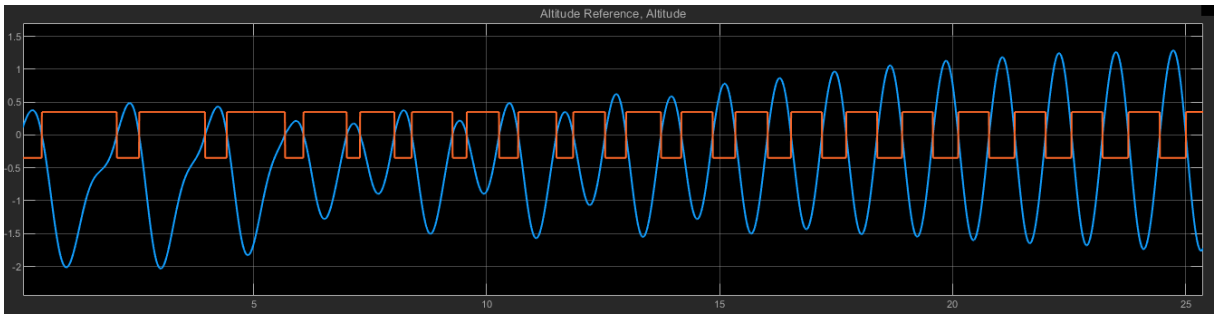
TÉCNICO LISBOA



UAV Relay-based Auto-tuning

Tuning the PID controller for a fixed-wing UAV via the relay method

Daniel de Schiffart

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisors: Marijn Hoogendoorn
Rita Maria Mendes de Almeida Correia da Cunha

Examination Committee

Chairperson: José Fernando Alves da Silva
Supervisor: Rita Maria Mendes de Almeida Correia da Cunha
Member of the Committee: Bruno João Nogueira Guerreiro

November 2019

Cover image · Relay feedback experiment with the desired altitude as input and commanded pitch angle as output

Acknowledgements

First, a few words of acknowledgement to those who made all of this possible.

To the NLR for having me and allowing me to develop this thesis project with their support, and my colleagues at the company for making me feel at home in a country I still struggle to call my own. Dank jullie allemaal!

To professor Rita, who accepted to supervise a thesis from a thousand kilometers away without even hesitating, and to Marijn, the toughest ping-pong opponent I could ask for, as patient as a supervisor can be even through all the language barriers, without whom this thesis probably wouldn't even be finished in the first place. Thanks for all the close ping-pong rounds (and the less close ones whose results I will not mention for my own sake).

To my friends and brothers in arms at IST, who dragged me through what felt like five hopeless years. Here's to finding out which one of us becomes a billionaire first.

To my family, to my mom who keeps supporting me from an ever-increasing number of kilometers away (two flights away already!), and my dad for always supporting me even after I just decided to crash at his home for these last six months and counting.

Abstract

This thesis presents the study of the viability of an automatic tuner for a fixed-wing UAV PID-based controller. The tuner is based on the relay feedback method, which is commonly applied in industrial processes and can be used to tune individual PID control loops of an aircraft of any kind, even during a flight, with little to no input from the pilot. In this thesis, a mathematical model of a fixed wing UAV is developed in parallel with a virtual relay auto-tuner. Both are implemented in Simulink®, and then tested together to determine the ideal values for a PID controller used to control this specific UAV model. The values obtained by the controller are then applied to the mathematical model of the unmanned aerial vehicle (UAV) and their results discussed upon.

Keywords: PID controlled flight, fixed-wing UAV, relay feedback, auto-tuning

Resumo

Esta tese estuda a viabilidade de um afinador automático para um controlador PID de um UAV de asa fixa através do método de relay. Este método, mais utilizado em processos industriais, pode ser utilizado para afinar *loops* individuais de um controlador PID de um avião arbitrário, mesmo durante o voo, com pouco ou nenhum *input* do piloto. Nesta tese é desenvolvido um modelo de um UAV em paralelo com um auto-afinador virtual baseado no método relay, ambos desenvolvidos em Simulink®, e depois testados em conjunto para determinar os parâmetros do controlador desse UAV. Os resultados obtidos pelo afinador são depois aplicados ao modelo do UAV e o seu efeito é observado e discutido.

Palavras-Chave: Voo controlado por PID, UAV de asa fixa, feedback por relay, auto-afinação

Contents

Contents ix

List of Figures xii

List of Tables xiv

Glossary xv

List of Abbreviations xvi

List of Symbols xvii

1 Introduction 1

1.1 Thesis Layout . . . 2

2 UAV Modelling 4

2.1 Coordinate Frames . . . 4

2.1.1 Transformation between coordinate frames . . . 5

2.1.2 The Earth-fixed frame . . . 7

2.1.3 Vehicle frame . . . 7

2.2 Equations of Motion . . . 8

2.2.1 Describing the state . . . 8

2.2.2 Applying the dynamics . . . 9

2.3 Forces and Moments . . . 12

2.3.1 Gravity . . . 12

2.3.2 Aerodynamics . . . 13

2.3.3 Thrust . . . 16

2.4 Full Equations of Motion . . . 16

2.5 Linearized Model . . . 17

3 Flight Control 19

3.1 Dynamic Flight Modes . . . 19

3.1.1 Longitudinal modes . . . 19

3.1.2 Lateral modes . . . 20
3.2 Flying and Handling Qualities . . . 20
3.2.1 Classifications . . . 21
3.2.2 Flying qualities . . . 22
3.2.3 Flying and handling quality requirements . . . 22
3.3 Feedback Loops . . . 25
3.3.1 *Elevator from Desired Pitch* . . . 26
3.3.2 *Aileron from Desired Roll* . . . 26
3.3.3 *Rudder from Y Acceleration* . . . 27

4 Auto-tuning via Relay Feedback **29**
4.1 The PID Controller . . . 29
4.1.1 Basic Functionality . . . 30
4.2 Ziegler-Nichols Tuning Rules . . . 32
4.2.1 Frequency Analysis . . . 32
4.2.2 Tuning rules . . . 33
4.3 Relay Feedback . . . 34
4.3.1 The relay component . . . 35
4.4 Improvements to the Relay Experiment . . . 37
4.4.1 Relay with hysteresis . . . 37
4.4.2 Saturation relay . . . 38
4.4.3 Relay with preload . . . 38
4.5 Locating Different Points . . . 39
4.5.1 Time-delayed relay response . . . 39
4.5.2 Relay with integrator . . . 40
4.5.3 Relay with hysteresis . . . 41
4.6 Tuning Methods . . . 41

5 Simulink® Implementation Platform **43**
5.1 Flight Simulation . . . 43
5.2 Relay Auto-tuner . . . 44
5.3 Implementation . . . 46

6 MyTwinDream Implementation **47**
6.1 UAV Characteristics . . . 47
6.1.1 Aerodynamic coefficients . . . 47
6.1.2 Thrust model . . . 47
6.2 Mathematical Model . . . 49
6.2.1 Linearized model . . . 49
6.2.2 Derived transfer functions . . . 50

6.3 UAV Dynamics . . . 50
6.3.1 Flying and handling qualities . . . 51
6.4 Auto-tuning Rules . . . 52
6.5 Tests . . . 53
6.5.1 Basic experiment . . . 54

7 Results and Discussion **56**
7.1 The Relay Experiment . . . 56
7.2 Tuning Results . . . 58
7.2.1 Step tests . . . 59
7.3 Conclusions . . . 61
7.4 Considerations Post-Work . . . 61
7.4.1 Future development . . . 62
7.4.2 Next steps . . . 62
7.4.3 Further reading . . . 62

Bibliography **64**

A Aircraft Parameters **67**
A.1 MyTwinDream parameters . . . 67

List of Figures

2.1 Rotation of a coordinate frame ϕ radians around one axis. 6
2.2 Reference frame used to calculate the dynamics on the UAV. 10
2.3 Control surfaces for a standard fixed-wing UAV. 15

3.1 Requirements for the UAV short-period damping. 23
3.2 Requirements for the short-period natural frequency requirements. 24
3.3 Requirements for the UAV phugoid damping. 25
3.4 Requirements for the UAV roll mode time constant for class I UAVs. 25

4.1 Basic feedback loop. 30
4.2 Basic feedback loop using PID control in standard form. 30
4.3 Basic feedback loop using PID control in parallel form. 31
4.4 Nyquist plot for an arbitrary feedback loop. 33
4.5 Feedback loop used for relay auto-tuning. 35
4.6 Relation between input and output for an ideal or *on-off* relay with output value d, without hysteresis applied. 35
4.7 Relay experiment example output over time. 36
4.8 Nyquist curve for the loop of an arbitrary system multiplied by its critical gain k_u. 37
4.9 Different types of relay and their input-output relation. 38
4.10 Feedback loop with a relay and a preload gain. 39
4.11 Feedback loop with a relay and an integrator used to determine a different point on the Nyquist curve. 40

5.1 The Simulink® block for the MTD UAV. 43
5.2 The overall structure used to simulate the flight of a UAV block. 44
5.3 The main layout of the inside of the relay auto-tuner block. 45
5.4 The frequency and amplitude measurement system of the relay feedback process. 45

6.1 The MyTwinDream UAV with its propeller blades removed. 48
6.2 SunnySky X2814-KV900 thrust force for a single engine based on throttle command and height. 48

6.3 Plot for the adjustment of the α parameter of the Åström and Hägglund tuning rule for the MTD aircraft. 53

7.1 Behaviour of the control surfaces and measured variables during the simulated relay feedback experiment. 57

A.1 Lift, drag, and moment coefficient to angle of attack α plots for the MTD aircraft. 68

List of Tables

3.1 Flight categories and examples of fitting flight scenarios. 22
3.2 Requirements for the UAV spiral mode for class I UAVs. 23
3.3 Requirements for the UAV dutch-roll properties for class I UAVs. 25

4.1 Ziegler-Nichols frequency-based tuning rules. 34

6.1 SunnySky X2814-KV900 characteristics for sea-level ($h = 0$ m) flight. 48
6.2 Trimmed level flight conditions for the MTD linearized model. 49
6.3 Tuning rules selected for this thesis. 52
6.4 Aircraft control surface sign conventions. 54

7.1 Relay experiment results for the MTD UAV. 58
7.2 Tuning results for the analysed feedback loops. 59

A.1 Parameters for the MTD aircraft. 67

Glossary

MATLAB®	Numerical computing software package used throughout the project
MicroPilot	Canadian autopilot manufacturer, provider of the autopilot used in this thesis project
model	Mathematical description of an aircraft and the relation between aerodynamic forces and moments and measurable flight and aircraft conditions
MP2128^{LRC2}	Model of the MicroPilot autopilot used in this thesis project
MyTwinDream	Small styrofoam model aircraft used for testing the developed framework
NLR	*Koninklijk Nederlands Lucht- en Ruimtevaartcentrum* or Royal Netherlands Aerospace Centre
SCALAiR	Short for *Scaled Aircraft Research*, the overarching NLR project in which this thesis project is included
Simulink®	Graphical programming environment integrated into MATLAB® used to develop and test models of the aircraft
SunnySky	Chinese manufacturer of the engines used in the MTD aircraft
tuning	Process of mathematically or experimentally determining the optimal gains for a PID controller based on pre-determined performance criteria
X2814-KV900	Model of the SunnySky engines used in the MTD aircraft

List of Abbreviations

AGL	height above ground level
DOF	degree of freedom
FOLPD	first-order lag plus time delay
HWIL	hardware-in-the-loop
LHP	left half-plane
MIMO	multiple-input, multiple-output
MTD	MyTwinDream
NED	north-east-down
PD	proportional-derivative
PI	proportional-integral
PID	proportional-integral-derivative
PIO	pilot-induced oscillations
RC	remote-controlled
RHP	right half-plane
RPV	remotely piloted vehicle
SAS	stability augmentation system
SFD	scaled flight demonstrator
SI	*système international (d'unités)*
SISO	single-input, single-output
SOSPD	second-order system plus time delay
UAV	unmanned aerial vehicle

List of Symbols

Latin alphabet symbols

a	Relay feedback oscillation amplitude
$\bar{a}$	Saturation relay input maximum
A_m	Ampltiude margin
b	Wing span
$\bar{c}$	Mean aerodynamic chord
d	Relay output
f_u	Ultimate frequency
g	Gravitational acceleration
J_{xx}	Moments and products of inertia
k	Saturation relay slope
k_p	Proportional gain (parallel form)
k_i	Integral gain
k_d	Derivative gain
k_{dd}	Feed-forward gain
k_c	Proportional gain (standard form)
k_u	Ultimate gain
m	Mass
p	Roll rate
q	Pitch rate
r	Yaw rate
S	Wing area
T_d	Derivative time
T_i	Integration time
T_u	Ultimate period
u	Body longitudinal velocity
v	Body lateral velocity
V_a	Airspeed

w Body down velocity

Greek alphabet symbols

α Angle of attack

β Sideslip angle

γ Flight path angle

δ_a Aileron deflection

δ_e Elevator deflection

δ_r Rudder deflection

δ_t Thrust throttle

ϵ Relay hysteresis

ζ Damping

θ Pitch angle

φ Phase shift

ρ Air density

ϕ Roll angle

ϕ_m Phase margin

χ Course angle

ψ Yaw angle

ω_n Natural frequency

ω_u Ultimate radial frequency

Non-dimensional coefficients

C_D Drag coefficient

C_L Lift coefficient

C_X Force coefficient along the x axis

C_Y Force coefficient along the y axis

C_Z Force coefficient along the z axis

C_l Moment coefficient around the x axis

C_m Moment coefficient around the y axis

C_n Moment coefficient around the z axis

Subscripts

d Two-dimensional drag

D Drag

dr Dutch-roll mode

l Two-dimensional lift

L Lift

m Moment

phu Phugoid mode

ref Reference or desired input

r Roll mode

sp Short-period mode

s Spiral mode

Chapter 1

Introduction

The concept of scaled flight testing allows for affordable testing of novel concepts to be implemented in large-scale aircraft without the costs involved in performing tests on the aircraft itself. The careful study and construction of scaled models of large-scale aircraft allows the results to be obtained and their conclusions to be transposed to their full-scale counterparts without a considerable loss in accuracy, even when testing within a wind tunnel. By using scaled fixed-wing unmanned aerial vehicles (UAVs), these models can be taken from the wind tunnel and flown in more realistic flight scenarios, while still fulfilling the low risk and cost premise of scaled flight testing.

However, flight control for a scaled UAV is often implemented with a lot less complexity than its full-scale counterparts. While the cause for this may be a multitude of factors ranging from budget restrictions to differences in objectives, this means that aside from the scaling factors used to relate multiple control variables between UAVs and full-size aircraft, the control methods used and implemented in the latter cannot be directly transposed and used in the former. Looking at the control technology available for different complexities, at the lower end of the spectrum enters the proportional-integral-derivative (PID) controller, which is according to Åström and Hägglund, the 'workhorse' of the control industry.

The usage of this type of controller in the control industry is near universal [25]. At lower control complexities, this controller fulfills most of the requirements necessary for every situation through a simple and affordable implementation, and is present as a building block for more complex control systems. In the scenario of flight control, it becomes the ideal controller to use in the type of scaled fixed-wing UAVs used in scaled flight testing for these very reasons.

However, PID tuning is a debated and diverse field. For lower risk scenarios, the trial-and-error approach is the most straightforward way to solve a control problem, at the cost of accuracy and time per tuning. Tuning based on a model of a process to tune is a more common approach, but does indeed require an accurate version of the model to tune and carries over errors within its implementation, often caused by the diverse approximations and linearizations performed.

In 1942, Ziegler and Nichols suggested an approach made tuning by hand much faster and more efficient than trial-and-error. They suggested taking a certain feedback loop and increasing its value until the system is brought to an unstable condition, and using the maximum value of the gain prior to this

point to set the values of the remaining parameters via fixed, but well tested, linear relations. While not fully automating the tuning of a specific feedback loop, this improved the control of most of the situations in which PID controllers were applied. With them being low cost and low risk implementations, often requiring far less accuracy than a well-tuned PID can provide, the approach was quickly adopted in many fields of application. Due to the generalized approach of this method based on empirical relations, many PID controllers within the industry were found being poorly tuned or not tuned at all, often being left with their factory parameters for their usage lifetime [3].

Tuning a PID controller automatically was the logical next step. Based on the work of Ziegler and Nichols, Åström and Hägglund proposed the relay auto-tuner in 1984, which makes use of a relay component and the properties used by the original Ziegler and Nichols rules to automate the tuning process of a PID controller. It uses the input and output of a process to tune its controller, similar to the aforementioned Ziegler and Nichols rules, but avoids bringing said process to a near unstable point, which is undesired in more delicate scenarios. In the time since, multiple improvements and adjustments have been made to this method, but the essential idea found its way to multiple commercial implementations of PID controllers.

In the aerospace industry, their implementations are overlooked. While some studies have been made in the past [24], the most complete study was made by Poksawat, Wang and Mohamed [22] in which the authors successfully find the dynamics of a UAV based on the relay experiment to tune said UAV, but the careful study and availability of accurate large-scale aircraft models and the hobby-like nature of UAV flight mean that often the tuning for these vehicles is done manually [16]. Some modern tuning algorithms are also surging in the field [30], most notably the algorithms developed for the open-source ArduPilot [29], but the relay approach suggested by Åström and Hägglund is sparsely tested and documented.

The project developed for this thesis is included in this context. A virtual model of a relay-based automatic PID tuner (henceforth referred to as relay auto-tuner) will be developed alongside (but independent from) mathematical models of different aircraft, and their results will be tested on simulated implementation scenarios. Conclusions will be taken on the viability of the implementation of relay-based auto-tuning on PID-controlled UAVs.

1.1 Thesis Layout

This thesis will first introduce all the theoretical concepts involved to carry out its intended goal in the first half, establishing the basis of implementation without depending on any particular implementation platform or target UAV or flying conditions. Chapter 2 focuses on developing the virtual UAV model used for any computational tests, approximating its properties as close as possible to theoretically perfect flight conditions. This model will be built based on several equations for a UAV's kinematics and dynamics, relating its movement through the air to the forces that act on it. Chapter 3 introduces classical control theory concepts and the way these relate to the flight of a UAV, and presents how these are analysed to provide requirements and goals for the design of a generic UAV controller. Section 4.1 gives a detailed explanation on the functioning of a PID controller to lay down the basis for how it can be tuned and how it

should be tuned. Chapter 4 contains the core of the presented concept, and is where the theory behind an automated PID tuner is introduced.

After the first part of the thesis introduces all the concepts, the second part focus purely on the platforms of implementation and their specific properties, how their properties changed the direction of the thesis project, the execution of the ideas presented previously and a discussion on how to interpret the results and use them to further develop the concepts here introduced in the future. Chapter 5 shows the developed platform to carry out the tests based on the theory presented, most notably Chapter 4, as a generic relay-based auto-tuning PID block was developed in Simulink® for implementation in this thesis, but could be very much used in other projects. Chapter 6 presents the MyTwinDream UAV, the main implementation platform for this thesis project, presents the data used in the Simulink® test platform, and writes the information on how the tests should be carried out for this particular UAV. Chapter 7 presents the results of the tests outlined in Chapter 6 and details how to interpret them, following that up with an in-depth discussion on the various challenges this project introduced and how it should be explored following the work carried out in this thesis.

Chapter 2

UAV Modelling

Although the planned auto-tuner does not depend on prior knowledge of the UAV and its processes, in order to implement and test the auto-tuner prior to actual implementation, a virtual dynamic model of the UAV to test has to be developed. This virtual model should use a set of initial flight conditions and control inputs to mathematically determine the aircraft's behaviour through time, effectively approximating how the real UAV might respond to different inputs and flight conditions in a real flight scenario.

A model of a UAV is based on expected behaviour from an aircraft based on typical aircraft layouts and previously studied and described physical relations between its subsystems and other frames of reference. These models are developed based on available literature discussing the topic, such as Beard and McLain [5], Stevens and Lewis [28], or Mulder et al. [17], and the development of a UAV model suitable for the project at hand will be described in this chapter. Later on, in Section 4.6, real aerodynamic and geometric UAV data will be used to create mathematical models of these UAVs to analyse their behaviour in certain simplified flight conditions.

2.1 Coordinate Frames

To create a virtual model of the UAV and its dynamics across physical space, it is important to define a frame of reference around which to measure said dynamics. In this first section, the UAV is considered as a single point of mass in space around which all other parts of the UAV translate and rotate around, acting as an on-board frame of reference, usually placed at the centre of gravity of the UAV. This point is then measured in a separate frame of reference at a further distance, to simulate its translation and rotation with relation to the Earth.

Since the UAV will be simplified as a point of mass until Section 2.5, any forces and moments applied on the UAV will be ignored except for the force of gravity. This approximation can be done based on the assumption that the UAV is a rigid body, meaning all its points are considered to be in the same position relative to each other at all times, and also that it is a body of constant mass, an assumption that can be held true by the nature of electric propulsion used in the UAV analysed in this project.

A number of different coordinate frames can be presented for an aircraft's position and rotation,

depending on the detail required on the kinematic and dynamic analysis and the information required. They will be presented and discussed in this section, after a short presentation on the transformation of vectors between any two related coordinate frames.

2.1.1 Transformation between coordinate frames

Consider a mass with position at point $\boldsymbol{p}$ with coordinates (p_x, p_y, p_z) within a specific coordinate frame A. The position of point $\boldsymbol{p}$ can be described using the unit vectors $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$ of coordinate frame A as

$$\boldsymbol{p} = p_x \boldsymbol{i} + p_y \boldsymbol{j} + p_z \boldsymbol{k} \,.$$

Introducing a second frame of reference B, the position of point $\boldsymbol{p}$ can be described in both frames similarly. To distinguish between both frames of reference, a vector is superscripted with its corresponding coordinate frame, and the same point can be represented as

$$\boldsymbol{p} = p_x^A \boldsymbol{i}^A + p_y^A \boldsymbol{j}^A + p_z^A \boldsymbol{k}^A \boldsymbol{p} = p_x^B \boldsymbol{i}^B + p_y^B \boldsymbol{j}^B + p_z^B \boldsymbol{k}^B \,.$$

With it being the same point, it can be shown that

$$p_x^A \boldsymbol{i}^A + p_y^A \boldsymbol{j}^A + p_z^A \boldsymbol{k}^A = p_x^B \boldsymbol{i}^B + p_y^B \boldsymbol{j}^B + p_z^B \boldsymbol{k}^B \,.$$

To remove the unit vectors on the right side, the dot product of the unit vector $(\boldsymbol{i}^B, \boldsymbol{j}^B, \boldsymbol{k}^B)$ is applied to the equation and changed into the matrix form to get

$$\begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} = \begin{bmatrix} \boldsymbol{i}^B \cdot \boldsymbol{i}^A & \boldsymbol{i}^B \cdot \boldsymbol{j}^A & \boldsymbol{i}^B \cdot \boldsymbol{k}^A \\ \boldsymbol{j}^B \cdot \boldsymbol{i}^A & \boldsymbol{j}^B \cdot \boldsymbol{j}^A & \boldsymbol{j}^B \cdot \boldsymbol{k}^A \\ \boldsymbol{k}^B \cdot \boldsymbol{i}^A & \boldsymbol{k}^B \cdot \boldsymbol{j}^A & \boldsymbol{k}^B \cdot \boldsymbol{k}^A \end{bmatrix} \begin{bmatrix} p_x^A \\ p_y^A \\ p_z^A \end{bmatrix} \,. \tag{2.1}$$

This describes succinctly the relation between both coordinate frames, with the matrix in the center being the transformation matrix between them.

If the transformation between two reference frames is a simple right-hand rotation along one axis described by an angle ϕ, like the one displayed in Figure 2.1 the matrix from Equation (2.1) can be simplified to get a *rotation matrix*, defined as

$$R_A^B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \,, \tag{2.2}$$

where the subscript defines the original reference frame and the superscript the new reference frame. If the same is done for a rotation to coordinate frame C about a different axis of A, in this case y, for an

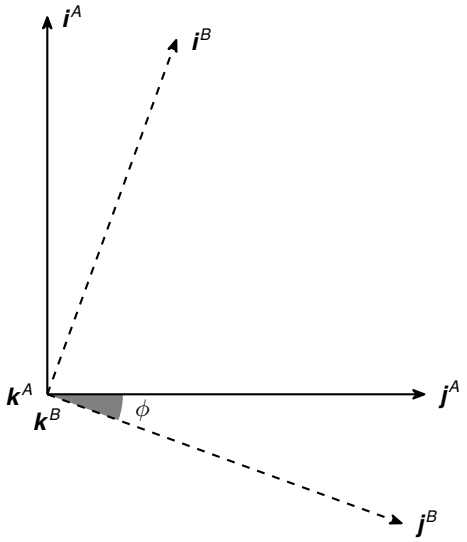


Figure 2.1: Rotation of a coordinate frame ϕ radians around one axis.

angle θ, the result becomes

$$R_A^C = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}. \tag{2.3}$$

The same can be done for the z axis, this time with angle ψ, to coordinate frame D, written as

$$R_A^D = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.4}$$

Should multiple rotations be required, like a coordinate frame E that is ϕ degrees around x and θ degrees around y of coordinate frame A, they can be written as a rotation from A to B and from B to E as a matrix multiplication

$$R_A^E = R_A^C R_A^B.$$

When all matrices are multiplied together, the standard three-degree of freedom (DOF) rotation matrix that uses the rotation matrices in Equations (2.2) to (2.4) is obtained, which is written as a multiplication of all matrices

$$R_A^F = R_A^B R_A^C R_A^D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.5}$$

which when written in full is

$$R_A^F = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix}. \tag{2.6}$$

This is the full rotation matrix between coordinate frames defined by the *Euler angles* ϕ, θ and ψ and will be used later on whenever rotation between coordinate frames is mentioned.

2.1.2 The Earth-fixed frame

The first and more common coordinate frame is the inertial or Earth-fixed frame, which considers a fixed point on the Earth's ground as its origin, with its ***i*** vector pointed to the Earth's north, ***j*** to east, and ***k*** pointed down to the center of the Earth. This is also the reason this coordinate frame is also known as north-east-down (NED).

This coordinate frame is commonly used to analyse the translation of an aircraft through the air, but also to analyse its orientation to the ground, as the gravity is always a multiple of the ***k*** vector of this frame, and angles such as the aircraft's pitch θ and flight path angle γ use the Earth's horizon as defined in this coordinate frame as reference.

2.1.3 Vehicle frame

The vehicle frame is a coordinate frame that follows the UAV around, but with varying types of orientation. Different frames can be defined for different uses, with the common factor that their origin is on the center of the UAV. Here, each variant will be built on top of the last, with each transformation matrix presented, until the body frame is reached.

The first variant shares the same orientation as the Earth-fixed frame. The second variant shares the same orientation as the Earth-fixed frame, with the exception of the yaw angle, so its rotation from the first frame is defined as

$$R_1(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.7}$$

where ψ is the yaw angle.

The third variant is the same as the previous variant with the addition of a pitch angle θ, so

$$R_2(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}. \tag{2.8}$$

The fourth variant, also called the body frame, adds the roll angle, the last of the Euler angles. So,

transforming from the third variant,

$$R_3(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}. \tag{2.9}$$

Combining all the previous matrices yields the matrix seen in Equation (2.6), which can be used to convert a point from the first variant of the vehicle frame to the body frame.

The stability frame adds an angle of attack α, or the angle the UAV trajectory makes with the wind. From the body frame,

$$R_4(\alpha) = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}. \tag{2.10}$$

The wind frame adds a sideslip angle to the stability frame, an angle around the z axis of the stability frame, for any lateral mismatch between the UAV's angle and the wind angle, meaning the conversion looks similar to the yaw matrix, or

$$R_5(\beta) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.11}$$

2.2 Equations of Motion

The final objective of this chapter is to obtain a mathematical model of the UAV's motion in the air to be adapted into a simulation environment and, consequently, perform the relay feedback test on it via said simulations. This motion is often mathematically described as a set of relations and equations known as the **equations of motion**. These describe the relation between the UAV's position and velocity and the forces and moments applied to it, originating from its interaction and the interaction of its parts with the world around it.

In this section these relations will be described in the terms typically used in the aerospace industry and the equations will be derived from said relations. The final relations will resemble typical full-sized aircraft equations, with only their values (discussed in Section 4.6) being of a smaller scale for their UAV implementations.

2.2.1 Describing the state

To obtain the kinematic conditions of a UAV in any given point in time, only its position, velocity, angle, and angular rates are required. This allows for full prediction of the UAV status indefinitely in the absence of any forces or moments, and will allow for the creation of he simulation model once the effect of forces and moments are taken into account.

The position is measured in the Earth-fixed frame of reference described in Section 2.1.2, and the velocity in the body frame of reference discussed in Section 2.1.3. So, the derivative of the position is obtained using Equation (2.6) giving

$$\begin{bmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{p}_d \end{bmatrix} = R_A^F \begin{bmatrix} u \\ v \\ w \end{bmatrix} . \tag{2.12}$$

The angular rates are obtained through the conversions given in Section 2.1.3. While p, q, and r are given in the body frame, the Euler angles ϕ, θ, and ψ are used in separate transformations. So,

$$\begin{aligned} \begin{bmatrix} p \\ q \\ r \end{bmatrix} &= \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + R_3(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R_3(\phi) R_2(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} , \end{aligned}$$

which is inverted to obtain the angular rates returning

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\theta\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 1 & \sin\phi\sec\theta & \cos\theta\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} . \tag{2.13}$$

2.2.2 Applying the dynamics

Next up the forces and moments that affect the UAV at any point in time and how these affect the motion of the UAV will be introduced.

The forces and moments are separated into three of each, one per axis, for a normal three-dimensional reference frame. The dynamics will be analysed in the vehicle reference frame, seen in Figure 2.2, so the forces and moments are separated into their effect on each of its axes. All the forces and moments acting on the UAV are summed together and applied in its center of mass, and separated into three components: f_x, f_y, and f_z, for the forces, and l, m, and n, for the moments.

The force f_x is in line with the x axis, so it will directly affect the u speed. Both of these concepts can be connected via Newton's second law, so that

$$f_x = m\dot{u} ,$$

so $\dot{u}$ can be written as [5]

$$\dot{u} = (rv - qw) + \frac{1}{m} f_x , \tag{2.14}$$

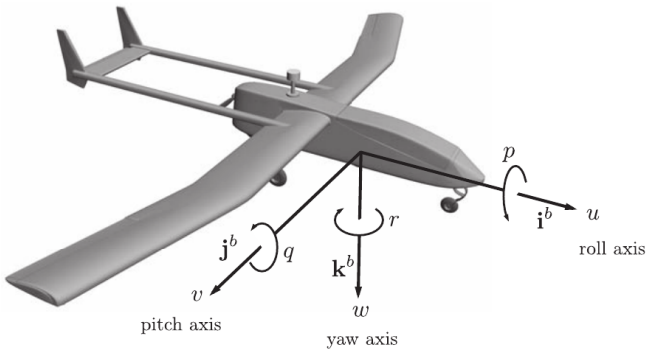


Figure 2.2: Reference frame used to calculate the dynamics on the UAV. Adapted from [5].

where the $(rv - qw)$ factor is added due to the body's angular velocity in relation to the Earth-fixed frame. v and w can be written in a similar way, as

$$\dot{v} = (pw - ru) + \frac{1}{m} f_y \tag{2.15}$$

$$\dot{w} = (qu - pv) + \frac{1}{m} f_z \,. \tag{2.16}$$

For the moments, a similar approach is taken, knowing that the vectorial sum of all moments applied to the body is equal to

$$\dot{\boldsymbol{h}} = \boldsymbol{m} \,, \tag{2.17}$$

where $\dot{\boldsymbol{h}}$ is the angular momentum vector. The rates are measured in the Earth-fixed frame, so to obtain the angular momentum in the body frame

$$\dot{\boldsymbol{h}^{\boldsymbol{B}}} + \omega^B_{B/I} \times \boldsymbol{h} = \boldsymbol{m}^B \,, \tag{2.18}$$

where the B superscript denotes the body frame of reference and B/I the body frame angular rate in relation to the inertial frame.

The angular momentum of a body is defined as the product of its inertia matrix $\boldsymbol{J}$ and its angular velocity vector ω, so

$$\boldsymbol{h} = \boldsymbol{J}\omega \,. \tag{2.19}$$

$\boldsymbol{J}$ is given by

$$\boldsymbol{J} = \begin{bmatrix} J_x & -J_{xy} & -J_{xz} \\ -J_{xy} & J_y & -J_{yz} \\ -J_{xz} & -J_{yz} & J_z \end{bmatrix} , \tag{2.20}$$

where the diagonal variables are the *moments of inertia* and the remaining variables are the *products of inertia*. The moments of inertia generally measure a body's tendency to rotate around a specific axis, and

are an intrinsic property of an aircraft, dependent of its shape and mass distribution.

Deriving Equation (2.19) returns

$$\dot{\boldsymbol{h}} = \boldsymbol{J}\dot{\omega}\,, \tag{2.21}$$

which when used in combination with Equation (2.18) yields

$$\boldsymbol{J}\dot{\omega} + \omega \times (\boldsymbol{J}\omega) = \boldsymbol{m}\,. \tag{2.22}$$

Rearranging this information the angular rate derivatives become

$$\dot{\omega} = \boldsymbol{J}^{-1}\left(-\omega \times (\boldsymbol{J}\omega) + \boldsymbol{m}\right)\,. \tag{2.23}$$

Since most UAV are symmetric by default, $\boldsymbol{J}$ can be heavily simplified as the asymmetric components are removed, giving

$$\boldsymbol{J} = \begin{bmatrix} J_x & 0 & -J_{xz} \\ 0 & J_y & 0 \\ -J_{xz} & 0 & J_z \end{bmatrix}\,, \tag{2.24}$$

which makes the task of inverting it much simpler, and ends up looking like

$$\boldsymbol{J}^{-1} = \begin{bmatrix} \frac{J_z}{J_x J_z - J_{xz}^2} & 0 & \frac{J_{xz}}{J_x J_z - J_{xz}^2} \\ 0 & \frac{1}{J_y} & 0 \\ \frac{J_{xz}}{J_x J_z - J_{xz}^2} & 0 & \frac{J_x}{J_x J_z - J_{xz}^2} \end{bmatrix}\,, \tag{2.25}$$

Now the angular rate derivatives can be written through Equation (2.23) as

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{J_z}{J_x J_z - J_{xz}^2} & 0 & \frac{J_{xz}}{J_x J_z - J_{xz}^2} \\ 0 & \frac{1}{J_y} & 0 \\ \frac{J_{xz}}{J_x J_z - J_{xz}^2} & 0 & \frac{J_x}{J_x J_z - J_{xz}^2} \end{bmatrix} \left(\begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix} \begin{bmatrix} J_x & 0 & -J_{xz} \\ 0 & J_y & 0 \\ -J_{xz} & 0 & J_z \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} + \begin{bmatrix} l \\ m \\ n \end{bmatrix} \right) \tag{2.26}$$

$$= \begin{bmatrix} \Gamma_1 pq - \Gamma_2 qr + \Gamma_3 l + \Gamma_4 n \\ \Gamma_5 pr - \Gamma_6 (p^2 - r^2) + \frac{1}{J_y} m \\ \Gamma_7 pq - \Gamma_1 qr + \Gamma_4 l + \Gamma_8 n \end{bmatrix}\,, \tag{2.27}$$

where the Γ variables were introduced to simplify writing. Their definition is given by

$$\begin{aligned} \Gamma_1 &= \frac{J_{xz}(J_x - J_y + J_z)}{J_x J_z - J_{xz}^2} & \Gamma_2 &= \frac{J_z(J_z - J_y) + J_{xz}^2}{J_x J_z - J_{xz}^2} & \Gamma_3 &= \frac{J_z}{J_x J_z - J_{xz}^2} \\ \Gamma_4 &= \frac{J_{xz}}{J_x J_z - J_{xz}^2} & \Gamma_5 &= \frac{J_z - J_x}{J_y} & \Gamma_6 &= \frac{J_{xz}}{J_y} \\ \Gamma_7 &= \frac{(J_x - J_y)J_x + J_{xz}^2}{J_x J_z - J_{xz}^2} & \Gamma_8 &= \frac{J_x}{J_x J_z - J_{xz}^2} \end{aligned}\,. \tag{2.28}$$

So with this information, the equations of motion can be written with the corresponding forces applied. First the position derivatives,

$$\begin{aligned}
\dot{p}_n &= (\cos\theta\cos\psi)u + (\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi)v + (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)w \\
\dot{p}_e &= (\cos\theta\sin\psi)u + (\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi)v + (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)w \\
\dot{p}_d &= -u\sin\theta + v\sin\phi\cos\theta + w\cos\phi\cos\theta\,,
\end{aligned}$$

the velocities

$$\begin{aligned}
\dot{u} &= rv - qw + \frac{1}{m}f_x \\
\dot{v} &= pw - ru + \frac{1}{m}f_y \\
\dot{w} &= qu - pv + \frac{1}{m}f_z\,,
\end{aligned}$$

the Euler angles

$$\begin{aligned}
\dot{\phi} &= p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\
\dot{\theta} &= q\cos\phi - r\sin\phi \\
\dot{\psi} &= q\sin\phi\sec\theta + r\cos\phi\sec\theta\,,
\end{aligned}$$

and the angular rates

$$\begin{aligned}
\dot{p} &= \Gamma_1 pq - \Gamma_2 qr + \Gamma_3 l + \Gamma_4 n \\
\dot{q} &= \Gamma_5 pr - \Gamma_6(p^2 - r^2) + \frac{1}{J_y}m \\
\dot{r} &= \Gamma_7 pq - \Gamma_1 qr + \Gamma_4 l + \Gamma_8 n\,.
\end{aligned}$$

2.3 Forces and Moments

The next step is determining the forces f_x, f_y, and f_z, and the moments l, m, and n, based on the aerodynamic properties of the UAV.

2.3.1 Gravity

The most basic force applied to the UAV at all times is the force of gravity. A simple model for gravity is used, a single force pointing down on the Earth-fixed frame. Due to the relatively low expected flight altitude, the acceleration due to gravity can be assumed to be constant throughout, and equal to the mean acceleration at the Earth's surface, or roughly $9.807\,\mathrm{m\,s^{-1}}$, a value henceforth represented by g. With

this in mind the force of gravity is determined using Newton's definition of force, so

$$f_g = mg\,,$$

where m is the UAV's mass.

With this in mind, the force now needs to be included in the equations of motion. The force of gravity can be simply represented by a positive force in the down direction in the Earth-fixed frame, so

$$f_g = \begin{bmatrix} 0 & 0 & mg \end{bmatrix}^{\mathsf{T}}\,.$$

To obtain this in the vehicle frame, the transformation from Equation (2.6) is applied, and so

$$\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = R_A^F \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} = \begin{bmatrix} -mg\sin\theta \\ mg\cos\theta\sin\phi \\ mg\cos\theta\cos\phi \end{bmatrix}\,. \tag{2.29}$$

2.3.2 Aerodynamics

The main components of the force are the aerodynamic effects that act on the UAV, which describe how it interacts with the air surrounding it and how this interaction generates the forces necessary to keep it in flight and how its behaviour and trajectory can be controlled. These descriptions use aeronautical notations and simplifications that will be briefly overviewed before applied to the equations of motion.

The aerodynamic force generated by an aircraft in flying conditions is caused by the pressure distribution throughout its entire body, most notably the wings, which are carefully designed to allow the aircraft to stay in flight. This pressure distribution can be carefully simulated to obtain accurate predictions of the forces acting on the entire aircraft, but for the context of designing a model to simulate its flight path in the air, the forces are associated with a scalar non-dimensional coefficient and the properties of the airflow around the aircraft.

The registered properties of the airflow are the air density ρ and airspeed V_a, and the response of an aircraft to it is summarized by various scalar coefficients, summarized into three coefficients for the forces and three for the moments. In this case, the aerodynamic forces and moments are written as

$$f_x = \frac{1}{2}\rho V_a^2 S C_X \tag{2.30}$$

$$f_y = \frac{1}{2}\rho V_a^2 S C_Y \tag{2.31}$$

$$f_z = \frac{1}{2}\rho V_a^2 S C_Z \tag{2.32}$$

$$l = \frac{1}{2}\rho V_a^2 S b C_l \tag{2.33}$$

$$m = \frac{1}{2}\rho V_a^2 S \bar{c} C_m \tag{2.34}$$

$$n = \frac{1}{2}\rho V_a^2 S b C_n\,, \tag{2.35}$$

where S represents the wing surface, b the wing span and $\bar{c}$ the mean aerodynamic chord. All the different ways the aerodynamics of the aircraft can be changed will be introduced into the model via the six coefficients C_X, C_Y, C_Z, C_l, C_m, and C_n.

Longitudinal forces and moments

The three main forces generated during steady flight are lift, drag, and pitching moment. Due to symmetry, these are aligned with the longitudinal forces and moments of the aircraft, along the x and z axis. These forces are determined by their own aerodynamic coefficients and their relation to the airflow, and are given by

$$L = \frac{1}{2}\rho V_a^2 S C_L \tag{2.36}$$

$$D = \frac{1}{2}\rho V_a^2 S C_D \tag{2.37}$$

$$m = \frac{1}{2}\rho V_a^2 S \bar{c} C_m\,, \tag{2.38}$$

where L and D represent the lift and drag forces respectively and m the moment around the aerodynamic center. These coefficients are related to the longitudinal aerodynamic coefficients C_X and C_Z by a two-dimensional coordinate conversion dependent on the angle of attack α given by

$$C_X = -C_D \cos\alpha + C_L \sin\alpha \tag{2.39}$$

$$C_Z = -C_D \sin\alpha - C_L \cos\alpha\,. \tag{2.40}$$

The values of C_L and C_D are form the core of the aerodynamic properties for steady level flight and will be given focus. Their values define the behaviour of the aircraft, so they are dependent on all the variables that can influence an aircraft's aerodynamics. In this thesis, the variables considered are the angle of attack α, the pitching moment q, but also the shape of the aircraft.

The shape of the aircraft and how the airflow affects the forces acting on it can be changed using the concept of *control surfaces*. These are parts of the surface of the aircraft that can be adjusted at will and change the aerodynamic forces and moments acting on it. These include, for a standard fixed-wing aircraft setup, the *elevator*, *ailerons*, and *rudder*, and are shown on Figure 2.3. Their effect on the aerodynamics of the aircraft is simplified into a direct dependency on their deflection angles δ_e, δ_a, and δ_r, respectively.

So with all things considered, if the effects of each variable on the aerodynamic coefficients are divided into their own linear subcoefficients then the lift and drag coefficients C_L and C_D can be given by

$$C_L = C_{L_0} + \frac{\partial C_L}{\partial \alpha}\alpha + \frac{\partial C_L}{\partial q}q + \frac{\partial C_L}{\partial \delta_e}\delta_e \tag{2.41}$$

$$C_D = C_{D_0} + \frac{\partial C_D}{\partial \alpha}\alpha + \frac{\partial C_D}{\partial q}q + \frac{\partial C_D}{\partial \delta_e}\delta_e\,, \tag{2.42}$$

where C_{L_0} and C_{D_0} represent the values of C_L and C_D when all the variables α, q and δ_e are zero. Making

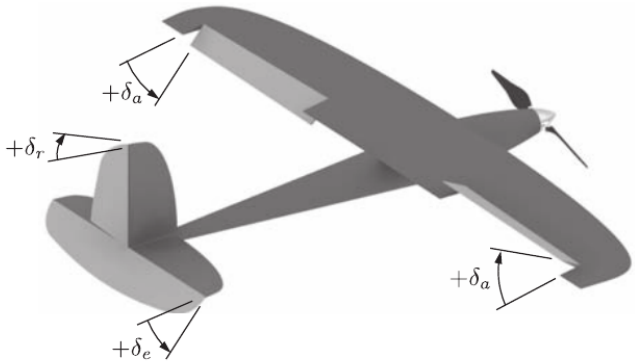


Figure 2.3: Control surfaces for a standard fixed-wing UAV. Adapted from [5].

the partial derivatives non-dimensional simplifies the calculations, so it can be written

$$C_L = C_{L_0} + C_{L_\alpha}\alpha + C_{L_q}\frac{\bar{c}}{2V_a}q + C_{L_{\delta_e}}\delta_e \tag{2.43}$$

$$C_D = C_{D_0} + C_{D_\alpha}\alpha + C_{D_q}\frac{\bar{c}}{2V_a}q + C_{D_{\delta_e}}\delta_e\,, \tag{2.44}$$

where the angle of attack and elevator deflection angles are already non-dimensional, and $\frac{\bar{c}}{2V_a}$ is a common factor for making angular rates non-dimensional. These coefficients are, in the case of this thesis project, generated by computer software, and C_{L_α} and C_{D_α} are replaced with non-linear variables $C_L(\alpha)$ and $C_D(\alpha)$ to allow for more accuracy in the calculation of these coefficients given by the software generation.

A similar procedure can be applied to the pitching moment coefficient C_m, which depends on the same variables, so

$$C_m = C_{m_0} + C_m(\alpha) + C_{m_q}\frac{\bar{c}}{2V_a}q + C_{m_{\delta_e}}\delta_e\,. \tag{2.45}$$

Lateral forces and moments

In a similar approach to the longitudinal forces, the coefficients can be separated into their separate components and analysed one by one. The lateral variables sideslip angle β, roll rate p, yaw rate r, and the control variables of aileron and rudder deflections δ_a and δ_r are used. So for lateral moments and forces,

$$C_Y = C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}\frac{b}{2V_a}p + C_{Y_r}\frac{b}{2V_a}r + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r \tag{2.46}$$

$$C_l = C_{l_0} + C_{l_\beta}\beta + C_{l_p}\frac{b}{2V_a}p + C_{l_r}\frac{b}{2V_a}r + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_r}}\delta_r \tag{2.47}$$

$$C_n = C_{n_0} + C_{n_\beta}\beta + C_{n_p}\frac{b}{2V_a}p + C_{n_r}\frac{b}{2V_a}r + C_{n_{\delta_a}}\delta_a + C_{n_{\delta_r}}\delta_r\,. \tag{2.48}$$

2.3.3 Thrust

The data for thrust force used in the simulations was assumed to be obtained from external sources, which would eventually be modelled into look-up tables of data and applied in the simulations as a single force, which is dependent on variables such as altitude and speed. The assumption is made that the engine producing the thrust force is in line with the UAV's x axis, such that the thrust force F_t can be simply be assumed to be a component of the f_x force.

A full description of the thrust force in this case can only be given on a per-case basis. For the MyTwinDream (MTD), check Section 6.1.2.

2.4 Full Equations of Motion

With all the previous sections taken into account, the full set of non-linear equations of motion used to simulate the aircraft can be written. Here are presented the position derivatives

$$\dot{p}_n = (\cos\theta\cos\psi)u + (\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi)v + (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)w \tag{2.49}$$

$$\dot{p}_e = (\cos\theta\sin\psi)u + (\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi)v + (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)w \tag{2.50}$$

$$\dot{p}_d = -u\sin\theta + v\sin\phi\cos\theta + w\cos\phi\cos\theta\,, \tag{2.51}$$

the velocities

$$\dot{u} = rv - qw - g\sin\theta + \frac{\rho V_a^2 S}{2m}\left(C_X(\alpha) + C_{X_q}\frac{\bar{c}}{2V_a}q + C_{X_{\delta_e}}\delta_e\right) \tag{2.52}$$

$$\dot{v} = pw - ru + g\cos\theta\sin\phi + \frac{\rho V_a^2 S}{2m}\left(C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}\frac{b}{2V_a}p + C_{Y_r}\frac{b}{2V_a}r + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r\right) \tag{2.53}$$

$$\dot{w} = qu - pv + g\cos\theta\cos\phi + \frac{\rho V_a^2 S}{2m}\left(C_Z(\alpha) + C_{Z_q}\frac{\bar{c}}{2V_a}q + C_{Z_{\delta_e}}\delta_e\right)\,, \tag{2.54}$$

the Euler angles

$$\dot{\phi} = p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \tag{2.55}$$

$$\dot{\theta} = q\cos\phi - r\sin\phi \tag{2.56}$$

$$\dot{\psi} = q\sin\phi\sec\theta + r\cos\phi\sec\theta\,, \tag{2.57}$$

and the angular rates

$$\dot{p} = \Gamma_1 pq - \Gamma_2 qr + \frac{1}{2}\rho V_a^2 Sb\left(C_{p_0} + C_{p_\beta}\beta + C_{p_p}\frac{b}{2V_a}p + C_{p_r}\frac{b}{2V_a}r + C_{p_{\delta_a}}\delta_a + C_{p_{\delta_r}}\delta_r\right) \tag{2.58}$$

$$\dot{q} = \Gamma_5 pr - \Gamma_6(p^2 - r^2) + \frac{\rho V_a^2 S\bar{c}}{2J_y}\left(C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{\bar{c}}{2V_a}q + C_{m_{\delta_e}}\delta_e\right) \tag{2.59}$$

$$\dot{r} = \Gamma_7 pq - \Gamma_1 qr + \frac{1}{2}\rho V_a^2 Sb\left(C_{r_0} + C_{r_\beta}\beta + C_{r_p}\frac{b}{2V_a}p + C_{r_r}\frac{b}{2V_a}r + C_{r_{\delta_a}}\delta_a + C_{r_{\delta_r}}\delta_r\right)\,, \tag{2.60}$$

where the values for C_X, C_Z, and their associated variables can be determined from Equations (2.39) and (2.40).

2.5 Linearized Model

The non-linear model of the aircraft written in the previous section allows us to implement it in different simulation platforms (such as Simulink®, as done in this thesis in Chapter 5) and run tests on it, namely the relay feedback experiment, to analyse its behaviour. However, to perform several different control analyses on the UAV, a simpler linear model is often desired. For this effect, the non-linear model can be linearized around a given trim point and studied around that point, with the restriction that any analyses done in this state will be valid for a given range around it.

A non-linear model of the UAV can be described in compact form as

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) \,. \tag{2.61}$$

If this model is linearized around a trim point, or a point where the system is in equilibrium, where the condition

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) = 0$$

proves true, the linearized version of Equation (2.61) can be written as

$$\dot{\boldsymbol{x}} = \frac{\partial f(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}} \boldsymbol{x} + \frac{\partial f(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}} \boldsymbol{u} \,. \tag{2.62}$$

The linearized model of the UAV is obtained using a Simulink® model based on the equations of motion of the previous section, and the Simulink® function `trim`. This will be further discussed in Chapter 5.

The state of the aircraft is described by the vector $\boldsymbol{x}$, which is based on the 12 equations of motion from Section 2.4, so it can written as

$$\boldsymbol{x} = \begin{bmatrix} p_n & p_e & p_d & u & v & w & \phi & \theta & \psi & p & q & r \end{bmatrix}^\mathsf{T} \tag{2.63}$$

and the input vector $\boldsymbol{u}$

$$\boldsymbol{u} = \begin{bmatrix} \delta_e & \delta_a & \delta_r & \delta_t \end{bmatrix}^\mathsf{T} \,. \tag{2.64}$$

Due to the complexity of a state-space model with 12 entries, the state-space model has been split into a longitudinal and a lateral model. This separation is done according to the frame of reference in Figure 2.2, where the longitudinal model includes movements along the x and z axes and rotations around the y axis, and the lateral model includes movements along the y axis and rotations around the x and z axes.

The obtained longitudinal model of the state-space model of the UAV is summarized as

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} X_u & Y_w & Y_q & -g\cos\theta & 0 \\ Z_u & Z_w & Z_q & -g\sin\theta & 0 \\ M_u & M_w & M_q & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sin\theta & -\cos\theta & 0 & u\cos\theta + w\sin\theta & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \\ h \end{bmatrix} + \begin{bmatrix} X_{\delta_e} & X_{\delta_t} \\ Z_{\delta_e} & 0 \\ M_{\delta_e} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_t \end{bmatrix}, \tag{2.65}$$

and the lateral model as

$$\begin{bmatrix} \dot{v} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_v & Y_p & Y_r & g\cos\theta\cos\phi & 0 \\ L_v & L_p & L_r & 0 & 0 \\ N_v & N_p & N_r & 0 & 0 \\ 0 & 1 & \cos\phi\tan\theta & q\cos\phi\tan\theta - r\sin\phi\tan\theta & 0 \\ 0 & 0 & \cos\phi\sec\theta & p\cos\phi\sec\theta - r\sin\phi\sec\theta & 0 \end{bmatrix} \begin{bmatrix} v \\ p \\ r \\ \phi \\ \psi \end{bmatrix} + \begin{bmatrix} Y_{\delta_a} & Y_{\delta_r} \\ L_{\delta_a} & L_{\delta_r} \\ N_{\delta_a} & N_{\delta_r} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_r \end{bmatrix}. \tag{2.66}$$

Chapter 3

Flight Control

In order to test and compare the effectiveness of the auto-tuner on its platform of implementation, a classical control study of the UAV based on the model from Chapter 2 is performed. This analysis concerns the relation between the modelled dynamics of the UAV and the control inputs, the study and design of feedback loops to close and control, and their stability and performance on a virtual flight. The information gathered here will be used to know where to apply the auto-tuner and analyse its performance by comparing it with the results herein obtained.

3.1 Dynamic Flight Modes

Every aircraft has a set of dynamic flight modes, a set of oscillatory behaviours associated with specific poles of the longitudinal and lateral systems. These oscillatory modes are used to describe how an aircraft reacts to disturbances on its trajectory or controls from steady undisturbed flight. The characteristics of each mode are used as a measure for certificating an aircraft's flying and handling qualities, and their study on the UAV models developed in this thesis will be explained in Section 3.2.

Traditional aircraft control literature describes five modes, two longitudinal and three lateral.

3.1.1 Longitudinal modes

The two longitudinal modes are known as the short-period mode and the phugoid mode, and describe oscillations of movement along the x and z axis and rotation around the y axis of the body frame. Both their oscillations are described using their frequency of oscillation, represented by ω_{sp} and ω_{phu}, and the rate at which their oscillations decay, represented by their damping ζ_{sp} and ζ_{sp}.

The short-period is described by a fast oscillation of the aircraft around its center of gravity that decays very quickly, within the order of a few seconds for large-scale aircraft. The phugoid, on the other hand, has a slower oscillation with far less damping. Its oscillation is often slow enough for the pilot to correct it manually should it interfere with the safe operation of the flight.

In the state-space system from Equation (2.65), the modes are associated with the eigenvalues from the $\boldsymbol{A}$ matrix, with each being associated with a complex pair of poles with equal real components and

symmetrical imaginary parts, with both poles having the same natural frequency and damping. So as an example, if the short-period is associated with the eigenvalue pair

$$\lambda = \sigma \pm j\omega \,,$$

the natural frequency and damping of this pair of poles would be defined as

$$\omega_n = |\lambda| = \sqrt{\sigma^2 + \omega^2} \tag{3.1}$$

$$\zeta = \frac{\sigma}{\omega_n} \,. \tag{3.2}$$

The short-period mode would be associated with the pair with the lowest natural frequency, and phugoid with the highest.

3.1.2 Lateral modes

The three lateral modes are known as the roll, spiral, and dutch-roll. They are described by movement along the y axis and rotation around the x and z axis.

The roll mode is the mode used to describe the behaviour of a change in roll angle. It is characterized by a time constant τ_r instead of a combination of natural frequency and damping. It is associated with the faster of the real poles of the lateral system such as that of Equation (2.66).

The spiral mode is an inherent mode that a UAV will enter if it is left flying with no control, which will cause it to dive and possibly crash. It is, by default, unstable, and is associated with the real pole closest to the center, sometimes even positive.

The dutch-roll, attributed to the complex pole pair, is described as a combination of oscillations of yaw and roll, generally slower than that of roll. Due to being associated as a pair of poles, natural frequency and damping apply again.

3.2 Flying and Handling Qualities

This section will cover the analysis of flying and handling qualities for the UAVs considered in this thesis. As autonomous flight does not have the same concept of flying and handling qualities of a manned aircraft, the analysis performed in this section will be carried out from the perspective of a remotely piloted vehicle (RPV), as the UAV of the project will also be flown in this mode.

The flying and handling qualities for a UAV differ slightly from the standard requirements for a manned aircraft (such as those discussed in *Flying Qualities of Piloted Aircraft* [7]). As a quick reference, a UAV does not have concerns regarding on-board pilot safety, and while there is concern for UAV reliability and recoverability and safety of grounded personnel, the bulk of flying qualities focus is on performance requirements. However, since analysis is done for a RPV, it does mean human factors such as pilot-induced oscillations (PIO) have to be considered.

3.2.1 Classifications

Before the actual rules and restrictions can be presented in Section 3.2.3, the definition of varying degrees of flying and handling qualities is presented. These are organized in a similar way to common literature on the topic [23, 7, 13], and define what it means for an aircraft to have good or bad flying qualities for different flying contexts and scenarios. These are defined by the UAV itself, the planned flight mission, the different flight phases, and the required or desired control characteristics.

UAV classification

There are multiple types of uses for UAV-based missions, which affect how the UAV has to be handled. Prosser and Wiler gathered a list of these uses.

Class I Small, light mini UAVs, such as surveillance, reconnaissance, targets, demonstration models, electronic warfare or harassment models;

Class II Low-maneuverability UAVs, such as those used in surveillance, reconnaissance, or other long endurance missions at high altitudes;

Class III Medium-maneuverability UAVs, similarly used in surveillance, reconnaissance and similar at low altitudes, low-level terrain following and avoidance missions, weapon delivery, and larger target UAVs;

Class IV High-maneuverability UAVs, such as interceptors and air-to-air combat targets.

Considering this description of UAV classes, considering the mission for this project, the models used fall into the Class I category, and this category will be used for any discussions in the remainder of this section.

Flight phase description

The different phases of a flight have been organized into four categories with similar handling quality requirements. These encompass all possibilities for a flight so that there are no gaps between any of them.

Adapted from Prosser and Wiler [23] is a list with these categories and a short description for each. Table 3.1 includes the most common examples for each category.

Category A Flight phases that require precision and rapid movement.

Category B Flight phases with slow, gradual movements and less precision required.

Category C Launch/recovery flight phases with similar requirements to category A.

Category D Launch/recovery flight phases with similar requirements to category B.

Table 3.1: Flight categories and examples of fitting flight scenarios. Adapted from [23].

Category A	Category B	Category C	Category D
Reconnaissance	Climb	Net capture	Approach
Antisubmarine search	Descent	Wave-off/go-around	Catapult take-off
Air-to-air combat	Cruise	Conventional landing	Parachute descent
Surface attack	Loiter	Conventional take-off	Approach to recovery
Weapon delivery	Emergency climb	Arresting gear landings	Midair parachute recovery
In-flight refueling	Emergency descent	Midair probe capture	
Target acquisition	Emergency deceleration		
Formation flying	Aerial delivery		
Terrain following			
Terrain avoidance			

3.2.2 Flying qualities

The quality of a given UAV's flight is separated into three levels with different characteristics for each analysed flying property. These levels share the analysed property's ability to fulfill the goals for which the UAV is designed. Ideally, the UAV should aim for level 1 in all conditions.

In case of autonomous control, the three levels are classified as follows, adapted from [23].

Level 1A UAV flying qualities are adequate to perform their corresponding mission phase.

Level 2A UAV flying qualities are adequate to perform their corresponding mission phase with some degradation of mission effectiveness.

Level 3A UAV flying qualities are adequate to recover the vehicle.

For manual control in a RPV mode, the three levels are described as such.

Level 1M Remote control is adequate to perform the corresponding mission phase.

Level 2M Remote control is adequate to perform the corresponding mission phase, with an increase in operator work load and decrease in mission effectiveness.

Level 3M Remote control is degraded but still adequate to recover vehicle.

For most cases, the separation into automatic and manual flying qualities is not relevant and it can be assumed that the levels on both versions are analogous to each other.

3.2.3 Flying and handling quality requirements

The core of the flying and handling modes are based on the characteristics of the dynamic flight modes discussed in Section 3.1. Their requirements will be presented in a mode-to-mode basis.

Short-period

The requirements for the short-period damping ζ_{sp} are given in Figure 3.1, and for the natural frequency $\omega_{n_{sp}}$ in Figure 3.2. Important to note is that the quality of $\omega_{n_{sp}}$ depends on the load-sensitivity factor $\frac{n}{\alpha}$

given in g per radians. This factor is simplified for trim conditions using the expression [9, 11]

$$\frac{n}{\alpha} = \frac{\rho V_0^2 S}{2mg} C_{L_\alpha} . \tag{3.3}$$

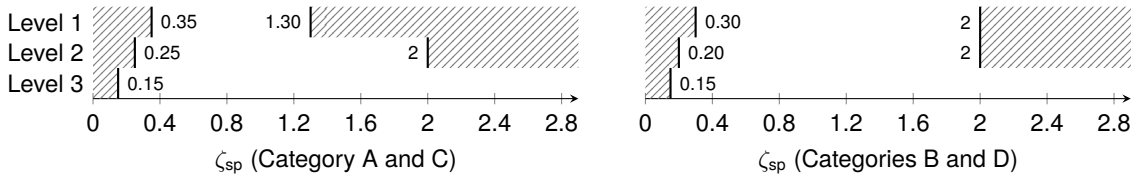


Figure 3.1: Requirements for the UAV short-period damping. The value of ζ_{sp} should remain within the indicated boundaries.

Phugoid

The phugoid requirements are presented in Figure 3.3. It should be noted that the damping condition for level 3 does not actually depend on the value of the phugoid damping ζ_{phu}, but on the value of T_2, which can be determined by

$$T_2 = \frac{\ln 2}{-\zeta_{phu}\omega_{n_{phu}}} , \tag{3.4}$$

where the damping ζ_{phu} is known to be negative due to the condition for level 2.

No specifications are given for the phugoid natural frequency $\omega_{n_{phu}}$.

Roll

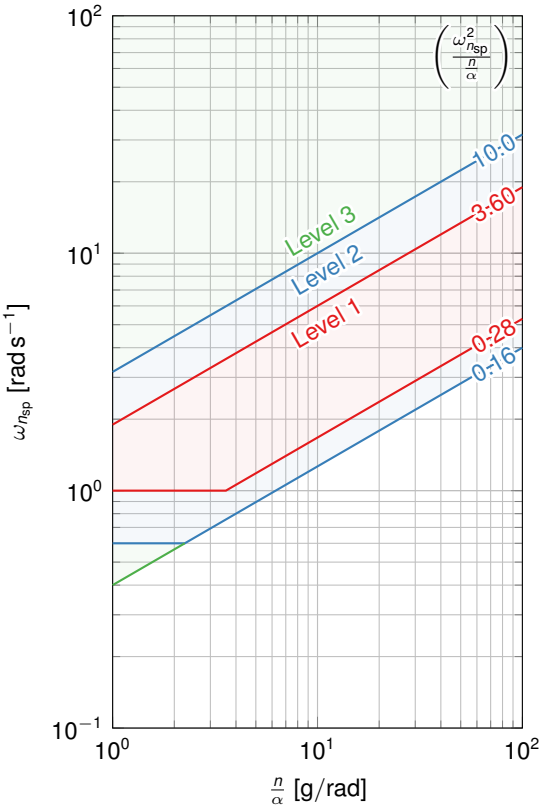
The roll requirements are listed in Figure 3.4. These are characterized by maximum values for the time constant τ_r of the roll mode.

Spiral

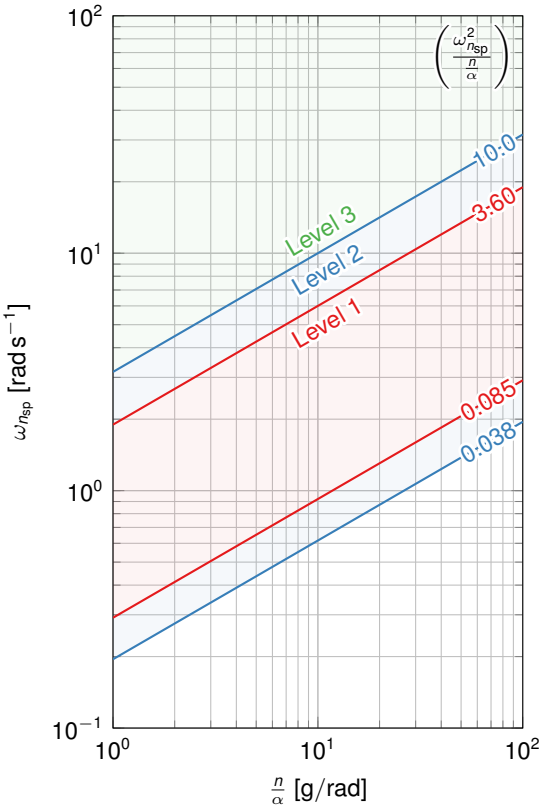
The spiral requirements are listed in Table 3.2. Due to the spiral mode usually being unstable by default, the restrictions for this mode lie in the UAV's response to a roll angle disturbance of up to 20 degrees taking more time than specified to double its value.

Table 3.2: Requirements for the UAV spiral mode for class I UAVs. The variable T measured is the time taken by the UAV to double its roll angle value following a disturbance of up to 20 degrees.

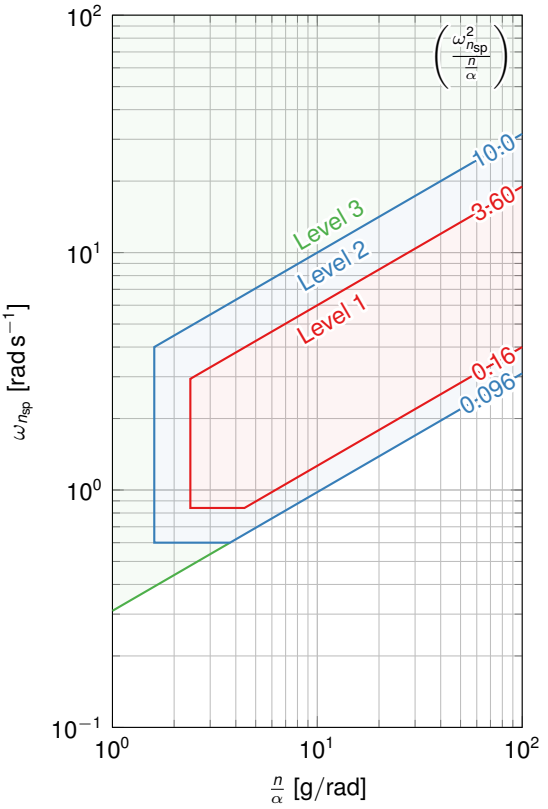
Level	Category A	Category B and C
1	$12\,\text{s} < T$	$20\,\text{s} < T$
2	$12\,\text{s} < T$	$12\,\text{s} < T$
3	$4\,\text{s} < T$	$4\,\text{s} < T$



(a) Category A.



(b) Category B.



(c) Category C.

Figure 3.2: Requirements for the short-period natural frequency requirements. The diagonal lines on the logarithmic scale are defined by the ratio in the upper-right corner. The areas outside of the graph can be obtained by extending the defined lines. Adapted from [23, 9].

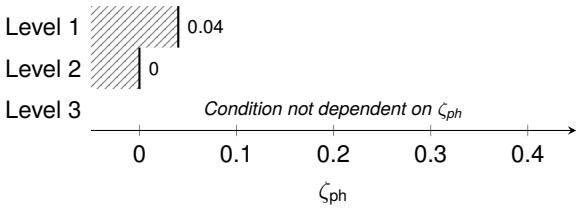


Figure 3.3: Requirements for the UAV phugoid damping. The condition for level 3 depends on the property T_2 defined in Equation (3.4), which is to be bigger than 55 seconds.

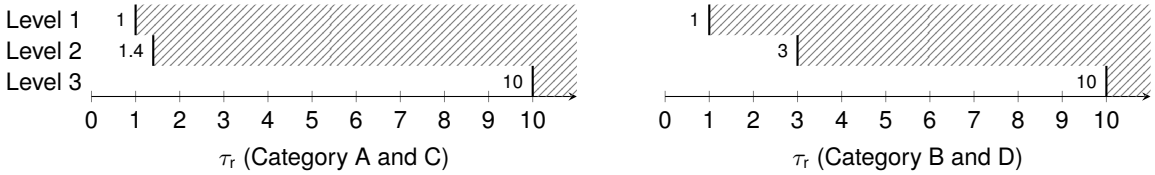


Figure 3.4: Requirements for the UAV roll mode time constant for class I UAVs.

Dutch-roll

The requirements for the properties of the dutch-roll mode can be seen in Table 3.3. These include restrictions for the natural frequency $\omega_{n_{dr}}$, damping ζ_{dr} and the product of both.

Table 3.3: Requirements for the UAV dutch-roll properties for class I UAVs.

Level	Category A	Category B	Category C
1	$0.19 < \zeta_{dr}$ $1 < \omega_{n_{dr}}$ $0.35 < \zeta_{dr}\omega_{n_{dr}}$	$0.08 < \zeta_{dr}$ $0.4 < \omega_{n_{dr}}$ $0.15 < \zeta_{dr}\omega_{n_{dr}}$	$0.08 < \zeta_{dr}$ $1 < \omega_{n_{dr}}$ $0.15 < \zeta_{dr}\omega_{n_{dr}}$
2	$0.02 < \zeta_{dr}$ $0.4 < \omega_{n_{dr}}$ $0.05 < \zeta_{dr}\omega_{n_{dr}}$	$0.02 < \zeta_{dr}$ $0.4 < \omega_{n_{dr}}$ $0.05 < \zeta_{dr}\omega_{n_{dr}}$	$0.02 < \zeta_{dr}$ $0.4 < \omega_{n_{dr}}$ $0.05 < \zeta_{dr}\omega_{n_{dr}}$
3	$0.02 < \zeta_{dr}$ $0.4 < \omega_{n_{dr}}$	$0.02 < \zeta_{dr}$ $0.4 < \omega_{n_{dr}}$	$0.02 < \zeta_{dr}$ $0.4 < \omega_{n_{dr}}$

3.3 Feedback Loops

To improve the stability and control performance of the UAV, a look at its control loops will be of focus in this section. These loops describe the relation between an input and an output variable in the aircraft, and the behaviour of the aircraft when the output variable is scaled and fed back to the input.

The PID loops are separated into levels of abstraction, with different sets of loops superposing others. This approach to controller design is called *loop closure*, referring to how some loops function with other loops inside to improve control. This is made possible because some loops are faster than others by a difference big enough that their behaviour can be, if well tuned, neglected when included inside the slower loops. The faster loops that do not include any other inside their structure are called the *inner loops*, and

are the most crucial to aircraft stability and flying and handling qualities. Only these inner loops will be analysed in this thesis.

The project described in this thesis includes its later implementation in the MicroPilot MP2128^{LRC2} autopilot, which restricts the adjustable feedback loops to the ones included in its hardware. The MicroPilot MP2128^{LRC2} has three inner control loops that will be the focus of this thesis project. Each of these loops represent a reference input value and the control variable that will be fed to the aircraft. The loops are, in no particular order, *Elevator from Desired Pitch*, *Aileron from Desired Roll*, and *Rudder from Y Accelerometer*.

3.3.1 *Elevator from Desired Pitch*

The *Elevator from Desired Pitch* loop regulates the behaviour of the elevator of the UAV based on the current desired pitch angle. The desired pitch angle is given either by the pilot or the outer loop around this loop, which is dependent on the active gain schedule in the case of the MP2128^{LRC2}.

Beard and McLain describe the relation between elevator and aircraft pitch on the body frame as a transfer function [5, p. 73] dependent on various flight conditions and aerodynamic coefficients in a much simpler fashion, represented as

$$\frac{\theta(s)}{\delta_e(s)} = \frac{a_{\theta_3}}{s^2 + a_{\theta_1}s + a_{\theta_2}}, \tag{3.5}$$

whose coefficients a_{θ_1}, a_{θ_2}, and a_{θ_3} are defined as

$$a_{\theta_1} = -\frac{\rho V_a^2 cS}{2J_y} C_{m_q} \frac{c}{2V_a} \tag{3.6}$$

$$a_{\theta_2} = -\frac{\rho V_a^2 cS}{2J_y} C_{m_\alpha} \tag{3.7}$$

$$a_{\theta_3} = \frac{\rho V_a^2 cS}{2J_y} C_{m_{\delta_e}}, \tag{3.8}$$

The transfer function for this loop can also be taken directly from the longitudinal state-space model from Equation (2.65).

3.3.2 *Aileron from Desired Roll*

This feedback loop regulates the aileron deflection based on the desired roll angle. Similarly to the previous loop, the roll angle is controlled by the pilot or an outer loop dependent on the gain schedule.

Beard and McLain describe the relation between aileron deflection and aircraft roll in the body frame as a transfer function [5, p. 69] dependent on various flight conditions and aerodynamic coefficients, represented as

$$\frac{\phi(s)}{\delta_a(s)} = \frac{a_{\phi_2}}{s(s + a_{\phi_1})}, \tag{3.9}$$

whose coefficients a_{ϕ_1} and a_{ϕ_2} are defined as

$$a_{\phi_1} = -\frac{1}{2}\rho V_a^2 SbC_{p_p}\frac{b}{2V_a} \tag{3.10}$$

$$a_{\phi_2} = \frac{1}{2}\rho V_a^2 SbC_{p_{\delta_a}}, \tag{3.11}$$

wherein the coefficients C_{p_p} and $C_{p_{\delta_a}}$, not usually described in literature, can be given as

$$C_{p_p} = \Gamma_3 C_{l_p} + \Gamma_4 C_{n_p} \tag{3.12}$$

$$C_{p_{\delta_a}} = \Gamma_3 C_{l_{\delta_a}} + \Gamma_4 C_{n_{\delta_a}}, \tag{3.13}$$

A more accurate but higher-order transfer function can be obtained from the lateral state-space model from Equation (2.66).

3.3.3 *Rudder from Y Acceleration*

This loop controls the rudder deflection based on the current value of the side acceleration, or the value of the derivative of the side speed $\dot{v}$. In the case of the MP2128$^{\text{LRC2}}$ the purpose of this loop is to regulate coordinated turns, and the desired side force is always zero, independent of gain schedule.

Although the previously cited source [5] does not describe the relation between the rudder deflection δ_r and the side acceleration $\dot{v}$ directly (described in the *MicroPilot Autopilot Instalation & Operation Manual* as *Rudder from Y Accelerometer*), it does describe the relation between the rudder deflection and the sideslip angle β as

$$\frac{\beta(s)}{\delta_r(s)} = \frac{a_{\beta_2}}{s + a_{\beta_1}}, \tag{3.14}$$

whose coefficients a_{β_1} and a_{β_2} are defined as

$$a_{\beta_1} = -\frac{\rho V_a S}{2m} C_{Y_\beta} \tag{3.15}$$

$$a_{\beta_2} = \frac{\rho V_a S}{2m} C_{Y_{\delta_r}}. \tag{3.16}$$

Given that the relation between side speed v and sideslip angle β can be given as [5, p. 20]

$$v = V_a \sin\beta,$$

and assuming β is small enough to approximate $\cos\beta \simeq 1$ then the side acceleration $\dot{v}$ is given by

$$\dot{v} = \dot{V}_a \sin\beta,$$

and the relation between both variables can be defined in the Laplace domain as

$$\beta(s) = \frac{\dot{v}(s)}{V_a s} .$$

This in turn gives us the transfer function for the rudder deflection from the measured side acceleration

$$\frac{\dot{v}(s)}{\delta_r(s)} = \frac{(a_{\beta_2} V_a)s}{s + a_{\beta_1}} . \tag{3.17}$$

Much like the previous loop, a transfer function can also be obtained from the lateral state-space model from Equation (2.66).

Chapter 4

Auto-tuning via Relay Feedback

Modern developments in PID study have brought a great number of tuning methods to the surface [20]. With the widespread implementation of the PID controller and the manual labour necessary to tune a PID controller, there is a great incentive to automate its tuning, and there have been many studies on the plausibility of a fully automated tuner, which would set the gains of PID controllers and its variations automatically based on the system it is implemented in. One very notable example is the work of Åström and Hägglund [1, 2] in which the authors describe the use of a relay in a feedback loop with a PID controller implemented to automatically determine certain properties of the process on which some tuning rules are based.

First, the PID controller and its functionality is presented. Then, the method described by Åström and Hägglund becomes the focus of this chapter, in which the tuning rules on which it is based are primarily presented.

4.1 The PID Controller

The PID controller is one of the most widely used controllers in several industries, in most part due to its simplicity and low cost, but also by its surprising capability and competence in different types of implementation. It is estimated that more than 95% of controllers in industrial process control are either a PID controller or a variation of it [3], and while numbers in the transport industry tend to be lower, it still shows how widely implemented and studied this form of control really is, and how it remains one of the most impactful forms of control industry-wide [25].

In the aerospace industry, the role of the PID is slowly having less prominence, as often the control process in modern commercial aircraft are far more complex to handle the requirements of such a delicately balanced system. Nonetheless, PID controllers can still be found in one form or another as parts of these larger control schemes, or as in the case of this thesis project, in the control loops of simpler, lower risk implementations, such as UAVs and other lower complexity aircraft.

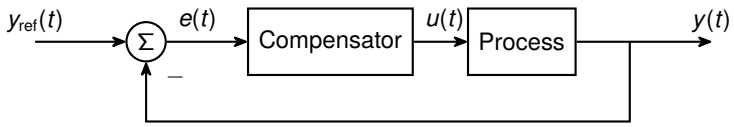


Figure 4.1: Basic feedback loop.

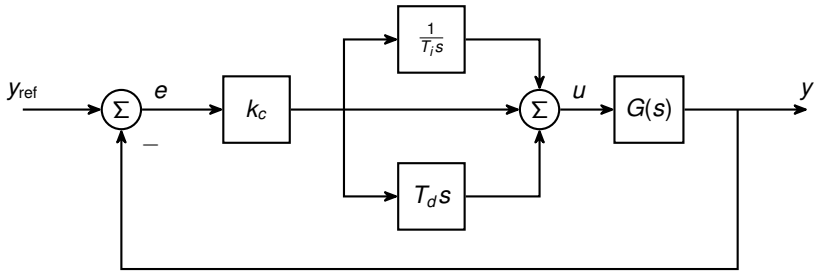


Figure 4.2: Basic feedback loop using PID control in standard form.

4.1.1 Basic Functionality

Take the feedback loop represented in Figure 4.1. In it a process and a compensator are represented, as well as the process variable $y(t)$, the set point $y_{\text{ref}}(t)$, the error $e(t)$ ($e(t) = y_{ref} - y(t)$), and the process input $u(t)$, all defined in the time domain. The core concept of feedback control is based on creating a relation between the error $e(t)$ and the process input $u(t)$ that performs according to pre-specified objectives (defined for our case, indirectly, in Section 3.2), a relation referred to as compensation. The most basic example of compensation is a gain k, which means that $u(t)$ takes the value of

$$u(t) = ke(t)\,,$$

defined at any point in time.

A PID controller is a slightly more complex type of compensator. It relates the error $e(t)$ to the process input $u(t)$ using the equation Åström and Hägglund [3, p. 64]

$$u(t) = k_c \left(e(t) + \frac{1}{T_i} \int_0^t e(t)\,\mathrm{d}t + T_d \frac{\mathrm{d}e(t)}{\mathrm{d}t} \right)\,, \tag{4.1}$$

a relation illustrated by Figure 4.2.

The PID controller has three main effects on the value of $u(t)$, proportional, integrative, and derivative, represented in Equation (4.1) by the three parameters k_c, T_i, and T_d. The first, called proportional action, scales $u(t)$ based on the value of the error at any given time, and follows much the same logic as a normal proportional compensator. It provides a steady correction of the error in the loop, with a faster response for higher values of the gain associated with it, but can cause instability should its value be too high. Thus, a well tuned controller should find the balance between speed and stability in the response.

The integral action, controlled by the variable T_i, scales the value of $u(t)$ based on an integration of the value of the error in a given period of time prior to the calculation. Its main objective is to correct the *steady state error* of the process, an error between its set point and output variables that is apparent when the process with feedback has reached equilibrium. This is achieved by scaling the error with time, an

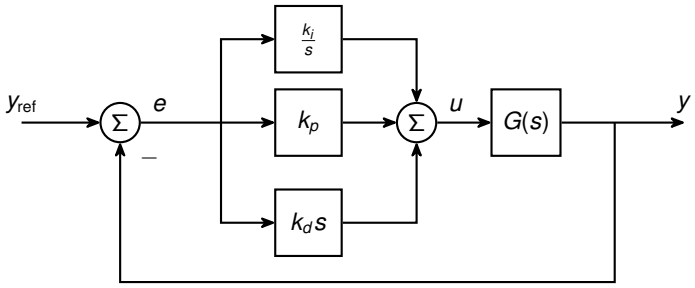


Figure 4.3: Basic feedback loop using PID control in parallel form.

effect caused by the integration of the error. A PID controller with no integrative action is the same as one for which $T_i = \infty$, as shown by Equation (4.1), and the steady state error for the controlled process is eliminated by any finite value of T_i. Lowering the value of the integral time T_i increases the speed of approach of the controlled variable, but also causes more oscillation.

The last effect, the derivative action, determined by the variable T_d, scales $u(t)$ based on the derivative of the error on any given time. This action increases the stability and speed of the loop by applying corrective action proportional to the *predicted* error of the process, which it obtains via the derivative of the current state of the process. While this increases the speed of convergence, it is particularly susceptible to noise in the process, which causes havoc to the prediction mechanism. Therefore, this action should be used with caution. A PID controller with no derivative action is the same as having a null derivative time $T_d = 0$, and increasing the value of the derivative time T_d is often associated with an increase in damping, but causes the damping to decrease significantly if T_d becomes too large.

Adapting a PID controller to any given feedback loop requires balancing all three of these effects in relation to each other and to the loop in which it is being implemented. The overall impact of each effect can be adjusted by their respective variables, k_c for proportional gain, T_i for the integrative effect, and T_d for the derivative effect. Determining the best values for all three parameters for a given scenario and objective is called tuning.

A variation on the PID mathematical relation described in Equation (4.1) is the one described as

$$u(t) = k_p e(t) + k_i \int_0^t e(t)\,\mathrm{d}t + k_d \frac{\mathrm{d}e(t)}{\mathrm{d}t}\,, \tag{4.2}$$

also well illustrated in Figure 4.3. Where as the previous form (described in Equation (4.1)) is called the *standard* form, this second form is called *parallel* form. In this case, each effect can be adjusted more independently, but the controller is essentially the same. The parameters are now k_p, k_i, and k_d, controlling proportional, integrative, and proportional action respectively, and relate to the parameters of the standard form via

$$\begin{aligned} k_p &= k_c \\ k_i &= \frac{1}{T_i} k_c \\ k_d &= T_d k_c\,. \end{aligned} \tag{4.3}$$

Laplace domain

Using the Laplace Transform, the PID controller can be described by the transfer function

$$G(s) = k_c \left(1 + \frac{1}{T_i s} + T_d s\right) , \tag{4.4}$$

or, in the parallel form,

$$G(s) = k_p + k_i \frac{1}{s} + k_d s , \tag{4.5}$$

This representation, seen in Figures 4.2 and 4.3, allows us to describe the behaviour of the PID controller in the frequency domain and apply it to the necessary process transfer functions to analyse and understand their behaviour in any implemented feedback loops, the focus of the following section.

4.2 Ziegler-Nichols Tuning Rules

Before introducing the relay-based auto-tuner and its associated tuning rules, an overview is done on the Ziegler-Nichols tuning rules on which they (and many others) are based, the system characteristics necessary to apply them, and how these are experimentally obtained.

4.2.1 Frequency Analysis

When a sinusoid input is fed to a single-input, single-output (SISO) closed loop system, the output will also be a sinusoid of equal frequency after a transient period, with a different phase and amplitude of oscillations [3]. This phase and amplitude is dependent on the system and the frequency of the input, so for any given system, the output of a sinusoidal input of frequency ω can be described by the difference in phase $\varphi(\omega)$ and the ratio of the amplitude of the input and output $a(\omega)$. In another way, a complex number can be defined

$$G(i\omega) = a(\omega) e^{i\varphi(\omega)} ,$$

such that its magnitude is the amplitude gain and its argument the phase shift, or

$$a(\omega) = |G(i\omega)|$$
$$\varphi(\omega) = \arg G(i\omega) .$$

Plotting this function in the complex plane for every value of ω yields what is called a *Nyquist plot*, similar to the one represented in Figure 4.4. This plot allows for the analysis of a determined system's stability by understanding the plot's behaviour around the *critical point*, the point $s = -1$. However, that analysis will not be performed in this thesis.

The lowest frequency at which the output lags the input by π radians is called the *ultimate frequency*

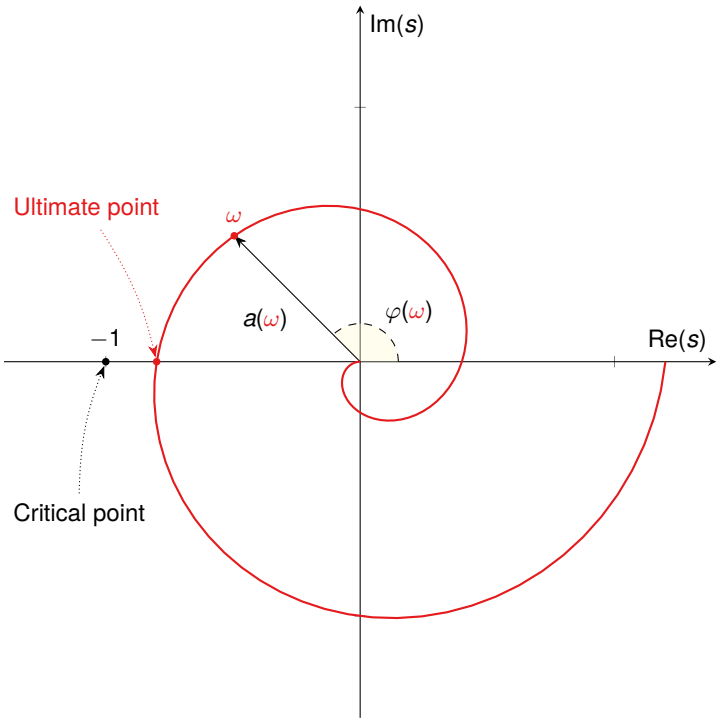


Figure 4.4: Nyquist plot[a] for an arbitrary feedback loop. Each point represents the amplitude gain a and phase lag φ for a specific system at a frequency ω. The ultimate point represents the frequency at which the system lags by $-\pi$ radians.

[a] The Nyquist plot is more commonly represented as a closed plot, symmetric around the real axis. However, due to not providing any extra information in the context of this thesis, half of the plot will be omitted throughout.

and is visible in Figure 4.4 as the point where the plot crosses the negative real axis, called the *ultimate point*. Should the gain at this point surpass unity (thus meaning the ultimate point is at the left of the critical point), the amplitude of the output will increase exponentially and the system will become unstable.

The Ziegler-Nichols PID tuning rules, introduced in 1942 [33], are based on this condition of the system. By introducing a proportional gain to the system, the gain at all points of the Nyquist plot can be adjusted. Setting its value at zero and steadily increasing it will eventually cause the system to oscillate at the ultimate frequency ω_u. The period of these oscillations is called the *ultimate period* T_u and is obtained through

$$T_u = \frac{2\pi}{\omega_u}. \tag{4.6}$$

The value of the proportional gain used to bring a system to its ultimate point is retrieved as the ultimate gain k_u.

4.2.2 Tuning rules

According to Ziegler and Nichols, the values of T_u and k_u can be used to determine the parameters of a PID controller to be put in place of the proportional controller introduced. As an example, in the case of a standard-type PID controller, the proportional gain k_p is given by

$$k_p = 0.6k_u\,,$$

the integrative parameter by

$$T_i = 0.5T_u\,,$$

and the derivative parameter by

$$T_d = 0.125T_u\,.$$

The values for a standard-type PID controller and variations can be found in Table 4.1.

Table 4.1: Ziegler-Nichols frequency-based tuning rules. Adapted from [33, 4].

Type	k_p	T_i	T_d
P	$0.5k_u$	—	—
PI	$0.4k_u$	$0.8T_u$	—
PID	$0.6k_u$	$0.5T_u$	$0.125T_u$

Tuning a PID controller was, for a long time, a manual effort, mostly done by trial and error. Since their introduction in 1942, the Ziegler-Nichols tuning rules are still in use today and form the base on which multiple other rules are built, including the relay auto-tuning method which will be analysed during this chapter. Their introduction also started an ongoing research focus on the PID controller, increasing the understanding of their behaviour exponentially since that time. In 2009, O'Dwyer published a list of 245 PID tuning rules developed in this time for many different types of processes and scenarios [20], a valiant gathering of information demonstrating the progress of the study of PID controllers since their appearance in the field. Some of these tuning rules will be analysed in Section 4.6 for their application in the current case.

4.3 Relay Feedback

A simple setup for a process and a PID controller, similar to the one found in Figure 4.3, can be used to manually determine the ultimate gain and frequency with which to tune the PID via Ziegler-Nichols. However, operating a process in oscillation near instability can be dangerous and introduces risks that in some scenarios should be avoided altogether.

Åström and Hägglund suggested [1] that by introducing a relay component to the loop in a similar way to Figure 4.5, both the ultimate gain and frequency can be obtained without putting the process in a near unstable state. Introducing a relay to the process causes the output signal to oscillate at the ultimate frequency, with the value of output being controlled by the relay output d, an adjustable parameter. These parameters can be used to model the system based on its frequency response, and most importantly, as will be explained next, can be used to determine the values of the ultimate gain k_u and frequency ω_u on which the tuning rules are based.

Special care must be taken not to select high relay amplitudes that may excite high frequency modes.

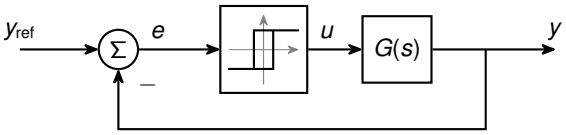


Figure 4.5: Feedback loop used for relay auto-tuning.

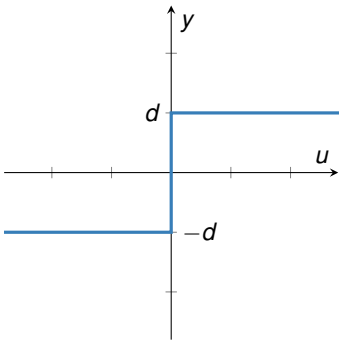


Figure 4.6: Relation between input and output for an ideal or *on-off* relay with output value d, without hysteresis applied.

4.3.1 The relay component

The basic principle of the functioning of a relay is described using the ideal or *on-off* relay, exemplified in Figure 4.6. An ideal relay outputs a value d when its input is higher than a specific set point (typically zero), and its opposite $-d$ if said input is lower than the same set point.

When receiving a symmetrical sinusoidal input $e(t)$, an ideal relay will output a square wave of the same frequency ω but different amplitude a, as exemplified by Figure 4.7. Decomposing the ideal relay using a Fourier series for a square wave, the relay output is seen to be symmetrical to the origin, nullifying the terms a_0 and a_n and returning

$$b_n = \frac{2}{\pi}\int_0^{\pi} d\sin(n\omega t)\,\mathrm{d}(\omega t) \tag{4.7}$$

$$= \frac{2}{\pi}\left[\frac{-d\cos(n\omega t)}{n}\right]_0^{\pi}. \tag{4.8}$$

For odd values of n,

$$b_n = \frac{2}{\pi}\left(\frac{d}{n} - \left(-\frac{d}{n}\right)\right) = \frac{4d}{n\pi}, \tag{4.9}$$

and for even values,

$$b_n = \frac{2}{\pi}\left(\left(-\frac{d}{n}\right) - \left(-\frac{d}{n}\right)\right) = 0. \tag{4.10}$$

Hence, the relay output $u(t)$ is, for odd values of n,

$$u(t) = \sum_{n=1}^{+\infty} \frac{4d}{n\pi}\sin(n\omega t). \tag{4.11}$$

Assuming that the system $G(s)$ attenuates the higher order harmonics of the square wave, it is

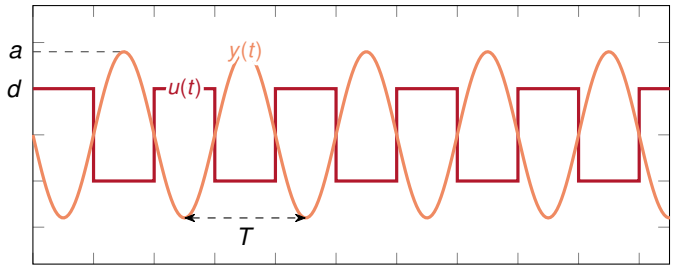


Figure 4.7: Relay experiment example output over time for the system represented in Figure 4.5. $u(t)$ represents the relay output and $y(t)$ the process output.

reasonable to approximate $u(t)$ by its first harmonic component, *i.e.*

$$u(t) \simeq \frac{4d}{\pi} \sin(\omega_u t) \,. \tag{4.12}$$

The describing function method can then be used to analyse the behaviour of the system, by representing the relay as a non-linear component $N(a)$ (see Figure 4.6). The non-linear component $N(a)$ describes the propagation of a sinusoid of amplitude a through the system and for that reason it is called the describing function. The definition of the describing function used is the ratio of the fundamental component of the output (the first element of the Fourier series, or $n = 1$) of the nonlinear element to the input. In this case,

$$\begin{aligned} N(a) &= \frac{\frac{4d}{\pi} \sin(\omega t)}{a \sin(\omega t)} \\ &= \frac{4d}{a\pi} \,. \end{aligned} \tag{4.13}$$

The system can only reach an oscillatory state if

$$G(j\omega_u) N(a) = -1 \,. \tag{4.14}$$

Since the describing function for a relay is positive real gain (see Equation (4.13)), oscillation only occurs at the ultimate frequency of $G(s)$, $\omega = \omega_u$.

This describing function can also be used to determine the ultimate gain of the process with which to tune a PID controller. As mentioned previously, the ultimate gain k_u stands for the proportional gain used to bring the gain of the loop to unity at the ultimate frequency ω_u, or to bring the Nyquist curve to an intersection with the critical point $s = -1$. In polar form,

$$\begin{aligned} G(j\omega_u) k_u &= -1 \\ k_u &= -\frac{1}{G(j\omega_u)} \,. \end{aligned} \tag{4.15}$$

For this system, the describing function can be used in place of the proportional gain as

$$G(j\omega_u) N(a) = -1 \,. \tag{4.16}$$

Reusing Equation (4.15), the value of the ultimate gain as a function of the amplitude of the oscillations of

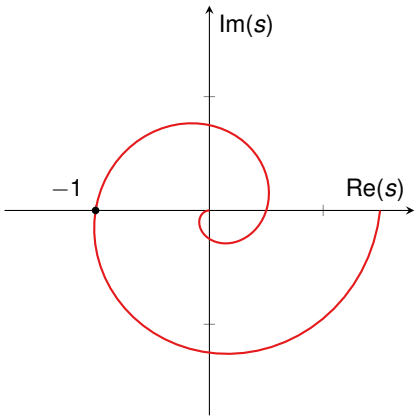


Figure 4.8: Nyquist curve for the system of Figure 4.4 multiplied by its critical gain k_u. Notice the intersection with the critical point at $s = -1$.

the system can be finally obtained via

$$k_u = \frac{4d}{a\pi}. \tag{4.17}$$

This result is valid for obtaining the gain at a phase delay of $-\pi$ radians, but can be adjusted for determining other points along the Nyquist curve. This will be looked at in Sections 4.4 and 4.6.

Part of the appeal of the relay auto-tuning method comes from the fact that in a tuning experiment using an ideal relay, the values of ω_u and a can easily be extracted from the output, with the value of k_u being easily calculated short thereafter. These values can then be used with Ziegler-Nichols tuning rules and other subsequent variants without inducing the system into near-unstable oscillations.

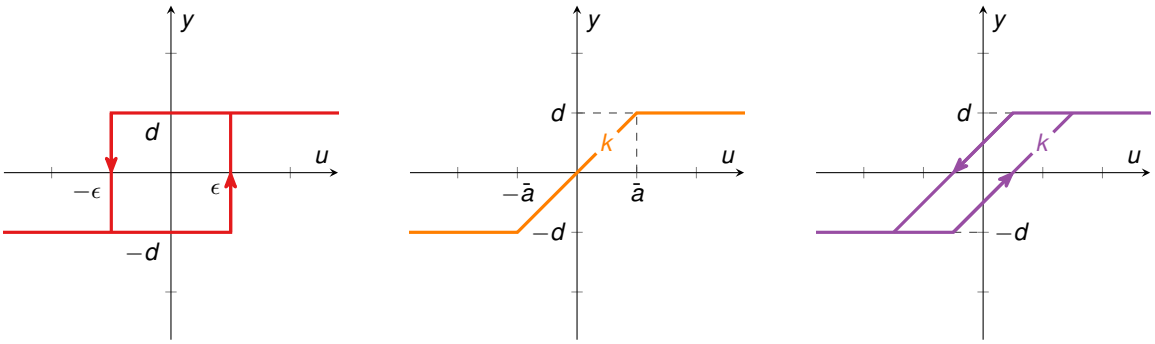
4.4 Improvements to the Relay Experiment

Although experiments with ideal relay can be successful in the determination of suitable values for k_u and ω_u, these values are often far from the real values for each system [32], errors stemming from linear approximations made from nonlinear components such as the relay. However, the accuracy of these values can be improved with the use of relays different from ideal. Not only that, but the location of Nyquist curve points can be desired, namely for the different tuning rules that will be discussed in Section 4.6, which cannot be obtained with the ideal relay.

This section will present some setup variants to the one discussed in Section 4.3, their effect on the results of the relay experiment and how these results can be interpreted and used for improving the quality of the obtained tuning parameters.

4.4.1 Relay with hysteresis

The ideal behaviour of a relay can often be undesired, most notably in the presence of noise, which will cause sudden switches between outputs if this noise causes the input to transition between the reference point. To avoid this problem, a relay with hysteresis can be introduced. A relay with hysteresis outputs d when the input surpasses a value ϵ and $-d$ if the input drops below $-\epsilon$. This behaviour is best understood



(a) Relay with hysteresis ϵ.

(b) Saturation relay with slope k and input limit $\bar{a}$.

(c) Saturation relay with hysteresis.

Figure 4.9: Different types of relay and their input-output relation.

in Figure 4.9a.

In the presence of a relay with hysteresis, the describing function $N(a)$ is determined to be [4]

$$N(a) = \frac{4d}{a\pi}\left(\sqrt{1-\frac{\epsilon^2}{a^2}} - j\left(\frac{\epsilon}{a}\right)\right). \tag{4.18}$$

By adjusting the value of ϵ, different points of the Nyquist curve between phase lags of $-\pi$ and $-\frac{\pi}{2}$ are determined [26], based on the describing function obtained. However, for small values of ϵ, Equation (4.17) can be used to determine the value of k_u at a slight cost of accuracy.

4.4.2 Saturation relay

As seen in Section 4.3, the value for the ultimate gain is obtained from an approximation to the first harmonic of the ideal relay output. This leads to a loss in accuracy as the output is approximated from a square wave to a sinusoidal.

A variation of the ideal relay used in auto-tuning is the saturation relay, which eases the transition between both states of the relay and improves accuracy in the identification of the ultimate gain and frequency. Between two outputs, $-\bar{a}$ and $\bar{a}$, around the reference point, the relay outputs a value between $-d$ and d linearly dependent on the slope k. This allows the output to equal its input within the range of $-\bar{a}$ and $\bar{a}$. A graphical representation of this behaviour is found in Figure 4.9b.

4.4.3 Relay with preload

A simple addition to the standard relay process is the consideration of a direct *feed-forward* signal connecting the relay input and output (alongside the relay) with a small gain, similar to Figure 4.10. This addition amplifies the fundamental frequency discussed in **??**. Considering the same sample input signal

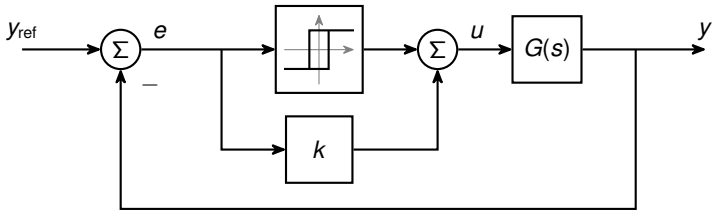


Figure 4.10: Feedback loop with a relay and a preload gain.

$e(t)$,

$$e(t) = a \sin \omega t \,, \tag{4.19}$$

the first harmonic of the relay output is described as

$$u(t) = \frac{4d}{\pi} + ka \,, \tag{4.20}$$

where k represents the value of the preload gain. Using the describing function from Equation (4.13), the ultimate gain is determined by

$$K_u = N(a) = \frac{4d}{a\pi} + k \,. \tag{4.21}$$

The usual values for the preload gain depend on the amplitude of the relay oscillations d, and take a value of around 0.2 to 0.3 times its value, determined by experimental studies [14].

4.5 Locating Different Points

Section 4.3 discussed the usage of an ideal relay in the identification of the frequency where the phase lag φ of a system is $-\pi$ radians, the aforementioned critical frequency ω_u, and the gain k_u required to bring the system to an oscillatory condition at this frequency. However, it is also relevant to be able to locate the frequency for different values of φ, namely for some of the tuning rules discussed in Section 4.6, which make use of this extra information to provide more robust PID parameters. These calculated frequencies for specific phase lag values are analogous to different points on the Nyquist curve (see Figure 4.4). The relay experiment can be adjusted to find these different points with the same overall procedure.

4.5.1 Time-delayed relay response

Using a time delay of T seconds in the relay path of the system, the frequency of oscillation ω is obtained for a specific phase shift φ [6, 14]. These three variables are related by

$$T = -\frac{\varphi}{\omega_u} \,. \tag{4.22}$$

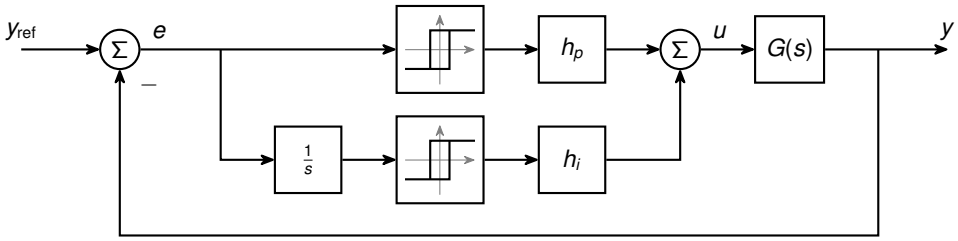


Figure 4.11: Feedback loop with a relay and an integrator used to determine a different point on the Nyquist curve.

So as an example, if the value of the frequency at a lag of $-\frac{\pi}{2}$ is necessary, the value of the time delay is required to be

$$T = \frac{\pi}{2\omega_c}.$$

Important to note for this method is that the value of ω_c needs to be determined beforehand in order to determine the necessary time delay.

4.5.2 Relay with integrator

A different approach to determining different points on the Nyquist curve is with the use of an integrator alongside the relay in the auto-tuning loop [27], as seen in Figure 4.11. The two parameters h_p and h_i control the mixing of the outputs of both the standard relay and the integrated relay, and determine which phase lag φ is used to measure the frequency in the experiment with the relation

$$\frac{h_i}{h_p} = \tan\varphi\,. \tag{4.23}$$

Their values can be adjusted freely to achieve the desired ratio from a specific phase lag. Considering

$$\tan\varphi = \frac{\sin\varphi}{\cos\varphi},$$

the values of h_i and h_p can be set to

$$\begin{aligned} h_i &= \sin\varphi \\ h_p &= \cos\varphi\,. \end{aligned} \tag{4.24}$$

This choice is arbitrary but keeps the values of both the parameters in a known and simple to implement mathematical relation to the value of φ.

If the case $h_p = 0$ and $h_i = 1$ is considered,

$$\varphi = \arctan\left(\frac{h_i}{h_p}\right) = \arctan\left(\frac{1}{0}\right) = \arctan\left(+\infty\right) = -\frac{\pi}{2}, \tag{4.25}$$

only the integrated relay is considered. In this case, the output of the loop will lag the input by an additional

$-\frac{\pi}{2}$ radians, dragging the oscillation point on the Nyquist curve to the intersection with the negative imaginary axis. The feedback loop will oscillate at the frequency $\omega_{-\frac{\pi}{2}}$, which per definition is smaller than the ultimate frequency ω_u, meaning it will take longer to reach the limit cycle and determine the correct value than the original unintegrated relay. The gain $k_{-\frac{\pi}{2}}$ can be determined in the same way as the original process, so Equation (4.17) can be used, replacing k_u with the gain for the current situation.

The balance between the value of the integrated relay output and the ideal relay output through the values of h_i and h_p allows the loop in Figure 4.11 to determine the frequency for phase lags φ different than $-\pi$ or $-\frac{\pi}{2}$. Hence, the gain for any arbitrary phase lag value can be equally determined through Equation (4.17).

Due to its flexibility, this method was opted to be implemented into the thesis project to auxiliate with the location of different points on the Nyquist curve.

4.5.3 Relay with hysteresis

The use of a relay with hysteresis, presented and discussed in Section 4.4.1, allows for the determination of points of phase lag φ between the $-\pi$ radians of the ideal relay and the $-\frac{\pi}{2}$ of the relay with integrator iteratively by adjusting the value of the relay hysteresis ϵ until the desired frequency of oscillation is obtained, as discussed by Scali and Costaggiu [26]. The point for a given hysteresis value ϵ is given by

$$-\frac{\pi}{4d}\sqrt{a^2-\epsilon^2}-j\frac{\pi}{4d}\epsilon\,, \tag{4.26}$$

which is derived from the describing function in Equation (4.18).

The authors also propose the use of this property to tune a proportional-integral (PI) controller based on predefined performance specifications, however such an approach lies beyond the scope of this thesis.

4.6 Tuning Methods

The final step in the relay auto-tuning procedure is the calculation of the PID controller parameters. This is done via tuning rules, mathematical formulas that relate a varying amount of parameters of the process being tuned and its tuning conditions to obtain values for k_c, T_i and T_d, and are not restricted to the context of relay auto-tuning. However, in that very context, the tuning rules used for the most part depend only on variables obtained in the relay experiment and even other desired closed-loop performance parameters, requiring no prior knowledge of the process whatsoever[1].

The tuning rules used in this thesis were collected from the *Handbook of PI and PID controller tuning rules* [20, p. 324], which in turn were gathered from multiple sources available prior to the date of its publication. As mentioned in the previous section, they consist only of the *Ziegler-Nichols type* of tuning rules. A large number of them were sampled and will be compared to each other further on for the specific

[1] Tuning rules that make use of prior knowledge of the process can be used to increase the accuracy of the tuning. However, they are not considered in this thesis.

UAVs considered, based on the comparison of their performance in the context of this thesis with the predefined requirements described in Section 3.2.

Yu discusses how there is not one single tuning rule perfect for every implementation, but instead argues how they depend on multiple factors surrounding the process, such as the process type, order, parameters, nonlinearities, uncertanties, and so forth [32]. Yu also separates the pool of available tuning rules into two main categories, the *Ziegler-Nichols type* tuning rules and the *model-based* tuning rules.

Ziegler-Nichols type tuning rules are described as rules based on the values of k_u and T_u, named after the original rules developed by Ziegler and Nichols [33]. The performance of these rules is, again, case dependent, but a good way to analyse the best tuning rules in each case is the process dead time to time constant ratio $\frac{D}{\tau}$, which the author uses to separate processes into time-constant-dominant processes and dead-time-dominant processes. The value of $\frac{D}{\tau}$ can be approximately determined via the shape of the relay feedback instead of relying on knowledge gathered previously.

The second type, *model-based*, uses an assumption of the model of the process to be tuned to obtain its properties and adapt the tuning parameters based on this theoretical model. These rules work best with an accurate model of the process and provide very good tunings, but require this model to be determined beforehand. This second type of rules will not be considered in this thesis, as the goal remains to obtain a tuning rule regardless of the UAV implemented.

Chapter 5

Simulink® Implementation Platform

To test the theory developed in Section 1.1, corresponding simulation models had to be developed. For this thesis the software package Simulink®, as a subset of MATLAB®, was used for most of the necessary implementations.

The two main components of this project, the UAV simulation and the relay auto-tuner, were developed separately in their own Simulink® libraries, being joined together afterwards to perform the necessary tests and provide the proof of concept. In this chapter, the method of implementation for each component is described.

5.1 Flight Simulation

The flight simulator uses the basis developed in Section 2.5 to provide a time-based simulation of the behaviour of a generic UAV, based on its aerodynamic and geometric data, and is based on the work of Beard and McLain [5]. The component that stores the UAV data is implemented as a library block, a simple independent block that receives the state of flight and commanded inputs and returns the forces and moments applied on the UAV. An example of this block used as an implementation of the MTD UAV used in Chapter 6 and implemented in Simulink® is found in Figure 5.1.

Inside this block are found the implementations of Equations (2.30) to (2.35). This block is then

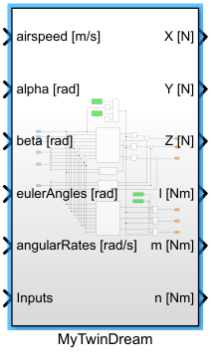


Figure 5.1: The Simulink® block for the MTD UAV, containing all the relations developed in Section 2.5 and adapted to the MTD dataset.

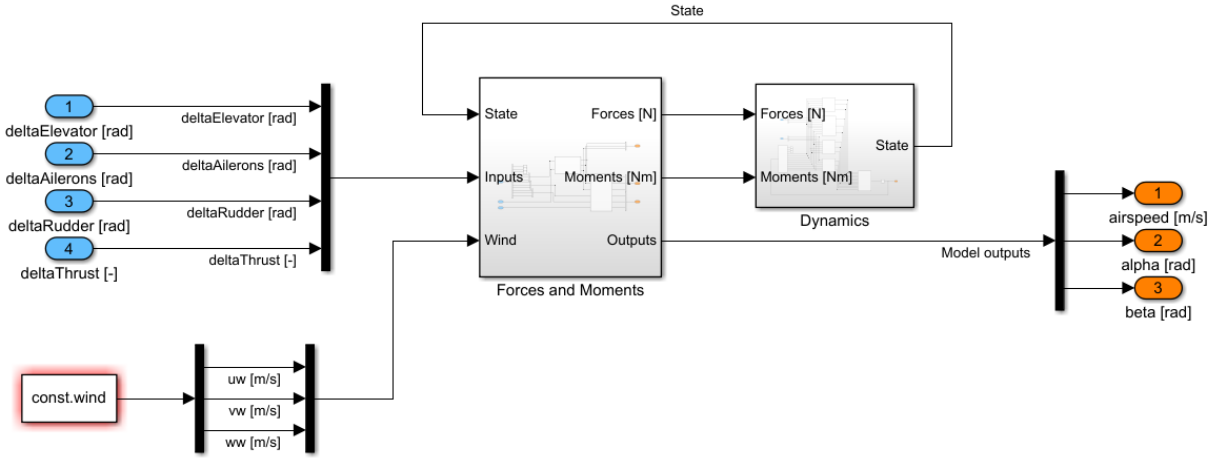


Figure 5.2: The overall structure used to simulate the flight of a UAV block.

implemented into the equations of motion from Section 2.4 to provide a state derivative, which is posteriorly integrated and fed back into the UAV model. The overview of this interaction is seen in Figure 5.2.

This model is also used in conjunction with the `trim` function to obtain a linearised state-space system of any UAV implemented with the system.

5.2 Relay Auto-tuner

The implementation of the relay auto-tuner described in Chapter 4 was done separately from the UAV. The core functioning of the auto-tuner is based on the work of Hornsey [14], but was further expanded on it to provide extra features and tuning rules to be used for testing with the UAV from Section 5.1. The main layout of the Simulink® model of the auto-tuner is found in Figure 5.3.

The core mechanism of the auto-tuner is the determination of the oscillation frequency and amplitude. This is achieved by a zero-crossing counter available in Simulink®, which counts the amount of time a certain signal has crossed the value zero incrementally since a defined starting point. The process variable is measured and subtracted to the equilibrium point defined by the user, which is ideally the process variable value in equilibrium at the start of the tuning. After this, the amount of zero-crossings is divided by the elapsed time since the start of the relay feedback process each second, returning a good approximation of the frequency of the relay switching frequency. The amplitude is measured by obtaining the maximum and minimum values the process variable reaches, and diving its value by two. Because the value might be bigger before it actually converges, an option to set the time of start of amplitude measurements is included. This system is shown in Figure 5.4.

After this, a pre-load block is applied to the gain should a non-zero value be set, and the values of ω_u and k_u are passed to the tuning rules.

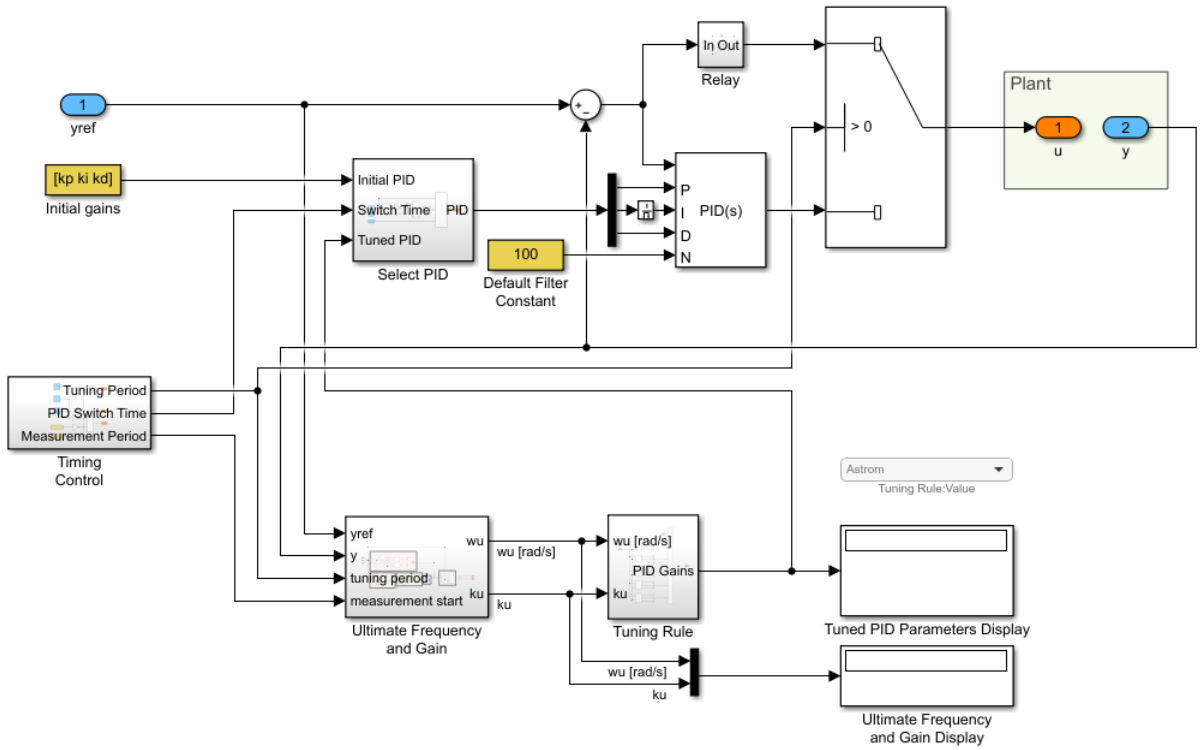


Figure 5.3: The main layout of the inside of the relay auto-tuner block.

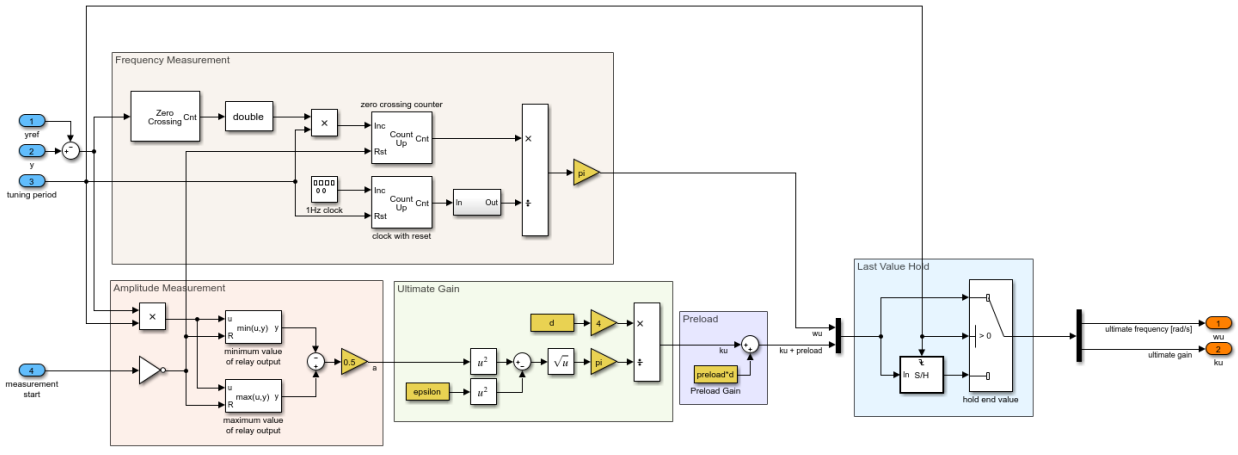


Figure 5.4: The frequency and amplitude measurement system of the relay feedback process. On dark purple, the frequency measuring component, and on pink, the amplitude measuring component. The light blue section holds the final value of the measurement when the tuning ends.

5.3 Implementation

Both components were placed in interaction with each other on a Simulink® model, to execute a flight simulation and test the relay feedback on the system. The initial conditions were determined by the trim algorithm described in Section 2.5. Each of the commanded inputs of the simulated UAV were placed in a feedback loop as defined in Section 3.3, with the exception of the thrust input, which was set to a fixed command, also determined by the trim algorithm.

Chapter 6

MyTwinDream Implementation

This chapter focuses on the implementation of all the topics discussed in a test scenario using the MyTwinDream UAV. Although the final objective of the *SCALAiR* project is the implementation of a scaled down version of an Airbus A320, the MTD allows for an initial testing platform with less risk involved, thus allowing for more experimentation.

6.1 UAV Characteristics

The MyTwinDream, henceforth referred to as MTD, is a small *off-the-shelf* styrofoam UAV used by the *SCALAiR* project to test the implementation of various features of the MicroPilot autopilot and to allow for the pilots the adapt to the hardware's implementation and characteristics. It includes two SunnySky X2814-KV900 propeller engines, one per wing, as well as two ailerons, elevators, and a single rudder for its control surfaces.

The UAV's full characteristic set as it is used in the project can be found in Appendix A.1.

6.1.1 Aerodynamic coefficients

The aerodynamic coefficients were determined via a computer simulation, using a mix of the XFLR5 [8] and AVL [10] programs with its geometric data to generate the dataset seen in Tables A.1b and A.1c and Figure A.1. This data was used to make the mathematical model of the MTD used in the simulations of this thesis project.

6.1.2 Thrust model

The thrust model was obtained via *PropCalc – Propeller Calculator Software* [18] and adapted into a lookup table of various dimensions similar to that seen in Table 6.1. A plot of the engine thrust force for a single engine can be found in Figure 6.2.

Figure 6.1: The MyTwinDream UAV with its propeller blades removed.

Table 6.1: SunnySky X2814-KV900 characteristics for sea-level (h = 0 m) flight.

Throttle [%]	Thrust [N]	RPM
0	0	0
13	0.1764	1200
19	0.392	1800
26	0.7056	2400
32	1.0976	3000
39	1.5876	3600
46	2.156	4200
52	2.8224	4800
59	3.5672	5400
66	4.41	6000
74	5.3312	6600
81	6.3504	7200
88	7.448	7800
96	8.6338	8400
100	9.31	8723

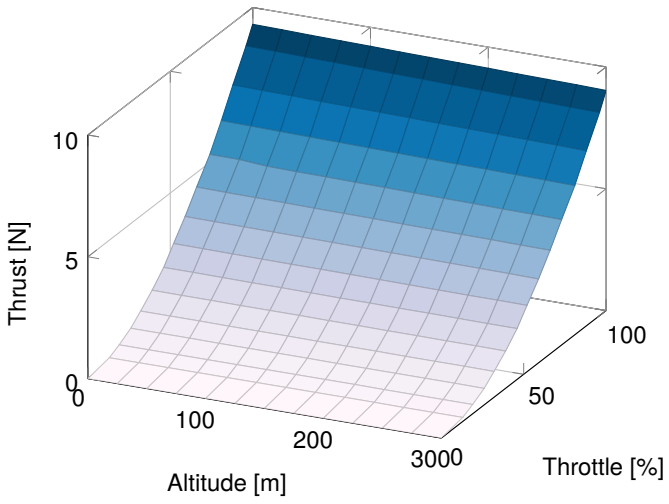


Figure 6.2: SunnySky X2814-KV900 thrust force for a single engine based on throttle command and height.

6.2 Mathematical Model

Using the data presented in Section 6.1, the MTD was developed into a mathematical model of its in-flight behaviour to be linearized and used in simulated testing based on the relations discussed in Section 2.5. This model was adapted into the Simulink® implementation platform where said tests could be run.

6.2.1 Linearized model

The program discussed in Chapter 5 was used to linearize the model. This procedure returns a state-space model of the form

$$\dot{x} = \boldsymbol{A}x + \boldsymbol{B}u$$
$$\dot{y} = \boldsymbol{C}x + \boldsymbol{D}u\,,$$

where the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ reflect the behaviour of the UAV. The flight conditions for this linearization are presented in Table 6.2.

Table 6.2: Trimmed level flight conditions for the MTD linearized model.

Property		Value	
Air density	ρ	1.225	$\mathrm{kg\,m^{-3}}$
Wind speed	V_w	0.0	$\mathrm{m\,s^{-1}}$
Airspeed	V_a	17.0	$\mathrm{m\,s^{-1}}$
Flight path angle	γ	0.0	rad

Considering a state vector similar to that of Equation (2.63) and input vector to that of Equation (2.64), the obtained $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices for the MTD at the conditions of Table 6.2 were obtained. These were decomposed into their longitudinal and lateral counterparts, with the dynamics of the aircraft position p_n, p_e and p_d dropped for not being necessary for the analysis that follows. The longitudinal model is

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.05814 & 0.3051 & 0.05462 & -9.807 \\ -1.182 & -7.863 & 15.13 & 0.0354 \\ -0.03191 & -8.839 & -10.96 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} -0.05165 & 4.14 \\ -14.31 & 0 \\ -171.8 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_t \end{bmatrix}, \tag{6.1}$$

which solely contains the variables concerning longitudinal flight. The lateral model, which in turn models only the variables relative to lateral flight, takes the form

$$\begin{bmatrix} \dot{v} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -0.3047 & 0.02019 & -16.78 & 9.807 & 0 \\ -1.021 & -18 & 3.638 & 0 & 0 \\ 1.722 & -0.5569 & -1.376 & 0 & 0 \\ 0 & 1 & -0.00361 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} v \\ p \\ r \\ \phi \\ \psi \end{bmatrix} + \begin{bmatrix} -0.7077 & 3.805 \\ 259.8 & 5.683 \\ 0 & -24.7 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_r \end{bmatrix}. \tag{6.2}$$

The initial trim conditions calculated by the linearization algorithm around which the model are linearized are represented by the state vector $\boldsymbol{x}$

$$\begin{aligned}\boldsymbol{x} &= \begin{bmatrix} p_n & p_e & p_d & u & v & w & \phi & \theta & \psi & p & q & r \end{bmatrix}^\mathsf{T} \\ &= \begin{bmatrix} 0 & 0 & 0 & 17.0 & 0 & -0.0613 & 0 & 0.0036 & 0 & 0 & 0 & 0 \end{bmatrix}^\mathsf{T}\end{aligned} \tag{6.3}$$

and the input vector $\boldsymbol{u}$

$$\boldsymbol{u} = \begin{bmatrix} \delta_e & \delta_a & \delta_r & \delta_t \end{bmatrix}^\mathsf{T} = \begin{bmatrix} -0.007439 & 0 & 0 & 0.265 \end{bmatrix}^\mathsf{T}. \tag{6.4}$$

All values are in *système international (d'unités)* (SI) units. Throttle is defined between 0 (no throttle) and 1 (full throttle).

6.2.2 Derived transfer functions

The models obtained in Equations (6.1) and (6.2) allow us to obtain transfer functions for the dynamics of the UAV, which are not used directly in the auto-tuning of the PID loops, but are used for various illustrative purposes in this thesis project.

The focus will be on the inner loops discussed in Section 3.3. The *Elevator from Pitch* transfer function obtained from Equation (6.1) looks like

$$\frac{\theta(s)}{\delta_e(s)} = \frac{-171.85(s+7.075)(s+0.1099)}{(s^2+0.03527s+0.4542)(s^2+18.84s+220.2)}, \tag{6.5}$$

the *Aileron from Roll* transfer function from Equation (6.2) is

$$\frac{\phi(s)}{\delta_a(s)} = \frac{259.78s(s^2+1.686s+29.3)}{s(s+17.96)(s-0.08412)(s^2+1.809s+30.8)}, \tag{6.6}$$

and the *Rudder from Y Acceleration* transfer function also obtained from Equation (6.2) is

$$\frac{\dot{v}(s)}{\delta_r(s)} = \frac{3.8049s^2(s+110)(s+18.45)(s-0.1021)}{s(s+17.96)(s-0.08412)(s^2+1.809s+30.8)}. \tag{6.7}$$

6.3 UAV Dynamics

The model developed in Section 6.2 can now be used to analyse the UAV's open-loop[1] flying and handling qualities based on the study done in Section 3.2. The qualities of the UAV are dependent on its dynamic modes and can be obtained from the longitudinal model, which describes the short-period and phugoid modes, and the lateral model, which describes the roll, spiral, and dutch-roll modes. The values describing these characteristics can be obtained from the eigenvalues from the $\boldsymbol{A}$ matrix from the corresponding state-space models, which can be found in Equations (6.1) and (6.2).

[1] Open-loop dynamics consider in this context only the behaviour of the MTD UAV in response to arbitrary control surface inputs, without any type of compensation.

The eigenvalues of a matrix are the values for λ that satisfy the equation

$$|\boldsymbol{A} - \lambda \boldsymbol{I}| = 0\,. \tag{6.8}$$

From the model in Equation (6.1), this means that the eigenvalues are obtained from

$$\begin{vmatrix} -0.05814 - \lambda & 0.3051 & 0.05462 & -9.807 \\ -1.182 & -7.863 - \lambda & 15.13 & 0.0354 \\ -0.03191 & -8.839 & -10.96 - \lambda & 0 \\ 0 & 0 & 1 & -\lambda \end{vmatrix} = 0\,. \tag{6.9}$$

In this project these values were determined using the damp function from the MATLAB® software package. For the case in Equation (6.9), the results were

$$\begin{aligned} \lambda_{\text{lon}} = &-0.0176 \pm 0.6737i \\ &-9.422 \pm 11.46535i \end{aligned} \tag{6.10}$$

Per the definitions given in Section 3.1, the short-period eigenvalues are the complex pair $-9.422 \pm 11.46535i$ and the phugoid eigenvalues are $-0.0176 \pm 0.6737i$.

The short-period dynamics are then obtained from these eigenvalues. The short-period natural frequency $\omega_{n_{\text{sp}}}$ is determined to be $14.84\,\text{rad}\,\text{s}^{-1}$ and its damping ζ_{sp} is 0.635. Analogously, the phugoid natural frequency $\omega_{n_{\text{phu}}}$ is $0.674\,\text{rad}\,\text{s}^{-1}$ and its damping ζ_{phu} is 0.0262.

For the lateral model in Equation (6.2), a similar approach can be taken. The eigenvalues are determined to be

$$\begin{aligned} \lambda_{\text{lat}} = &-17.9593 \\ &-0.9044 \pm 5.476i \\ &0.0841 \\ &0.0 \end{aligned} \tag{6.11}$$

In this case, the roll eigenvalue is -17.9593, the spiral eigenvalue is 0.0841 and the dutch-roll eigenvalues are the complex pair $-0.9044 \pm 5.476i$. The eigenvalue at the origin is not attributed to any mode and can thus be ignored.

The dutch-roll and roll dynamics used in the study of flying and handling qualities can thus be obtained from these results. For the dutch roll, its natural frequency $\omega_{n_{\text{dr}}}$ is $5.55\,\text{rad}\,\text{s}^{-1}$ and its damping ζ_{dr} is 0.163. For the roll dynamics, its time constant τ_{r} is found to be 0.0557 s.

6.3.1 Flying and handling qualities

Following the discussion handled in Section 3.2, the flying and handling qualities of the UAV can be determined.

For category A flight phases, short-period can be seen to equate to level 3, phugoid to level 2, roll to level 1, and dutch-roll to level 2. For category B flight phases, short-period is level 3, phugoid level 2, roll level 1, and dutch-roll level 1. Spiral mode had to be tested separately in the Simulink® model of the MTD, but after giving the UAV a roll angle ϕ disturbance of 10° its value doubled to 20° after 6.615 s, giving the spiral mode a level 3. All in all, this means that the MTD can be classified as a level 3 UAV for category A and B flight phases.

6.4 Auto-tuning Rules

With the transfer functions for the studied loops given in Section 6.2.2 and the flying and handling qualities of the MTD given in Section 6.3.1, the targets for the implemented PID controller parameters and therefore the implemented tuning rules can be defined. For this scenario, their restrictions will be established based on the flying and handling qualities discussed in Section 3.2 and the effect of each PID parameter on the poles of the MTD's dynamic modes.

Using the aforementioned transfer functions and Equation (4.4), the effect of a specific set of PID parameters on the process transfer function can be determined, as well as the poles, and depending on the loop, the behaviour of the UAV dynamic modes. Knowing that, for a process $G(s)$ and a compensator $C(s)$, the transfer function of the system with feedback is

$$H(s) = \frac{G(s)C(s)}{1 + G(s)C(s)},$$

MATLAB® can be used to analyse various sets of PID parameters and determine their effect on the flying and handling qualities of the UAV. This was achieved using a mix of the MATLAB® functions `damp`, `feedback`, and `pidstd`, which would return a transfer function with the effect of the feedback in place with the poles for each corresponding dynamic mode adjusted accordingly. With this information, the calculation of flying and handling qualities was automated using `damp` to allow for mass testing of different PID parameters and, consequently, tuning rules.

For the MTD UAV, a total of 44 tuning rules were adapted from the *Handbook of PI and PID controller tuning rules* [20] and tested on the MTD transfer functions to observe their effect on the UAV's flying and handling qualities. Out of all 44, 3 tuning rules were shown to satisfy level 1 flying and handling qualities for level flight and are presented in Table 6.3.

Table 6.3: Tuning rules selected for this thesis. Adapted from [20].

Author	k_c	T_i	T_d
Pettit and Carr [21]	$0.5k_u$	$1.5T_u$	$0.167T_u$
Fuxiang and Zhixiong [12]	$0.27k_u$	$2.40T_u$	$1.32T_u$
Luyben and Luyben [15]	$0.46k_u$	$2.20T_u$	$0.16T_u$

An additional rule not included in Table 6.3 was chosen for this thesis, based on the work of Åström and Hägglund [2], which has an additional phase margin ϕ_m parameter, accompanied by a rule-specific α

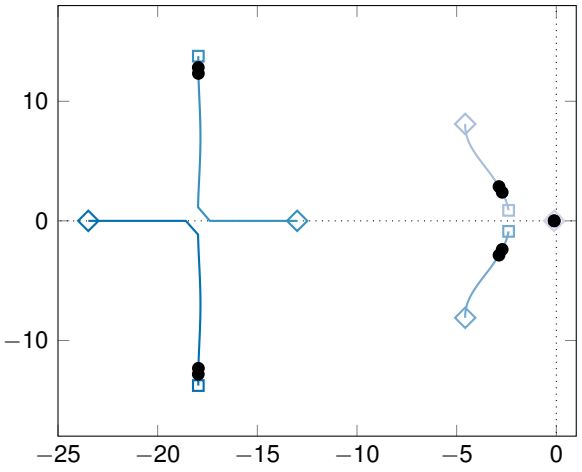


Figure 6.3: Poles of the short-period mode of the MTD for the range of values of 1 to 10 of the α parameter of the Åström and Hägglund tuning rule for the MTD aircraft. The value of α starts at 1 at the diamonds and climbs to 10 at the squares. Represented in black are the pole locations for the case of α = 4 and α = 5.

parameter. The proportional gain for a standard PID controller is given by

$$k_c = k_u \cos \phi_m \,. \tag{6.12}$$

The relation between T_i and T_d is given as a linear ratio α

$$T_i = \alpha T_d \,, \tag{6.13}$$

a ratio that is used to determine the value of T_d from the ultimate frequency ω_u through

$$T_d = \frac{\tan \phi_m + \sqrt{\frac{4}{\alpha} + \tan^2 \phi_m}}{2\omega_u} \,. \tag{6.14}$$

T_i is then determined by Equation (6.13). The value of α most commonly used is 4 [2], for unclear reasons.

This fourth tuning rule, chosen due to its simplicity and importance in the very first implementations of relay-based auto-tuners, fell short of level 1 short-period mode due to having the natural frequency of its poles be far too high. However, studying its adjustable parameter α and its effect on these poles, a diagram was plotted, presented in Figure 6.3, and a value of 5 was chosen based on the position of the short-period poles, which would now satisfy the level 1 flying and handling quality requirements without compromising any other, bringing up the total of selected tuning rules that satisfy the outlined requirements to four.

6.5 Tests

The MTD will implement the MP2128^{LRC2} autopilot as its controller. Having the UAV's properties considered in the previous section, the auto-tuner developed in Chapter 5 can be used in conjunction with the mathematical model of the UAV to determine the gains to be used in each of the autopilot's loops. However, the auto-tuner must be adjusted to fit its target, especially concerning the sizing of its outputs.

6.5.1 Basic experiment

The developed auto-tuner has support for a number of properties to be adjusted. A basic experiment is first done, which can later be supported with expanded properties and techniques for improved accuracy or different tuning approach. This first basic experiment requires the definition of the relay output d, the duration of the tuning experiment, and the point in time after which the amplitude measurement begins. These values do not require calculations and can be mere guesses, but they require realistic values given their context. As such, an understanding of the physical properties of their implementation helps in these definitions.

Three PID loops are tested. For each loop, a single gain schedule is tested, solely at the condition of level flight. In addition, four different tuning rules are obtained. Considering that for each of these tests, loops, and rules, a single set of PID parameters is obtained, a total of 36 PID parameters are obtained for the basic experiments.

An important detail is the definition of initial PID values, which need to be used before activating the auto-tuning relay experiment to bring the UAV to a trimmed condition. As previously mentioned, the relay auto-tuning is best performed from a state of equilibrium. The linearization returns initial trim conditions, seen in Equations (6.3) and (6.4), which are used as initial conditions for the UAV simulation. Therefore, the simulation starts in equilibrium, meaning the relay auto-tuning can and is indeed started at zero seconds, meaning the loops require no initial PID parameters.

The three inner loops listed in Section 3.3 are considered as the first example.

Elevator from Pitch

For the *Elevator from Pitch* loop, the relay is placed where a compensator would be, similar to Figure 4.5, thus outputting the value of the process input, in this case the elevator deflection δ_e. The value of d has to be large enough to cause a measurable effect on the UAV, while not being too large to cause instability, and while staying within the UAV limits. Considering the actuator limits, $-30°$ to $+15°$, a good starting point for the value of d would be $10°$.

The sign of d has also to be carefully reconsidered in order to prevent unstable feedback. Looking at the conventions in Table 6.4, the mismatch between the elevator deflection and its effect on the pitch moment is visible, with positive elevator deflections producing negative pitch moments. So for this scenario, the sign of d is established as negative, leaving its final value as $-10°$. This can be proved by the static gain of the loop's corresponding transfer function in Equation (6.5), which has a negative value of -1.335.

Table 6.4: Aircraft control surface sign conventions. Adapted from Beard and McLain [5, p. 42], Nelson [19, p. 62], and the *MicroPilot Autopilot Instalation & Operation Manual*, p. 184.

	Deflection	**Sense**	**Primary effect**
Elevator	Trailing edge down	Positive	Negative pitching moment
Rudder	Trailing edge left	Positive	Negative yawing moment
Ailerons	Left-wing trailing edge down	Positive	Positive rolling moment

The duration of tuning and time of measurement are arbitrary and can be as large as necessary, as long as other variables don't affect the tuning during that time period (*e. g.* wind gusts). However, since ideally the process has to settle into the oscillation amplitude first, some time is necessary before measurements can be taken place. Additionally, a good measurement should count for about more than three oscillations of the process. With this in mind, a guess can be taken on how long the tuning should take. The inner dynamics of a UAV are considerably fast in comparison to the industrial processes studied by PID literature [3, 14], so a tuning can be concluded within a relatively short time period.

For all inner loop tunings considered in this thesis, the tuning period was established at 10 s, with measurements starting at 5 s.

Aileron from Roll

For the *Aileron from Roll* loop, whose limits are $-25°$ to $25°$, $20°$ was used. The sign of d for this scenario is positive (see Table 6.4), so $d = +20°$.

Rudder from Y Acceleration

For the *Rudder from Y Acceleration* loop, whose limits are $-30°$ to $30°$, a lower value for d had to be chosen due to the sensitive nature of this loop. A value of $3°$ was used. The sign of d is this case is more tricky to establish since its static gain is zero (from Equation (6.7)), and Table 6.4 does not correlate directly the effect of the rudder on the UAV's side force. However, knowing that the aerodynamic coefficient $C_{Y_{\delta_r}}$, which correlates the effect of the rudder on the side force, is positive, a positive deflection on the rudder can be assumed to yield a positive side force. Hence, $d = +3°$.

Also due to the linear nature of the simulation of this loop (refer to Equation (2.53)), coupled with an absence of noise in the experiments and good speed of the actuators implemented, the oscillations of this loop will be very fast, very small, and dependent on the duration of each simulation step. To avoid his scenario but still obtain usable ultimate frequency and gain values, a hysteresis was applied to the relay, which will hold the relay output until its input crosses a symmetrical treshold. For this scenario, and since the relay hysteresis (described in Section 4.4.1) is applied based on the measured input, the chosen value for the relay hysteresis ϵ is 1 g, which is sufficient to cause the *Rudder from Y Accelerometer* loop to oscillate without risking causing any major damage.

For every experiment, the loops are tuned in the order listed here.

Chapter 7

Results and Discussion

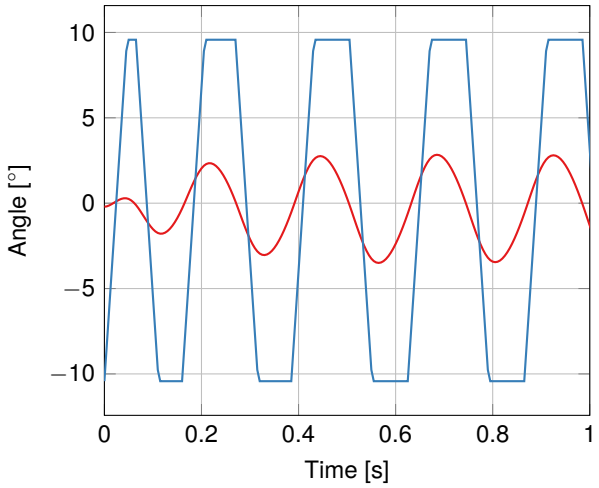
Following the presentation of the implementation and simulations for the MTD UAV in the previous chapter, the results were obtained and will be presented and discussed in full in this chapter. The second part of the chapter discusses what could be added to the experiments carried out in the progress of this thesis to improve the accuracy of the results, what additional theory can be studied and possibly implemented in addition to what was already included and all the possibilities and objectives considered in the scope of this thesis but ultimately not carried out for various reasons.

7.1 The Relay Experiment

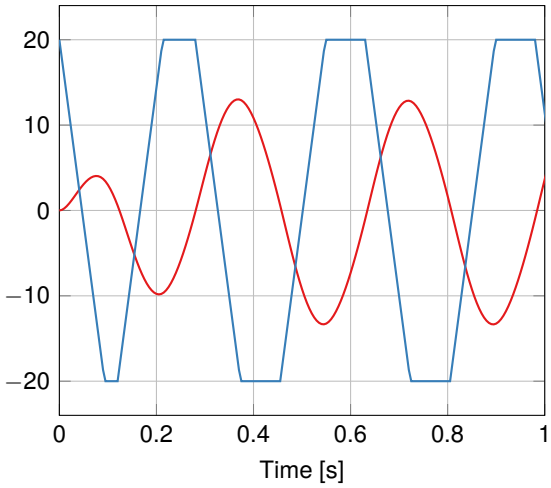
The simulation described in Section 6.5 was thus run in the simulated environment described in Chapter 5. The results were afterwards extracted from Simulink® and stored for comparison with each other. The data gathered for each simulation step includes the time, set point, measured variable, command output, ultimate frequency, ultimate gain, and the three PID parameters calculated.

The first second of the relay experiment for each of the three loops can be seen in Figure 7.1. As the tuning is only applied after the end of the experiment (which is outside of the time scope represented), these results are independent of the tuning rule applied. Various differing behaviours can be seen in these three plots, but the first analysis can be focused on the *Elevator from Pitch* loop, represented in Figure 7.1a. Since the simulation starts with the UAV in a trimmed condition, the oscillation of the pitch angle first requires just about three full oscillations to reach its full amplitude, and at this point the relay auto-tuner is ready to determine the ultimate frequency and gain. Also because of the initial trim condition, the relay is seen to oscillate not exactly between $-d$ and $+d$, but between their differences with the set point, which was calculated to be different than zero. Not only that, but its output is not a perfectly square-type wave, but rather one with a ramping of its signal around the switching points. This happens due to the implementation of actuator models, which restrict the maximum rate of change of the elevator deflection δ_e.

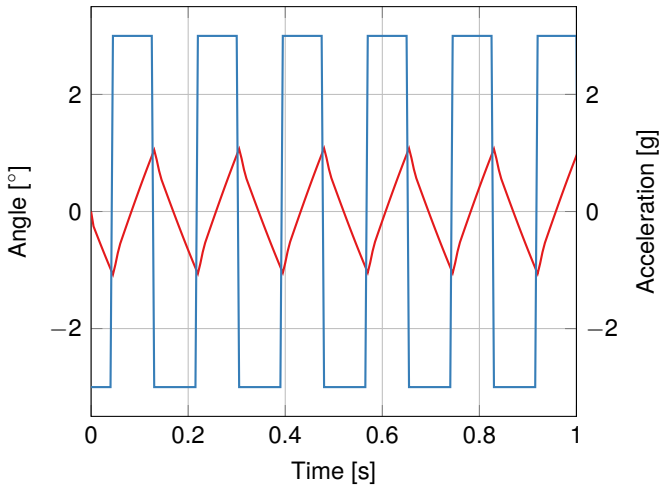
The behaviour of the second loop, *Ailerons from Roll*, visible in Figure 7.1b, is very similar to the *Elevator from Pitch*, with longer and bigger oscillations than its elevator counterpart. This can be explained



(a) *Elevator from Pitch* loop.



(b) *Ailerons from Roll* loop.



(c) *Rudder from Y Accelerometer* loop.

Figure 7.1: Behaviour of the control surfaces during the simulated relay feedback experiment. The red line is the process variable, and the blue line is the commanded variable, controlled by a relay.

by the higher value of the set relay output d, which leads to these bigger oscillations. However, since the value of the outputted PID parameters depends on both amplitude and frequency, adjusting the value of d will not affect the final values of the PID parameters. In this case, a bigger value for d was chosen as it allows the experiment to obtain more conclusive results.

The last loop, *Rudder from Y Acceleromter*, is, in this scenario, the oddest of the three. The shape of the Y acceleration does not take a near-sinusoidal form as its other counterparts, but rather a triangular shape. Also to note is that its waveform changes direction when it reaches 1 g. All these effects are caused by the presence of a positive hysteresis ϵ of 1 g, which causes the waveform to obtain its characteristic shape. The value of the hysteresis is well visible in Figure 7.1c, in an interval in which the value of the Y acceleration remains steady until its next point of inversion. Yu [32] has an excellent chapter discussing the waveforms of the relay auto-tuning experiment and how their shape can help identify the process behind the relay auto-tuner (should this be unknown).

Table 7.1: Relay experiment results for the MTD UAV.

Loop	d	ϵ	ω_u	k_u
Elevator from Pitch	$-10°$	0	25.761 06	−3.678 26
Ailerons from Roll	20°	0	17.907 08	1.818 44
Rudder from Y Acceleration	3°	1 g	35.814 15	0.187 24

7.2 Tuning Results

As mentioned in Section 6.5, 36 PID parameters were obtained, 12 for each loop. The results obtained are those immediately obtained at the end of the relay experiment described in Section 6.5.

From the experiments visible in Section 7.1, it can be asserted that all three loops eventually reach a periodic oscillation with frequency equal to the ultimate frequency and an ultimate oscillation amplitude used to determine the ultimate gain. In the case of the *Rudder from Y Accelerometer* the frequency obtained, while it isn't the ultimate frequency, it can be used to determine the ultimate frequency. The results of these relay feedback experiments can be found in Table 7.1.

Important to note is the difference in the gain signals, which do match up with the discussion in Section 6.5. Had any of the signals of the d been wrong, the oscillations would not occur and the value of the measured variable would increase until the UAV spirals out of control. Equally notable is the fact that while the oscillations are approximately within the same range, the gain for the *Rudder from Y Acceleration* loop is quite lower. This is not due to a lower value for the relay output d, which is normalized by the amplitude of the oscillations via Equation (4.17), but due to a mixture of a higher effectiveness of the rudder on its corresponding loop variable, and the presence of a hysteresis factor, which affects the value of the descriptive function (and consequently the ultimate gain) via Equation (4.18).

Once the ultimate frequency and gain are obtained, a tuning rule can be applied to obtain the values of the required PID parameters. The four tuning rules presented in Section 6.4 were used on the results from Table 7.1, and the corresponding PID parameters were obtained, presented in Table 7.2.

Immediately apparent is the similarity between some of the results between tuning rules, namely the relation between Åström and Hägglund and Pettit and Carr. From Section 6.4, knowing that the phase margin ϕ_m specification used by the Åström and Hägglund tuning rule is 60° (or $\frac{\pi}{3}$ radians), the proportional gain is then known to be $k_u \cos 60°$, or $0.5k_u$, identical to the proportional gain of Pettit and Carr.

Another identifiable oddity is the discrepancy between the values returned by the Fuxiang and Zhixiong rule comparing to the remaining values. In comparison, and looking at Table 6.3 and of course Table 7.2, the Fuxiang and Zhixiong provides a PID controller with far less proportional action, close to half that of the remaining rules, but balances it out by having a slight increase in integrative action and a substantially more aggressive derivative action. While this in theory increases the damping of the system in response to oscillations brought about by the integrative action, as discussed in Section 4.1, a more aggressive derivative action also increases any loop's sensitivity to incoming noise. While no noise was modelled in the simulations run in this thesis project, it could become a problem in practical implementations.

Table 7.2: Tuning results for the analysed feedback loops.

(a) Tuning results for the *Elevator from Pitch* loop of the MTD.

Author	k_c	T_i	T_d
Pettit and Carr	−1.8391	0.3658	0.0407
Fuxiang and Zhixiong	−0.9931	0.5853	0.3219
Luyben and Luyben	−1.6920	0.5365	0.0390
Åström and Hägglund	−1.8391	0.3572	0.0714

(b) Tuning results for the *Ailerons from Roll* loop.

Author	k_c	T_i	T_d
Pettit and Carr	0.9092	0.5263	0.0585
Fuxiang and Zhixiong	0.4899	0.8421	0.4631
Luyben and Luyben	0.8365	0.7719	0.0561
Åström and Hägglund	0.9092	0.5139	0.1027

(c) Tuning results for the *Rudder from Y Accelerometer* loop.

Author	k_c	T_i	T_d
Pettit and Carr	0.0936	0.2631	0.0292
Fuxiang and Zhixiong	0.0502	0.4210	0.2315
Luyben and Luyben	0.0855	0.3859	0.0280
Åström and Hägglund	0.0936	0.2569	0.0513

7.2.1 Step tests

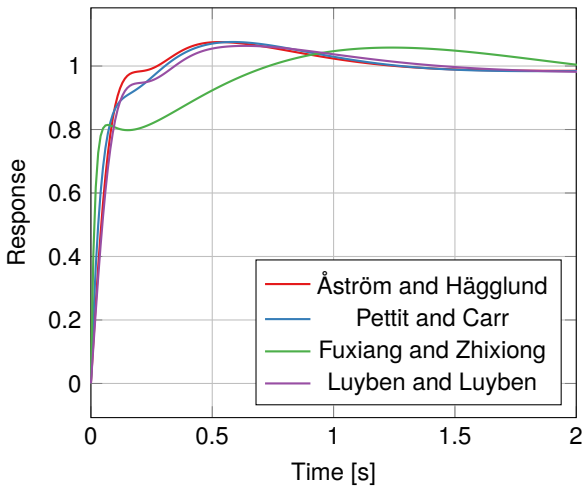
Once the final results for the PID parameters were determined, a comparison of the performance of each set of parameters and, consequently, of each tuning rule in the context of the MTD was carried out. By taking the transfer functions for each inner loop of the MTD as determined in Section 6.2.2 and applying each set of PID parameters as described in Section 6.4, performance of the resulting SISO transfer functions can be analysed.

In his context, the step response for each transfer function was done within MATLAB® using the `step` function. Figure 7.2 displays the results obtained for each loop and tuning rule.

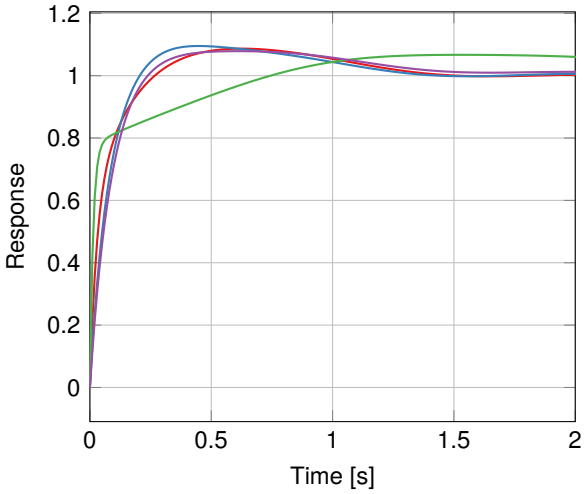
The *Elevator from Pitch* loop performance, shown in Figure 7.2a, shows quite the similarity between the step responses obtained by each rule, with the exception being the Fuxiang and Zhixiong, which behaves quite uniquely, consequence of its increased integrative and derivative actions. It is the fastest of the rules to reach its first peak at around 0.8, and then corrects its own steady state error, but due to the increased integrative action, some slower oscillation is induced afterwards and corrected more slowly due to a reduced proportional action, making it the slowest set of parameters to converge.

Between the remaining sets of rules, Åström and Hägglund appears to provide the fastest peak and settling time but by a very small margin. The remaining rules perform appropriately and with very little distinction from this last one.

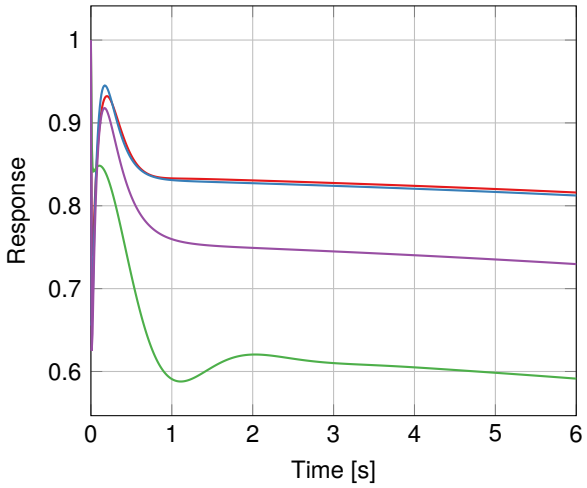
From the *Ailerons from Roll* loop performance, shown in Figure 7.2b, once again the exception is the Fuxiang and Zhixiong rule, which shows the fastest peak time and damping with the smallest overshoot,



(a) *Elevator from Pitch* loop.



(b) *Ailerons from Roll* loop.



(c) *Rudder from Y Accelerometer* loop.

Figure 7.2: Step tests using the transfer functions for each loop determined in Section 6.2.2 with the PID parameters determined via the relay feedback test.

but slowest settling time, having considerable oscillations ($\simeq 0.05$) even after the two seconds displayed. The remaining parameters now have slightly bigger differences between their responses, with Pettit and Carr now having the fastest peak time but bigger overshoot in comparison to the other two.

The *Rudder from Y Accelerometer* provides the most interesting step response. Not only does it not reach the given target of 1, but it also slowly decays away from said target back to zero. Since the analysis is performed here using the linearized transfer functions from Section 6.2.2, this slow decay cannot be caused by other effects of the UAV, but it can be attributed to integral action of the implemented PID controller, with is further evidenced by the fact that the tuning rules with more integrative action are diverging from the target 1 the fastest, as can be seen in Figure 7.2c. This points to the fact that the PID controller assumes a steady-state error, meaning it wants to lead the input signal back to zero. As the *Rudder from Y Acceleration* loop is used to maintain the acceleration along the y axis zero, this does not

pose as much of an issue.

With these results, it can be asserted that for the tuning rules that satisfy good flying and handling quality requirements, Fuxiang and Zhixiong provides the fastest peak, smallest overshoot, and highest damping properties, but its reliance on integrative and derivative action means its settling time is eventually sacrificed, and could pose a problem once noise models are introduced. For a more reliable alternative, between the remaining tuning rules, due to the similarity in their results, the Åström and Hägglund rule would be the preferred choice for general implementation scenarios, as it provides much the same properties but allows for the specification of a phase margin and provides an extra tuning parameter that regulates the influence of integrative and derivative action, should so be necessary.

7.3 Conclusions

The mathematical model of the MTD UAV was successfully developed based on its geometrical and physical data, and a trim point was found for a level flight condition. Alongside this development, the relay auto-tuner was developed on the Simulink® platform to interact with any given process that it was implemented in, being validated every step of the way.

In being developed and validated separately, the interaction between both components eventually determined the outcome of the plan this thesis set out to accomplish, which was successful for the tests that were carried out with both platforms. Relay feedback was achieved for all input-output pairings tested for the MTD, and sets of parameters for desired PID controllers were obtained and compared to each other based on their performance. With this, the thesis showed that not only is a relay-based auto-tuner theoretically possible for small-scale PID-controlled UAV, but that it satisfies the required conditions for good flying and handling qualities and that there is room for improvement and adjustment in a case-by-case approach should more requirements come to play for different missions.

7.4 Considerations Post-Work

Following the literature study undertaken along the period of development of this thesis, a large number of different possibilities to improve or follow up the initial plan were discovered and while some of these improvements were laid out on the finished work and taken proper conclusions on, namely a great amount of alternative and improved tuning rules [20], the scope of said possibilities is quite large and very interesting to be further investigated. In this section are laid out some improvements on the original design that can be implemented based on the knowledge herein presented and some interesting literature on which future work following these improvements can be based, mostly concerning modifications to the relay auto-tuner presented in Chapter 4.

7.4.1 Future development

The transition between the auto-tuning experiment and the application of tuned gains to a PID controller needs to be adjusted to make the transition smoother and avoid undesired spikes in either the input or output values due to the switch being made in a moment when either of these values are changing at a higher rate than desired, often due to the sharp on-off behaviour of a relay. This can be achieved by making the transition dependent on the behaviour of the output, making sure it occurs at a point where the ultimate frequency is changing at a rate below a certain threshold (meaning the ultimate point has been reached) and the output value of the process as well (meaning the transition won't be as abrupt). As it stands the transition and application of gains is done based on time alone.

Another development point could be to explore the outer loops of the controller. The dynamics of the outer loops are different enough from the inner loops to provide a challenge to the rules and analyses applied here, and considering the flying and handling qualities are at that point, ideally, fulfilled to give a level 1 UAV, the requirements for the outer loops, while less stringent and less crucial to the stability of the UAV, would have to be reconsidered, and possibly adapted to the mission of the UAV on a case-by-case basis.

7.4.2 Next steps

The main shortcoming of the project was the lack of practical implementation. Due to a number of difficulties encountered with converting the theoretical values of gains into the platform of implementation, this topic was left unexplored. However, it could be a very interesting project to see the implementation done here on a computer simulation and have its results tested with the MTD UAV or any other UAV adapted into the Simulink® platform, and even though in this scenario any faults and imperfections in the calculated model of the UAV could carry over to the final results, it could be the real test of the viability of such a procedure in the tuning of a UAV.

Another possibility would be the practical implementation of the tuning procedure itself during the flight. Instead of deploying the relay auto-tuner on a virtual simulation of a flight with a theoretical model of the target UAV, implementing the auto-tuning capability in a *one-click* procedure could test the viability of the study done here in actual UAV and automate the tuning process of a PID-controlled UAV.

7.4.3 Further reading

Since the introduction of the relay auto-tuner by Åström and Hägglund in 1984 [1], the experiment has seen multiple developments on the original premise, with different variations on the multiple parts of the experiment with varying results for different contexts of utilization. Due to the sheer amount of improvements on the experiment developed since its inception, a lot of alternatives to the techniques implemented in this thesis were not experimented with. This section cites some of the possibilities considered.

Wang, Hang and Bi [31] suggests a decomposition of the input and output signals on the relay within the time it takes the frequency response to stabilise to obtain points of different frequency on the Nyquist

curve based on the number of samples obtained and, given a sufficient amount of said samples, recreate an approximation of the entire Nyquist curve between 0 and $-\pi$ radians of phase lag. This possibility can be used instead of the techniques mentioned in Section 4.5 for tuning rules which make use of this additional information for tuning of different characteristics.

A whole other use for the relay experiment is process identification, mentioned in more encompassing literature [3, 32]. This uses more information obtained through the experiment to obtain a simplified mathematical model of the process being tested, usually in the form of a transfer function. This procedure can be used for some of the dynamics in the aircraft that can be reliably reduced to second-order SISO models and tuned mathematically via more common methods [28, 19, 5].

Yu [32] suggests identifying the shape of the relay feedback resulting from an experiment to obtain the order and characteristics of a process. This approach loses accuracy as the order of the process is bigger, however, considering the approximations of a maximum of second-order done for the dynamics of level flight, it is a possibility worth exploring to enhance the quality of the tunings discussed previously coupled with model-based tuning rules appropriate for each scenario.

O'Dwyer [20] has a great assembly of tuning rules in his *Handbook of PI and PID controller tuning rules*, some of which were used along this thesis. However, tuning rules based on additional process information or other determinable variables, which could yield more interesting results, were seldom looked at, requiring more modifications to the original relay auto-tuner than time could afford. Not only that, but the same source includes a lot more tuning rules depending on the applied process, which could be identified in the way discussed in the paragraph prior, or by having an accurate model of the UAV being flown, a step which lied beyond the objectives laid out for this thesis. This could open the possibility of using accurate models of the flown UAV to complement the relay auto-tuning, which could lead to more accurate and satisfactory tunings.

The simulation executed in this thesis project developed with MATLAB® and Simulink® was of a continuous nature, with each simulation step being calculated based on all previously obtained and calculated data and the signals in between each step interpolated based on the steps surrounding them. In an analog implementation of a relay auto-tuner, this type of simulation is sufficient and more accurate to represent the real system given that all its components are accurately modelled. However, given the more widespread implementation of digital controllers, a discrete simulation of the system would be more accurate and practical for most implementations in selected programming languages and other platforms, including the MicroPilot platform used in this thesis. A discretization of the models developed here could help for easier implementation in a digital system. Åström and Murray [4, pp. 11–21] give a brief description in the implementation of a digital PID controller in pseudocode, while Åström and Hägglund [3, p. 93] give an more in-depth discussion on the discretization of a PID controller and the consequences thereof.

Bibliography

[1] Karl Johan Åström and Tore Hägglund. 'Automatic Tuning of Simple Regulators'. In: *IFAC Proceedings Volumes* 17.2 (1984), pp. 1867–1872. DOI: 10.1016/S1474-6670(17)61248-5.

[2] Karl Johan Åström and Tore Hägglund. 'Automatic Tuning of Simple Regulators with Specifications on Phase and Amplitude Margins'. In: *Automatica* 20.5 (1984), pp. 645–651. DOI: 10.1016/0005-1098(84)90014-1.

[3] Karl Johan Åström and Tore Hägglund. *PID controllers. Theory, design and tuning*. 2nd ed. Instrument Society of America, 1995. ISBN: 1-55617-516-7.

[4] Karl Johan Åström and Richard M. Murray. *Feedback Systems. An Introduction for Scientists and Engineers*. Version v3.0j. 18th Aug. 2019. URL: http://www.cds.caltech.edu/~murray/FBS (visited on 25/09/2019).

[5] Randal W. Beard and Timothy W. McLain. *Small unmanned aircraft. Theory and practice*. Jan. 2012. ISBN: 978-0-691-14921-9. URL: http://uavbook.byu.edu/doku.php (visited on 01/04/2019).

[6] YangQuan Chen, ChuanHua Hu and Kevin L Moore. 'Relay feedback tuning of robust PID controllers with iso-damping property'. In: *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*. Vol. 3. IEEE. 2003, pp. 2180–2185. DOI: 10.1109/CDC.2003.1272941.

[7] Department of Defense. *Flying Qualities of Piloted Aircraft. Department of Defense Handbook*. Tech. rep. Version MIL-HDBK-1797. Department of Defense, 19th Dec. 1997.

[8] André Deperrois. *XFLR5*. Version 6.47. 7th July 2019. URL: http://www.xflr5.tech/xflr5.htm (visited on 10/10/2019).

[9] Dante A. DiFranco. *Flight Investigation of Longitudinal Short Period Frequency Requirements and PIO Tendencies*. Tech. rep. Version AFFDL-TR-66-163. Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, June 1967.

[10] Mark Drela and Harold Youngren. *AVL*. Version 3.36. 23rd Feb. 2017. URL: http://web.mit.edu/drela/Public/web/avl/ (visited on 10/10/2019).

[11] Tyler Foster and Jerry Bowman. 'Dynamic Stability and Handling Qualities of Small Unmanned-Aerial Vehicles'. In: 43rd AIAA Aerospace Sciences Meeting and Exhibit (10th Jan. 2005).

[12] Chen Fuxiang and Yang Zhixiong. 'Self-Tuning PM Method and its Formulas Deduction in PID Regulators'. In: *Zidonghua Xuebao/Acta Automatica Sinica* 19.6 (1993), pp. 736–740.

[13] John Hodgkinson. *Aircraft Handling Qualities*. Blackwell, 1999. ISBN: 0-632-03816-0.

[14] Stephen Hornsey. 'A Review of Relay Auto-tuning Methods for the Tuning of PID-type Controllers'. In: 5.2 (2012): *Reinvention: an International Journal of Undergraduate Research*. ISSN: 1755-7429. URL: https://warwick.ac.uk/fac/cross_fac/iatl/reinvention/archive/volume5issue2/hornsey (visited on 14/06/2019).

[15] Michael L. Luyben and William L. Luyben. *Essentials of Process Control*. McGraw-Hill, 1997.

[16] MicroPilot. *MicroPilot Autopilot Instalation & Operation Manual*. Version MP2128^{LRC2}. MicroPilot Inc. 4th Feb. 2013.

[17] J.A. Mulder et al. *Flight Dynamics*. Lecture Notes AE3202. Technische Universiteit Delft, 24th Mar. 2013.

[18] Markus Müller. *PropCalc – Propeller Calculator Software*. Version P7.16c. 25th Mar. 2019. URL: https://www.ecalc.ch/motorcalc.php (visited on 26/07/2019).

[19] Robert C. Nelson. *Flight stability and automatic control*. 2nd ed. WCB/McGraw Hill New York, 1998. ISBN: 0-07-046273-9.

[20] Aidan O'Dwyer. *Handbook of PI and PID controller tuning rules*. 3rd ed. Imperial College Press, 2009. ISBN: 1-84816-242-1.

[21] John W. Pettit and Douglas M. Carr. 'Self-tuning controller'. 4669040. 26th May 1987.

[22] Pakorn Poksawat, Liuping Wang and Abdulghani Mohamed. 'Automatic Tuning of Attitude Control System for Fixed-Wing Unmanned Aerial Vehicles'. In: *IET Control Theory and Applications* 10.17 (July 2016), pp. 2233–2242. DOI: 10.1049/iet-cta.2016.0236.

[23] Charles F. Prosser and Curtiss D. Wiler. *RPV Flying Qualities Design Criteria*. Tech. rep. Version AFFDL-TR-76-125. AFFDL/FGC, Dec. 1976.

[24] Kuldeep Rana et al. 'Aircraft's pitch axis control using relay feedback method'. In: *International Conference on Advances in Engineering, Technology and Sciences* (July 2014).

[25] Tariq Samad. 'A survey on industry impact and challenges thereof'. In: *IEEE Control Systems Magazine* 37.1 (2017), pp. 17–18. DOI: 10.1109/MCS.2016.2621438.

[26] C. Scali and R. Costaggiu. 'Relay with Hysteresis for Monitoring and Controller Design'. In: *IFAC Proceedings Volumes* 31.11 (1998), pp. 565–570. DOI: 10.1016/S1474-6670(17)44986-X.

[27] Kristian Soltesz and Tore Hägglund. 'Extending the relay feedback experiment'. In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 13173–13178. DOI: 10.3182/20110828-6-IT-1002.00066.

[28] Brian L. Stevens and Frank L. Lewis. *Aircraft Control and Simulation. Dynamics, Controls Design, and Autonomous Systems*. English. New York: Wiley-Blackwell, 1992. ISBN: 978-1-118-87098-3.

[29] ArduPilot Dev Team. *Automatic Tuning with AUTOTUNE*. 1st Jan. 2017. URL: http://ardupilot.org/plane/docs/automatic-tuning-with-autotune.html (visited on 11/09/2019).

[30] Nurbaiti Wahid and Nurhaffizah Hassan. ‘Self-tuning fuzzy PID controller design for aircraft pitch control’. In: *2012 Third International Conference on Intelligent Systems Modelling and Simulation*. IEEE. 2012, pp. 19–24. DOI: 10.1109/ISMS.2012.27.

[31] Qing-Guo Wang, Chang-Chieh Hang and Qiang Bi. ‘A Technique for Frequency Response Identification from Relay Feedback’. In: *IEEE Transactions on Control Systems Technology* 7.1 (1999), pp. 122–128. DOI: 10.1109/87.736766.

[32] Cheng-Ching Yu. *Autotuning of PID Controllers. A Relay Feedback Approach*. 2nd ed. Springer Science + Business Media, 2006. ISBN: 1-84628-036-2.

[33] John G. Ziegler and Nathaniel B. Nichols. ‘Optimum Settings for Automatic Controllers’. In: *ASME* 64.11 (1942), pp. 759–768.

Appendix A

Aircraft Parameters

This chapter presents the data utilised in the practical simulations and calculations of Section 4.6.

A.1 MyTwinDream parameters

The data obtained for the MyTwinDream UAV was calculated using the XFLR program with the known geometric properties. The thrust data was obtained via *PropCalc – Propeller Calculator Software* [18].

Table A.1: Parameters for the MTD aircraft.

(a) Geometric and mass data.

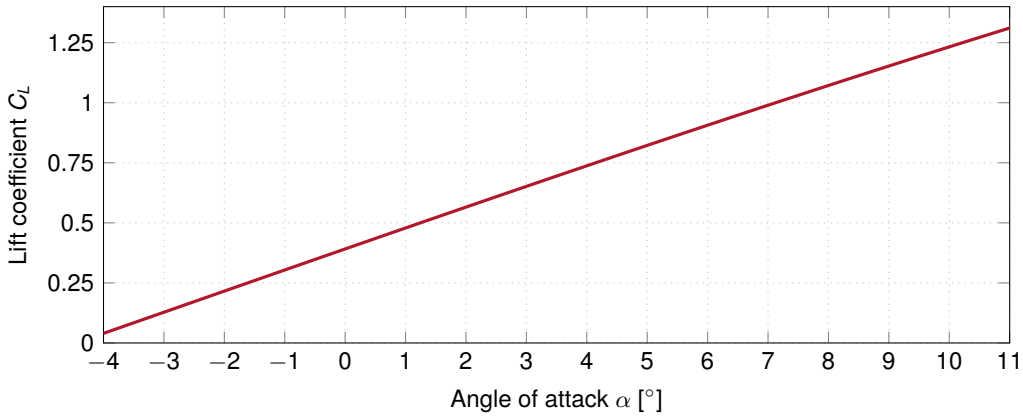
Property	Value	Unit
m	3.92	kg
b	1.8	m
S	0.4736	m^2
$\bar{c}$	0.2637	m
J_{xx}	0.213	$kg\,m^2$
J_{yy}	0.171	$kg\,m^2$
J_{zz}	0.350	$kg\,m^2$
J_{xy}	0.040	$kg\,m^2$
J_{xz}	0.000	$kg\,m^2$
J_{yz}	0.000	$kg\,m^2$

(b) Longitudinal derivatives.

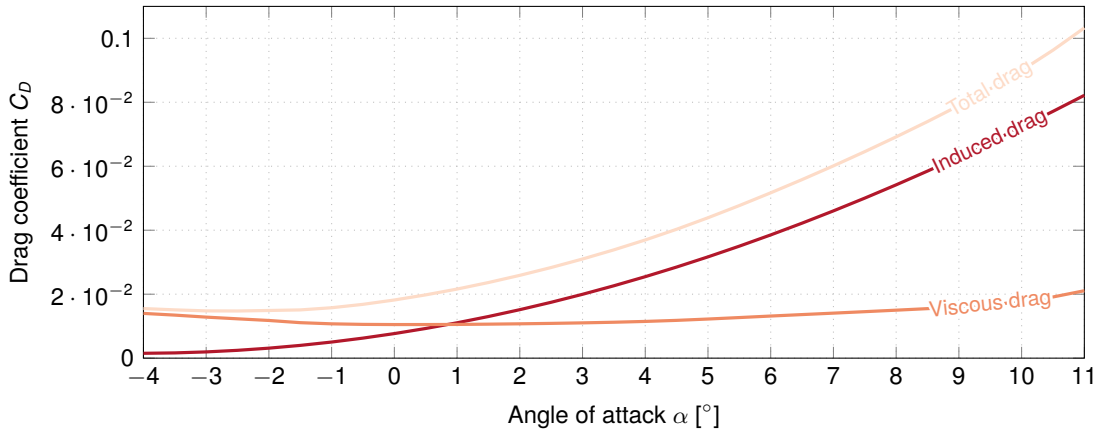
Derivative	Value
$C_{L_{\dot{\alpha}}}$	0.0000
C_{L_q}	9.0790
$C_{L_{\delta_a}}$	0.0000
$C_{L_{\delta_e}}$	0.0094
$C_{L_{\text{gear}}}$	0.0000
$C_{D_{\dot{\alpha}}}$	0.0000
C_{D_q}	0.0000
$C_{D_{\delta_a}}$	0.0000
$C_{D_{\delta_e}}$	0.0000
$C_{D_{\text{gear}}}$	0.0000
$C_{m_{\dot{\alpha}}}$	0.0000
C_{m_q}	−10.929
$C_{m_{\delta_a}}$	0.0000
$C_{m_{\delta_e}}$	−0.0232
$C_{m_{\text{gear}}}$	0.0000

(c) Lateral derivatives.

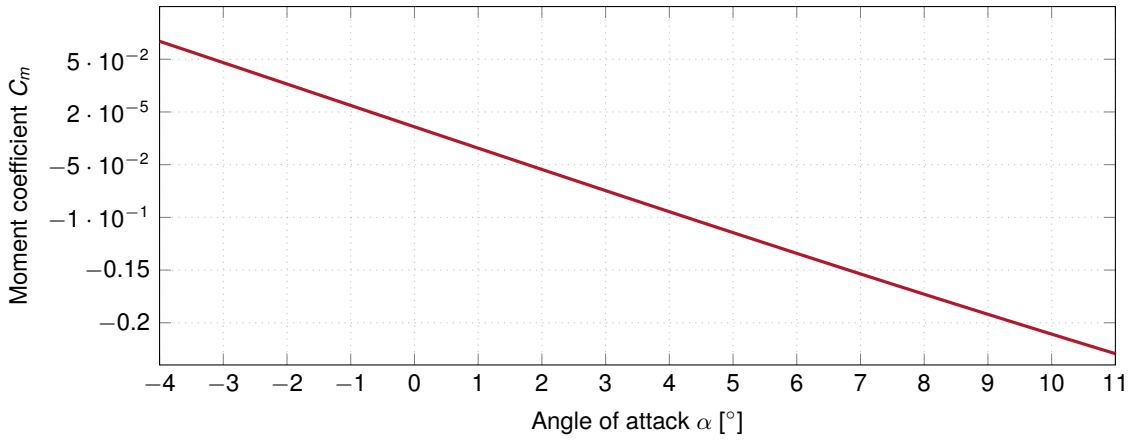
Derivative	Value
C_{Y_β}	−0.195
C_{Y_p}	0.0580
C_{Y_r}	0.1560
$C_{Y_{\delta_a}}$	0.0266
$C_{Y_{\delta_r}}$	0.1432
C_{l_β}	−0.0245
C_{l_p}	−0.4800
C_{l_r}	0.0970
$C_{l_{\delta_a}}$	−0.3667
$C_{l_{\delta_r}}$	0.0080
C_{n_β}	0.0679
C_{n_p}	−0.0244
C_{n_r}	−0.0603
$C_{n_{\delta_a}}$	0.0000
$C_{n_{\delta_r}}$	−0.0573



(a) Lift coefficient.



(b) Drag coefficient.



(c) Moment coefficient.

Figure A.1: Lift, drag, and moment coefficient to angle of attack α plots for the MTD aircraft.

From: "Hoogendoorn, Marijn" <Marijn.Hoogendoorn@nlr.nl>
Subject: **RE: Declaration of public discussion**
Date: 20 November 2019 at 16:36:02 GMT
To: Daniel de Schiffart <daniel.de.schiffart@gmail.com>
Cc: Rita Cunha <rita@isr.tecnico.ulisboa.pt>

To whom it may concern,

I supervised the graduate internship that Daniel de Schiffart completed at NLR in 2019.
I reviewed the thesis named "UAV Relay-based Auto-tuning, Tuning the PID controller for a fixed-wing UAV via the relay method" and conclude that all topics presented in the thesis can be discussed and published publicly without objections from NLR

Kind regards,

Marijn Hoogendoorn

Dedicated to innovation in aerospace

NLR - Royal Netherlands Aerospace Centre

Ir. M.L. (Marijn) Hoogendoorn
Junior R&D Engineer
Flight Test & Certification
p) +31 88 511 34 50
e) marijn.hoogendoorn@nlr.nl
i) www.nlr.org