatrix} \tag{3.52}$$

$$\begin{bmatrix} \dot{U} \\ \dot{V} \\ \dot{W} \end{bmatrix} = \begin{bmatrix} U \\ V \\ W \end{bmatrix} \times \begin{bmatrix} P \\ Q \\ R \end{bmatrix} + \frac{1}{m}\begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \mathbf{R_{il}^T}\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \frac{1}{m}\begin{bmatrix} F_{dx} \\ F_{dy} \\ F_{dz} \end{bmatrix} \tag{3.53}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \mathbf{S}(\mathbf{\Psi}) \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \tag{3.54}$$

$$\begin{bmatrix}\dot{P}\\ \dot{Q}\\ \dot{R}\end{bmatrix} = \mathbf{J}^{-1}\left(-\begin{bmatrix}P\\ Q\\ R\end{bmatrix} \times \mathbf{J}\begin{bmatrix}P\\ Q\\ R\end{bmatrix} + \begin{bmatrix}\tau_\phi\\ \tau_\theta\\ \tau_\psi\end{bmatrix}\right) \tag{3.55}$$

The state variables vector $\mathbf{x} \in \mathbb{R}^{12}$ is then defined as:

$$\mathbf{x} = \begin{bmatrix}\phi & \theta & \psi & P & Q & R & U & V & W & x & y & z\end{bmatrix}^T \tag{3.56}$$

The equations above can be manipulated to represent the quadrotor dynamics in the state-space form, so one can obtain $\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$, $\dot{P}$, $\dot{Q}$, $\dot{R}$, $\dot{U}$, $\dot{V}$, $\dot{W}$, $\dot{x}$, $\dot{y}$, $\dot{z}$.

From (3.52), we have:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix} = \begin{bmatrix}\mathbf{c}\psi\mathbf{c}\theta & \mathbf{c}\psi\mathbf{s}\theta\mathbf{s}\phi - \mathbf{s}\psi\mathbf{c}\phi & \mathbf{c}\psi\mathbf{s}\theta\mathbf{c}\phi + \mathbf{s}\psi\mathbf{s}\phi\\ \mathbf{s}\psi\mathbf{c}\theta & \mathbf{s}\psi\mathbf{s}\theta\mathbf{s}\phi + \mathbf{c}\psi\mathbf{c}\phi & \mathbf{s}\psi\mathbf{s}\theta\mathbf{c}\phi - \mathbf{c}\psi\mathbf{s}\phi\\ -\mathbf{s}\theta & \mathbf{c}\theta\mathbf{s}\phi & \mathbf{c}\theta\mathbf{c}\phi\end{bmatrix}\begin{bmatrix}U\\ V\\ W\end{bmatrix} \tag{3.57}$$

$$\begin{cases}\dot{x} = U(\cos\psi\cos\theta) + V(\cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi) + W(\cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi)\\ \dot{y} = U(\sin\psi\cos\theta) + V(\sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi) + W(\sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi)\\ \dot{z} = U(-\sin\theta) + V(\cos\theta\sin\phi) + W(\cos\theta\cos\phi)\end{cases} \tag{3.58}$$

Likewise, from (3.53):

$$\begin{bmatrix}\dot{U}\\ \dot{V}\\ \dot{W}\end{bmatrix} = \begin{bmatrix}U\\ V\\ W\end{bmatrix} \times \begin{bmatrix}P\\ Q\\ R\end{bmatrix} + \begin{bmatrix}T_x/m\\ T_y/m\\ T_z/m\end{bmatrix} + \begin{bmatrix}\mathbf{c}\psi\mathbf{c}\theta & \mathbf{s}\psi\mathbf{c}\theta & -\mathbf{s}\theta\\ \mathbf{c}\psi\mathbf{s}\theta\mathbf{s}\phi - \mathbf{s}\psi\mathbf{c}\phi & \mathbf{s}\psi\mathbf{s}\theta\mathbf{s}\phi + \mathbf{c}\psi\mathbf{c}\phi & \mathbf{c}\theta\mathbf{s}\phi\\ \mathbf{c}\psi\mathbf{s}\theta\mathbf{c}\phi + \mathbf{s}\psi\mathbf{s}\phi & \mathbf{s}\psi\mathbf{s}\theta\mathbf{c}\phi - \mathbf{c}\psi\mathbf{s}\phi & \mathbf{c}\theta\mathbf{c}\phi\end{bmatrix}\begin{bmatrix}0\\ 0\\ g\end{bmatrix} + \begin{bmatrix}F_{dx}/m\\ F_{dy}/m\\ F_{dz}/m\end{bmatrix} \tag{3.59}$$

$$\begin{bmatrix}\dot{U}\\ \dot{V}\\ \dot{W}\end{bmatrix} = \begin{bmatrix}VR - WQ\\ WP - UR\\ UQ - VP\end{bmatrix} + \begin{bmatrix}T_x/m\\ T_y/m\\ T_z/m\end{bmatrix} + \begin{bmatrix}(-\sin\theta)g\\ (\cos\theta\sin\phi)g\\ (\cos\theta\cos\phi)g\end{bmatrix} + \begin{bmatrix}F_{dx}/m\\ F_{dy}/m\\ F_{dz}/m\end{bmatrix} \tag{3.60}$$

$$\begin{cases} \dot{U} = VR - WQ - g(\sin\theta) + \frac{F_{dx}+T_x}{m} \\ \dot{V} = WP - UR + g(\cos\theta\sin\phi) + \frac{F_{dy}+T_y}{m} \\ \dot{W} = UQ - VP + g(\cos\theta\cos\phi) + \frac{F_{dz}+T_z}{m} \end{cases} \tag{3.61}$$

From (3.54):

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \tag{3.62}$$

$$\begin{cases} \dot{\phi} = P + Q(\sin\phi\tan\theta) + R(\cos\phi\tan\theta) \\ \dot{\theta} = Q(\cos\phi) + R(-\sin\phi) \\ \dot{\psi} = Q(\sin\phi\sec\theta) + R(\cos\phi\sec\theta) \end{cases} \tag{3.63}$$

From (3.55):

$$\begin{bmatrix} \dot{P} \\ \dot{Q} \\ \dot{R} \end{bmatrix} = \begin{bmatrix} 1/J_x & 0 & 0 \\ 0 & 1/J_y & 0 \\ 0 & 0 & 1/J_z \end{bmatrix} \left(- \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \times \begin{bmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \end{bmatrix} + \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \right) \tag{3.64}$$

If we define the skew-symmetric matrix of $\mathbf{\Omega} = \begin{bmatrix} P & Q & R \end{bmatrix}^T$:

$$S_k(\mathbf{\Omega}) = \begin{bmatrix} 0 & -R & Q \\ R & 0 & -P \\ -Q & P & 0 \end{bmatrix} \tag{3.65}$$

We can use the property $S_k(\mathbf{\Omega}) \cdot J\mathbf{\Omega} = \mathbf{\Omega} \times J\mathbf{\Omega}$. Thus:

$$\begin{bmatrix} \dot{P} \\ \dot{Q} \\ \dot{R} \end{bmatrix} = \begin{bmatrix} 1/J_x & 0 & 0 \\ 0 & 1/J_y & 0 \\ 0 & 0 & 1/J_z \end{bmatrix} \left(\begin{bmatrix} RQ(J_y - J_z) \\ RP(J_z - J_x) \\ QP(J_x - J_y) \end{bmatrix} + \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \right) \tag{3.66}$$

$$\begin{cases} \dot{P} = \frac{J_y - J_z}{J_x} RQ + \frac{\tau_\phi}{J_x} \\ \dot{Q} = \frac{J_z - J_x}{J_y} RP + \frac{\tau_\theta}{J_y} \\ \dot{R} = \frac{J_x - J_y}{J_z} QP + \frac{\tau_\psi}{J_z} \end{cases} \tag{3.67}$$

With the systems of equation (3.58), (3.61), (3.63), (3.67), we have the state equations:

$$\begin{cases} \dot{\phi} = P + Q(\sin\phi\tan\theta) + R(\cos\phi\tan\theta) \\ \dot{\theta} = Q(\cos\phi) + R(-\sin\phi) \\ \dot{\psi} = Q(\sin\phi\sec\theta) + R(\cos\phi\sec\theta) \\ \dot{P} = \frac{J_y - J_z}{J_x} RQ + \frac{\tau_\phi}{J_x} \\ \dot{Q} = \frac{J_z - J_x}{J_y} RP + \frac{\tau_\theta}{J_y} \\ \dot{R} = \frac{J_x - J_y}{J_z} QP + \frac{\tau_\psi}{J_z} \\ \dot{U} = VR - WQ - g(\sin\theta) + \frac{F_{dx} + T_x}{m} \\ \dot{V} = WP - UR + g(\cos\theta\sin\phi) + \frac{F_{dy} + T_y}{m} \\ \dot{W} = UQ - VP + g(\cos\theta\cos\phi) + \frac{F_{dz} + T_z}{m} \\ \dot{x} = U(\cos\psi\cos\theta) + V(\cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi) + W(\cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi) \\ \dot{y} = U(\sin\psi\cos\theta) + V(\sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi) + W(\sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi) \\ \dot{z} = U(-\sin\theta) + V(\cos\theta\sin\phi) + W(\cos\theta\cos\phi) \end{cases} \tag{3.68}$$

The state equations may be written in the form $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$. Considering an equilibrium point $\bar{\mathbf{x}} \in \mathbb{R}^{12}$:

$$\bar{\mathbf{x}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bar{x} & \bar{y} & \bar{z} \end{bmatrix}^T \tag{3.69}$$

When applying $\bar{\mathbf{x}}$ to the state equations, the reciprocal input $\mathbf{u} = \begin{bmatrix} T_z & \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^T$ will be $\bar{\mathbf{u}} = \begin{bmatrix} -mg & 0 & 0 & 0 \end{bmatrix}^T \in \mathbb{R}^4$. This operation point would correspond to a hover state. Note that this particular value of $\bar{\mathbf{u}}$ represents the necessary force to compensate for the weight acting over the drone [Sabatino 2015].

Linearizing by Taylor Series approximation:

$$A = \left.\frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}\right|_{\substack{\mathbf{x}=\bar{\mathbf{x}}\\ \mathbf{u}=\bar{\mathbf{u}}}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -g & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ g & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{3.70}$$

$$B = \left.\frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\right|_{\substack{\mathbf{x}=\bar{\mathbf{x}}\\ \mathbf{u}=\bar{\mathbf{u}}}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{J_x}\frac{\partial \tau_\phi}{\partial T_z} & \frac{1}{J_x} & 0 & 0 \\ \frac{1}{J_y}\frac{\partial \tau_\theta}{\partial T_z} & 0 & \frac{1}{J_y} & 0 \\ \frac{1}{J_z}\frac{\partial \tau_\psi}{\partial T_z} & 0 & 0 & \frac{1}{J_z} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1/m & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.71}$$

Considering possible additive disturbances caused by, for example, drag forces $\mathbf{d} =$

$\begin{bmatrix} F_{dx} & F_{dy} & F_{dz} \end{bmatrix}^T \in \mathbb{R}^3$:

$$D = \left.\frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{d}}\right|_{\substack{\mathbf{x}=\bar{\mathbf{x}}\\ \mathbf{u}=\bar{\mathbf{u}}}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1/m & 0 & 0 \\ 0 & 1/m & 0 \\ 0 & 0 & 1/m \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.72}$$

We have then the linear model $\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} + D\mathbf{d}$, which results in:

$$\begin{cases} \dot{\phi} = P \\ \dot{\theta} = Q \\ \dot{\psi} = R \\ \dot{P} = \frac{T_z}{J_x}\frac{\partial \tau_\phi}{\partial T_z} + \frac{\tau_\phi}{J_x} \\ \dot{Q} = \frac{T_z}{J_y}\frac{\partial \tau_\theta}{\partial T_z} + \frac{\tau_\theta}{J_y} \\ \dot{R} = \frac{T_z}{J_z}\frac{\partial \tau_\psi}{\partial T_z} + \frac{\tau_\psi}{J_z} \\ \dot{U} = -g\theta + \frac{F_{dx}}{m} \\ \dot{V} = g\phi + \frac{F_{dy}}{m} \\ \dot{W} = \frac{F_{dz}-T}{m} \\ \dot{x} = U \\ \dot{y} = V \\ \dot{z} = W \end{cases} \tag{3.73}$$

With T_x, T_y, T_z, τ_ϕ, τ_θ e τ_ψ defined by (3.13) and (3.14).

A second possibility for this analysis would be to consider the inputs as $\mathbf{u} = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 \end{bmatrix}^T$, where ω_i represents the actuators rotation speed. If we use the mixing matrix in (3.16) to obtain the conversion from mechanical conjugates to ω_i, we would obtain a different B matrix, given by:

$$B = \left.\frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\right|_{\substack{\mathbf{x}=\bar{\mathbf{x}}\\ \mathbf{u}=\bar{\mathbf{u}}}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{-\alpha_T d_x}{J_x}2\omega_1 & \frac{-\alpha_T d_x}{J_x}2\omega_2 & \frac{\alpha_T d_x}{J_x}2\omega_3 & \frac{-\alpha_T d_x}{J_x}2\omega_4 \\ \frac{-\alpha_T d_x}{J_y}2\omega_1 & \frac{-\alpha_T d_x}{J_y}2\omega_2 & \frac{\alpha_T d_x}{J_y}2\omega_3 & \frac{\alpha_T d_x}{J_y}2\omega_4 \\ \frac{-\alpha_Q}{J_z}2\omega_1 & \frac{\alpha_Q}{J_z}2\omega_2 & \frac{-\alpha_Q}{J_z}2\omega_3 & \frac{\alpha_Q}{J_z}2\omega_4 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{2\alpha_T\omega_1}{m} & \frac{2\alpha_T\omega_2}{m} & \frac{2\alpha_T\omega_3}{m} & \frac{2\alpha_T\omega_4}{m} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.74}$$

In order to approach the conversion from mechanical conjugates into actuator commands through 3.13 and 3.14, without approximations, considering aerodynamic effects, the calculation turns analytically complex very quickly. Hence, in this thesis, we proposed a neural network to perform this step in the control allocation, which will be detailed with simulation and experimental results in Chapter 4.

3.5.1 Stability, Observability and Controllability Analysis

Stability

Classically, a system is considered stable if the eigenvalues of the matrix A have a negative real part. It will be considered marginally stable if the eigenvalues have a null real part.

Thus, it is necessary to solve the equation:

$$|\lambda I - A| = 0 \tag{3.75}$$

Given the dimension of our matrix A shown previously in (3.70), the calculation was made using MATLAB and resulted in $\lambda = 0$ for all states, that is, the system is marginally stable.

Observability

In general terms, in order to know what happens inside a given system, it must be observable.

Consider the system:

$$\dot{\mathbf{x}} = A\mathbf{x} \tag{3.76}$$

where A is given by (3.70), and

$$\mathbf{y} = C\mathbf{x} \tag{3.77}$$

with dimensions $\mathbf{x} \in \mathbb{R}^{12}$, $\mathbf{y} \in \mathbb{R}^{12}$, $A \in \mathbb{R}^{12\times12}$, $C \in \mathbb{R}^{12\times12}$. The observability matrix is given by:

$$O = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{11} \end{bmatrix} \tag{3.78}$$

The system will be observable if, and only if, the observability matrix $O \in \mathbb{R}^{144\times12}$ has full rank. Once more, the computation on MATLAB showed that all states of the system are observable.

Controllability

Similarly, if one needs to modify a system by means of a control input, this system must be controllable.

A system will be controllable if, and only if, the controllability matrix has full rank. This matrix is given by $C_o \in \mathbb{R}^{12\times 48}$:

$$C_o = \begin{bmatrix} B & AB & A^2B & \cdots & A^{11}B \end{bmatrix} \tag{3.79}$$

Where B is given by (3.71). The results after computation on MATLAB showed that all the system states are controllable.

3.6 Final Considerations

In this chapter, the fundamental concepts of the quadrotor mathematical model were presented. These concepts will be needed for further comprehension of the contributions and experiments performed for this thesis. We have shown the equations that constitute the quadrotor model, highlighting the state variables which will be used in Chapter 4.

We have also presented the modeling of the main aerodynamic effects that have an influence on the quadrotor flight and how they modify the classic dynamic model. It has been shown that the previously neglected components of the thrust force appear in the force model and have influence in the *roll*, *pitch* and *yaw* torques.

Furthermore, we have made explicit the simplification made by the classic mixing matrix, which disregards blade flapping effects that lead to a thrust force rotation, which has also been modeled in this chapter, as one of the contributions of this research.

Finally, the presented model was linearized by Taylor Series approximation, analyzed around an equilibrium point and verified for stability, observability and controllability.

Chapter 4

Control Allocation with Aerodynamic Effects

In this Chapter, the basic cascade control structure for a quadrotor will be presented in Section 4.1. This structure is used for position and attitude control. Even if the controller algorithm is not the focus of this research, it is necessary to understand its principles since it is an important tool used in the system. This knowledge will also assist in the comprehension of where the control allocation contribution will operate.

In Section 4.2, we define some mathematical notation used throughout the Chapter as well as some concepts that were adopted in this research. In Section 4.3, we discuss control allocation and its role in control systems, and we also present the project and training of the neural network we used for our control allocation considering aerodynamic effects, the NNCA.

Finally, Sections 4.4.1 and 4.4.2 discuss simulation and real flight experiments results that compare the classic mixing matrix control allocation (MMCA) with the proposed neural network control allocation (NNCA) approach.

The contributions of this Chapter were published in [Madruga et al. 2022].

4.1 Basic Quadrotor Control Structure

Although the quadrotor control algorithm is not the main focus of this research, one needs to understand its principles in order to comprehend the operation of the aircraft and, consequently, the role of this thesis' contributions, especially in the control allocation step (which we consider to be separate from the controller itself, as shown in Figure 4.1).

The quadcopter usually works with a cascade control structure, where with the output of the first controller provides the set point for the second controller, In this case, the outer loop consists of the *Position Controller*, while the *Attitude Controller* forms the inner loop.

The path commands (x, y) are generated in the *Trajectory Generator*, which sends them to the *Position Controller*. This controller in turn outputs the *roll* (ϕ) and *pitch* (θ) commands to the *Attitude Controller*. The altitude (z) command comes directly from the *Trajectory Generator*, and the *yaw* (ψ) controller is independent. The respective adjustments in the angular velocities of the motors to achieve the desired behavior in ϕ, θ, ψ and z will then be sent to the quadcopter through the *Control Allocation*. The sensing system must be adequate for good state feedback and, thereafter, an accurate calculation of the error.

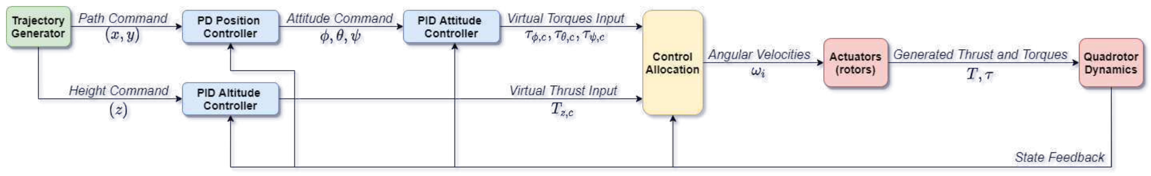

Figure 4.1: Block diagram for the quadrotor control strategy.

We have the following expressions for the attitude PID controller, where e_ϕ and e_θ represent the roll and pitch errors, respectively; P is the angular velocity around the x-axis in the body-frame, and Q is the angular velocity around the y-axis in the body-frame:

$$\tau_{\phi_c} = K_p e_\phi + K_i \int e_\phi \, dt + K_d \dot{e}_\phi \tag{4.1}$$

$$\tau_{\theta_c} = K_p e_\theta + K_i \int e_\theta \, dt + K_d \dot{e}_\theta \tag{4.2}$$

As aforementioned, the attitude controller commands are generated by the PD trajectory controller, shown in (4.3) and (4.4). Note that we need to perform a rotation around the z-axis (ψ) during the error calculation because the body and inertial frames are not necessarily aligned in the same direction.

$$\theta_{cmd} = K_p(\cos(\psi)e_x + \sin(\psi)e_y) + K_d v_x \tag{4.3}$$

$$\phi_{cmd} = K_p(-\sin(\psi)e_x + \cos(\psi)e_y) + K_d v_y \tag{4.4}$$

The yaw control is made separately, as in (4.5), where e_ψ represents the yaw error and R is the angular velocity around the z-axis in the body-frame:

$$\tau_{\psi_c} = K_p e_\psi + K_d \dot{e}_\psi \tag{4.5}$$

Finally, the height control is performed through a PID, as shown below. Table 4.1 shows the control gains used in the experiments throughout this thesis.

$$T_{z_c} = K_p e_z + K_i \int e_z \, dt + K_d \dot{e}_z \tag{4.6}$$

Controller Gains	**Position PD** [K^x K^y]	**Attitude PID** [K^θ K^ϕ]	**Altitude/Height PID** [K^z]	**Yaw PD** [K^ψ]
Proportional (K_p)	$[-0.24\ 0.24]$	$[0.013\ 0.01]$	$[0.8]$	$[0.004]$
Integral (K_i)	-	$[0.01\ 0.01]$	$[0.24]$	-
Derivative (K_d)	$[0.1\ -0.1]$	$[0.002\ 0.003]$	$[0.5]$	$[0.0012]$

Table 4.1: Controller gains used in the experiments throughout this thesis.

Since we aim to compare two different control allocation methods in this chapter, the following assumption is naturally required:

Assumption 2. *For the same controller design, simply replacing the traditional mixing matrix control allocation with the neural network control allocation here proposed does not harm the control project.*

In order to verify this assumption, we designed a simple controller for the quadrotor system with the mixing matrix and tested it to see if the project criteria were maintained when replacing it with the novel neural network control allocation we propose further ahead.

We set the project criteria of 1s rising time (t_r), which is the time where the output first reaches the reference, and 3s settling time (t_s), defined as the time where the output is within a 1% error when compared to the steady state. With this, we specified the corresponding proportional (K_p), integral (K_i) and derivative (K_d) attitude controller gains, shown in Table 4.2.

The controllers were implemented in discrete-time, with the Bogacki-Shampine method, which is a requirement for it to be deployed to the hardware for real flight experiments afterward. The step size was 5 ms, which was chosen to match the sampling time of the quadrotor during flight experiments. Other parameters necessary for the simulation, such as constants that model the aircraft, were obtained from the documentation provided by the manufacturer.

The obtained step response behaviors are presented in Fig. 4.2 and detailed in Table 4.2, for the traditional mixing matrix control allocation and the novel neural network control allocation. We can infer, comparing both step responses, that **Assumption 2** is in fact true, as the difference between them is negligible. Therefore, it is possible to use the new control allocation approach and apply it to more elaborate maneuvers to observe the results. Once again, we reiterate that the PD-PID controllers are used here merely as a tool to perform the experiments and simulations, and are not the focus of the contributions of this work.

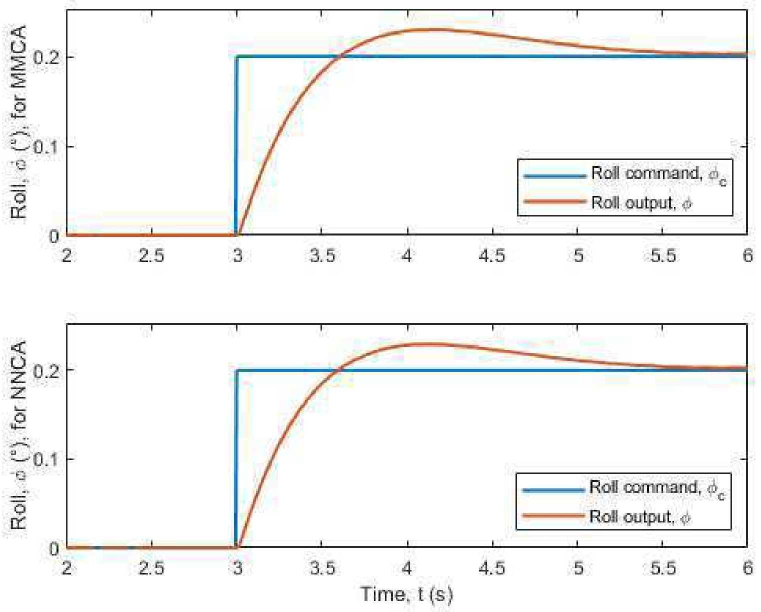

Figure 4.2: Roll step responses for the control system using the traditional mixing matrix (MMCA), on the top, and the proposed network control allocation (NNCA), on the bottom.

	Rise time (t_r)	**Settling time (t_s)**	**Overshoot**
Mixing Matrix	0.605 s	3 s	14.9%
Neural Network	0.6 s	3 s	14.6%
K_p	0.0092		
K_i	0.0097		
K_d	0.003		

Table 4.2: Step response characteristics.

4.2 Method Overview and Notation

In our control allocation, the objective is to produce actuator commands that will generate resultant torques and thrusts on the UAV body as close as possible to the requested control signals, also called virtual control inputs. In this sense, we provide here an overview of the notations used throughout this thesis, for a better understanding of the upcoming Sections.

Definition 2. *The virtual control inputs* $[\mathbf{T}_c, \boldsymbol{\tau}_c] = \begin{bmatrix} T_c & \tau_{\phi_c} & \tau_{\theta_c} & \tau_{\psi_c} \end{bmatrix}^T \in \mathbb{R}^4$ *are the commanded signals at the output of the controller, where* $\mathbf{T}$ *represents the thrust force, and* $\boldsymbol{\tau}$*, the torques.*

Remark 1. *In each of the four actuators* i*,* $i = 1, 2, 3, 4$*, there are the body-frame thrusts,* $\mathbf{T}_i^B \in \mathbb{R}^3$*, and torques,* $\boldsymbol{\tau}_i \in \mathbb{R}^3$*, as well as their components* $\begin{bmatrix} T_{x_i} & T_{y_i} & T_{z_i} \end{bmatrix}^T$ *and* $\begin{bmatrix} \tau_{\phi_i} & \tau_{\theta_i} & \tau_{\psi_i} \end{bmatrix}^T$*, respectively.*

Definition 3. $\mathbf{u} = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 \end{bmatrix}^T \in \mathbb{R}^4$ *is the vector of angular velocities sent as actuator commands by the control allocator. These velocities should produce the desired resultant thrust and torques.*

Definition 4. *The vector* $[\mathbf{T}, \boldsymbol{\tau}] = \begin{bmatrix} T & \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^T \in \mathbb{R}^4$ *represents the real resultant mechanical conjugates established at the UAV body. In this thesis, they were calculated by means of experimental data and dynamic model equations. Again,* $\mathbf{T}$ *represents the thrust force, and* $\boldsymbol{\tau}$*, the torques.*

Assumption 3. *If F is a virtual function that describes the transformation of the actuator commands ω_i to real thrust and torques, and $\widetilde{F}$ is its model, we have:*

$$\begin{bmatrix} \mathbf{T} \\ \boldsymbol{\tau} \end{bmatrix} = \widetilde{F}(\mathbf{u}) \tag{4.7}$$

Hence, the desired control allocation must perform the inverse operation, using the virtual control inputs:

$$\mathbf{u} = \widetilde{F}^*(\mathbf{T}_c, \boldsymbol{\tau}_c), \tag{4.8}$$

where $\widetilde{F}^*$ represents the function which performs the inverse operation of $\widetilde{F}$.

Using this notation, the control allocation algorithm goal is to achieve $\mathbf{T} = \mathbf{T}_c$ and $\boldsymbol{\tau} = \boldsymbol{\tau}_c$. Having $\begin{bmatrix} \mathbf{T} \\ \boldsymbol{\tau} \end{bmatrix} = F(\widetilde{F}^*(\mathbf{T}_c, \boldsymbol{\tau}_c))$, the composition of the inverse mappings with the real system behavior functions must be equal to identity [Chebbi et al. 2020]. See Fig. 4.3.

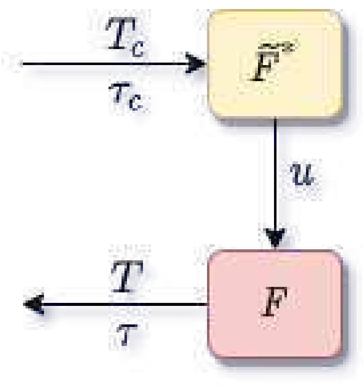

Figure 4.3: Data flow from the virtual control inputs $\mathbf{T}_c, \boldsymbol{\tau}_c$ to the real mechanical conjugates applied on the UAV body $\mathbf{T}, \boldsymbol{\tau}$. $\widetilde{F}^*$ can represent the control allocation block.

Remark 2. *$\widetilde{F}^*$ can be interpreted as the control allocator. In this thesis, it can refer to the classic mixing matrix or the proposed neural network, and we will compare the performance of both methods.*

Remark 3. *$\widetilde{F}$ represents the quadcopter dynamic model considering aerodynamic effects such as blade flapping that were detailed in Chapter 3.*

As has been stated in Chapter 3, when adding the aerodynamic consideration to the classic quadrotor model, the previously invertible relation provided by the classic allocation matrix is lost. That is, $\widetilde{F}^*$ is not simply the matrix F^{-1} anymore. Several control allocation methods, as will be detailed below, use optimization methods in order to allocate the control effort or, in this case, perform the inverse operation of F. Throughout the process of this research, many numeric methods have been tested in an effort to find $\widetilde{F}^*$, since the analytical inversion was not trivial, even using tools such as MATLAB's symbolic math toolbox. These methods included interior point, trust-region, active-set and sequential quadratic programming, but none achieved the desired result.

That motivated the research to move towards the usage of a function-fitting neural network as a tool to provide us with the necessary inverse operation of F. Note that the artificial neural network (ANN) in this work is merely a tool to reach a goal. This thesis does not intend to contribute to the state-of-the-art of ANN architectures, so once a function-fitting feed-forward neural network fit our objectives, it was used as part of the solution, and the research moved forward with it, not finding the need to pursue other neural network configurations such as recurrent or convolutional ANNs, though that could be addressed in future works. It is also worth mentioning that a necessary constraint for this thesis was that the neural network configuration was kept relatively simple because of platform restrictions regarding computational capacity.

That being said, the contribution at this point of the research lies in a new control allocation approach with a well-established method in the literature, and a solution to the main problem of how to include aerodynamic effects into the control system, following a more faithful model for the quadrotor and guaranteeing a more trustworthy setting of the commanded control effort, without the need of modifying the controller algorithm itself or even the controller tuning.

4.3 Neural Network Control Allocation

Generally speaking, a control allocation solves an undetermined, typically constrained, system of equations [Härkegård 2004]. The input to the control allocation block is the total control effect to be produced, or the virtual control input $\tau_c(t) \in \mathbb{R}^k$. Its output, on the other hand, is the achieved actuator command $u(t) \in \mathbb{R}^m$, where $m > k$ if the system is over-actuated, or $m < k$ if the system is under-actuated. That is, with a given $\tau_c(t)$, we search $u(t)$ such that:

$$g(u(t)) = \tau_c(t), \tag{4.9}$$

where $g : \mathbb{R}^m \to \mathbb{R}^k$ is the mapping from the achieved actuator commands to the virtual control inputs in the system to be controlled [Härkegård 2003]. In the control allocation literature, (4.9) usually becomes:

$$B_m u(t) = \tau_c(t), \tag{4.10}$$

with B_m being called the control effectiveness matrix of dimensions $k \times m$ and rank k. Fig. 4.4 illustrates the control allocation process. It must transform the virtual inputs τ_c into commands u sent to the actuators such that they generate the allocated inputs $\tau = \tau_c$.

It is important to stress that, when it comes to the process of control allocation, the terminology used depends on the application in mind, so the control allocator can also be known as control mixer, control blender, control selector, or control distributor [Härkegård 2003]. Therefore, the control effectiveness matrix can be referenced by several notations. In quadcopter applications, it is usually called mixing matrix, referencing the control mixer; the control inputs are the thrusts and torques, and the actuator commands are the angular velocities ω_i for each rotor.

The traditional formulation of the linear control allocation problem is the system (4.11), which consists of (4.9) with the application of the actuator constraints [Johansen & Fossen 2013], [Härkegård 2003].

$$\begin{cases} B_m u = \tau_c, \\ u_{min} < u < u_{max}. \end{cases} \tag{4.11}$$

There are three possible outcomes when solving (4.11):

1. There is an infinite number of solutions;

2. There is one unique solution;

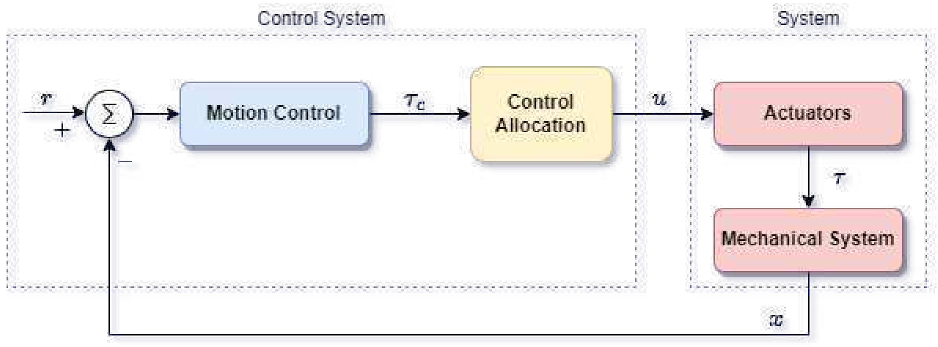

Figure 4.4: Control system structure including control allocation. The **motion control** specifies a virtual control effort τ_c that must be produced, based on the reference signal r and the state feedback x. The **control allocation** distributes this control demand among the **actuators**, which generate the actual allocated control effort τ. If the allocation is successful, then $\tau_c = \tau$.

3. No solution exists.

Many proposed control allocation methods correspond to different ways of computing the solution for a specific allocation objective, which most of the time is to avoid actuator saturation, rather than other objectives. The majority of well-established control allocation algorithms depend on numerical optimization to solve the control allocation problem.

Next, we briefly mention and describe the main classic allocation methods according to [Härkegård 2003]. All of these alternatives can be solved using numerical algorithms:

- Optimization-based Control Allocation methods rely on a pragmatic interpretation of the control allocation problem, which states: for the system (4.11), determine a feasible control input so that this system is satisfied. If there are several solutions, pick the optimal one.

- In Direct Control Allocation, the choice of control input is made unique by geometric reasoning.

- In Daisy Chain control allocation, the allocation suite is divided into groups, which are successively employed to generate the total control effort.

The main problem approached when using control allocation in the literature is the saturation of actuators in over-actuated systems. The principles of control allocation are however general and can be used to optimize other disturbances in the systems, from power or fuel consumption to aerodynamic effects [Johansen & Fossen 2013], which is the case of this research.

4.3.1 Neural Network Project and Training

The problem of allocating the proper angular velocities to the rotors, which will generate the commanded virtual thrust and torques, is solved with an analytical inversion of (3.12) and (3.14), which transforms the angular velocities into thrust and torques, $\omega_i \rightarrow [\mathbf{T}, \boldsymbol{\tau}]$.

We need therefore a solution to perform the opposite operation, $\widetilde{F}^*$ (as described in Section 4.2), from the virtual control inputs to the rotor velocities, $[\mathbf{T}_c, \boldsymbol{\tau}_c] \rightarrow \omega_i$, considering the aerodynamic effects introduced in Chapter 3. This inverse transformation would derail the invertible relation of the classic mixing matrix and hinder its usage. In this work, we propose the use of a neural network to perform the function fitting for finding $\widetilde{F}^*$, and use it as the control allocation.

Definition 5 (Neural Network Control Allocation (NNCA))**.** *The control allocation performed by means of a function fitting neural network that provides a transformation $\widetilde{F}^*$ from the plant input $[\mathbf{T}_c, \boldsymbol{\tau}_c] = \begin{bmatrix} T_{z_c} & \tau_{\phi_c} & \tau_{\theta_c} & \tau_{\psi_c} \end{bmatrix}^T$ to the actuator commands $\mathbf{u} = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 \end{bmatrix}^T$, taking into account aerodynamic effects as described in equations (3.12) and (3.14).*

That is, as in Section 4.2:

$$\mathbf{u} = \widetilde{F}^*(\mathbf{T}_c, \boldsymbol{\tau}_c), \tag{4.12}$$

In our specific case, the equations also depend on some system states, namely $\mathbf{v}$ and $\boldsymbol{\Omega}$, so we have:

$$\mathbf{u} = \widetilde{F}^*(\mathbf{T}_c, \boldsymbol{\tau}_c, \mathbf{x}), \tag{4.13}$$

Remark 4. *The control allocation step in a system may or may not need state feedback to perform the conversion from virtual controller commands to actuator commands. In the quadrotor's case, there are states in the aerodynamic effects equations, so state feedback is* *needed to perform the inverse operation, as shown in Figure 4.5. The system states used here will be detailed further in this Chapter.*

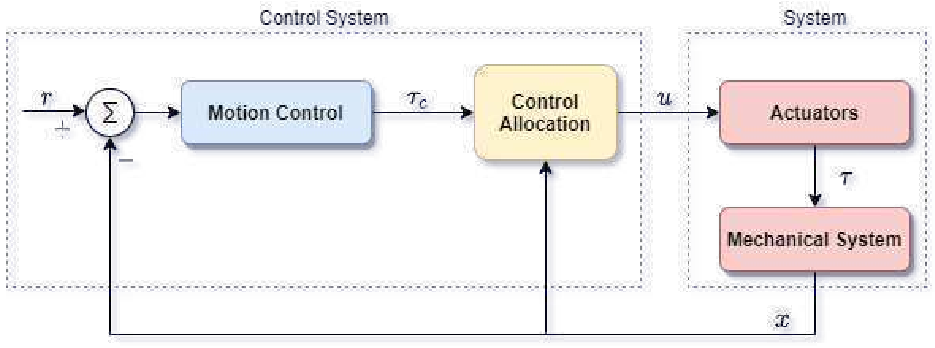

Figure 4.5: Control allocation with state feedback.

Function fitting is the process of training a neural network on a set of inputs with the aim of producing an associated set of target outputs. After the network has been created with the desired hidden layers and the training algorithm, the neural network must be trained using a set of training data. Once the neural network has fitted the data, it forms a generalization of the input-output relationship. The trained network can therefore be used to generate outputs for a new set of inputs that were not used in training if the training set was appropriate. Neural networks are good at fitting functions. There is proof that a fairly simple neural network can fit any practical function [Hornik, Stinchcombe & White 1989], [Gorban & Wunsch 1998].

The proposed function fitting neural network is composed of a two-layer feed-forward network with sigmoid hidden neurons and linear output neurons. This structure was chosen since it can fit multi-dimensional mapping problems arbitrarily well, provided consistent data and enough neurons in its hidden layer are given. In this case, 20 neurons were used for the first step of the training, and 10 neurons for the second one, as described below. The diagram for this neural network can be seen in Figure 4.6.

The neural network control allocation approach has a two-step training process in order to take into account the aerodynamic effects, as well as achieving desirable training performance and mean squared error (MSE) order. The second step of the training process is independent of the mixing matrix.

In the mathematical model presented in Chapter 3, we do not represent uncertainties caused by nonlinear terms. Nonetheless, the curve fitting provided by the neural network

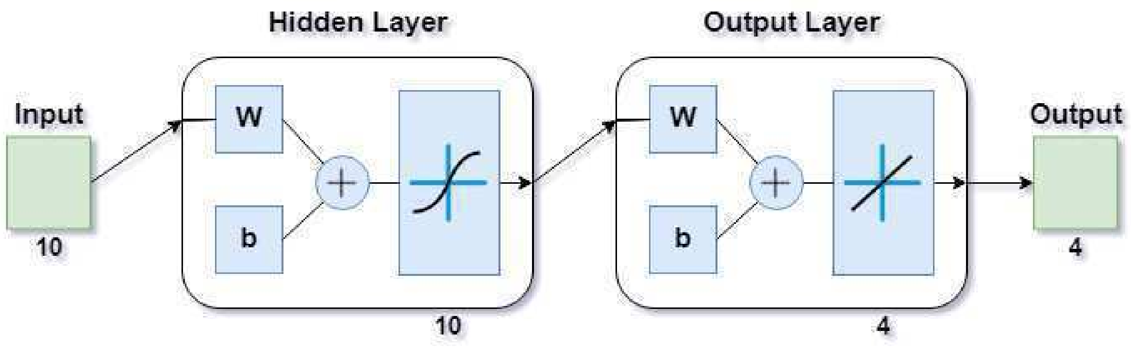

Figure 4.6: Neural network diagram showing the hidden layer with sigmoid neurons and the output layer with linear neurons. $\mathbf{W}$ and **b** represent the weights and biases, respectively. The network has 10 inputs, which are the mechanical conjugates and some of the plant states, and 4 outputs, which correspond to the actuators' velocities.

imposes an approximation error influenced by uncertain nonlinear terms.

With the purpose of minimizing the effect of these uncertainties, we aimed for a certain MSE order based on recent works in the literature regarding curve-fitting using neural networks [Gonzalez-Diaz et al. 2020], [Li, Wibowo & Guo 2019], [El-Melegy 2013]. To achieve this goal, we used our two-step training process, as will be described next. The references found in the literature used radial basis function (RBF) networks for the curve-fitting. Considering that, we first trained RBF neural networks using the same mentioned two-step training method. Our objective was to compare the MSE of these RBF results to the MSE of the network structure we are using and see if they agree with the results found in the literature [Gonzalez-Diaz et al. 2020], [Li, Wibowo & Guo 2019], [El-Melegy 2013].

For the RBF networks, the training process was the same as below, creating a two-layer network with radial basis neurons in the first layer and linear neurons in the second. In the first training, the RBF network obtained an **MSE of** $\mathbf{4.2 \times 10^{-2}}$. In the second training, the RBF network achieved an **MSE of** $\mathbf{6.0297 \times 10^{-4}}$. These error orders are expected, according to the literature [Gonzalez-Diaz et al. 2020], [Li, Wibowo & Guo 2019], [El-Melegy 2013]. If we compare these results with the ones of our proposed network, detailed in Table 4.3, it is possible to conclude that not only are they within the expected MSE order of previous works, but our results are slightly better than the MSE obtained with the RBF networks and the same training data.

The complete training for our NNCA is described next and summarized in Table 4.3.

		1^{st} **training: Bayesian-regularization** (20 **neurons,** 64004 **samples)**	2^{nd} **training: Levenberg-Marquadt** (10 **neurons,** 34136 **samples)**
Sample division	**Dataset**	**Mean-squared error**	**Mean-squared error**
70%	**Train**	3.28812×10^{-2}	4.4857×10^{-4}
15%	**Validation**	-	5.98463×10^{-4}
15%	**Test**	4.09559×10^{-2}	4.50975×10^{-4}

Table 4.3: MSE for the neural network control allocation training process.

- **First Training**

  1. An open-loop simulation of the quadrotor system without any controller was performed, using pseudo-random binary sequence (PRBS) pulses with amplitude 0.2 as mechanical conjugates ($\mathbf{T}_c, \boldsymbol{\tau}_c$) to collect the corresponding angular velocities outputs (ω_i). Other variables ($\mathbf{v}$ and $\boldsymbol{\Omega}$) were also collected for training, as detailed in Fig. 4.9.
  2. The collected angular velocities data were applied to the aerodynamic effects equations (3.12) and (3.14) aiming to collect the real mechanical conjugates ($\mathbf{T}, \boldsymbol{\tau}$) established at the aircraft.
  3. The input-output data ($\mathbf{T}, \boldsymbol{\tau}, \boldsymbol{\Omega}, \mathbf{v} \rightarrow \omega_1, \omega_2, \omega_3, \omega_4$) were then used for the first training step, with the Bayesian-regularization algorithm to update weight and bias values.

With the first training, we obtained a mean squared error of the order 10^{-2}, as detailed in Table 4.3. The performance of this first training using Bayesian Regularization is shown in Figure 4.7. To refine the network and obtain an MSE order of at least 10^{-3} as in the mentioned recent works found in the literature [Gonzalez-Diaz et al. 2020], [Li, Wibowo & Guo 2019], [El-Melegy 2013], the second training was performed, using the results of the first one.

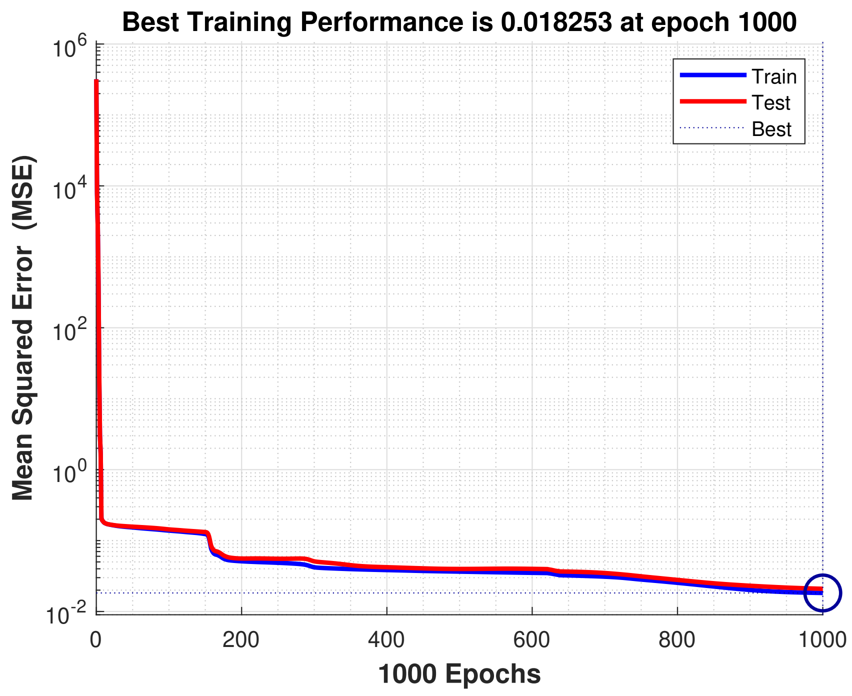

Figure 4.7: Performance of the first training step, made in open-loop, using the Bayesian Regularization method.

- **Second Training**

  1. A new set of input-output data were collected from a closed-loop simulation with the first trained network as the control allocation method and a PD-PID error controller.
  2. 3-2-1-1 maneuvers were used as attitude commands, and the network obtained in the first step converted the virtual control commands ($\mathbf{T}_c, \boldsymbol{\tau}_c$) into angular velocities (ω_i).
  3. Once more, the collected angular velocities data were applied to the aerodynamic effects equations (3.12) and (3.14) aiming to collect the real mechanical conjugates ($\mathbf{T}, \boldsymbol{\tau}$) established at the aircraft.
  4. The input-output data ($\mathbf{T}, \boldsymbol{\tau}, \boldsymbol{\Omega}, \mathbf{v} \rightarrow \omega_1, \omega_2, \omega_3, \omega_4$) were then used for the second neural network training, with the Levenberg-Marquadt algorithm to update weight and bias values. This time, we obtained the MSE of the order 10^{-4}.

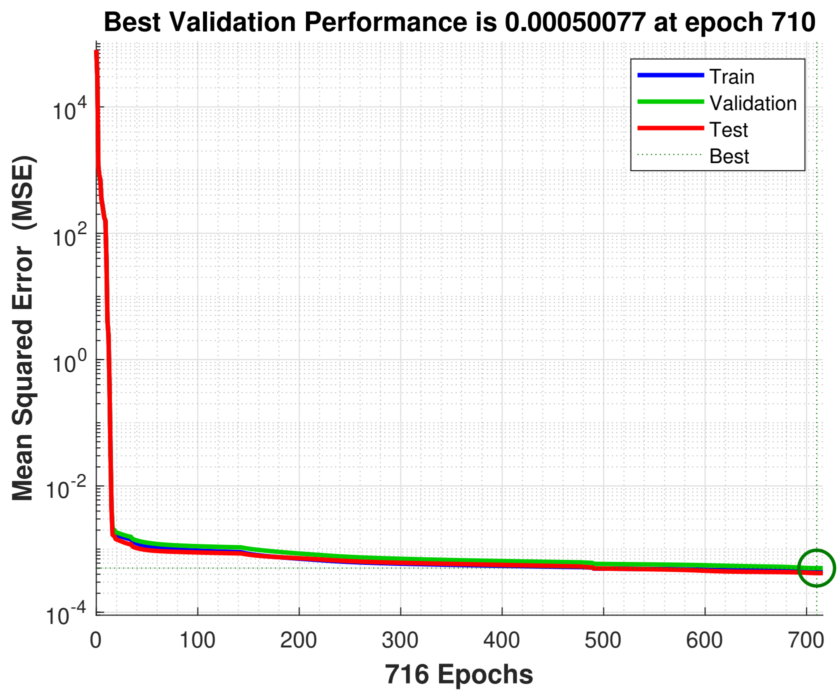

Figure 4.8: Performance of the second training step, made in closed-loop, using the Levenberg-Marquadt algorithm.

The performance of the second training using the Levenberg-Marquadt algorithm is shown in Figure 4.8. Bayesian Regularization also updates the weight and bias values according to Levenberg-Marquardt optimization. However, it then minimizes a combination of squared errors and weights to determine the correct combination so as to produce a network that generalizes well.

The primary application of the Levenberg–Marquardt algorithm is in the least-squares curve fitting problem. That is, given a set of m empirical pairs (x_i, y_i) of independent and dependent variables, find the parameters β of the model curve $f(x, \beta)$ so that the sum of the squares of the deviations $S(\beta)$ is minimized:

$$\hat{\beta} \in \arg\min_{\beta} S(\beta) \equiv \sum_{i=1}^{m} [y_i - f(x_i, \beta)]^2 \tag{4.14}$$

In this case, the empyrical pairs consisted of the vectors of the input elements $x_i = (T_{zi}, \tau_{\phi_i}, \tau_{\theta i}, \tau_{\psi_i}, \Omega_{\phi_i}, \Omega_{\theta i}, \Omega_{\psi_i}, v_{xi}, v_{y_i}, v_{zi})$, and the vectors of the output elements $y_i = (\omega_{1i}, \omega_{2i}, \omega_{3i}, \omega_{4i})$, as detailed further.

Both training steps were carried out for a maximum of 1000 epochs or until the validation error failed to decrease for 6 iterations, and we obtained Regression values of $R = 0.99999$ for the two steps. Regression values measure the correlation between outputs and targets. An R value near 1 means that the variables are strongly related, while an R-value near 0 means that they have little to no impact on each other.

The whole training process was performed offline, meaning that during the flight this allocation method requires low computational power, because the network is already trained and no update is made online. The classic mixing matrix was only used in the first step of the training to collect data.

While the mixing matrix control allocation uses only the thrust and torques to perform the matrix inversion and obtain the squared angular velocities, more variables were added to the neural network training process for the purpose of obtaining a better fit, according to the aerodynamic effects modeled in Chapter 3. See Fig. 4.9, which details the input and output variables used in training. Therefore we used 10 input elements $(T_z, \tau_\phi, \tau_\theta, \tau_\psi, \Omega_\phi, \Omega_\theta, \Omega_\psi, v_x, v_y, v_z)$, with T_z as the z-axis component of the trust T, which comes from the height controller, to obtain 4 output elements $(\omega_1, \omega_2, \omega_3, \omega_4)$. During the UAV flight, these additional state variables ($\mathbf{v}$ and $\mathbf{\Omega}$) are obtained from sensing.

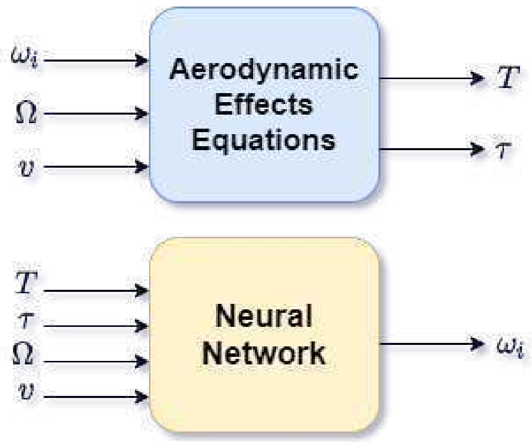

Figure 4.9: Block diagrams representing the variables used for neural network training and therefore for neural network control allocation.

4.4 Results

In order to validate the proposed approaches, we performed simulations and experimental tests with the mini-drone PARROT Mambo, which has a hardware support package from MATLAB/Simulink. This quadrotor is represented in Figure 3.1. It is equipped with a downward-facing camera for positioning using optical-flow odometry, an ultrasonic sensor, a barometric sensor, and an inertial measurement unit (IMU). All these sensors are used for stabilization purposes. The mini-drone dimensions are 180 mm $\times$ 180 mm $\times$ 40 mm, weighting 70 g. Power is provided by a GiFi Power 3.7 V, 600 mAh, 2.2 Wh, 15c Li-Po battery, enabling a flight time of 7-10 minutes, depending on the nature of the task to be performed and the accessories (load) implemented.

The applied control strategy for the quadcopter operation consisted of the PD-PID controller mentioned in Section 4.1, implemented in discrete-time, with the Bogacki-Shampine method and a step size of 5 ms. In Fig. 4.1, the **actuators (rotors)** and **quadrotor dynamics** blocks represent the mini-drone, while the other blocks correspond to the algorithm embedded in the aircraft.

A scheme illustrating the interaction between the user, quadrotor and MATLAB/Simulink support package (which contains the simulator) is shown in Figure 4.10, while the main Simulink environment interface is shown in Figure 4.11. The PARROT Mambo can receive commands either via computer or joystick. If any algorithm modification is done in MATLAB/Simulink, the firmware in the drone needs to be updated in order to deploy the modifications made by the user to the hardware. For the experiments performed throughout this research, we have used MATLAB/Simulink to generate flight commands and fly the drone. Our aircraft was autonomous, that is, it did not have a pilot commanding via joystick.

We have evaluated the proposed method by comparing the mechanical conjugates at the output of the controller (the virtual control commands $\mathbf{T}_c, \boldsymbol{\tau}_c$) to those established at the aircraft ($\mathbf{T}, \boldsymbol{\tau}$), which were calculated using the inflow and blade flapping equations. We also analyzed changes in the controller performance. The neural network was able to closely match $\mathbf{T}, \boldsymbol{\tau}$ and $\mathbf{T}_c, \boldsymbol{\tau}_c$, while the classic control allocation with the mixing matrix could not achieve the same performance. The error between the two mechanical conjugates ($\boldsymbol{\tau} - \boldsymbol{\tau}_c$

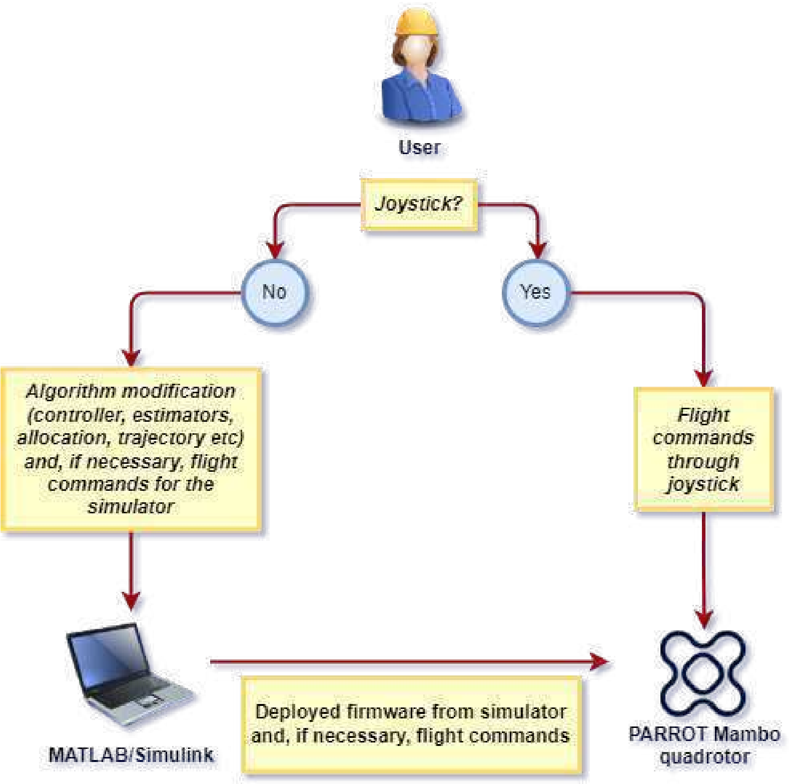
User
Joystick?
No
Yes
Algorithm modification (controller, estimators, allocation, trajectory etc) and, if necessary, flight commands for the simulator
Flight commands through joystick
MATLAB/Simulink
Deployed firmware from simulator and, if necessary, flight commands
PARROT Mambo quadrotor

Figure 4.10: Scheme for user-drone-simulator connection.

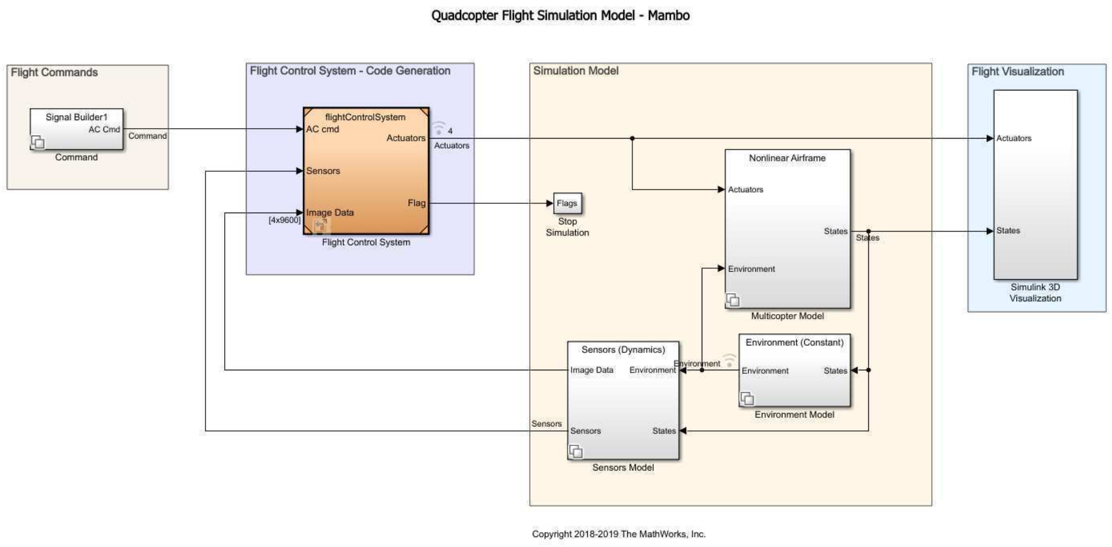

Figure 4.11: Simulink environment interface for the PARROT Mambo support package.

and $\mathbf{T} - \mathbf{T}_c$), called allocation error, was also determined in both cases. We were able to enhance the performance of the PD-PID controller with the new control allocation, without altering the controller gains, that is, the tuning was the same for both the traditional mixing matrix and our proposed neural network control allocations.

4.4.1 In Silico Simulation Tests

The simulation environment has the advantage of making possible tests that would not be trivial to perform in a real flight experiment, for several reasons: the size of the room, walls as obstacles, dangerous maneuvers, damaging equipment etc.

The following test consisted of an attitude maneuver called 3-2-1-1, where opposing step commands are sent to one of the attitude axes of the quadrotor for 3 s, 2 s, 1 s e 1 s again. It is a mildly aggressive maneuver, depending on the attitude angle command, naturally, and, because of the lack of position commands, can culminate with the drone hitting the surrounding walls. Hence, we took advantage of the simulation environment to perform this test preliminarily, while ultimately using a proper trajectory in the real flight experiment in Section 4.4.2.

The simulation was performed applying a *roll* 3-2-1-1 maneuver, with an $11°$ command. The control technique used here is as described in Section 4.1 for the attitude, that is, a PID controller. Since there is no position command in this test, there is also no position controller, i.e. the user sends the angle commands directly to the attitude controller.

The flight performance results for the classic mixing matrix and the proposed neural network can be seen in Figures 4.12 to 4.16. With the same PID gains in both flights, the controller working with the mixing matrix cannot follow the reference for the roll maneuver in Figure 4.12, showing a great steady-state error. In Figure 4.13, we can see that the performance is greatly improved with the new control allocation that considers the aerodynamic effects.

Likewise, Figures 4.14 and 4.15 show the pitch commands performance, for the mixing matrix and the neural network respectively. In Figures 4.16 and 4.17, the same is shown for the yaw commands. Even though in the pitch case the results seem marginally worse for the neural network, the yaw angle gives us a better behavior for the new proposed control allocation once more, and since the improvement was very significant in the maneuver axis (i.e. roll), we consider the data to be favorable to the neural network control allocation.

Furthermore, Figures 4.18 and 4.19 show the comparison between the virtual torque commands and the established roll torque in the quadrotor, for both control allocations. One can see, in figure 4.18, that the actuators cannot generate the torque commands that the controller is demanding for a reasonable amount of time. This is what we call an allocation error. On the other hand, in Figure 4.19, it is shown that the new control allocation sends commands to the actuators that are possible to achieve, so the controller commands are better established. This means that with the consideration of the aerodynamic effects in the control allocation, the controller is capable of providing commands more suitable to the actuators' capabilities, resulting in a more suitable control effort.

The mean squared error (MSE) calculations for the attitude angles are shown in Table 4.4. As the graphics already indicated, the error for the neural network control allocation was significantly smaller for the roll angle. The null MSE for the yaw angle should be taken carefully, since this result is from a simulated environment.

	MSE Roll	**MSE Pitch**	**MSE Yaw**
Mixing Matrix	21.0879	3.0465×10^{-4}	2.0663
Neural Network	15.1296	1.4×10^{-3}	0

Table 4.4: Attitude mean squared errors for the NNCA simulation results. The smallest errors are highlighted.

Finally, one could argue that it would be possible to improve the mixing matrix system performance by adjusting the controller gains. However, even if an improvement is possible by drastically increasing the controller gains, we were not able to eliminate the steady-state error completely for the mixing matrix case, because of the allocation error. Also, since we are comparing two control allocation algorithms, and not controller algorithms, we judged it to be relevant that the same controller was used with the same tuning in both cases, to make explicit the benefits brought by the newly proposed method.

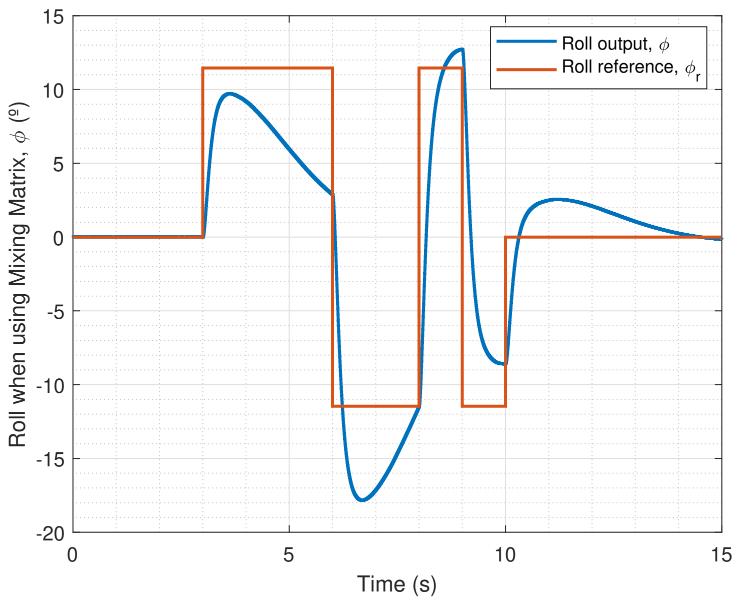

Figure 4.12: Roll performance for the classic mixing matrix control allocation (MMCA).

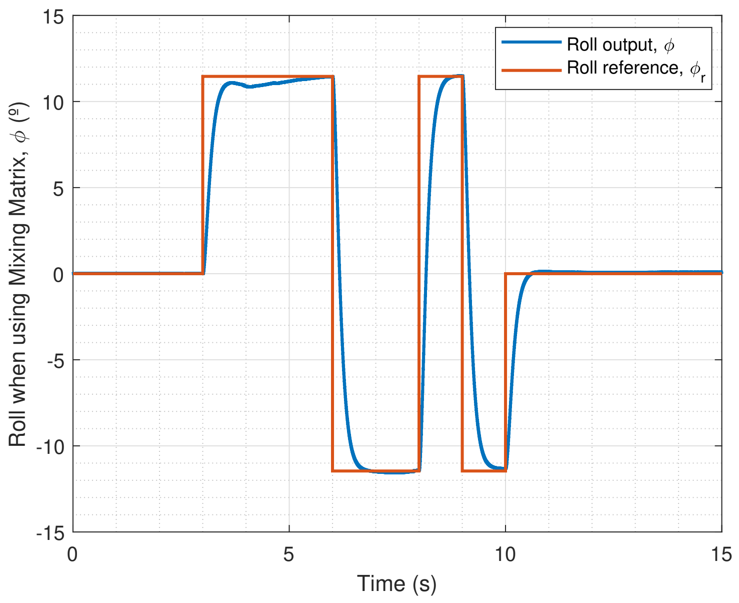

Figure 4.13: Roll performance for the neural network control allocation (NNCA).

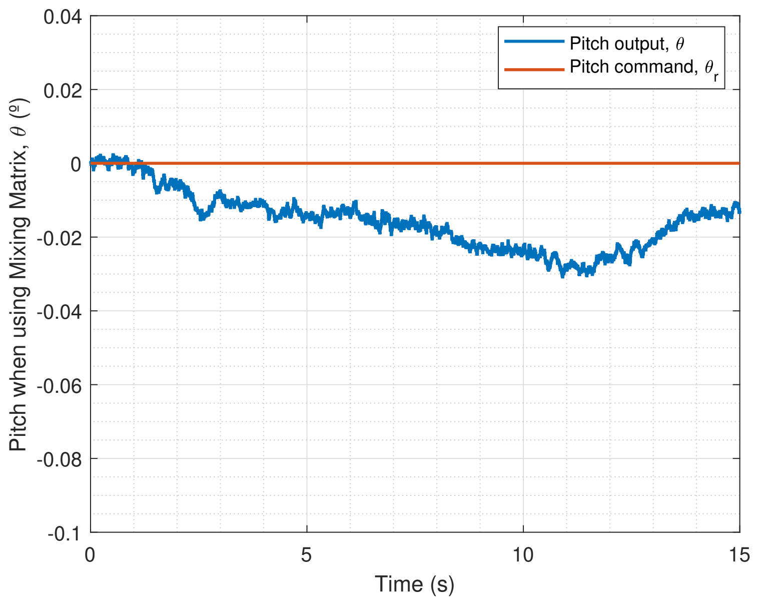

Figure 4.14: Pitch performance for the classic mixing matrix control allocation (MMCA).

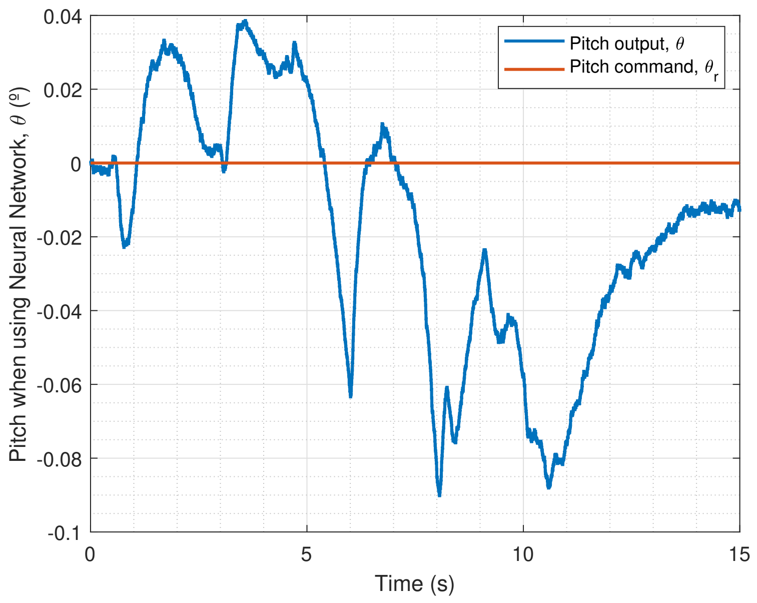

Figure 4.15: Pitch performance for the neural network control allocation (NNCA).

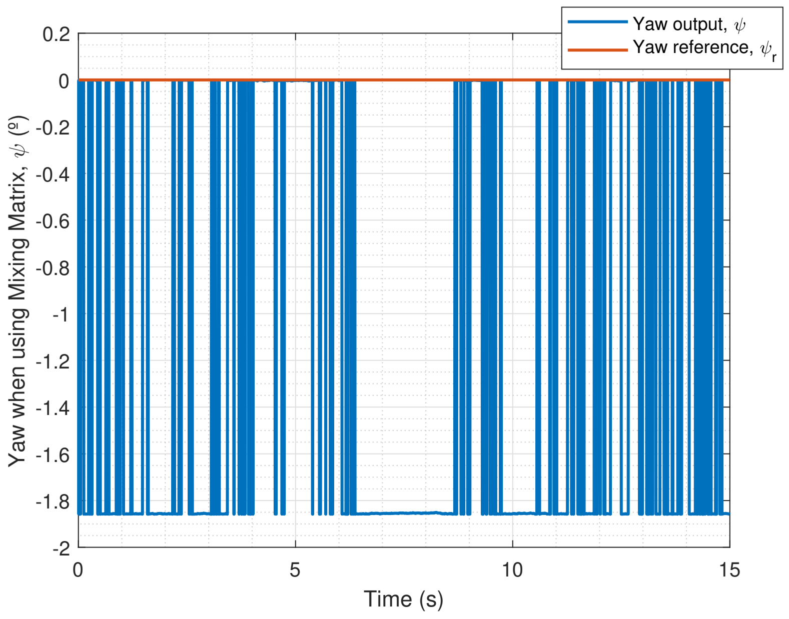

Figure 4.16: Yaw performance for the classic mixing matrix control allocation (MMCA).

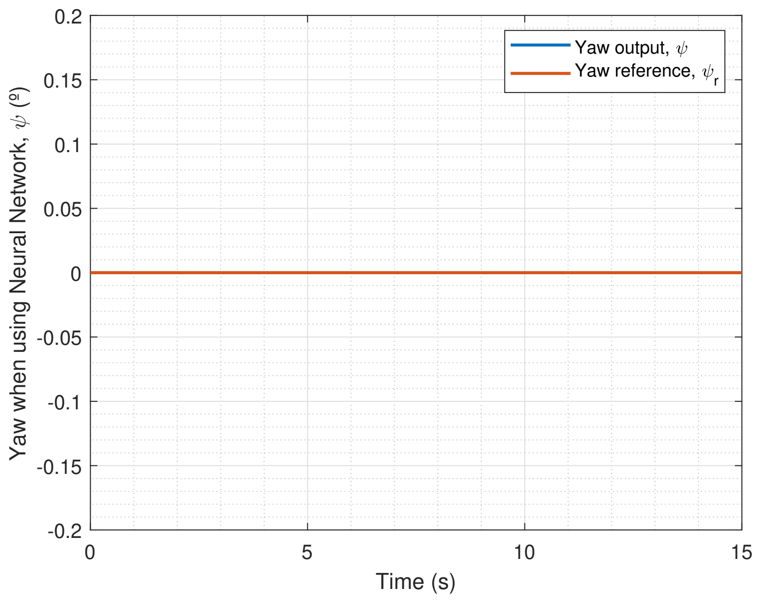

Figure 4.17: Yaw performance for the neural network control allocation (NNCA).

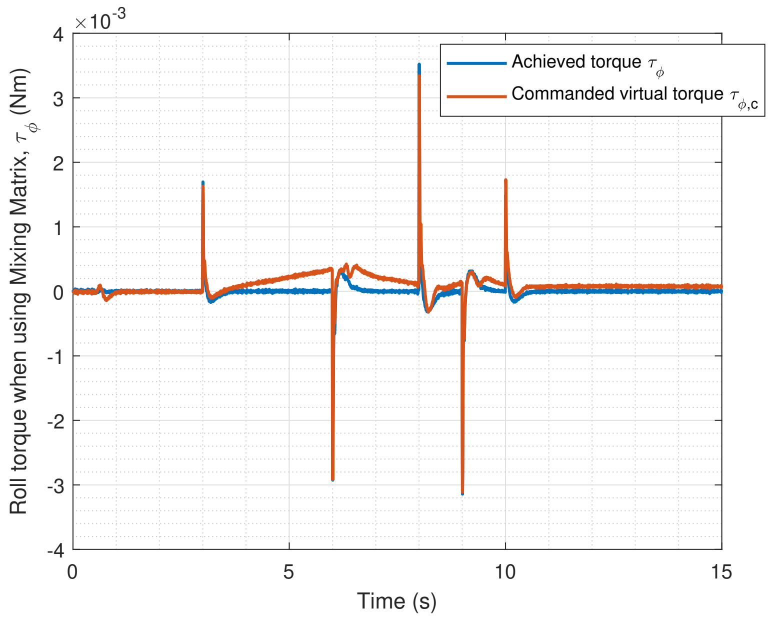

Figure 4.18: Comparison between virtual commanded torque and achieved torque for the classic mixing matrix control allocation (MMCA).

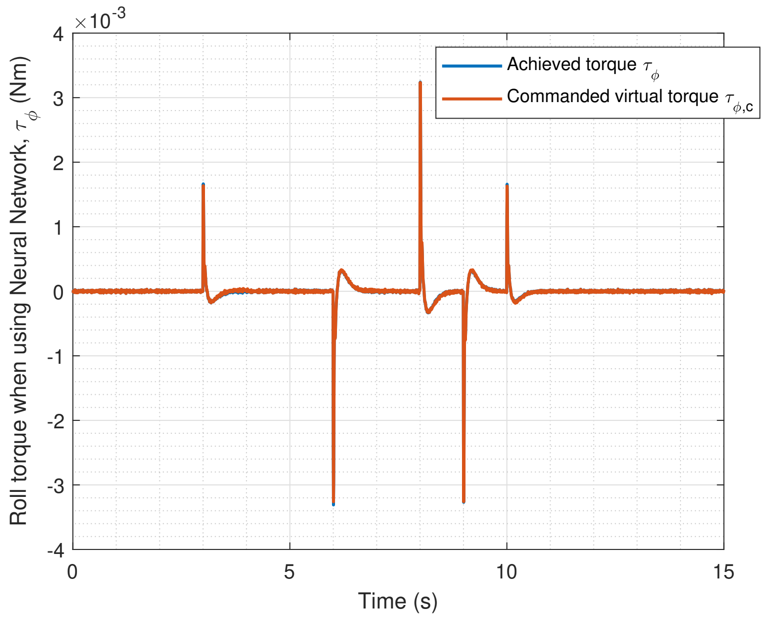

Figure 4.19: Comparison between virtual commanded torque and achieved torque for the neural network control allocation (NNCA).

4.4.2 UAV Experimental Tests

For the real flight test scenario, our position reference consisted of a set of waypoints for the quadrotor to follow. They formed a square with a 1.5 m side and a transition radius of 0.3 m to maneuver through the corners, as well as take-off and landing commands. The experiment consisted of the mini-drone taking-off in the bottom-left corner of the square. Then the UAV performs a pitch forward movement until it reaches the next corner, then a roll lateral movement, a pitch backward movement, and then again a roll lateral movement. This path is repeated once more, and then the mini-drone lands, resulting in a total of 9 waypoints [0 0 1; 1.5 0 1; 1.5 1.5 1; 0 1.5 1; 0 0 1; 1.5 0 1; 1.5 1.5 1; 0 1.5 1; 0 0 1]. Here the z-axis value 1 represents the height (1 m) at which the drone navigates along the square path. Fig. 4.20 shows the trajectory reference and output for the complete path of the waypoint following, with take-off and landing, highlighting the aircraft positioning in black, for clarity's sake. The yaw orientation reference of the quadrotor was fixed throughout the experiment.

The test scenario was planned for a mini-drone, which is a type of aircraft especially susceptible to aerodynamic effects influence, as aforementioned. The path was designed to induce mildly abrupt roll and pitch maneuvers (the corners of the square, that is, the waypoints) that would also allow an adequate observation of said effects. The behavior of the quadrotor throughout the trajectory however was not intended to be as aggressive as mentioned in previous works in the literature, because we wanted to highlight the importance of the consideration of aerodynamic effects in the control allocation even without aggressive maneuvers. Note the trajectory used in this experiment differs from the maneuvers used for the neural network training explained in Section 4.3.1, hence the training does not need to be repeated for each specific maneuver. Furthermore, Table 3.3 shows some constants relative to the quadcopter in use, which were obtained from the documentation made available by the manufacturer. These constants are used in the aerodynamic effects modeling in Table 3.2.

Fig. 4.20 shows the three-dimensional representation of the commanded and followed trajectory, with take-off and landing, for the mixing matrix and neural network control allocations. Fig. 4.21 and Fig. 4.22 show the trajectory projection in the xy-plane of the first loop for the same respective cases. We omit the aircraft's second loop through the square in Fig. 4.21 and Fig. 4.22 to facilitate visualization. We set a *transition radius* as $0.3m$, shown

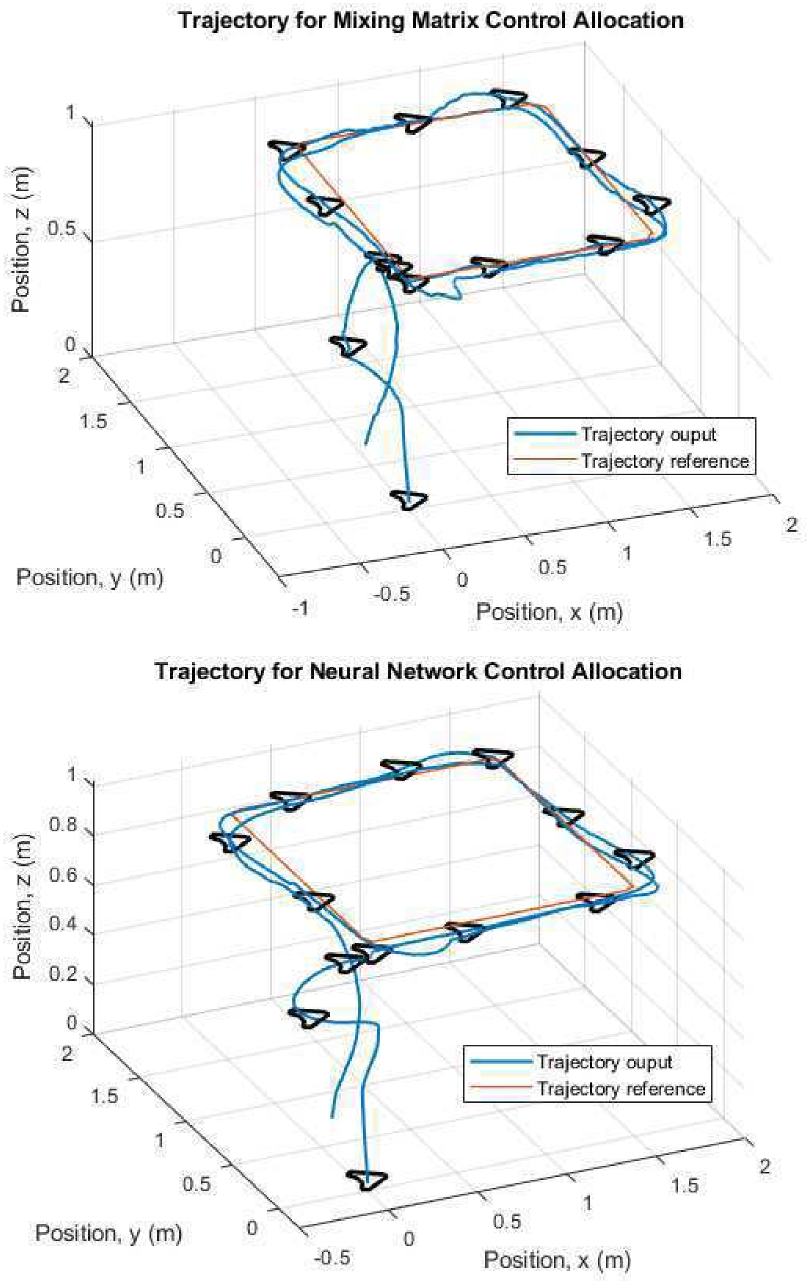

Figure 4.20: Trajectory reference and output for both loops of the waypoint following, with take-off and landing, highlighting the aircraft positioning, for the sake of clarity.

as the red circle in the figures. It means that when the drone is approaching a waypoint, this value is used as the radius around the waypoint where the drone starts moving toward the next point. If the transition radius is set as $0.3m$, and the drone is $0.3m$ away from the waypoint, the drone starts moving towards the next waypoint. The accuracy of reaching a specified waypoint increases with a decrease in the value of the transition radius. Notice that the path following is more smoothly achieved with the neural network control allocation approach, while with the mixing matrix some stuttering is present when pursuing the commanded trajectory. This is emphasized in Fig. 4.23, where some parts of the trajectory are highlighted to show the difference in path following for the mixing matrix and neural network control allocations.

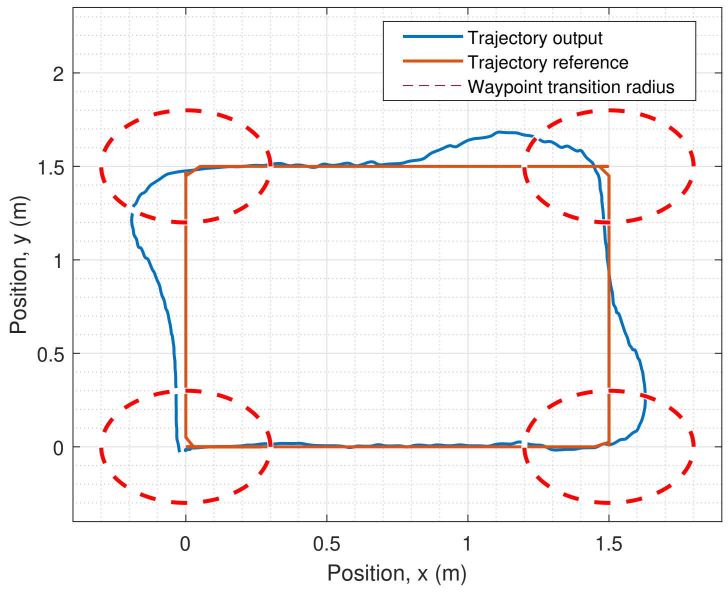

Figure 4.21: Projection (x, y) of trajectory reference and output when using the mixing matrix control allocation.

The respective mean squared error (MSE) calculations for each axis are shown in Table 4.5, for angular and linear positions. For every case except the yaw angle and position z, the MSE is smaller when using the neural network, especially in the case of the roll angle, where we had the most improvement when using the new control allocation. This could be due to the fact that the yaw and z rate were limited in the considered experiment. Also, the yaw state feedback is not as accurate as the others, because it is done using only the gyroscope, in contrast with the other angles which can rely on sensor fusion for a more precise estimation (i.e. gyroscope and accelerometer data).

Yaw control performance could be further investigated by applying more aggressive maneuvers in this axis in future work. The performed maneuvers were chosen based on what is most used in typical quadrotor flights, where yaw and height are mostly kept constant after reaching steady-state flight.

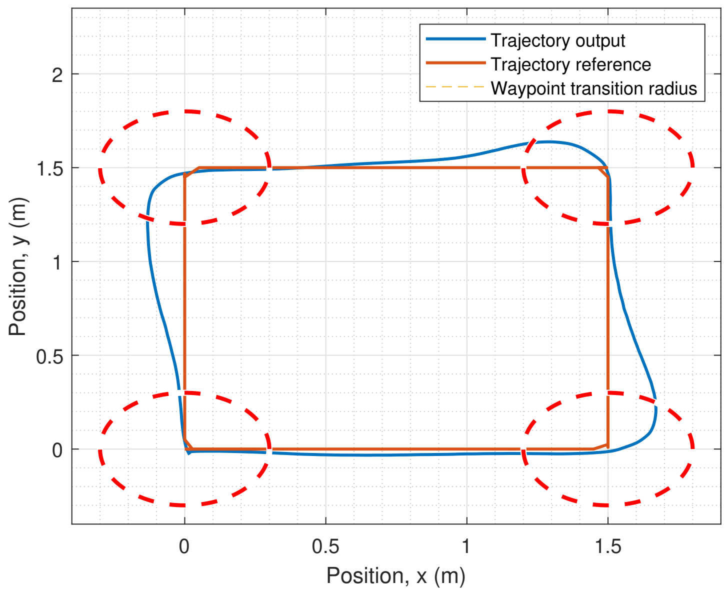

Figure 4.22: Projection (x, y) of trajectory reference and output when using the neural network control allocation.

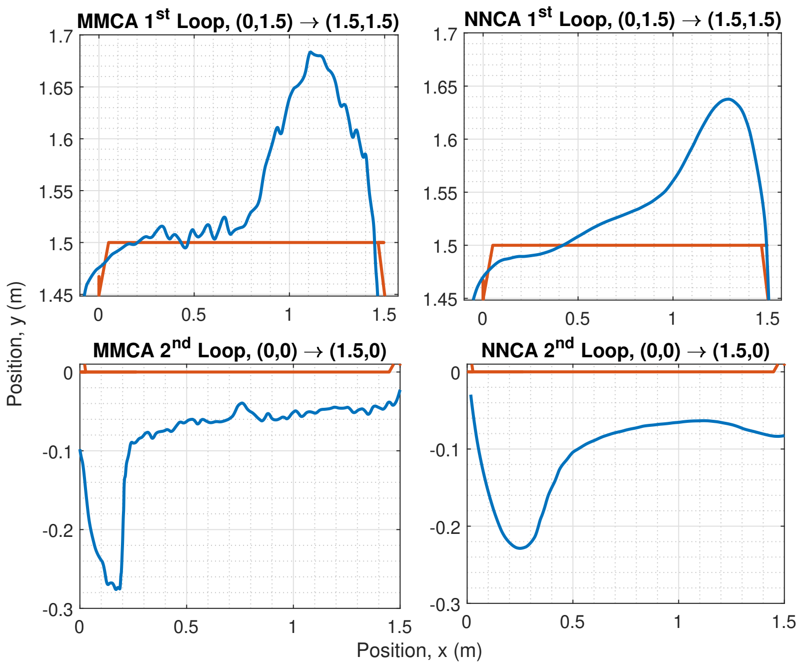

Figure 4.23: Augmented parts of the trajectory to show the difference in path following behavior for the mixing matrix (MMCA) and neural network (NNCA) control allocations.

The performance of the UAV during the roll angle maneuver of the squared path can be seen in Fig. 4.24, for the mixing matrix, and in Fig. 4.25, for the neural network control

	MSE Roll	**MSE Pitch**	**MSE Yaw**
Mixing Matrix	7.3017×10^{-4}	4.7728×10^{-4}	3.2070×10^{-6}
Neural Network	1.4444×10^{-4}	4.2851×10^{-4}	7.2123×10^{-5}
	MSE x	**MSE y**	**MSE z**
Mixing Matrix	0.0372	0.0347	0.0163
Neural Network	0.0300	0.0300	0.0176

Table 4.5: Mean squared errors for the angular and linear positions for both mixing matrix and neural network control allocations. The smallest errors are highlighted.

allocation. The mixing matrix approach shows a much noisier behavior and a higher amplitude variation when compared to the neural network control allocation approach, where the adherence of the output to the commands is much more reliable, stable, and therefore preferable for the quadcopter flight. The difference between the reference signals in both cases is due to the consideration of the aerodynamic effects in the control allocation algorithm.

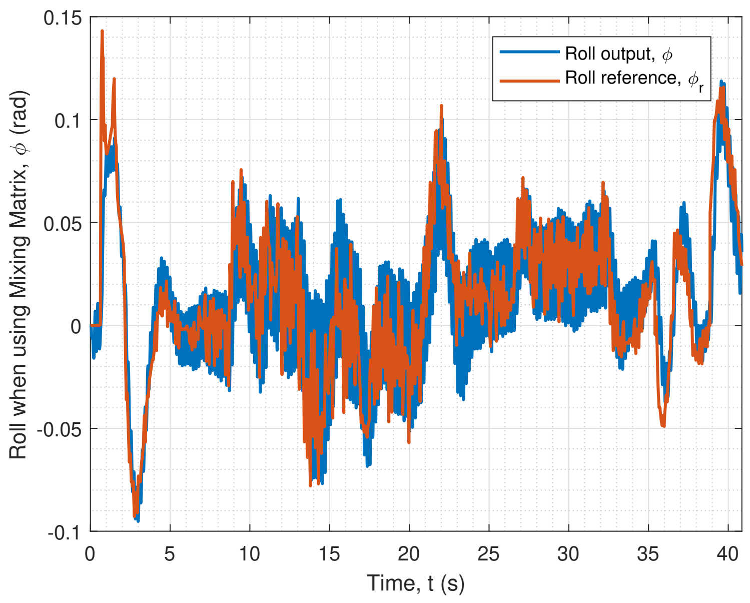

Figure 4.24: Roll commands and output when using the mixing matrix control allocation.

Fig. 4.26 and Fig. 4.27 show the comparison between the commanded virtual roll torques, which are commanded at the output of the controller. These figures also show the achieved roll torques at the aircraft, which were calculated using (3.14) with experimen-

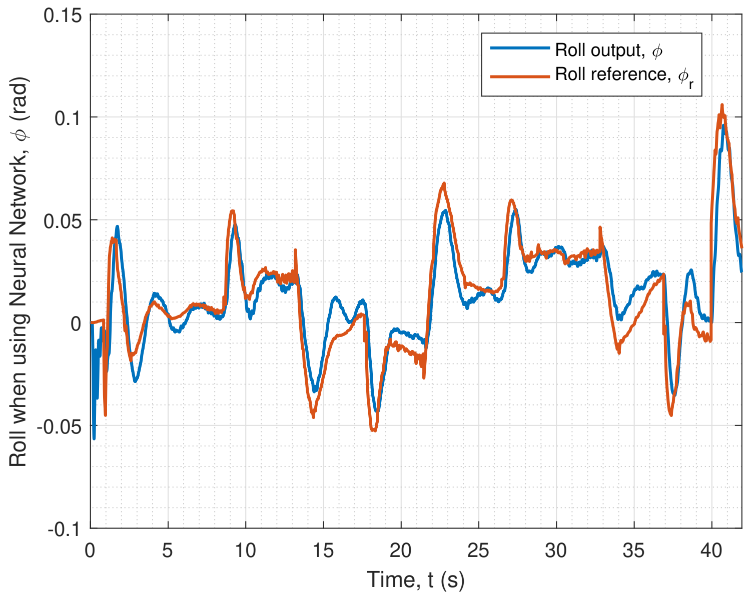

Figure 4.25: Roll commands and output when using the neural network control allocation.

tal flight data. Fig. 4.26 refers to the experiment with the mixing matrix, while Fig. 4.27 uses the neural network control allocation. The amplitude of the roll torques when using the neural network is smaller, requiring less effort from the quadrotor. Also, a comparison between the allocation errors for both cases is shown in Figures 4.28, 4.29 and 4.30 proving that the neural network allocation considering aerodynamic effects significantly reduces the allocation error.

Likewise, Fig. 4.31 and Fig. 4.32 present the corresponding comparison for commanded virtual thrusts and achieved thrusts, for mixing matrix and neural network control allocations, respectively. We note a similar behavior as the roll torque case mentioned above. There are fewer noisy oscillations when using the neural network, as well as smaller amplitudes. Fig. 4.33 shows the allocation errors for the two methods, where the error when applying the neural network control allocation is several times smaller (in the order of 10^{-1}) than the traditional alternative with the mixing matrix.

Finally, Fig. 4.34 and Fig. 4.35 show the thrust $\mathbf{T}^B$ components on the x and y axes, T_x and T_y, during the quadrotor flight when using the mixing matrix, which considers these components to be zero or negligible. Not only these components do exist, but they are not of negligible order when compared to the order of the system torques they have an influence on. For instance, Fig. 4.35 shows T_y which has an influence on the roll torque, as in (3.20).

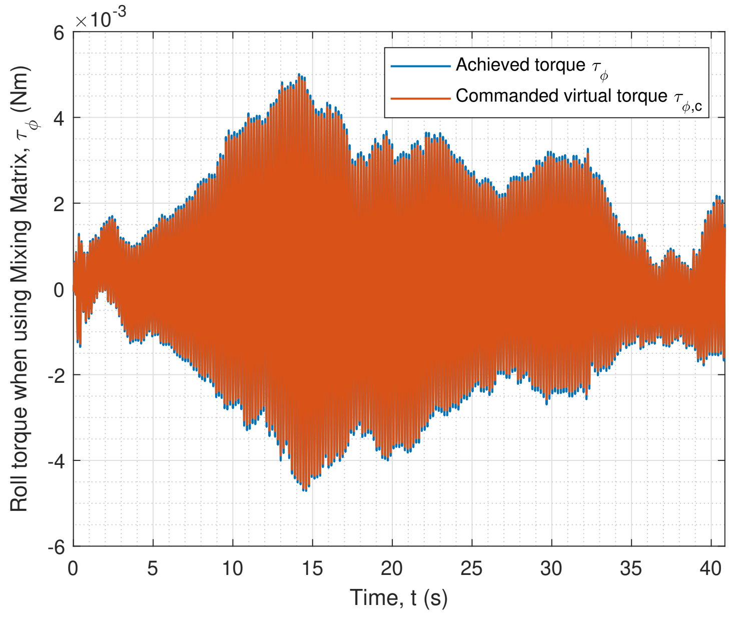

Figure 4.26: Comparison between commanded roll torques at the control output and those achieved at the plant for the mixing matrix control allocation.

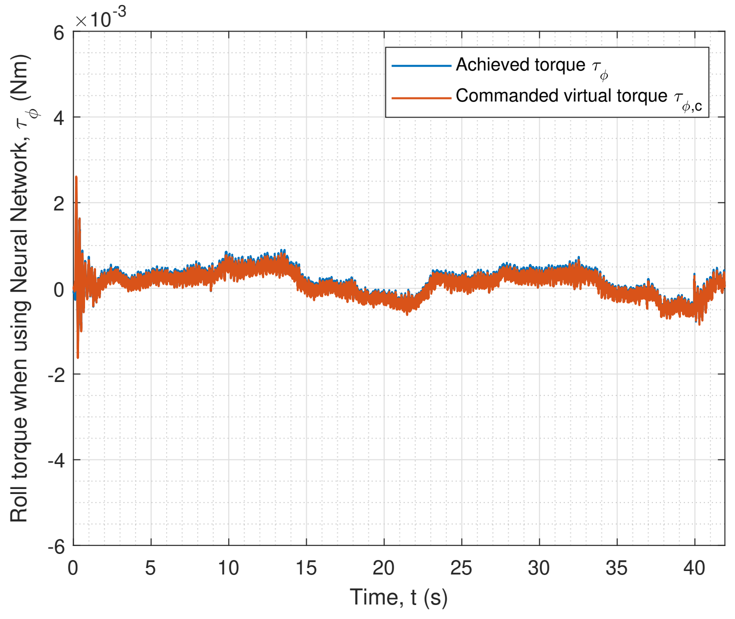

Figure 4.27: Comparison between commanded roll torques at the control output and those achieved at the plant for the neural network control allocation.

Note that the behavior of the roll torques seen in Fig. 4.26 greatly resembles that of T_y in Fig. 4.35.

Measures were also taken during the flight experiments to minimize the error due to

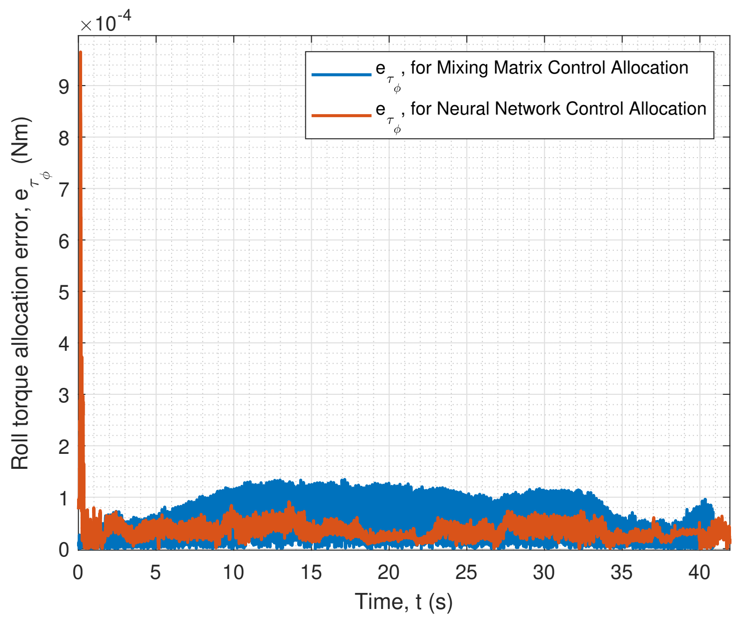

Figure 4.28: Comparison between the allocation errors for the achieved roll torques (τ_ϕ) at the plant for both the mixing matrix (in blue) and neural network (in red) control allocations.

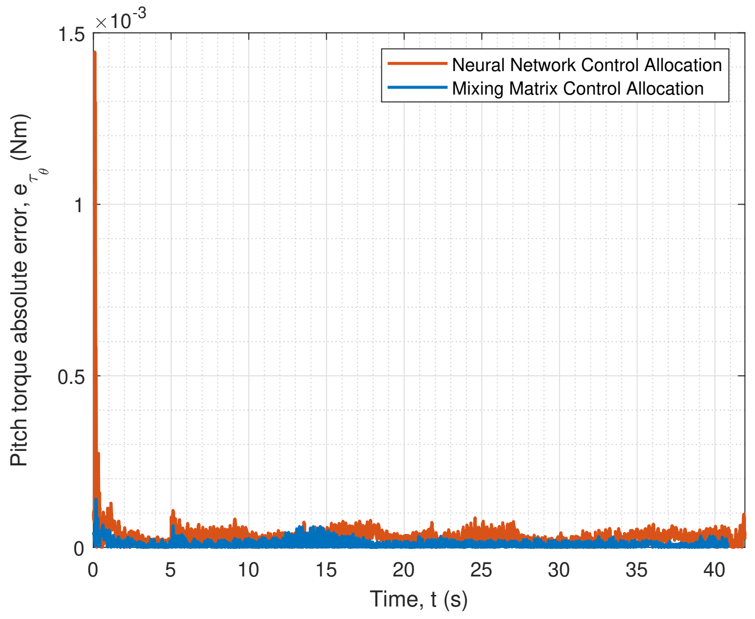

Figure 4.29: Comparison between the allocation errors for the achieved pitch torques (τ_θ) at the plant for both the mixing matrix (in blue) and neural network (in red) control allocations.

nonlinear uncertainties, since they were performed in a controlled indoor environment. The uncertain nonlinear terms were not directly treated in the modeling presented in this paper, as it was not the focus of this work. However, the practical results discussed here already

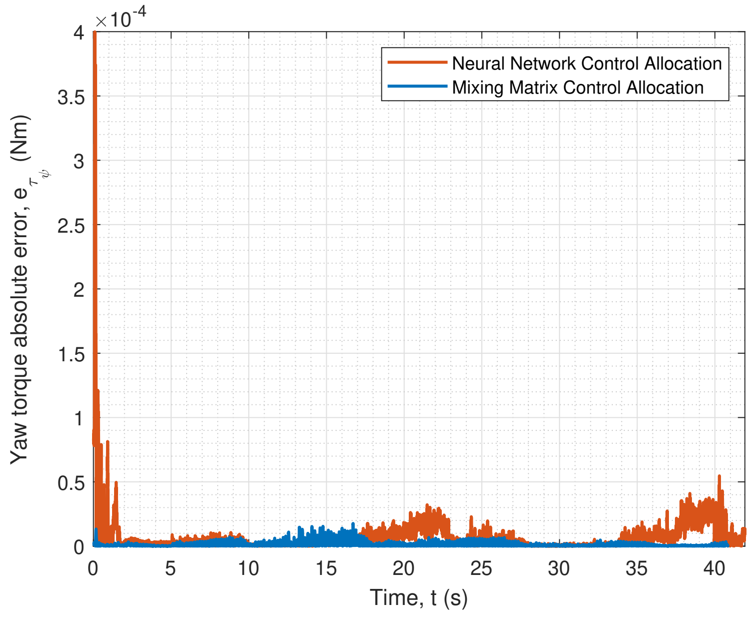

Figure 4.30: Comparison between the allocation errors for the achieved yaw torques (τ_ψ) at the plant for both the mixing matrix (in blue) and neural network (in red) control allocations.

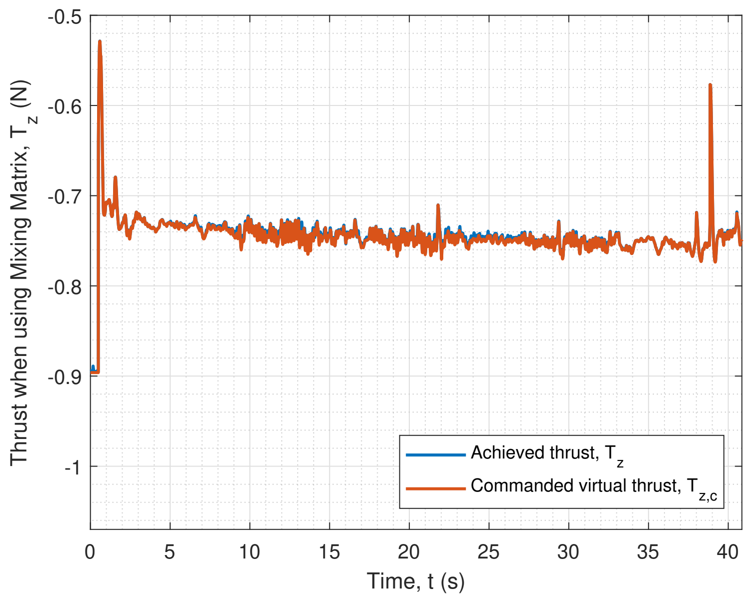

Figure 4.31: Comparison between the commanded thrust at the control output and that achieved at the plant for the mixing matrix control allocation.

show the advantages of the proposed method. In future work, an approach considering these uncertainties could be included.

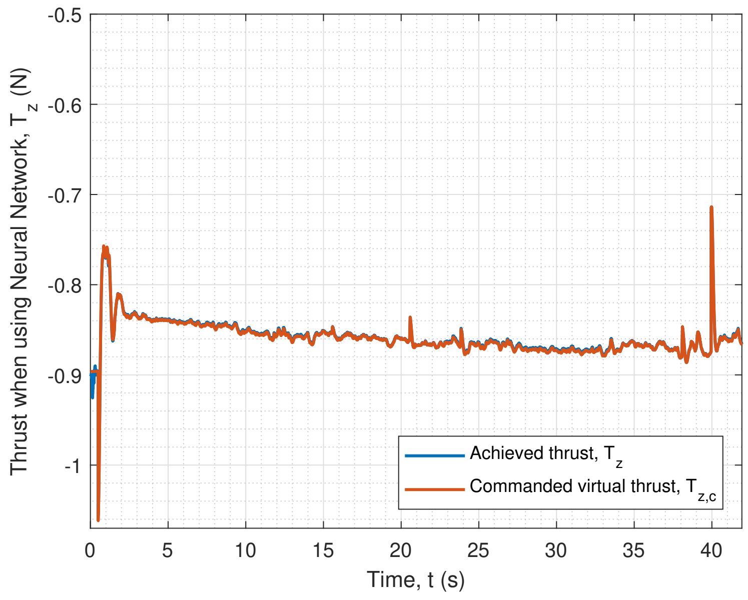

Figure 4.32: Comparison between the commanded thrust at the control output and that achieved at the plant for the neural network control allocation.

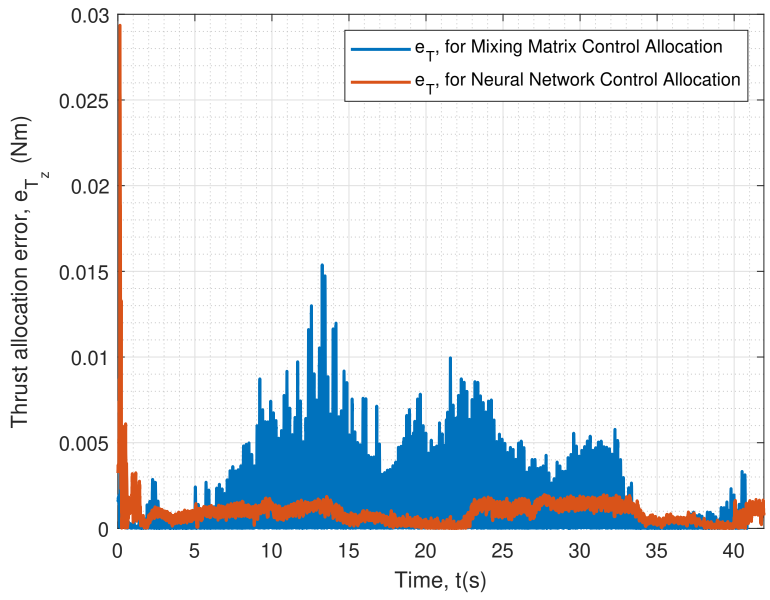

Figure 4.33: Comparison between the allocation errors for thrusts, in the z-axis (T_z), achieved at the plant for both the mixing matrix (in blue) and neural network (in red) control allocations.

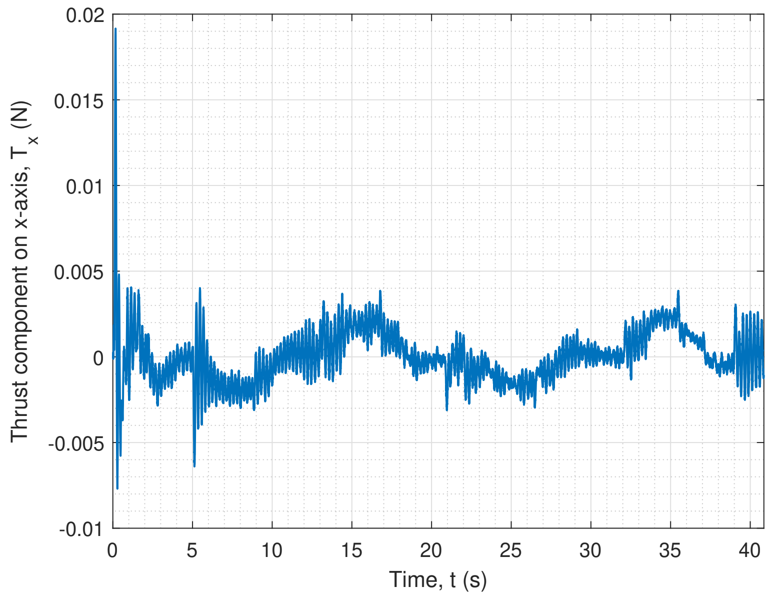

Figure 4.34: Thrust component in the x-axis, T_x, when using the mixing matrix control allocation.

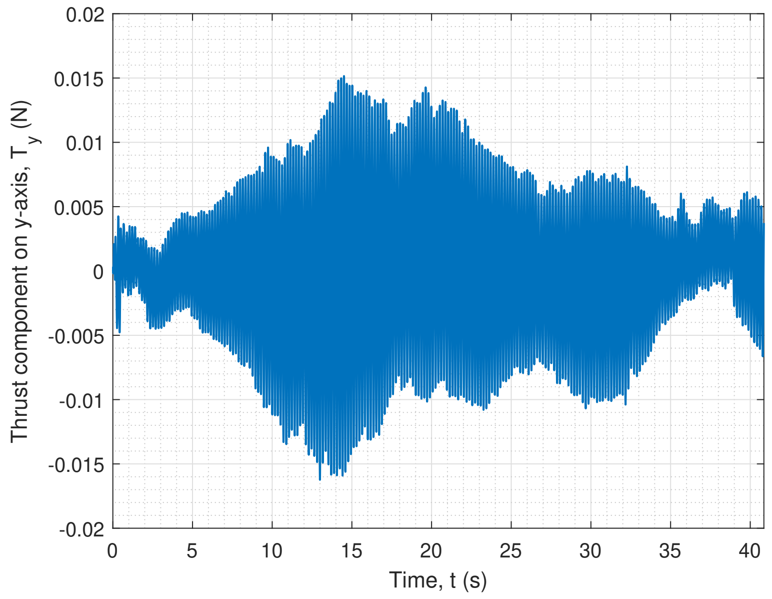

Figure 4.35: Thrust component in the y-axis, T_y, when using the mixing matrix control allocation.

4.5 Final Considerations

In this Chapter, the basic cascade control structure for a quadrotor UAV has been presented, and the necessity of the control allocation step has been shown. In order to take into consideration the aerodynamic effects, it has been concluded that the classic mixing matrix is not accurate, since it carries an allocation error. In this sense, we proposed that the aerodynamic effects could be taken into account by applying a more refined control allocation process that used a function-fitting neural network to transform virtual control inputs into actuator commands.

The network training was performed offline, requiring very low computational power when compared to other optimization-based control allocation algorithms. The neural network required a two-step training process that was independent of the classic simplified allocation method on its second step. The first training was performed in an open-loop system, via PRBS pulses. A second training was executed with the purpose of refining the network in an equivalent closed-loop system, and in this case, the allocation was implemented using the open-loop trained neural network alongside the PD-PID error controller.

The real flight results showed that the aircraft behavior was smoother when using the neural network control allocation approach, the allocation error was smaller and the signals were less noisy, that is, the consideration of aerodynamic effects in the system interfered directly and positively with the quadrotor performance. Finally, the performance of the controller also improved with the use of the neural network control allocation, as well as the quality of the thrust and torque signals.

Chapter 5

Fault Detection and Diagnosis

In this chapter, we propose an approach for estimating the loss of effectiveness of UAV actuators while also considering the presence of the previously discussed aerodynamic effects.

In Sections 5.1 and 5.1.1, some basic concepts for the subject will be presented, including definitions and the main types of fault that can affect a quadrotor. Section 5.2 will address how the modeling of the loss-of-effectiveness (LOE) actuator faults is done throughout this thesis. In Section 5.3, we present the developed fault detection and diagnosis scheme based on a Reduced Order Extended Kalman Filter Tolerant to Aerodynamic Effects (ROEKF-TAE) capable of detecting LOE faults in more than one actuator at the same time.

Simulation and experimental results are shown in Sections 5.4.1 and 5.4.2, where the same simulation tool as mentioned earlier was used, as well as the PARROT Mambo micro drone platform for flight tests. The results are compared to those of a traditional EKF, as well as to those of a state-of-the-art adaptive Kalman Filter available in the literature. Final considerations are discussed in Section 5.5.

The scientific paper that gathered the contributions of this Chapter can be found in [Madruga et al. 2023].

5.1 Basic Concepts

When dealing with UAVs, safety should be a major concern, since crashing a drone can endanger infrastructure or harm people, which is important in the case of missions such as package delivery or image capturing [Sun 2020]. Therefore, it is important to know when a fault occurs in the system, even if it is not major yet, since it could lead to a total failure. There are several types of faults that will be described later on, but we highlight beforehand the fact that the failure of the rotor system is a fundamental problem to address [Sun 2020].

UAVs with vertical take-off and landing are designed with several rotors with fixed-pitch blades as actuators of the system. The rotor system generates thrust lift and control torques. In this sense, rotor failures threaten the flight because they can highly change the flight dynamics. The most popular type of multi-rotor drone is especially vulnerable to rotor (or actuator) failures because it does not have rotor redundancy since a minimum of four actuators is required to maintain full control authority of thrust and three-axis attitude of a multi-rotor drone [Sun 2020]. Besides, adding more rotors implies higher costs, more complexity, lower energy efficiency, and more potential failure modes.

To clarify the proper definitions [Ducard 2009]:

- A **Fault** is defined as a deviation of one or more characteristics of the system from its usual standard condition. It is an abnormal behavior of the system, even if apparently it does not affect the overall functioning, but it may eventually lead to complete failure. A fault may be small and hidden, hard to detect and estimate.

- A **Failure** is the permanent interruption of the system's ability to perform a required task. It may result from one or more faults, it terminates the operation of a unit in the system. For example, an actuator is considered failed if it can no longer be used in a controlled manner.

When dealing with faults in a system, there is usually a module called FDI (Fault Detection and Isolation/Identification) or FDD (Fault Detection and Diagnosis) to monitor the aircraft components' status. The FDI/FDD system informs another supervision module of the existence or severity of the fault and then the supervisor can decide which action to take regarding it. In this Chapter, we focus on developing an FDD system capable of **detecting**, **estimating**, and **locating** an actuator fault, meaning it can **detect its occurrence**, **estimate its intensity** and **locate the faulty actuator(s)**. In order to avoid possible performance issues and misidentification of faults for other disturbances, the aerodynamic effects modeling will be included in the FDD system formulation.

5.1.1 Fault Types

There are several places where a fault can affect the aircraft, from structural to sensing damage. Here we define the three main types of fault:

- **Sensor fault:** a multi-rotor UAV depends on several sensors in order to fly. accelerometers, barometers, gyroscopes for odometry, as well as cameras and/or GPS for location. A faulty sensor leads to errors either by sending false information or no information at all. One may deal with this kind of fault by having a redundant sensor or by reconstructing the absent measurement with knowledge of the plant and data of the remaining sensors.

- **Actuator fault:** we call actuator, in the case of multi-rotor aircraft, the motor + propeller set, also called the rotor. An actuator fault can be caused by propeller damage, electric failure, or mechanical disturbances. The flight performance is instantaneously affected, and immediate actions must be taken in order to maintain vehicle and flight integrity.

- **Structural damage:** any other structural damage that is not covered by actuator faults, though not all structural damage is as critical. The same observations made for actuator faults are valid here.

5.1.2 Types of Actuator Faults

Given that in this work we focus specifically on actuator faults, here are the main forms in which an actuator might be compromised in a multirotor UAV:

- **Lock-in-place:** during a Lock-in-place fault, the actuator "freezes" at a certain condition and does not respond anymore to control commands.

- **Hard-over:** the Hard-over fault is characterized by the actuator only assuming its maximum and minimum position (in our case, velocities), regardless of the subsequent commands. The response is only limited by motor saturation.

- **Float:** a Float fault occurs when the actuator has a change of moment ($\delta\tau$ or δ T) equal to zero and does not contribute to achieving the controller commands, that is, it "floats" permanently in the state it was when the fault happened.

- **Loss-of-effectiveness/Loss-in-effectiveness (LOE/LIE):** a Loss-of-effectiveness fault is characterized by a reduction in the actuator gain when compared to its nominal value. In our case, i.e. multi-rotor UAVs, that means a decrease in the effectiveness of generating the thrust and torque mechanical conjugates.

The different types of actuator faults mentioned are shown in Figure 5.1. We assume, for the sake of simplicity, that the transfer functions of the actuators are equal to one.

In Table 5.1 below, t_{Fi} represents the instant in time when the fault occurs; $\alpha^i \in [\overline{\alpha}^i, 1]$ is the effectiveness factor, or fault factor; and $\overline{\alpha}^i > 0$ denotes the minimum effectiveness that maintains the flight possible.

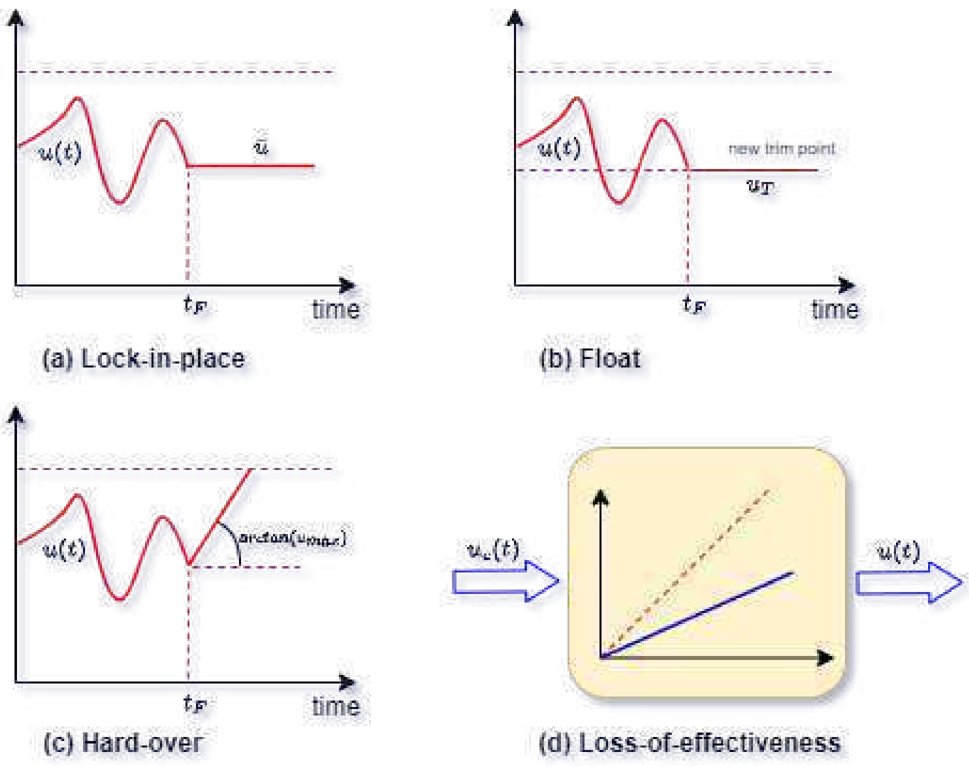

Figure 5.1: Fault types. [Bošković & Mehra 2003].

$u_i(t) =$	$u_c(t)$	$\alpha^i(t) = 1$	for all $t \geq 0$	**NO FAILURE**
	$\alpha^i(t)u_c(t)$	$0 < \overline{\alpha}^i \leq \alpha^i(t) < 1$	for all $t \geq t_{Fi}$	**Loss-of-effectiveness**
	0	$\alpha^i(t) = 0$	for all $t \geq t_{Fi}$	**Float**
	$u_{ci}(t_{Fi})$	$\alpha^i(t) = 0$	for all $t \geq t_{Fi}$	**Lock-in-place**
	$(u_{i_{MIN}})$ **or** $(u_{i_{MAX}})$	$\alpha^i(t) = 0$	for all $t \geq t_{Fi}$	**Hard-over**

Table 5.1: Mathematical characterization of possible fault types.

The fault detection system must be robust against other external disturbances, model uncertainties and sensing noise. Namely, it should not provide false alarms while still being sufficiently sensitive to detect small faults before they lead to failure.

The flight in certain conditions might affect the detection of loss-of-effectiveness faults as a result of the presence of aerodynamic effects, since they naturally attenuate thrust and torque generation by the rotors, even if no fault has occurred. As previously stated in this research, this happens mostly in high-speed conditions and with small aircraft. A challenge of the current state-of-the-art is to validate fault detection schemes that are efficient under these conditions [Sun et al. 2018].

5.2 Modeling of LOE Actuator Faults

In this thesis, the actuator consists of the motor and propeller set, as shown in Figure 3.1. If the propeller is damaged, or if the motor is prevented from spinning as expected, the actuator may not reach the desired rotation speed. Consequently, the desired thrust cannot be generated [Madruga et al. 2022]. From now on, we will focus on characterizing and modeling loss-of-effectiveness faults.

As established, this section will deal with the modeling of loss of effectiveness actuator faults which are recognized as the most common type of fault [Zhong et al. 2018]. The modeling of this type of fault is made with the inclusion of a proportional fault factor to the actuator's rotation speed, and this factor is treated as a stochastic process to be estimated by the FDD. This approach is similar to what can be found in the state-of-the-art literature, as in [Avram, Zhang & Muse 2017], [Baldini et al. 2020], [Moghadam & Caliskan 2015], [Schijndel, Sun & Visser 2021], [Zhong et al. 2018]. Here this model will be used in the

FDD estimator, along with the consideration of the relevant aerodynamic effects in order to detect, localize and estimate the fault intensity.

It is reasonable to assume that the vehicle takes off with "healthy" actuators since tests can easily be performed before a mission to guarantee so. A loss of effectiveness fault can be caused by propeller damage, rust, minor electrical faults, lack of lubrication etc. All of these result in the expected thrust being attenuated because the actuator can't achieve the proper rotation speed. However, coming from different sources, it is difficult to predict the appropriate behavior of this fault factor. This motivates the stochastic nature of the LOE fault factor modeling.

Definition 6. $\mathbf{u} = \begin{bmatrix}\omega_1 & \omega_2 & \omega_3 & \omega_4\end{bmatrix}^T \in \mathbb{R}^4$ *is the vector of actuator velocities sent as actuator commands. These velocities should produce the desired resultant thrust and torques.*

Definition 7. *Let* α^i*,* $i = 1, 2, 3, 4$*, represent the loss of control effectiveness factor or, for the sake of simplicity, the fault factor. If a fault occurs in the* ith *actuator, the* ith *control input can be represented as:*

$$\omega_i^f = \alpha^i \omega_i, \tag{5.1}$$

where ω_i *represents the expected rotation speed for the actuator, and* ω_i^f *is the achieved velocity due to the occurrence of a fault.*

These types of faults are defined as *partial loss of control effectiveness* and are recognized as the most common type of actuator fault [Zhong et al. 2018].

Remark 5. α^i *is bounded as* $0 < \overline{\alpha}^i \leq \alpha^i \leq 1$*, where* $\overline{\alpha}^i$ *represents the lower bound that keeps the quadrotor controllable. If* $\alpha^i < \overline{\alpha}^i$*, the aircraft will become uncontrollable and it may result in a flight crash [Zhong et al. 2018].*

Here, $\alpha^i = 1$ represents the normal expected behavior of the actuator, and $\alpha^i < 1$ means that there is a loss-of-effectiveness fault in the ith actuator. $\alpha^i = 0$ represents the extreme case where the ith actuator stops completely, which is out of the scope of this thesis. That is because when a single rotor fails completely on a quadrotor UAV, the whole physics of the flight changes, and the controller needs to be altered in order to maintain flight, due to the lack of rotor redundancy. There is a limit to the fault factor value that keeps the drone in flight without changing the controller technique to a fault-tolerant one, which in turn changes the scope of the problem completely. Since in this work we focus only on fault detection, not on the controller, we chose α^i values that could represent a significant fault without jeopardizing the flight of the UAV. Nevertheless, we can indeed use our approach to fault detection when a single rotor completely fails, provided that we have a fault-tolerant controller to maintain flight. Furthermore, the dynamics of actuator faults are typically unknown, but they can be modeled as stochastic processes:

$$\alpha^i_{k+1} = \alpha^i_k + w^\alpha_k \tag{5.2}$$

where w^α_k is a zero-mean white noise sequence satisfying the condition that will be shown in (5.5a).

Remark 6. *One could argue that, since a change in the velocity ω_i of the actuators results in a correspondent change in the final thrust and torques that ultimately make the quadrotor fly, the fault modeling shown in (5.1) could be made in terms of the aforementioned mechanical torques instead of ω_i. However, associating an individual fault factor to each ω_i and therefore to each motor allows us to locate the faulty actuator using the changes in its correspondent fault factor α^i.*

Remark 7. *Take-off and landing are omitted from the figures of the simulation and real UAV flight experiment results since they consist of transient behavior that might make the readability of the graphs difficult. Furthermore, the FDD method here proposed is designed to work in steady state which comprises most of the flight.*

With the LOE fault modeling in mind, one might need to revisit the quadrotor model from Chapter 3, where all actuator velocities will be attenuated by the fault factor, so Table 3.2 is modified and rewritten as Table 5.2 below. Note that this implies changes in the thrusts $\mathbf{T^{B,i}}$ and torques Q_i.

Once more, $\mathbf{r}^{\mathrm{B}}_{p,i}$ refers to the i^{th} propeller position with respect to the center of mass of the UAV, θ_{b0} is the equivalent blade pitch at the rotor axis, and θ_{b1} is the washout of a linear twist blade [Pounds, Mahony & Corke 2010]. The constant γ_{aero} is the non-dimensional

Relative air speed at propeller i: $\mathbf{v}^{\mathrm{B}}_{ra,i}$	$\Omega \times \mathbf{r}^{\mathrm{B}}_{p,i} + \mathbf{R}^{-1}_{li}\mathbf{v}$
Planar components: μ_i	$\sqrt{{v^{\mathrm{B}}_{x,ra,i}}^2 + {v^{\mathrm{B}}_{y,ra,i}}^2} / \left\lvert \alpha^i \omega_i \frac{d}{2} \right\rvert$
Non-dimensionalized normal inflow: l_i	$\lvert v^{\mathrm{B}}_{z,ra,i} \rvert / \left\lvert \alpha^i \omega_i \frac{d}{2} \right\rvert$
Sideslip azimuth relative to x-axis x: j_i	$\arctan \frac{v^{\mathrm{B}}_{x,ra,i}}{v^{\mathrm{B}}_{y,ra,i}}$
Sideslip rotation matrix: $\mathbf{J}_{ss,i}$	$\begin{bmatrix} \cos(j_i) & -\sin(j_i) \\ \sin(j_i) & \cos(j_i) \end{bmatrix}$
Longitudinal flapping: $\beta_{lf,i}$	$\mathbf{J}^{\mathrm{T}}_{ss,i} \begin{bmatrix} ((\frac{8}{3}\theta_{b0} + 2\theta_{b1}) - 2l_i)/(1/\mu_i - \mu_i/2) \\ 0 \end{bmatrix}$
$a_{1,s,i}$	$\beta_{x,lf,i} - 16Q/(\gamma_{aero} \lvert \alpha^i \omega_i \rvert)$
$b_{1,s,i}$	$\beta_{y,lf,i} - 16P/(\gamma_{aero} \lvert \alpha^i \omega_i \rvert)$

Table 5.2: Aerodynamic effects on total thrust $\mathbf{T}^{\mathrm{B}}_i$ considering the possibility of LOE actuator faults.

Lock Number [Leishman 2006]. Here, α^i, $i = 1, 2, 3, 4$, is the fault factor applied to ω_i with the goal of modeling LOE actuator faults. If $\alpha^i = 1$, the actuator is healthy and the Table 5.2 is again reduced to Table 3.2.

As mentioned in Chapter 2, to the extent of this research, there has been no solution to fault detection and diagnosis taking aerodynamic effects into account, especially with real flight experiment validation. These available solutions could then be subject to false positives in the fault detection system, since the disturbance of a fault, depending on its intensity, might be confused with aerodynamic effects and vice-versa. This can happen because, as seen in Chapter 3, the blade flapping tilts the thrust force, making available to the quadrotor upwards movement only a component of that original vector. That is, it *attenuates* the thrust force, which is also what an LOE fault does. So it is important not to mistake a real fault for a natural aerodynamic behavior in a healthy motor.

5.3 Fault Detection and Diagnosis Estimator

As previously stated, our estimator is based on an Extended Kalman Filter. Once more, according to [Okada et al. 2021], Kalman filter-based techniques provide a low false alarm rate, a short delay in detecting the fault, robustness with respect to model uncertainties, isolation of simultaneous faults, and the possibility of fault estimation. Besides, for the case of multiplicative faults, as in the scope of this thesis, the use of parameter estimation techniques stands out. In Chapter 2, Table 2.1 makes a summary of the comparison between other types of solutions and the one here proposed.

Let us first consider the quadrotor model described in equations (3.5) to (3.8) and (3.12) to (3.14). If we allow for it to be affected by actuator faults as modeled in Section 5.2, we will have $\alpha^i \neq 1$. In order to detect and estimate the faults (i.e. the fault factors α^i), we propose a Reduced Order Extended Kalman Filter Tolerant to Aerodynamic Effects (ROEKF-TAE), where the fault factors α^i are considered as augmented states of the system, so that they can be estimated. A fault in the ith actuator can be detected and located by monitoring changes in its α^i, while the estimation of α^i's value indicates the intensity of that loss-of-effectiveness. The ROEKF-TAE main contribution is its capability of detecting, localizing and estimating simultaneous LOE actuator faults while considering the presence of aerodynamic effects and consequently not mistaking them for LOE faults, as will be verified by real flight experiments in Section 5.4.2.

In this context, the augmented system state vector to be estimated is given by

$$\mathbf{x} = \begin{bmatrix} P & Q & R & U & V & W & \alpha^1 & \alpha^2 & \alpha^3 & \alpha^4 \end{bmatrix}^{\mathrm{T}},$$

and thus, from (3.6) and (3.8) and considering the fault factors α^i as in Section 5.2, we derive the following nonlinear model for designing the ROEKF-TAE estimation:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \mathbf{y} = \mathbf{C}\mathbf{x}, \tag{5.3}$$

where

$$\mathbf{f}=\begin{bmatrix} \frac{J_y-J_z}{J_x}RQ+\frac{\tau_\phi}{J_x} \\ \frac{J_z-J_x}{J_y}RP+\frac{\tau_\theta}{J_y} \\ \frac{J_x-J_y}{J_z}QP+\frac{\tau_\psi}{J_z} \\ -g(\sin\theta)+\frac{T_x}{m} \\ g(\cos\theta\sin\phi)+\frac{T_y}{m} \\ g(\cos\theta\cos\phi)+\frac{T_z}{m} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \mathbf{C}=\begin{bmatrix} 1&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0 \end{bmatrix}.$$

Remark 8. *Note that the terms related to θ and ϕ in this model are treated as time-varying quantities, measured by sensors, while the fault factors α^i, $i = 1, 2, 3, 4$ and $\mathbf{u}$ are embedded as parts of the mechanical torques τ_ϕ, τ_θ, τ_ψ, T_x, T_y, T_z which are calculated by means of (3.12), (3.13) and (3.14), and thus are omitted here for the sake of readability.*

Furthermore, the ROEKF-TAE only uses the state equations for P, Q, R, U, V, W, and α^i shown in (5.3) and does not require all the twelve states in (3.5) to (3.8) to estimate the fault factors α^i.

5.3.1 Filter Equations

For the sake of simplicity, let the aforementioned system be written as:

$$\mathbf{x}_{k+1} = \mathbf{F}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k, \tag{5.4a}$$

$$\mathbf{y}_k = \mathbf{C}\mathbf{x}_k + \nu_k. \tag{5.4b}$$

where $\mathbf{F}(\mathbf{x}_k, \mathbf{u}_k)$ is the discrete-time approximation of the continuous-time equations given in (5.3), and $\mathbf{w}_k = \begin{bmatrix}\mathbf{w}_k^x & \mathbf{w}_k^\alpha\end{bmatrix}^{\mathrm{T}}$ and ν_k are uncorrelated zero-mean white Gaussian noise sequences with covariances $\mathbf{Q}_k$ and $\mathbf{R}_k$, respectively. The noise sequences $\mathbf{w}_k^x$, $\mathbf{w}_k^\alpha$ and ν_k satisfy the following [Zhong et al. 2018]:

$$E\left\{\begin{bmatrix}\mathbf{w}_k^x\\ \mathbf{w}_k^\alpha\\ \nu_k\end{bmatrix}\begin{bmatrix}\mathbf{w}_l^x\\ \mathbf{w}_l^\alpha\\ \nu_l\end{bmatrix}^{\mathrm{T}}\right\} = \left[\begin{bmatrix}\mathbf{Q}_k^x & 0 & 0\\ 0 & \mathbf{Q}_k^\alpha & 0\\ 0 & 0 & \mathbf{R}_k\end{bmatrix}\right], \tag{5.5a}$$

$$\mathbf{Q}_k = \begin{bmatrix}\mathbf{Q}_k^x(\cdot) & 0_{6\times 4}\\ 0_{4\times 6} & \mathbf{Q}_k^\alpha(\cdot)\end{bmatrix}, \tag{5.5b}$$

where E is the expectation function.

The ROEKF-TAE equations are shown below. They can be used for state estimation of discrete-time nonlinear systems in this form. For the filter implementation, the aforementioned equations (5.3) were discretized using the Forward Euler method.

$$\hat{\mathbf{x}}_{k+1|k} = \mathbf{F}(\mathbf{x}_{k|k}, \mathbf{u}_k) \tag{5.6a}$$

$$\mathbf{P}_{k+1|k} = \mathbf{A}_k\mathbf{P}_{k|k}\mathbf{A}_k^{\mathrm{T}} + \mathbf{Q}_k \tag{5.6b}$$

$$\hat{\mathbf{y}}_{k+1|k} = \mathbf{C}\hat{\mathbf{x}}_{k+1|k} \tag{5.6c}$$

$$\mathbf{K}_{k+1} = \mathbf{P}_{k+1|k}\mathbf{C}^{\mathrm{T}}(\mathbf{C}\mathbf{P}_{k+1|k}\mathbf{C}^{\mathrm{T}} + R_{k+1})^{-1} \tag{5.6d}$$

$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + \mathbf{K}_{k+1}(\mathbf{y}_{k+1} - \hat{\mathbf{y}}_{k+1|k}) \tag{5.6e}$$

$$\mathbf{P}_{k+1|k+1} = (\mathbf{I} - \mathbf{K}_{k+1}\mathbf{C})\mathbf{P}_{k+1|k} \tag{5.6f}$$

$$\mathbf{y}_{k+1|k+1} = \mathbf{C}\hat{\mathbf{x}}_{k+1|k+1}, \tag{5.6g}$$

where $\mathbf{A}_k$ is the Jacobian given by $\mathbf{A}_k = \frac{\partial \mathbf{F}}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}_{k|k}}$, which is updated at every time-step.

Furthermore, here $\mathbf{P}$ is the state covariance, $\mathbf{K}$ is the Kalman gain, $\mathbf{I}$ is the identity matrix, $\mathbf{Q}_k$ is the process covariance matrix, $\mathbf{R}_k$ is the measurement covariance matrix, $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ are the estimated values of $\mathbf{x}$ and $\mathbf{y}$.

Remark 9. *The system given in (5.4a)-(5.4b) is observable since the observability matrices are of full rank.*

Furthermore, if only parts of a quadrotor system are observable, and since we use a Reduced Order EKF, we do not need to use all of the systems states to estimate and detect the faults, so it is still possible to use our FDD if the states used for the estimation are available (P, Q, R, U, V, W). And they usually are, because a controller would need feedback on these states (and others) in order to make the quadrotor fly.

5.3.2 Determining $\mathbf{Q}_k$ and $\mathbf{R}_k$ covariance matrices

The identification of covariances $\mathbf{Q}_k$ and $\mathbf{R}_k$ is not usually straightforward, this being known as one of the limitations of using Kalman filter-based strategies. However, there are several optimization methods available in the literature that propose a solution to this issue.

Assumption 4. *The covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$ are constant, that is, they do not change during flight and satisfy (5.5a) and (5.5b).*

For the implementation of the proposed technique, we used the method derived in [Bavdekar, Deshpande & Patwardhan 2011], and applied in [Fang C. C. de Visser & Holzapfel 2022], in order to find adequate covariance values for our system beforehand. The optimization was performed offline, before the system operation, and the values of $\mathbf{Q}_k$ and $\mathbf{R}_k$ remain constant during the whole flight. The approach is based on the maximum-likelihood principle and consists of minimizing the equation (27) present in the cited reference [Bavdekar, Deshpande & Patwardhan 2011]. Here, the optimization problem was solved using a sequential quadratic programming strategy.

If we define, from Equations (5.6e) and (5.6d), respectively, the innovation sequence $\mathbf{e}_{k+1}$ as a zero mean Gaussian white noise sequence:

$$\mathbf{e}_{k+1} = \mathbf{y}_{k+1} - \hat{\mathbf{y}}_{k+1|k} \tag{5.7}$$

and its covariance Σ_{k+1} given by:

$$\Sigma_{k+1} = \mathbf{C}\mathbf{P}_{k+1|k}\mathbf{C}^{\mathrm{T}} + \mathbf{R}_{k+1} \tag{5.8}$$

Assumption 5. *One can assume that the individual elements of* $\mathbf{w}_k$ *and* ν_k *are uncorrelated with each other. It is reasonable to assume this when the noise sources are arising from independent physical variables [Bavdekar, Deshpande & Patwardhan 2011]. As a consequence, the decision variables,* $\mathbf{Q}_k$ *and* $\mathbf{R}_k$ *can be considered diagonal matrices. Moreover, one imposes the positivity constraint on the covariance matrices* $\mathbf{Q}_k$ *and* $\mathbf{R}_k$*.*

Assumptions 4 and 5 are necessary due to this system having a considerable order, and the resulting optimization problem would become complex in terms of computations [Bavdekar, Deshpande & Patwardhan 2011]. A large number of decision variables would also imply that a data set with a large N is necessary to reduce the variance errors in the estimates.

According to [Fang C. C. de Visser & Holzapfel 2022], from the several studies that have attempted to estimate the noise covariances in a Kalman Filter framework, one can mention the Myers and Tapley (MT) approach [Myers & Tapley 1976], which estimates the covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$ with a covariance matching technique using the innovation in the EKF. [Mendonça & Góes 2007] applied the MT approach, designing two parallel Kalman filters to estimate the $\mathbf{Q}_k$ and $\mathbf{R}_k$ covariances, improving the main filter's performance. [Bavdekar, Deshpande & Patwardhan 2011] proposed an extended expectation-maximization (EM) method, based on the maximum-likelihood principle, to identify $\mathbf{Q}_k$ and $\mathbf{R}_k$. In [Myers & Tapley 1976] and [Mendonça & Góes 2007], the covariances are estimated online, dynamically, while [Bavdekar, Deshpande & Patwardhan 2011] tunes them for the filter offline, before the flight.

A comparison of different adaptive filter tuning methods for the noise covariances in a Kalman Filter is made in [M.R. Mohan M.S. 2016], having concluded that the extended EM method is a better option for estimating them than the MT approach. Finally, a dynamic covariance matrix would imply a high computational demand during flight, which our current platform does not provide, so the technique developed in [Bavdekar, Deshpande & Patwardhan 2011] was considered the most adequate for this research.

This results in the following optimization problem:

$$(\hat{\mathbf{Q}}, \hat{\mathbf{R}}) = \underset{(\mathbf{Q},\mathbf{R})}{\arg\min} \left[\sum_{i=1}^{N} \log(\det(\mathbf{\Sigma}_i)) + \mathbf{e}_i^T \mathbf{\Sigma}_i^{-1} \mathbf{e}_i \right] \tag{5.9}$$

Subject to:

Model equations (5.4a)-(5.4b),

Filter equations (5.6a)-(5.6g),

and

$$\begin{aligned}
&\mathbf{Q} = \mathbf{diag}\begin{bmatrix}\delta_1 & \delta_2 & \ldots & \delta_{n_w}\end{bmatrix}\\
&\mathbf{R} = \mathbf{diag}\begin{bmatrix}\gamma_1 & \gamma_2 & \ldots & \gamma_r\end{bmatrix}\\
&0 < a_L \leq \delta_i \leq a_H,\ \mathbf{for}\ i = 1, 2, 3..., n_w\\
&0 < b_L \leq \gamma_i \leq b_H,\ \mathbf{for}\ i = 1, 2, 3..., r
\end{aligned} \tag{5.10}$$

Upper and lower bounds must be imposed ensuring that the optimization problem does not become numerically ill-conditioned during the iterations and the positive definiteness of the matrices is maintained. These bounds can be empirically found by taking into consideration the nature of the physical variable involved [Bavdekar, Deshpande & Patwardhan 2011].

Following the notation on (5.5b), we have obtained, for simulation data:

$$Q_k^x = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0\\ 0 & 0.5 & 0 & 0 & 0 & 0\\ 0 & 0 & 0.5 & 0 & 0 & 0\\ 0 & 0 & 0 & 5 & 0 & 0\\ 0 & 0 & 0 & 0 & 5 & 0\\ 0 & 0 & 0 & 0 & 0 & 5 \end{bmatrix} \tag{5.11}$$

$$Q_k^\alpha = \begin{bmatrix} 8.8\times10^{-7} & 0 & 0 & 0\\ 0 & 8.8\times10^{-7} & 0 & 0\\ 0 & 0 & 8.8\times10^{-7} & 0\\ 0 & 0 & 0 & 8.8\times10^{-7} \end{bmatrix} \tag{5.12}$$

and

$$R = \begin{bmatrix} 0.01 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.01 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.09 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.09 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.09 \end{bmatrix} \tag{5.13}$$

And for real flight data:

$$Q_k^x = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 \end{bmatrix} \tag{5.14}$$

$$Q_k^\alpha = \begin{bmatrix} 5.075 \times 10^{-7} & 0 & 0 & 0 \\ 0 & 5.075 \times 10^{-7} & 0 & 0 \\ 0 & 0 & 5.075 \times 10^{-7} & 0 \\ 0 & 0 & 0 & 5.075 \times 10^{-7} \end{bmatrix} \tag{5.15}$$

and

$$R = \begin{bmatrix} 0.01 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.01 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.09 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.09 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.09 \end{bmatrix} \tag{5.16}$$

Since this method was merely an available tool used in this research, we encourage the reader to check the reference [Bavdekar, Deshpande & Patwardhan 2011] if they wish further details on this step.

5.4 Results

In this Section, we present the simulation and real flight experiment results for the proposed ROEKF-TAE used as the FDD system tolerant to aerodynamic effects. We continue to use the same micro quadrotor detailed in Chapter 4, the PARROT Mambo, along with the MATLAB/Simulink support package. For these tests, we have collected the mini drone's mean velocities in the inertial frame both in simulation and real robot results. They were approximately $v_x = 0.15$ m/s; $v_y = 0.18$ m/s; $v_z = 0.07$ m/s, while the maximum velocities were approximately $v_x = 0.32$ m/s; $v_y = 0.38$ m/s; $v_z = 0.62$ m/s. The power is provided by a GiFi Power 3.7 V, 600 mAh, 2.2 Wh, 15c Li-Po battery.

In the experiments and simulations results shown below for this Chapter's contributions, the proposed FDD was implemented within the control architecture presented in Fig. 5.2. This is the same control architecture as seen in Chapter 4, already with the new control allocation approach and the aforementioned PID controllers.

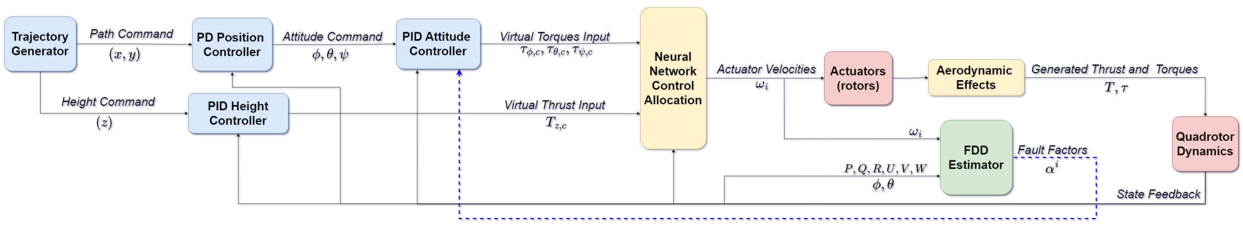

Figure 5.2: Block diagram for the quadrotor system. The blue blocks are the PARROT Mambo micro-drone controllers and path generators. The yellow blocks represent the NNCA and aerodynamic effects detailed in Chapter 4. The red blocks are the UAV dynamics and the green block is our proposed FDD approach. The blue dashed line may be considered in future work, where the PID Attitude controller can be replaced by a fault-tolerant approach.

Note that the *FDD Estimator* block, which is the main proposed approach in this work, takes as inputs the state feedback from the states it needs to estimate the fault factors, as well as the commanded actuator velocities ω_i. Here it is assumed that the actuator dynamics is fast enough to consider that the commanded ω_{ic} equals the established ω_i.

Finally, it is important to mention that the feedback loop from the *FDD Estimator* block output to the controller may be considered as a follow-up work with the proposal of a fault-tolerant controller. The goal of this research so far is to detect, estimate and locate LOE faults. The focus is not on the controller algorithm itself.

5.4.1 In Silico Simulation Tests

For Simulation tests, the PARROT Mambo Support package for Simulink was used, since this is the same quadrotor used in flight experiments. Also, unlike most simulation models used in the literature, the one used here has aerodynamic effects implemented as in Table 3.2 and 5.2. The necessary control strategy was implemented in discrete time, with the Bogacki-Shampine method, which is a requirement for it to be deployed to the PARROT Mambo hardware, and a step size of 5ms, chosen to match the sampling time of the quadrotor during flight experiments.

The test consisted of a 60 second flight, with take-off and landing. The drone takes off, follows a 0.5m radius circular trajectory, and lands. The loss-of-effectiveness faults were artificially introduced in the actuators through attenuation of the motor commands, inserted with gains that represent the fault factors α_k^i, ranging between 0 and 1, as previously stated.

In this trial, all motors start the flight in a healthy state ($\alpha_k^i = 1$, $i = 1, 2, 3, 4$) up to 20 s, $k \in [1, 4000)$, of the operation time, when motors 1 and 4 are simultaneously attenuated to work at 95% of the commanded velocity ($\alpha_k^1 = \alpha_k^4 = 0.95$, $k \in [4000, 8000)$). Furthermore, at 40 s of the operation time, the fault is increased to 80% ($\alpha_k^1 = \alpha_k^4 = 0.8$, $k \in [8000, 10400]$), as can be seen in the black dashed lines in Figure 5.3 and 5.6. These values were chosen based on empirical tests that showed the acceptable LOE range that still allowed the drone to fly, since the scope of this work does not propose to compensate for the faults, only to detect, estimate and locate them.

Note that, even if we have assumed the fault factors α^i to be constant in (5.3), their value is changed twice mid-flight, in both simulation and experimental tests, to show that our method can still detect and estimate the fault with some degree of a time-varying loss-of-effectiveness.

The comparison between four different estimators is shown in Figures 5.3 to 5.6. In green, a traditionally implemented EKF without tolerance to aerodynamic effects; in red, an adaptive state-of-the-art EKF [Zhong et al. 2018]; in blue, the ROEKF-TAE (reduced order EKF tolerant to aerodynamic effects) proposed in this thesis; and in pink, the adaptive factor

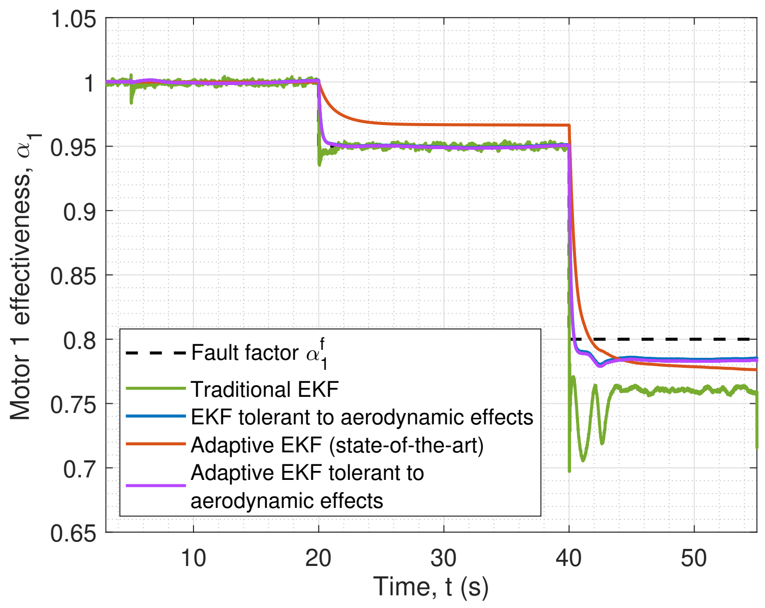

Figure 5.3: Fault factor and estimation for motor 1, which starts working at 95% capacity at 20s, along with motor 4, and then both at 80% capacity at 40s, while the other motors stay healthy.

from [Zhong et al. 2018] added to our proposed filter.

Considering quantitative and qualitative analysis, the traditional EKF (green line) might be considered the one with the worst performance in general, due to the fact it has no modifications in order to improve performance in this system. The signals in this case are visibly noisier than their other counterparts, with the worst transient behavior and overshoot when the fault value changes. In Table 5.3 it is also shown that it has the worst mean absolute error (MAE) for motors 1 and 3 when compared to the reference. Here, α_k^i is the induced loss-of-effectiveness in the motor i, $\hat{\alpha}_k^i$ is the estimated fault factor, i.e., $\hat{\alpha}_k^i = \hat{\mathbf{x}}_{k|k}(6+i)$, $i = 1, 2, 3, 4$, and $n = 10400$ is the number of samples in our case, for both simulation (Table 5.3) and experimental results (Table 5.4).

However, contrary to what one might have predicted, the Adaptive EKF introduced in [Zhong et al. 2018] actually worsened the mean value of the estimation in many parts, even if its signal is smoother. Most notably, in motors 2 and 3 (Figs. 5.4 and 5.5), which were not affected by the faults introduced in this simulation, we can see that the results from the AEKF (red line) deviate the most from the adopted healthy value of 1. In fact, the AEKF estimation is the farthest one from the reference for the first fault, that is, between the 20s

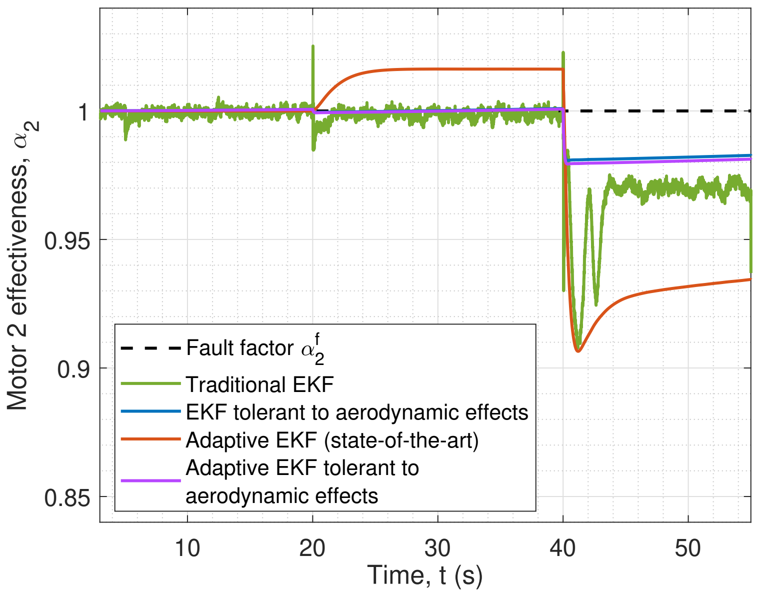

Figure 5.4: Fault factor and estimation for motor 2, which stays healthy through the flight, along with motor 3, while motors 1 and 4 fail.

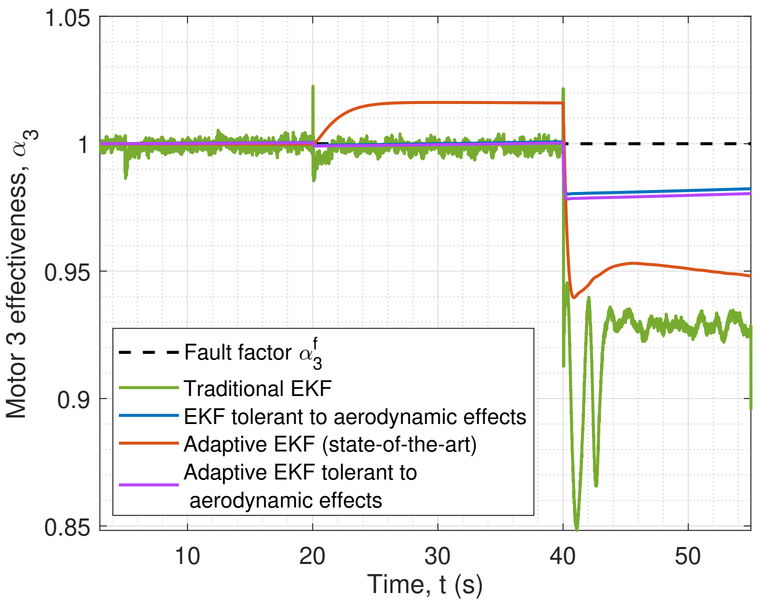

Figure 5.5: Fault factor and estimation for motor 3, which stays healthy through the flight, along with motor 2, while motors 1 and 4 fail.

and 40s, for all motors. After the second fault occurs at the 40s, it has the worst estimation for the faulty motor 4 and for the healthy motor 2. This behavior culminated in this filter having the worst average MAE on Table 5.3.

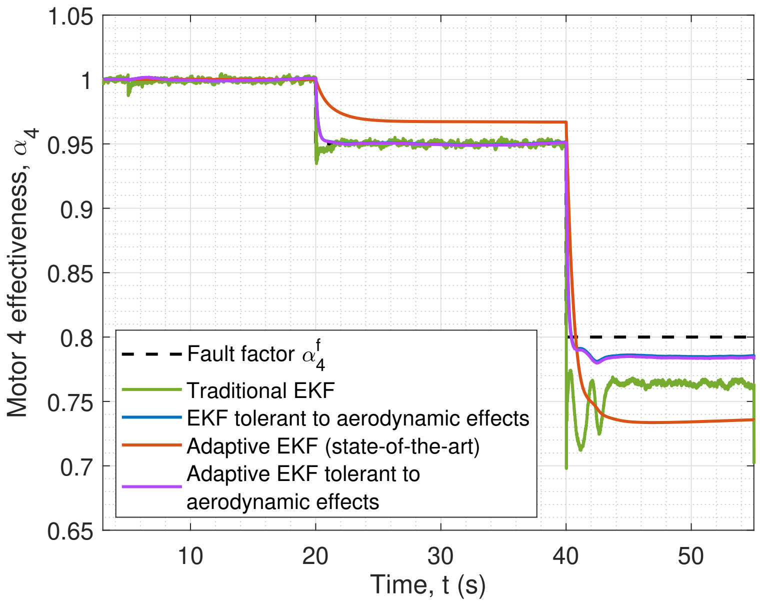

Figure 5.6: Fault factor and estimation for motor 4, which starts working at 95% capacity at 20s, along with motor 1, and then both at 80% capacity at 40s, while the other motors stay healthy.

MAE = $\frac{1}{n}\Sigma_{k=1}^{n}\lvert\alpha_k^i - \hat{\alpha}_k^i\rvert$	**Motor 1**	**Motor 2**	**Motor 3**	**Motor 4**	**Average**
Classic EKF	2.5%	1.84%	3.18%	2.36%	2.47%
ROEKF-TAE	0.76%	0.65%	0.67%	0.73%	0.7%
Adaptative EKF	1.73%	2.98%	2.29%	3.48%	2.62%
Adaptative ROEKF-TAE	0.83%	0.71%	0.74%	0.78%	0.76%

Table 5.3: Normalized Mean Absolute Errors for Simulation Results. The smallest errors are highlighted.

It is worth noting that the results for the AEKF presented here deviate from the ones shown in [Zhong et al. 2018]. This is due to the presence of aerodynamic effects modeling in our simulator, which was lacking in the simulator used by the authors in [Zhong et al. 2018], as it is in most of the available literature. Hence the influence of such effects is not evident in their results as it is in the ones presented here using a more realistic simulator.

The FDD system proposed here consists of the ROEKF-TAE represented with a blue line in Figures 5.3 to 5.6. One can see it has the best estimation overall, with the smallest MAEs in Table 5.3. The average MAE for our proposed filter is more than three times smaller than those for the traditional EKF and the state-of-the-art AEKF. Moreover, its estimation is barely affected by the faults in the other motors, contrary to the AEKF, as can be seen in Figure 5.4 and 5.5. After the 40s mark, when the fault increases to 80%, the estimation is not as precise as in the first fault before. However, the estimated value is still within acceptable thresholds so as to not detect a fault that never occurred (in this work, we adopted a threshold of 5%, but that can change from platform to platform and has to be determined empirically). This has happened with the AEKF or the traditional EKF in some moments. See Figure 5.4 and 5.5, detecting a loss in effectiveness of more than 5% in a healthy motor, and Figure 5.6 with the estimation of a larger fault than it was in reality. The results demonstrate that our approach is more precise and it converges faster (i.e., it detects the fault faster) than the compared approaches.

The adaptive factor from [Zhong et al. 2018] was also applied to the ROEKF-TAE proposed in this thesis. It is represented with a pink line in the Figures. However, the presence of this adaptive factor has shown no improvements in the results when added to our ROEKF-TAE approach. On the contrary, in some cases (Figure 5.5) it is possible to see that our proposed ROEKF-TAE was slightly worsened by the addition of the adaptive factor.

Remark 10. *The four filters were separately "tuned" for their respective covariance values using the method cited in 5.3.2. That said, only the traditional EKF had a different set of covariances.*

Remark 11. *The same NNCA as in Chapter 5 is used in the real flight experiments and simulations from this Chapter. The network did not need to be retrained for the case of the actuator faults.*

5.4.2 UAV Experimental Tests

The design of the flight experiment was similar to the simulation. As such, it consisted of a 60 second flight with a 0.5m radius orbital trajectory and 1.5m height. The LOE faults were also introduced in the same manner as in the simulation, that is, artificially attenuating the motor commands, inserting gains that represent the fault factors α_k^i, ranging between 0 and 1. A video of this flight experiment can be seen in https://youtu.be/9Msirpfe-SE and is illustrated in Figure 5.7. We encourage the reader to refer to the video, and to the graphs to be shown next for better visualization.

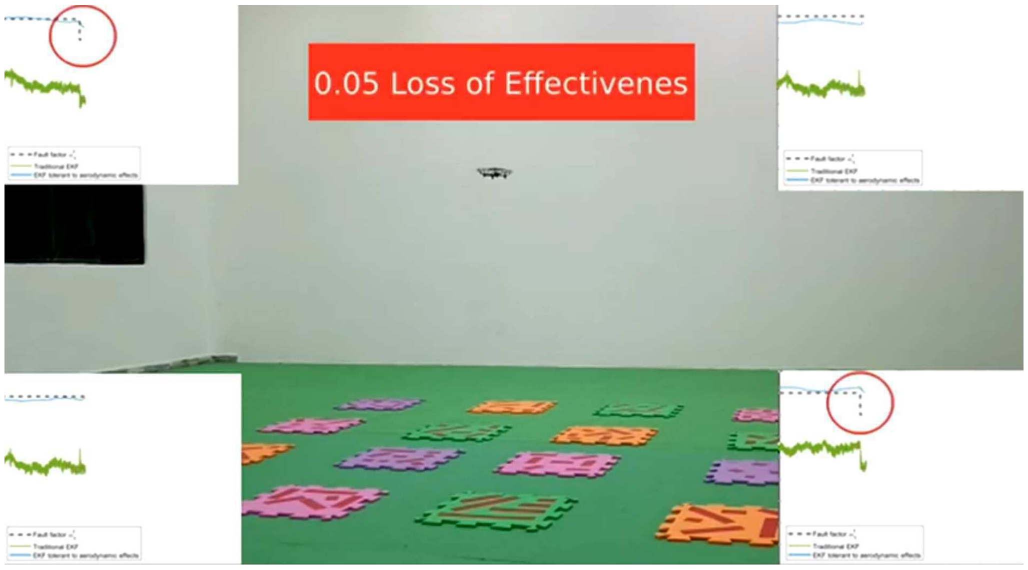

Figure 5.7: Flight experiment for the ROEKF-TAE.

For the real flight, the motors start in a healthy state ($\alpha_k^i = 1$) until 20 s (i.e. $k = 4000$) of the operation time, when motors 1 and 4 are simultaneously attenuated to work at 95% of the commanded velocity ($\alpha_k^1 = \alpha_k^4 = 0.95$). Here, at 40 s (i.e. $k = 8000$) of operation time, the fault is increased to 85% ($\alpha_k^1 = \alpha_k^4 = 0.85$). In real flight conditions, the drone could not recover from an 80% LOE fault as in simulation, so a fault of 85% was used for the flight experiments, since the objective of this research is not for the system to recover from faults, but to detect and locate them, as previously stated.

During the flight, the input data for the FDD Estimator were collected and saved in the flight log. They were provided to the Estimator for analysis post-flight, due to the memory limitations of our flight platform. The proposed algorithm is not computationally demanding since it consists of an EKF-based approach, but the processing capabilities of our available platform, the Parrot Mambo mini quadrotor, were insufficient.

The fact that the analysis is made *a posteriori* does not have any influence on the conclusions about the performance of the FDD System, since this analysis is made using exactly the same data that would be provided to the estimator online, with all the noises and natural real-flight disturbances. On account of the previously mentioned physical limitations, the data were simply stored for post-processing allowing the evaluation of the technique with

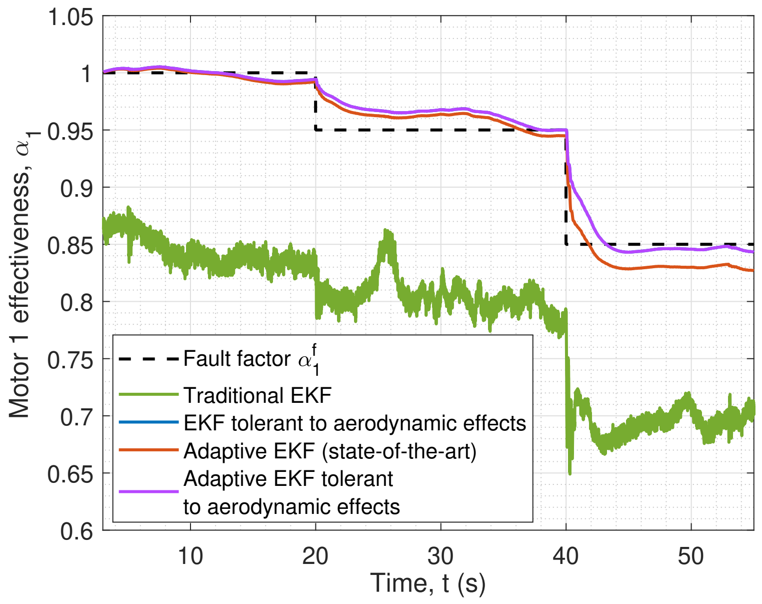

Figure 5.8: Fault factor and estimation for motor 1, for real flight experiments, which starts working at 95% capacity at 20s, along with motor 4, and then both at 85% capacity at 40s, while the other motors stay healthy.

real flight data.

Moreover, the *a posteriori* analysis of the data allowed us to apply a set of covariances better suited to the filter in real flights. We applied real-flight data to the optimization algorithm shown in Section5.3.2. That way we were not limited to simulation data and were able to achieve a more realistic estimator. We performed additional flights in order to collect real-flight data for optimization. Naturally, this tuning was achieved using data from a different flight than the one shown here to analyze the results.

Figures 5.8 to 5.11 depict the results of the fault factors estimation for the flight experiments. Once more, we compare the four estimators that were shown in the simulation results in Section 5.4.1, with the Mean Absolute Errors presented in Table 5.4. The color scheme remains the same as previously stated in Section 5.4.1.

Qualitatively speaking, one can note that the traditional EKF (green line) does not convert well its behavior from simulation to experimental tests, being too disturbed by the noise and vibrations from the real world and thus not estimating the fault factors correctly. Quantitatively, it shows an average MAE of 16.39% in Table 5.4, which is high above the other estimators tested here.

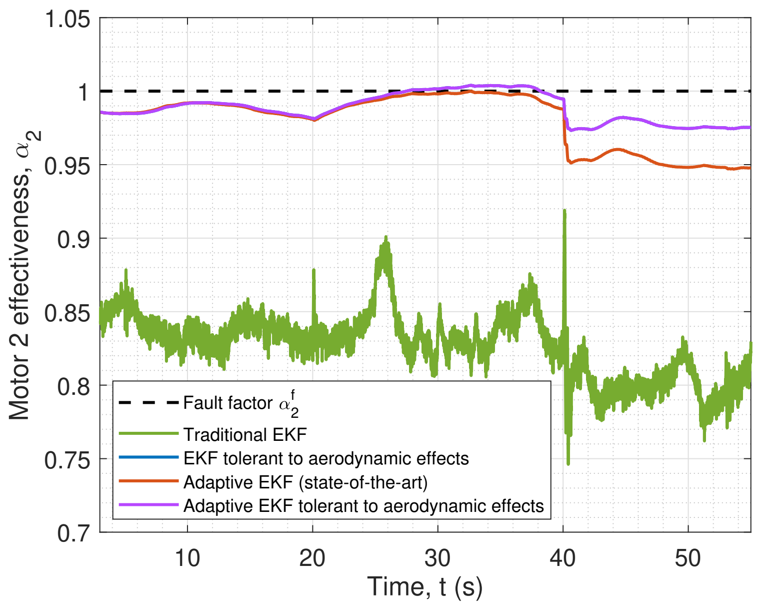

Figure 5.9: Fault factor and estimation for motor 2, for the real flight experiments. This motor stays healthy through the flight, along with motor 3, while motors 1 and 4 fail.

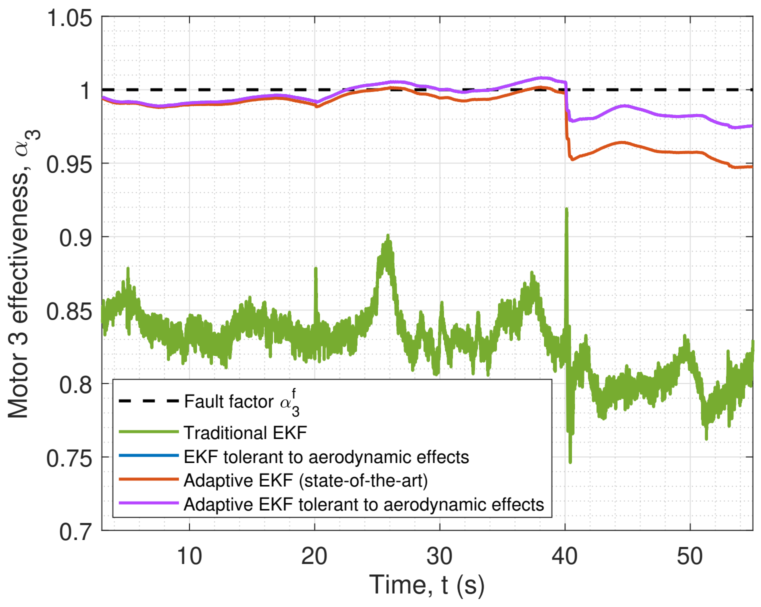

Figure 5.10: Fault factor and estimation for motor 3, for the real flight experiments. This motor stays healthy through the flight, along with motor 2, while motors 1 and 4 fail.

Regarding the adaptive filter from [Zhong et al. 2018], in a real experiment, we can perceive that its adaptive factor greatly improved the behavior when compared to the traditional EKF, providing a more reliable detection when compared to the reference, contrary to its

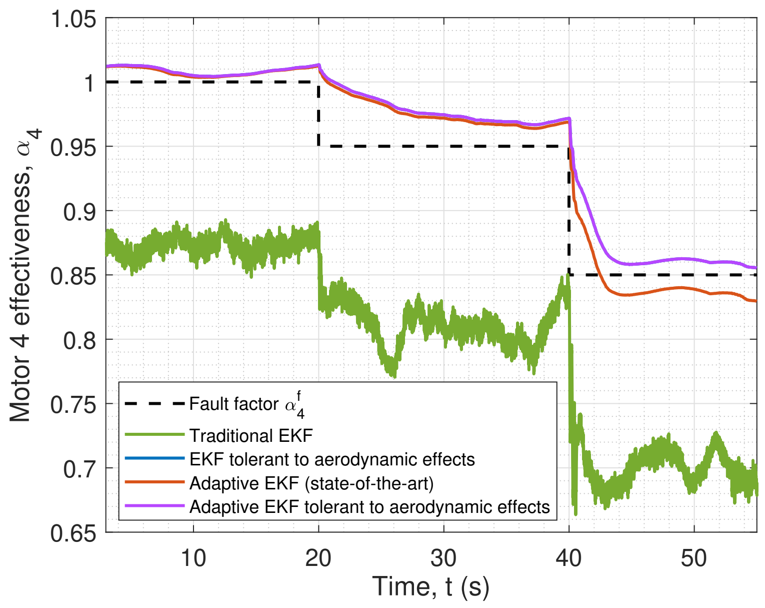

Figure 5.11: Fault factor and estimation for motor 4, for real flight experiments, which starts working at 95% capacity at 20s, along with motor 1, and then both at 85% capacity at 40s, while the other motors stay healthy.

simulation behavior.

Still, the ROEKF Tolerant to Aerodynamic Effects (in blue, under the pink curve) is the one closest to the LOE value, allowing a smaller threshold for fault detection. Table 5.4 shows it has the smallest MAE. It is worth noting that here in the flight experiments the EKF tolerant to aerodynamic effects with and without adaptive factors were practically identical in behavior, showing once more that the consideration of said effects makes the adaptive factor unnecessary, and thus saving computational costs.

MAE = $\frac{1}{n}\Sigma_{k=1}^{n}\lvert\alpha_k^i - \hat{\alpha}_k^i\rvert$	**Motor 1**	**Motor 2**	**Motor 3**	**Motor 4**	**Average**
Classic EKF	16.28%	17.24%	17.24%	14.80%	16.39%
ROEKF-TAE	1.08%	1.25%	0.9%	2.08%	1.33%
Adaptative EKF	1.28%	1.99%	1.67%	1.89%	1.71%
Adaptative ROEKF-TAE	1.08%	1.25%	0.9%	2.08%	1.33%

Table 5.4: Normalized Mean Absolute Errors for Experimental Results. The smallest errors are highlighted.

5.5 Final Considerations

In this Chapter, the fault detection problem and its main characteristics were presented, with the main focus on the detection of loss-of-effectiveness actuator faults. The formulation of the actuator fault problem was shown while including them in the aerodynamic effects modeling previously described.

A Fault Detection and Diagnosis system tolerant to aerodynamic effects for quadrotors based on a Reduced Order Extended Kalman Filter Tolerant to Aerodynamic Effects (ROEKF-TAE) has been proposed. The appropriate covariance matrices for the filter were chosen according to an optimization algorithm using a sequential quadratic programming strategy.

By estimating the fault factors corresponding to each actuator, the FDD was capable to detect, isolate and identify simultaneous loss-of-effectiveness faults, while taking into account aerodynamic effects such as blade-flapping. Simulation and real experimental results were presented, as well as comparisons with a state-of-the-art solution. Our proposed approach had a smaller mean absolute error when compared to the reference and a higher capability of identifying with precision the faulty actuator(s).

Chapter 6

Conclusion and Future Work

This research has proposed to develop a new fault detection and diagnosis scheme while contributing to the compensation of aerodynamic effects disturbances in quadrotor UAV systems. The contributions were made through the control allocation step and a fault detection and diagnosis scheme, both modified from their classic implementations in order to be tolerant to aerodynamic effects such as blade flapping and induced drag. The motivations, methods, goals and contributions behind this research were shown in Chapter 1, while the related works were discussed in Chapter 2, highlighting what is lacking in the current state-of-the-art literature in which we hope to have filled a gap with this thesis contributions.

For the comprehension of the basic concepts, the quadrotor mathematical modeling was presented in Chapter 3, as well as the aerodynamic effects that modify the classic model. It was shown analytically in Chapter 3 and experimentally through the results in later Chapters that the blade flapping effects cannot be neglected in the flight control system of a quadrotor, as is usually done in current literature, as discussed in Chapter 2. Chapter 3 also provided an analysis of the thrust force rotation around the rotor axis while under aerodynamic effects, which is a theoretical contribution of this thesis. Matters of system stability, observability and controllability were also discussed in this chapter.

In Chapter 4, we proposed the first modification of the quadrotor system in order to take into account the aerodynamic effects: a new control allocation, which consists of replacing the classic allocation matrix (or mixing matrix) by a fitting function neural network which converts the mechanical conjugates from the control input into actuator commands. After performing flight experiments to validate this technique, the control input has been compared to the thrust and torques generated by the actuators and established at the aircraft, in the NNCA case and in the classic allocation matrix case. The allocation error was much smaller for the NNCA, meaning the actuators execute the appropriate control effort demanded by the controller, while that was not possible for the mixing matrix.

The same NNCA was then applied to the flight tests and simulations in Chapter 5, in which we have developed a fault detection and diagnosis scheme for loss-of-effectiveness actuator faults, based on a ROEKF-TAE. The experiments and simulations for the proposed FDD consisted of failing two actuators at the same time, while following a circular trajectory. This is already a step forward from the current literature (besides the consideration of aerodynamic effects in the fault estimator) which usually only performs simulations or flight experiments for FDDs in hover state. The novel technique performed better when compared to other options in classic and state-of-the-art literature, as we have shown by means of graphical analysis and error calculation. The ROEKF-TAE had more precision when distinguishing between LOE faults effects and aerodynamic disturbances, when compared to the other estimators. It also identified with a greater level of confidence which actuators were truly defective and which were not.

This research has then concluded that the consideration of aerodynamic effects throughout the development of a quadrotor system is not negligible. Aerodynamic effects such as blade flapping and induced drag naturally hinder actuator performance, and their influence might even be mistaken for an actuator fault. Although here we have considered the control allocation and the FDD, other parts of the system might benefit from the consideration of those disturbances. The focus of this thesis did not rely on the controller algorithm but on the control allocation step and the fault detection and diagnosis application. Besides that, the inclusion of the NNCA into any other quadrotor's allocation step should significantly improve its flight performance. Figure 6.1 makes a synthesis of the logical steps that guided this research.

6.1 Contributions

The main contribution from an academic point of view is the compilation of the research results into scientific papers to be published in relevant journals in this area of knowledge.

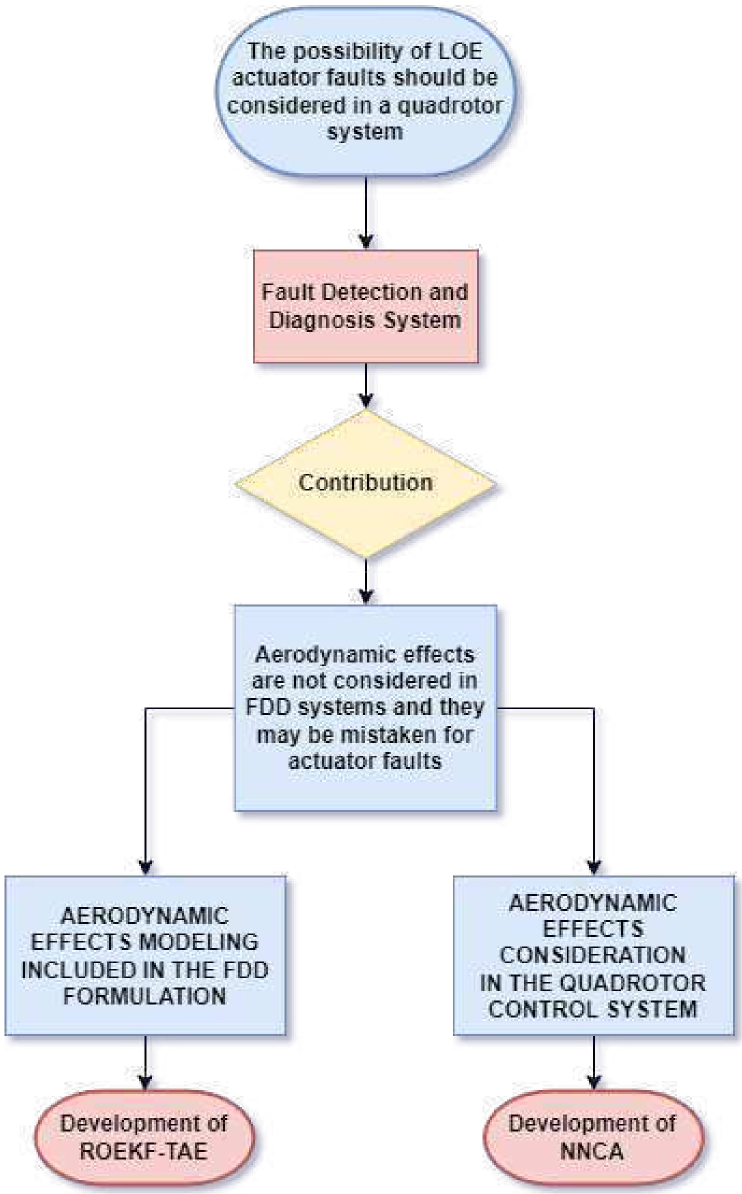
The possibility of LOE actuator faults should be considered in a quadrotor system
Fault Detection and Diagnosis System
Contribution
Aerodynamic effects are not considered in FDD systems and they may be mistaken for actuator faults
AERODYNAMIC EFFECTS MODELING INCLUDED IN THE FDD FORMULATION
AERODYNAMIC EFFECTS CONSIDERATION IN THE QUADROTOR CONTROL SYSTEM
Development of ROEKF-TAE
Development of NNCA

Figure 6.1: Research steps.

A first paper with the contributions from Section 3.4.1 and Chapter 4, regarding the thrust force rotation and the compensation of aerodynamic effects by means of the neural network control allocation (NNCA) has been published as:

- S. P. Madruga, A. H. B. M. Tavares, S. O. D. Luiz, T. P. do Nascimento and A. M. N. Lima, "Aerodynamic Effects Compensation on Multi-Rotor UAVs Based on a Neural Network Control Allocation Approach," in IEEE/CAA Journal of Automatica Sinica, vol. 9, no. 2, pp. 295-312, February 2022, doi: 10.1109/JAS.2021.1004266. [Madruga et al. 2022]

The **main contributions** of this paper were:

1. We proposed the use of a classic neural network as a **novel control allocation approach** that takes into account the aerodynamic effects.
2. The proposed method can be used to **improve the performance of quadrotor controllers in general**, since it removes the error introduced by the classic allocation matrix.
3. Our experimental results verified that **neglecting aerodynamic effects in the control allocation has a direct negative impact** in the flight performance.
4. **Several aspects of the system were improved** with the new control allocation, including the decrease in the allocation error, the increase in the controller performance, and better overall signal quality.

A second paper regarding the fault detection and diagnosis contributions from Chapter 5 has been accepted for publication in the IEEE Robotics and Automation Letters. This paper was partly produced in partnership with Technische Universität München during the 6 months of the sandwich period of this doctorate research, financed by CAPES.

- S. P. Madruga, T. P. Nascimento, F. Holzapfel and A. M. N. Lima, "Estimating the Loss of Effectiveness of UAV Actuators in the Presence of Aerodynamic Effects," in IEEE Robotics and Automation Letters, vol. 8, no. 3, pp. 1335-1342, March 2023, doi: 10.1109/LRA.2023.3238184. [Madruga et al. 2023] Video available at: https://youtu.be/9Msirpfe-SE.

As for the main contributions of this second paper, we can cite:

1. The proposed FDD scheme from Chapter 5 is able to **correctly identify faults in more than one actuator concurrently** in the presence of aerodynamic disturbances.

2. **Real flight experiments were conducted with a more complex maneuver/trajectory** instead of simply hover as has been seen in most of the examples available in the literature.

3. The consideration of the aerodynamic effects in the estimator **eliminated the need to use an adaptive factor** as in [Zhong et al. 2018].

4. The new filter has **reduced order**, that is, it does not need all state measurements to successfully estimate the fault factor.

6.2 Future work

The proposed Fault Detection and Diagnosis system was not integrated into the controller algorithm of the quadrotor. It is effective for detecting, estimating and isolating the fault, but the next direct step for the research would be its integration with the control system, either by means of a fault-tolerant controller approach to be used together with our ROEKF-TAE fault detector, or an alarm system algorithm to warn the user of the fault. That would solve the main limitation of this work, which is the mitigation of the detected faults.

This fault detection algorithm focused on loss-of-effectiveness actuator faults. The inclusion of other fault-detection techniques in the system in order to detect the other mentioned types of actuator faults could be an improvement brought by further research in future work. Besides that, a more in-depth study of the LOE fault factor characterization could also be performed in terms of electrical motor modeling, considering DC motors, which are used in multi-rotor UAVs.

The same can be said about the aerodynamic effects considered in the mathematical formulation of this thesis. The main focus was on blade-flapping and induced drag, which affect the actuator's air inflow during flight. However, there are several other aerodynamic disturbances in any flight system. So the inclusion of further aerodynamic consideration, especially in the Control Allocation step, is a way to further deepen the contributions. Other types of neural network architectures could also be implemented for the NNCA, investigating if there are any gains in performance that can be achieved with their use.

Real flight experiments with a wind tunnel could also help in the better characterization of the aerodynamic effects that affect flight behavior. In which conditions they are more relevant, and what is an exact size or speed threshold to better observe or not these effects.

Finally, one could point out that all the experiments here were performed with a platform built for indoor flight, which is in fact a valid and common application for quadrotor UAVs. Nonetheless, outdoor flights might expose an aircraft to other types of disturbances, especially aerodynamic ones, when we take into consideration wind, landing etc. So as a way to further expand the results here obtained, the developed techniques could be applied, adapted and tested with a drone suited for outdoor flight and the most relevant observed aerodynamic disturbances.

Bibliography

[Achermann et al. 2019]Achermann, F. et al. Learning to predict the wind for safe aerial vehicle planning. In: *2019 International Conference on Robotics and Automation (ICRA)*. [S.l.: s.n.], 2019. p. 2311–2317.

[Aishwarya & Jayanand 2016]AISHWARYA, V.; JAYANAND, B. Estimation and control of sensorless brushless dc motor drive using extended kalman filter. In: *2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT)*. [S.l.: s.n.], 2016. p. 1–7.

[Alex, Daniel & Jayanand 2016]ALEX, S. S.; DANIEL, A. E.; JAYANAND, B. Reduced order extended kalman filter for state estimation of brushless dc motor. In: *2016 Sixth International Symposium on Embedded Computing and System Design (ISED)*. [S.l.: s.n.], 2016. p. 239–244.

[Alves 2019]ALVES, J. G. *Control Allocation Applied to Robots Subject to Input Constraints*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2019.

[Ansari, Bajodah & Hamayun 2019]ANSARI, U.; BAJODAH, A.; HAMAYUN, M. Quadrotor control via robust generalized dynamic inversion and adaptive non-singular terminal sliding mode. *Asian Journal of Control*, v. 21, 07 2019.

[Ansari & Bajodah 2019]Ansari, U.; Bajodah, A. H. Quadrotor control using neuro-adaptive robust generalized dynamic inverasion. In: *2019 8th International Conference on Systems and Control (ICSC)*. [S.l.: s.n.], 2019. p. 1–6.

[Avram, Zhang & Muse 2017]AVRAM, R. C.; ZHANG, X.; MUSE, J. Quadrotor actuator fault diagnosis and accommodation using nonlinear adaptive estimators. *IEEE Transactions on Control Systems Technology*, v. 25, n. 6, p. 2219–2226, 2017.

[Avram, Zhang & Muse 2017]AVRAM, R. C.; ZHANG, X.; MUSE, J. Quadrotor actuator fault diagnosis and accommodation using nonlinear adaptive estimators. *IEEE TCST*, v. 25, n. 6, p. 2219–2226, 2017.

[Baldini et al. 2020]BALDINI, A. et al. Estimation of actuator faults in quadrotor vehicles: from theory to validation with experimental flight data. In: *2020 ICUAS*. [S.l.: s.n.], 2020. p. 1249–1256.

[Bavdekar, Deshpande & Patwardhan 2011]BAVDEKAR, V. A.; DESHPANDE, A. P.; PATWARDHAN, S. C. Identification of process and measurement noise covariance for state and parameter estimation using extended kalman filter. *J. of Proc. Contr.*, v. 21, n. 4, p. 585–601, 2011. ISSN 0959-1524.

[Bošković & Mehra 2003]BOŠKOVIĆ, J. D.; MEHRA, R. K. Failure detection, identification and reconfiguration in flight control. In: ____. *Fault Diagnosis and Fault Tolerance for Mechatronic Systems:Recent Advances*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003. p. 129–167. ISBN 978-3-540-45737-4. Disponível em: <https://doi.org/10.1007/3-540-45737-2$_5$>.

[Bouktir, Haddad & Chettibi 2008]BOUKTIR, Y.; HADDAD, M.; CHETTIBI, T. Trajectory planning for a quadrotor helicopter. In: *2008 16th Mediterranean Conference on Control and Automation*. [S.l.: s.n.], 2008. p. 1258–1263.

[Bresciani 2008]BRESCIANI, T. *Modelling, Identification and Control of a Quadrotor Helicopter*. Tese (Doutorado) — Lund University, 10 2008.

[Bristeau et al. 2009]Bristeau, P. et al. The role of propeller aerodynamics in the model of a quadrotor uav. In: *European Control Conference (ECC)*. [S.l.: s.n.], 2009. p. 683–688.

[Cai et al. 2017]CAI, G. et al. A framework of frequency-domain flight dynamics modeling for multi-rotor aerial vehicles. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, v. 231, n. 1, p. 30–46, 2017.

[Cao et al. 2019]Cao, S. et al. Adaptive incremental nonlinear dynamic inversion control based on neural network for uav maneuver. In: *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. [S.l.: s.n.], 2019. p. 642–647.

[Chebbi et al. 2020]CHEBBI, J. et al. Novel model-based control mixing strategy for a coaxial push-pull multirotor. *IEEE Robotics and Automation Letters*, v. 5, n. 2, p. 485–491, 2020. Disponível em: <https://doi.org/10.1109/LRA.2019.2963652>.

[Davis & Pounds 2017]Davis, E.; Pounds, P. E. I. Direct sensing of thrust and velocity for a quadrotor rotor array. *IEEE Robotics and Automation Letters*, v. 2, n. 3, p. 1360–1366, 2017.

[Dierks & Jagannathan 2008]DIERKS, T.; JAGANNATHAN, S. Neural network output feedback control of a quadrotor uav. In: *2008 47th IEEE Conference on Decision and Control*. [S.l.: s.n.], 2008. p. 3633–3639.

[Dierks & Jagannathan 2010]DIERKS, T.; JAGANNATHAN, S. Output feedback control of a quadrotor uav using neural networks. *IEEE Transactions on Neural Networks*, v. 21, n. 1, p. 50–66, 2010.

[Doman & Oppenheimer 2002]DOMAN, D.; OPPENHEIMER, M. Improving control allocation accuracy for nonlinear aircraft dynamics. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. [S.l.: s.n.], 2002. p. 1–10.

[Doman, Oppenheimer & Sigthorsson 2010]DOMAN, D.; OPPENHEIMER, M.; SIGTHORSSON, D. Dynamics and control of a biomimetic vehicle using biased wingbeat forcing functions: Part ii - controller. In: . [S.l.: s.n.], 2010. v. 34. ISBN 978-1-60086-959-4.

[Ducard 2009]DUCARD, G. *Fault-tolerant Flight Control and Guidance Systems: Practical Methods for Small Unmanned Aerial Vehicles*. [S.l.: s.n.], 2009. ISBN 978-1-84882-561-1.

[Ducard & Hua 2011]DUCARD, G.; HUA, M.-D. Discussion and practical aspects on control allocation for a multi-rotor helicopter. In: *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. [S.l.: s.n.], 2011. p. 95–100.

[El-Melegy 2013]EL-MELEGY, M. T. Random sampler m-estimator algorithm with sequential probability ratio test for robust function approximation via feed-forward neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, v. 24, n. 7, p. 1074–1085, 2013. Disponível em: <https://doi.org/10.1109/TNNLS.2013.2251001>.

[Fang C. C. de Visser & Holzapfel 2022]FANG C. C. DE VISSER, D. M. P. X.; HOLZAPFEL, F. Wind and airflow angle estimation using an adaptive extended rauch-tung-striebel smoother. In: *AIAA 2022-1399, Session: Estimation III*. [S.l.]: Aerospace Research Central, 2022.

[Fourlas & Karras 2021]FOURLAS, G. K.; KARRAS, G. C. A survey on fault diagnosis and fault-tolerant control methods for unmanned aerial vehicles. *Machines*, v. 9, n. 9, 2021. ISSN 2075-1702.

[Freeman et al. 2013]FREEMAN, P. et al. Model-based and data-driven fault detection performance for a small uav. *IEEE/ASME Transactions on Mechatronics*, v. 18, n. 4, p. 1300–1309, 2013.

[Gonzalez-Diaz et al. 2020]GONZALEZ-DIAZ, R. et al. Neural-network-based curve fitting using totally positive rational bases. *Mathematics*, v. 8, n. 12, 2020. ISSN 2227-7390. Disponível em: <https://doi.org/10.3390/math8122197>.

[Gorban & Wunsch 1998]Gorban, A. N.; Wunsch, D. C. The general approximation theorem. In: *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36227)*. [S.l.: s.n.], 1998. v. 2, p. 1271–1274 vol.2.

[Guo, Jiang & Zhang 2018]GUO, Y.; JIANG, B.; ZHANG, Y. A novel robust attitude control for quadrotor aircraft subject to actuator faults and wind gusts. *IEEE/CAA JAS*, v. 5, n. 1, p. 292–300, 2018.

[Härkegård 2003]HÄRKEGÅRD, O. *Backstepping and control allocation with applications to flight control*. Linköping: Linköpings Universitet, 2003. ISBN 91-7373-647-3.

[Harshavarthini et al. 2020]HARSHAVARTHINI, S. et al. Non-fragile fault alarm-based hybrid control for the attitude quadrotor model with actuator saturation. *IEEE Transactions on Circuits and Systems II: Express Briefs*, v. 67, n. 11, p. 2647–2651, 2020.

[Hoffmann et al. 2007]HOFFMANN, G. et al. Quadrotor helicopter flight dynamics and control: Theory and experiment. In: *AIAA Guidance, Navigation and Control Conference and Exhibit*. [S.l.: s.n.], 2007. p. 1–20.

[Hornik, Stinchcombe & White 1989]HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, v. 2, n. 5, p. 359 – 366, 1989. ISSN 0893-6080.

[Huang et al. 2009]Huang, H. et al. Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. In: *2009 IEEE International Conference on Robotics and Automation*. [S.l.: s.n.], 2009. p. 3277–3282.

[Hussein & Nemah 2015]HUSSEIN, M. T.; NEMAH, M. N. Modeling and control of quadrotor systems. In: *2015 3rd RSI International Conference on Robotics and Mechatronics (ICROM)*. [S.l.: s.n.], 2015. p. 725–730.

[Härkegård 2004]HäRKEGåRD, O. Dynamic control allocation using constrained quadratic programming. *Journal of Guidance Control and Dynamics*, v. 27, p. 1–27, 2004.

[Iannace, Ciaburro & Trematerra 2019]IANNACE, G.; CIABURRO, G.; TREMATERRA, A. Fault diagnosis for uav blades using artificial neural network. *Robotics*, v. 8, n. 3, 2019. ISSN 2218-6581.

[Inc. 2021]INC., T. M. *Simulink Support Package for Parrot Minidrones version: 1.1 (R2021a)*. Natick, Massachusetts, United States: The MathWorks Inc., 2021. Disponível em: <https://https://www.mathworks.com/help/supportpkg/parrot/index.html>.

[Jiang, Pourpanah & Hao 2020]Jiang, F.; Pourpanah, F.; Hao, Q. Design, implementation, and evaluation of a neural-network-based quadcopter uav system. *IEEE Transactions on Industrial Electronics*, v. 67, n. 3, p. 2076–2085, 2020.

[Johansen 2004]Johansen, T. A. Optimizing nonlinear control allocation. In: *IEEE Conference on Decision and Control (CDC)*. [S.l.: s.n.], 2004. v. 4, p. 3435–3440.

[Johansen & Fossen 2013]JOHANSEN, T. A.; FOSSEN, T. I. Control allocation - a survey. *Automatica*, v. 49, n. 5, p. 1087 – 1103, 2013. ISSN 0005-1098.

[Kantue & Pedro 2018]Kantue, P.; Pedro, J. O. Grey-box modelling of an unmanned quadcopter during aggressive maneuvers. In: *2018 22nd International Conference on System Theory, Control and Computing (ICSTCC)*. [S.l.: s.n.], 2018. p. 640–645.

[Kantue & Pedro 2018]Kantue, P.; Pedro, J. O. Nonlinear identification of an unmanned quadcopter rotor dynamics using rbf neural networks. In: *2018 22nd International Conference on System Theory, Control and Computing (ICSTCC)*. [S.l.: s.n.], 2018. p. 292–298.

[Karlina & Indriawati 2020]KARLINA, D. L.; INDRIAWATI, K. Fault tolerant control for speed sensorless of dc motor. In: *2020 ICoSTA*. [S.l.: s.n.], 2020. p. 1–5.

[Kirchengast, Steinberger & Horn 2018]KIRCHENGAST, M.; STEINBERGER, M.; HORN, M. Control allocation under actuator saturation: An experimental evaluation. *IFAC-PapersOnLine*, v. 51, n. 25, p. 48 – 54, 2018. ISSN 2405-8963. 9th IFAC Symposium on Robust Control Design ROCOND 2018.

[Lee et al. 2020]LEE, J.-y. et al. Fault classification and diagnosis of uav motor based on estimated nonlinear parameter of steady-state model. *IJMERR*, p. 22–31, 01 2020.

[Leishman 2006]LEISHMAN, G. *Principles of Helicopter Aerodynamics with CD Extra*. [S.l.]: Cambridge University Press, 2006. (Cambridge aerospace series). ISBN 9780521858601.

[Li, Wibowo & Guo 2019]LI, M.; WIBOWO, S.; GUO, W. Nonlinear curve fitting using extreme learning machines and radial basis function networks. *Computing in Science Engineering*, v. 21, n. 5, p. 6–15, 2019. Disponível em: <https://doi.org/10.1109/MCSE.2018.2875323>.

[Ma, Xu & Yang 2021]MA, H.-J.; XU, L.-X.; YANG, G.-H. Multiple environment integral reinforcement learning-based fault-tolerant control for affine nonlinear systems. *IEEE Trans. on Cyb.*, v. 51, n. 4, p. 1913–1928, 2021.

[Madruga et al. 2023]MADRUGA, S. P. et al. Estimating the loss of effectiveness of uav actuators in the presence of aerodynamic effects. *IEEE Robotics and Automation Letters*, v. 8, n. 3, p. 1335–1342, 2023.

[Madruga et al. 2022]MADRUGA, S. P. et al. Aerodynamic effects compensation on multi-rotor uavs based on a neural network control allocation approach. *IEEE/CAA JAS*, v. 9, n. 2, p. 295–312, 2022.

[Mahony, Kumar & Corke 2012]MAHONY, R.; KUMAR, V.; CORKE, P. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics Automation Magazine*, v. 19, n. 3, p. 20–32, Sept 2012. ISSN 1070-9932.

[Mendonça & Góes 2007]MENDONçA, E. M. H. C. B. D.; GóES, L. C. S. Adaptive stochastic filtering for online aircraft flight path reconstruction. *ARC Journal of Aircraft*, v. 44, n. 5, p. 1546–1558, 2007.

[Moghadam & Caliskan 2015]MOGHADAM, M.; CALISKAN, F. Actuator and sensor fault detection and diagnosis of quadrotor based on two-stage kalman filter. In: *2015 5th Australian Control Conference (AUCC)*. [S.l.: s.n.], 2015. p. 182–187.

[Monteiro, Lizarralde & Liu Hsu 2016]Monteiro, J. C.; Lizarralde, F.; Liu Hsu. Control allocation: Enhancing the performance of quadrotors high-level controllers. In: *XXI Congresso Brasileiro de Automática (CBA)*. [S.l.: s.n.], 2016.

[Monteiro, Lizarralde & Liu Hsu 2016]Monteiro, J. C.; Lizarralde, F.; Liu Hsu. Optimal control allocation of quadrotor uavs subject to actuator constraints. In: *American Control Conference (ACC)*. [S.l.: s.n.], 2016. p. 500–505.

[M.R. Mohan M.S. 2016]M.R. MOHAN M.S., N. N. e. a. A. A heuristic reference recursive recipe for adaptively tuning the kalman filter statistics part-1: formulation and simulation studies. *Indian Academy of Sciences Sādhanā*, v. 41, n. 4, p. 1473–1490, 2016.

[Myers & Tapley 1976]MYERS, K.; TAPLEY, B. Adaptive sequential estimation with unknown noise statistics. *IEEE Transactions on Automatic Control*, v. 21, n. 4, p. 520–523, 1976.

[Nan et al. 2022]NAN, F. et al. Nonlinear mpc for quadrotor fault-tolerant control. *IEEE RAL*, v. 7, n. 2, p. 5047–5054, 2022.

[Nascimento & Saska 2019]NASCIMENTO, T.; SASKA, M. Position and attitude control of multi-rotor aerial vehicles: A survey. *Annual Reviews in Control*, v. 48, 08 2019.

[Niemiec & Gandhi 2016]NIEMIEC, R.; GANDHI, F. Effects of inflow model on simulated aeromechanics of a quadrotor helicopter. In: *Proceedings of the 72nd American Helicopter Society Annual Forum*. [S.l.: s.n.], 2016. p. 1–13.

[Ning, Wlezien & Hu 2017]NING, Z.; WLEZIEN, R. W.; HU, H. An experimental study on small uav propellers with serrated trailing edges. In: *47th AIAA Fluid Dynamics Conference*. [S.l.: s.n.], 2017. p. 1–17.

[Okada et al. 2021]OKADA, K. F. Ávila et al. A survey on fault detection and diagnosis methods. In: *2021 14th IEEE IINDUSCON*. [S.l.: s.n.], 2021. p. 1422–1429.

[Omari et al. 2013]OMARI, S. et al. Nonlinear control of vtol uavs incorporating flapping dynamics. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. [s.n.], 2013. p. 2419–2425. Disponível em: <https://doi.org/10.1109/IROS.2013.6696696>.

[Oppenheimer, Doman & Sigthorsson 2010]OPPENHEIMER, M.; DOMAN, D.; SIGTHORSSON, D. Dynamics and control of a biomimetic vehicle using biased wingbeat forcing functions: Part i - aerodynamic model. In: *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*. [S.l.: s.n.], 2010.

[Park et al. 2020]PARK, J.-H. et al. Real-time quadrotor actuator fault detection and isolation using multivariate statistical analysis techniques with sensor measurements. In: *2020 20th International Conference on Control, Automation and Systems (ICCAS)*. [S.l.: s.n.], 2020. p. 33–37.

[Pounds 2007]POUNDS, P. *Design, Construction and Control of a Large Quadrotor Micro Air Vehicle*. Australian National University, 2007. Disponível em: <https://books.google.com.br/books?id=hBDqSAAACAAJ>.

[Pounds, Mahony & Corke 2006]POUNDS, P.; MAHONY, R.; CORKE, P. Modelling and control of a quad-rotor robot. 12 2006.

[Pounds, Mahony & Corke 2010]POUNDS, P.; MAHONY, R.; CORKE, P. Modeling and control of a large quadrotor robot. *Control Engineering Practice*, v. 18, p. 691–699, 07 2010.

[Pounds et al. 2004]POUNDS, P. et al. Towards dynamically-favourable quad-rotor aerial robots. In: BARNES, N.; AUSTIN, D. (Ed.). *Proceedings of the 2004 Australasian Con-*

ference on Robotics and Automation. Australia: The Australian Robotics and Automation Association Inc., 2004. p. 1–10.

[Powers et al. 2013]POWERS, C. et al. *Influence of Aerodynamics and Proximity Effects in Quadrotor Flight*. Heidelberg: Springer International Publishing, 2013. 289–302 p.

[Rashad et al. 2020]Rashad, R. et al. Fully actuated multirotor uavs: A literature review. *IEEE Robotics Automation Magazine*, v. 27, n. 3, p. 97–107, 2020.

[Reddinger & Gandhi 2017]REDDINGER, J.-P.; GANDHI, F. Neural network and machine learning allocation of redundant controls for power optimization on a compound helicopter. In: *AHS International 73rd Annual Forum & Technology Display, Fort Worth, Texas, USA*. [S.l.: s.n.], 2017. p. 1–13.

[Riether 2016]RIETHER, F. *Agile quadrotor maneuvering using tensor-decomposition-based globally optimal control and onboard visual-inertial estimation*. Tese (Doutorado) — Massachusetts Institute of Technology, 2016.

[Rot, Hasan & Manoonpong 2020]ROT, A. G.; HASAN, A.; MANOONPONG, P. Robust actuator fault diagnosis algorithm for autonomous hexacopter uavs. *IFAC-PapersOnLine*, v. 53, n. 2, p. 682–687, 2020. ISSN 2405-8963. 21st IFAC World Congress.

[Rudin, Ducard & Siegwart 2020]RUDIN, K.; DUCARD, G. J. J.; SIEGWART, R. Y. Active fault-tolerant control with imperfect fault detection information: Applications to uavs. *IEEE Transactions on Aerospace and Electronic Systems*, v. 56, n. 4, p. 2792–2805, 2020.

[Sabatino 2015]SABATINO, F. *Quadrotor control: modeling,nonlinear control design, and simulation*. Tese (Doutorado) — KTH Electrical Engineering, 2015.

[Schijndel, Sun & Visser 2021]SCHIJNDEL, B. A. S. van; SUN, S.; VISSER, C. de. Fast fault detection on a quadrotor using onboard sensors and a kalman filter approach. *CoRR*, abs/2102.06439, 2021. Disponível em: <https://arxiv.org/abs/2102.06439>.

[Schneider et al. 2012]SCHNEIDER, T. et al. Fault-tolerant control allocation for multirotor helicopters using parametric programming. In: . [S.l.: s.n.], 2012.

[Shi et al. 2019]Shi, G. et al. Neural lander: Stable drone landing control using learned dynamics. In: *2019 International Conference on Robotics and Automation (ICRA)*. [S.l.: s.n.], 2019. p. 9784–9790.

[Smeur, Höppener & Wagter 2017]SMEUR, E.; HÖPPENER, D.; WAGTER, C. Prioritized control allocation for quadrotors subject to saturation. In: *International Micro Air Vehicle Conference and Flight Competition 2017*. [S.l.: s.n.], 2017. p. 37–43.

[Smeur, Höppener & Wagter 2017]SMEUR, E.; HöPPENER, D.; WAGTER, C. D. Prioritized control allocation for quadrotors subject to saturation. In: . [S.l.: s.n.], 2017.

[Stolk 2017]STOLK, R. Minimum drag control allocation for the innovative control effector aircraft: Optimal use of control redundancy on modern fighters. In: . [S.l.: s.n.], 2017.

[Sun 2020]SUN, S. *Quadrotor Fault Tolerant Flight Control and Aerodynamic Model Identification*. Tese (Doutorado) — Delft Univ. of Technology, 2020.

[Sun et al. 2021]SUN, S. et al. Autonomous quadrotor flight despite rotor failure with onboard vision sensors: Frames vs. events. *IEEE RAL*, v. 6, n. 2, p. 580–587, 2021.

[Sun et al. 2018]SUN, S. et al. High-speed flight of quadrotor despite loss of single rotor. *IEEE RAL*, v. 3, n. 4, p. 3201–3207, 2018.

[Veras et al. 2019]VERAS, F. C. et al. Eccentricity failure detection of brushless dc motors from sound signals based on density of maxima. *IEEE Access*, v. 7, p. 150318–150326, 2019.

[Verberne & Moncayo 2019]Verberne, J.; Moncayo, H. Robust control architecture for wind rejection in quadrotors. In: *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. [S.l.: s.n.], 2019. p. 152–161.

[Wang et al. 2020]WANG, B. et al. Real-time fault detection for uav based on model acceleration engine. *IEEE Transactions on Instrumentation and Measurement*, v. 69, n. 12, p. 9505–9516, 2020.

[Wang & Puig 2016]WANG, Y.; PUIG, V. Zonotopic extended kalman filter and fault detection of discrete-time nonlinear systems applied to a quadrotor helicopter. In: *2016 3rd Conference on Control and Fault-Tolerant Systems (SysTol)*. [S.l.: s.n.], 2016. p. 367–372.

[Wang, Zhang & Han 2016]Wang, Y.; Zhang, H.; Han, D. Neural network adaptive inverse model control method for quadrotor uav. In: *2016 35th Chinese Control Conference (CCC)*. [S.l.: s.n.], 2016. p. 3653–3658.

[Waters et al. 2013]WATERS, S. et al. Control allocation performance for blended wing body aircraft and its impact on control surface design. *Aerospace Science and Technology*, v. 29, p. 18–27, 08 2013.

[Xu et al. 2018]XU, F. et al. Mixed active/passive robust fault detection and isolation using set-theoretic unknown input observers. *IEEE Transactions on Automation Science and Engineering*, v. 15, n. 2, p. 863–871, 2018.

[Xulin & Yuying 2018]XULIN, L.; YUYING, G. Fault tolerant control of a quadrotor uav using control allocation. In: *2018 Chinese Control And Decision Conference (CCDC)*. [S.l.: s.n.], 2018. p. 1818–1824.

[Zhang et al. 2021]ZHANG, X. et al. Time-domain frequency estimation with application to fault diagnosis of the uavs blade damage. *IEEE TIE*, p. 1–1, 2021.

[Zhao, Liu & Lewis 2021]ZHAO, W.; LIU, H.; LEWIS, F. L. Data-driven fault-tolerant control for attitude synchronization of nonlinear quadrotors. *IEEE Transactions on Automatic Control*, v. 66, n. 11, p. 5584–5591, 2021.

[Zhong et al. 2018]ZHONG, Y. et al. Robust actuator fault detection and diagnosis for a quadrotor uav with external disturbances. *IEEE Access*, v. 6, p. 48169–48180, 2018.