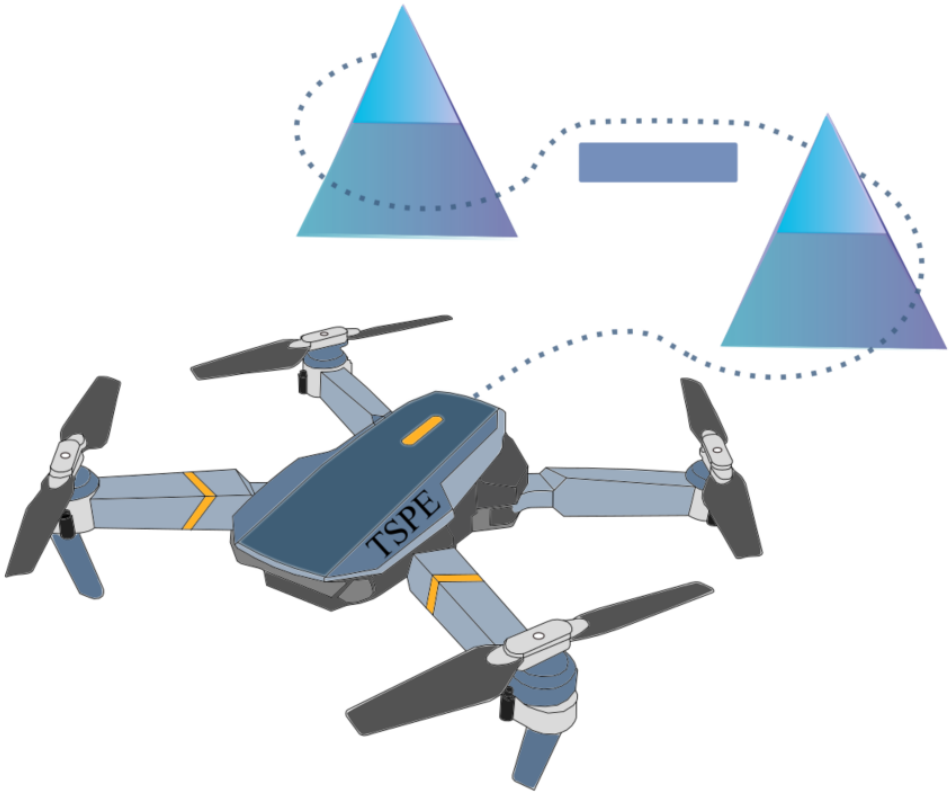
AALBORG UNIVERSITY
STUDENT REPORT

DEPARTMENT OF ENERGY
TECHNOLOGY

Quadcopter control exposed to ground, ceiling, wall and wind disturbances



P10 MASTER PROJECT
GROUP - MCE4-1030
AALBORG UNIVERSITY
DATE 15.11.2021

AALBORG UNIVERSITY
STUDENT REPORT

Master Project in M.Sc. at
Department of Energy Technology
Mechatronic Control Engineering
Pontoppidanstræde 103 -
9220 Aalborg

Title:

Quadcopter control exposed to ground, ceiling, wall and wind disturbances

Project:

Master Project

Project period:

July 2021 - November 2021

Project group:

MCE4-1030

Authors:

Tinus Stengaard Pedersen

Supervisor:

Petar Durdevic Løhndorf

Pages in total: 74
Appendix: A-C
Finalised: November 15, 2021

Abstract:

Drones are gaining interest in many fields and are used to simplify advanced tasks. For this purpose, robust control and knowledge of disturbance effects have to be raised, discussed, and analysed. This thesis investigates the impacts of noise, wind, ground, ceiling and wall effects. These disturbances are obtained from earlier studies and will be tested with three different control strategies to investigate the influence of a simple controller contrary to an advanced controller. Further, to mitigate the influences from noise, an Extended Kalman filter was created to avert the effect of noise.
This thesis highlights how important robust control is when using drones for industrial purposes. PD, SMC and ISMC controllers are used to demonstrate how a quadcopter reacts to the disturbances and how it influences the control. Different tests showed that sliding control behaves more robust against disturbances and validated linear control to be unusable in several cases. This thesis concludes that advanced control must be used when challenging environments are met, and excellent performance is of interest.

Preface

This Master thesis was written and created at Aalborg University in Denmark, at the Department of Energy Technology, specialising in Mechatronic Control Engineering. The project was created and accomplished with guidance from my supervisor Peter Durdevic Løhndorf.

Reading Guide

This thesis is written in the program Overleaf using Overleaf.com. The thesis's hyperlinks appear clickable, directing to bibliography, internal sections, equations, etc. A nomenclature is inserted at the beginning with the respective symbols used and their descriptions. Further, a table of content is presented in the beginning. All calculations and simulations are performed in Matlab, while all figures are created in the program Inkscape.

Aalborg University, November 15, 2021

Tinus Stengaard Pedersen
<Tspe16@student.aau.dk>

Summary

Droner florerer verden rundt og bliver brugt til mange professionelle opgaver, der kræver avanceret kontrol og mobilitet. I 2018 eksisterede der 10.000 kommercielle droner registreret i Europa ifølge SESARS senest publicerede rapport, og dette tal forventes at stige til 200.000 i 2025 og yderligere til 395.000 i 2035. Forøgelsen af droner skaber en god baggrund for at øge den viden som findes indenfor dette felt. Denne rapport udforsker nogle af de effekter som en drone oplever, når den er tæt på jorden, et loft, en væg, eller bliver udsat for vind. Dertil bliver der tilføjet støj til outputtene for at studere, om støj har en påvirkning.

En model af en quadcopter drone er blevet kreeret med det formål at teste hvordan den vil reagere på disse vigtige effekter. Modellen er blevet udarbejdet med viden fra tidligere studier for at efterligne en anerkendt drone, så simuleringerne kommer så tæt på virkeligheden som muligt. Med en udarbejdet model klar til test, skal kontrol opbygges for dronen. Her er der taget udgangspunkt i at forstå forskellene mellem simpel lineær kontrol og ikke lineær kontrol, og hvordan de reagere forskelligt. Til dette formål er der udarbejdet tre forskellige kontrolteorier. Alle kontrolteorier er blevet udsat for samme effekter og sammenlignet i et resultatafsnit, hvor forskellene på kontrolteorierne bliver synliggjort. De tre kontrolteorier er: PD, SMC og ISMC. Teorien bag kontrolteorierne bliver beskrevet i sektion 5 hvorefter en tuning af disse er ønsket. Her bliver en Hyperparameter Optimisation metode brugt for at opnå optimale resultater for alle kontrollerer. Denne metode går ind og udforsker forskellige værdier for alle kontrolteorierne hvorefter den finder de mest optimale værdier. Med udarbejdelsen af denne metode blev de tre kontrollere færdiggjort og klar til test.

Først blev et Extented Kalman filter brugt for at mindske støjen i system, hvilket resulterede i minimering af støjen i en sådan grad, at den ingen betydning havde. Med denne forstyrrelse udfaset, lå fokus på jord, loft, væg og vind effekterne. Disse effekter er blevet fremstillet ved hjælp af en sammenlægning af mange tidligere studiers efterforskning indenfor dette område.

Med kontrol og forstyrrelserne på plads kunne resultaterne blive lavet. Resultaterne viste hvordan sliding-mode kontrollen opnår større robusthed imod de valgte forstyrrelser, men også at lineær kontrol forbliver stabil med store fejlmarginer. Den lineære kontrol bliver kategoriseret ubrugelig i denne rapport, med udgangspunktet at kompensere for de valgte effekter, hvor i stedet SMC og ISMC bliver erklæret brugbare i forhold til jord, loft og væg effekter. Ud fra denne rapport kan det konkluderes, at ikke lineær kontrol er nødvendigt, hvis optimal performance er ønsket ved brug i vanskelige processer.

Aalborg University, November 15, 2021

Nomenclature

Symbol	Description	Unit
A	Area	$[m^2]$
A	Frictional coefficient	$[-]$
α	Tuning gain	$[-]$
c	Tuning gain	$[-]$
C_D	Dimensionless constant	$[-]$
D	Drag coefficient	$[-]$
e	Error	$[-]$
F_D	Frictional force / drag force	$[N]$
G	Gravity	$[m/s^2]$
$_i$	Contains several parameters	$[-]$
I	Current in ampere	$[A]$
I_0	No-load current	$[A]$
I	Moment of inertia	$[kg \cdot m^2]$
k	Thrust coefficient	$[-]$
k	Covariance function (Kernel)	$[-]$
K_k	Kalman gain	$[-]$
K_T	Back-emf constant	$[Vs/rad]$
K_v	Back electromotive force	$[V/m/s]$
L	Bound	$[-]$
l, L	Length	$[m]$
m, M	Mass	$[kg]$
n	Number of elements	$[-]$
N_i	Measurement noise	$[-]$
P	Power	$[W]$
ρ	Air density	$[kg/m^3]$
r	Radius	$[m]$
r	Reference	$[-]$
R_m	Motor resistance	$[\Omega]$
s, S	Sliding surface	$[-]$
τ	Torque	$[Nm]$
T	Thrust	$[N]$
T	Sampling time	$[s]$
T_f	Thrust factor	$[\%]$
ν	Control term	$[-]$
v, V	Velocity	$[m/s]$
V	Lyapunov candidate	$[-]$
vh	Induced velocity	$[m/s^{-1}]$

Symbol	Description	Unit
μ	Mean vector	$[-]$
ω	Angular velocity	$[RPM]$
ω_i	Process noise	$[RPM]$
z	Altitude command	$[m]$
ϕ	Roll command	$[^\circ]$
θ	Pitch command	$[^\circ]$
ψ	Yaw command	$[^\circ]$
σ	Standard deviation	$[-]$
σ	Variance	$[-]$

Abbreviation	Description
AWGN	Additive white Gaussian noise
BLDC	Brushless DC motor
BO	Bayesian optimisation
CASA	Civil Aviation Safety Authority
CFD	Computational fluid dynamics
DC	Direct current
EI	Expected improvement
EIP	Expected improvement plus
EKF	Extended Kalman Filter
FFA	Federal Aviation Administration
GP	Gaussian process
GPR	Gaussian process regression
ISMC	Integral Sliding mode control
LQG	Linear-Quadratic-Guassian
LQI	Linear-Quadratic-Integral
LQR	Linear-Quadratic-Regulator
MMA	Motor Mixing Algorithm
MPC	Model Predictive Control
PD	Proportional Derivative
PID	Proportional integral derivative
PMSM	Permanent Magnet Synchronous Motor
SESAR	Single European Sky ATM Research
SMC	Sliding mode control
UAV	Unmanned aerial vehicle
VTOL	Vertical take-off and landing

Subscripts	Description
B_{sf}	Sigmoid function
sgn	sign function
$C(\eta, \dot{\eta})$	Coriolis term
ξ	Denotes linear states
η	Denotes angular states
q	Denotes linear and angular states
f	Prediction function
G	Transfer function
f	Translational forces
τ	Rotational forces
J	Cost function
$\mathcal{K}$	Kinetic energy
K	Covariance function
$\mathcal{L}$	Lagrangian
$\mathcal{P}$	Potential energy
R	Rotational matrices
sgn	Sign function
u	Inputs or nonlinear controller
V_B	Linear velocity vector
ν	Rotational velocity vector
W_η	Transformation matrix
μ	Mean vector
Φ	Cumulative distribution
ϕ	Probability distribution
ϕ, θ and ψ	Roll, Pitch and Yaw

Contents

Introduction 1

Contents

The motivation and background for investigating this thesis will be explained in this chapter. Here state of the art will be created, setting a baseline for research in this field. The state of the art will lead to the objectives for this thesis and problem statement.

1.1 Overview

The flying vehicle drone or unmanned aerial vehicle(UAV) has gained interest from countless users worldwide. The earned interest comes from hobbyists, industries and ordinary citizens, who have learned to use this technical component for straightforward tasks, complicated tasks, competitions and fun.

The drone can be traced back to 1935, where the first comparable vehicle flying by a low-cost radio controller was invented. This plane was called Havilland DH82B "Queen Bee" and was used in the military to create realistic gunnery training [1]. Since then, the development of "drones" has existed primarily in the military sector until the 21st century. Now the interest has expanded to several other branches. Because of this raised interest in drones, additional scientific research is built to support the mathematics behind such vehicles to expand the knowledge further. Due to the expanded knowledge in this field, drones appear in several industries such as transport, surveillance, photographing, high demanding military operations and several other cases. The thesis will investigate how drones are affected by the disturbances called: ground effect, partial ground effect, ceiling effect, wall effect, wind gusts and noise. The general theory is used to build a model with information from earlier studies and actual drones. The model will contain a linear and nonlinear part enabling the possibility of linear and nonlinear control. The thesis will investigate disturbance methods researched in several papers to

simulate actual disturbances and visualise the effects. The focus concentrates on how a quadcopter reacts to disturbances and how different control laws steer the drone when exposed to these effects. From this study, the know-how of the disturbance rejection will be heightened.

To obtain these simulations and theories, an investigation of the types of drones already existing and their aptitude for different tasks will be presented in a state of the art section, where natural disturbance estimations will be investigated. The state of the art will lead to the objectives of this thesis and the problem statement.

1.1.1 What is a drone

The name "drone" can be traced back to old English, referencing a male honeybee whose only goal is to mate with the queen bee. This reference is from the 16th century, where it also was used to describe lazy humans. The reference can be compared to the present time where drones are used in tasks leaving the human with a joystick that does the job.

This technical phenomenon is associated with an aircraft and is a flying machine controlled through a joystick or autonomously by having implemented a pre-programmed system, followed by a fixed set of rules, or controlled through implemented self-learning algorithms. A standard electric drone is primarily made of these elements: Frame, propellers, motor, electronic speed controller, flight controller, battery, antenna, receiver and sensors as illustrated in figure 1.1.

When the drone is built with all rigid and electrical components, the next step will be to fly it. The user will need information about the drone to predict where it is and which commands to perform. This information is gathered from sensors chosen by the user, and the outputs will interact with the controller's performance. From the choosing of sensors, different control strategies can be investigated or purchased from open-source controller companies. If the user implements video cameras, the drone could be controlled in the first person with pictures as feedback. If instead a GPS module is implemented, the possibilities of navigating the drone through global coordinates are achievable.

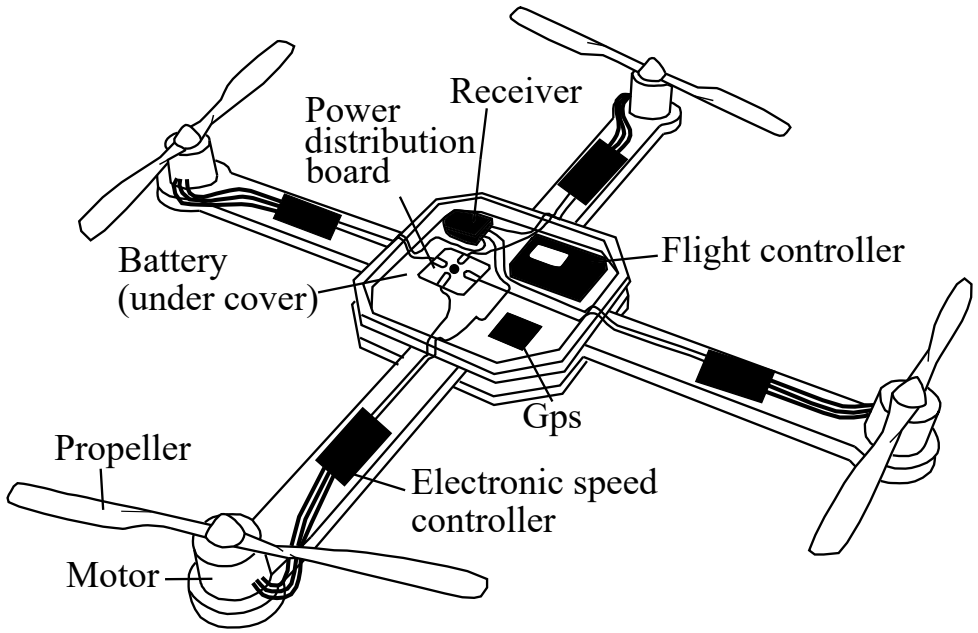


Figure 1.1: Representation of primary parts in an ordinary electric drone

1.2 Scope

In 2018 existed around 10.000 commercial drone units in Europe, according to SESAR's latest published report [2]. The number of drones is predicted to increase to 200.000 commercial drones in 2025 and further on to 395.000 in the year 2035. The rapid increase of this market provides a basis to advance the technology in this area tremendously, and to build upon this knowledge, the basic structure for drones need to be described and reviewed. As commercial drones multiply, the know-how and optimisation possibilities should be studied, where much research has already been performed. Here, the market has to be examined to know where knowledge can be expanded, and sufficient research exists. The state of the art will describe what a drone is, divide the subject into categories and further explain several ways to boost knowledge within the profession.

1.2.1 Types of drones

The industry, the branches of drones and what categorises them will be explained in this section. As a standard four major groups exist, named: *Multi Rotor Drones*, *Fixed Wing Drones*, *Single Rotor Helicopter* and *Fixed Wing Hybrid VTOL* [3]. Figure 1.2 illustrates these four types.

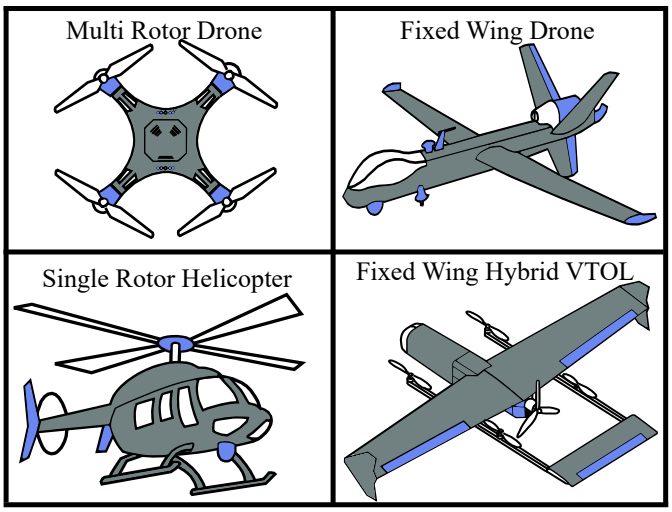


Figure 1.2: Illustration of the four major groups of drones

Multi Rotor Drone

The multi rotor drone is the most common used drone and is classified by the number of rotors. Meaning if having three rotors, the name is "Tricopter", having four, it is a "Quadcopter", eight an "Octocopter", etc. Out of these classifications, the quadcopter is the most popular choice, but it is also one of the less safe choices. If one of the four propellers fail, the quadcopter will crash. Either six or eight propellers can compensate for this. If using an octocopter, three propellers could fail, and the drone would still be capable of continuing, depending on the task and load applied. The downside of using several propellers occurs in the efficiency and flight time as a slower rotation, and larger propeller has increased efficiency compared to faster spinning and smaller ones. Using more propellers uses more electricity, and the flight time is compromised so that longer distance tasks can not be accomplished.

This drone is the cheapest choice and is the easiest to manufacture [3]. The multi rotor drone has the advantages of its accessibility, manoeuvrability and ease of use whilst vertical take-off and land(VTOL) and hover are achievable. Furthermore, it is capable of obtaining stability in hover mode, even at relative wind disturbances with simple control theory [4]. Unlike its benefits, this drone has a reduced range depending on the number of propellers and the tasks it fulfils, while it also has a low lifting capacity compared to single rotor drones.

Fixed Wing Drone

The fixed-wing drone is, as its name suggests, a drone with a set of fixed wings. This is illustrated in figure 1.2. This drone is primarily used to cover larger land areas while flying up to 10 times longer than multi-rotor drones, depending on its tasks. The structure of this type of drone makes it reject weather conditions more desirable than multi rotor drones, as wind and rain do not have the largest influence on battery drainage. Also, the structure prevents failure, as this type of drone is equipped with a natural gliding capability in the event of technical failure. This drone is by means great for mapping, surveying and firefighting over larger areas. The downside of this type of drone is that it cannot hover or VTOL, meaning it can not drop objects with high precision if it carries a payload. Also, this drone needs equipment to launch, either a catapult, runway or other technology which requires space or could damage the drone for each take-off. It also needs landing gear to avoid damages when landing and, in most cases, a runway. This drone can move forward with high speed, but it can not move backwards or stop in the air[5].

Single Rotor Helicopter

If gas-powered, the single rotor helicopter is referred to as just a "helicopter" in public and is a drone with one rotor. It has some of the same advantages as the multi rotor drone, such as VTOL, hover and gaining stability in the air, but diverges as it has this large rotor operating with an increased danger but raised efficiency. Because of the raised efficiency, the helicopter has an increased flight time and heavy payload capability, as it is built to be solid and durable. An electric type has lowered flight time, requires time to recharge, and is heavy compared to gas but is still ahead on efficiency, contrary to multi-rotor drones. The gas-powered helicopter has no charging time, increased endurance and lowered weight. Instead, the electrical is silent, has a little more power for its weight, is environmentally friendly and safer if crashing. The significant drawbacks for helicopters are their danger, making it difficult to legalise their use in many workplaces, their complexity when trying to create disturbance rejecting control, their vibrations and their price.

Fixed Wing Hybrid VTOL

This type of drone combines the multi rotor- and fixed-wing drone advantages, making it capable of VTOL and hover while also having great endurance and a long floating time. The research on this drone is deficient, as most articles are about the structure of this type and not its use. The advantages of this drone could benefit tasks of carrying a payload over more considerable distances while still dropping it with precision, but the disadvantages will then be its size and efficiency as it will decrease with its multi-functions. Implementing wings on multi-rotor drones will make the structure more receptive to disturbances as wind gusts, increasing the complexity of obtaining stability and disturbance rejecting. Also, an increased weight will lower the battery time if used as a multi-rotor for simple tasks, where the improved endurance is linked to its floating as a fixed-wing drone.

1.2.2 Summary

The multi rotor drone is a general drone used by ordinary citizens and found in many homes. If looking at the United States, the Federal Aviation Administration(FFA) had 1.782.479 drones registered in January 2021, and more than 81% of these were multi-rotor drones. Also, FFA has awarded 208.010 remote pilot certificates [6]. In Denmark exists 484 active companies working with drones, where most of them use multi rotor drones as well, meaning that a multi rotor drone is the most commonly used one [7].

Some of the projects are about visualising defects in either wind turbines, solar panels or high voltage cables. If looking at inspecting the high voltages cables, which follows a 4900 kilometres distance, the electricity companies have previously used helicopters or manually controlled drones [8]. The multi rotor drone did not include long endurance, and lots of time was lost during recharge. While the helicopters were costly and dangerous, the multi rotor was still preferred. New technology will use high voltage cables to keep the drones charged for flying the distance in one stretch. The drones investigated here are multi rotor drones as most companies prefer that drone, and because of that, this type is chosen to study further. Moreover, they are looking at making these drones intelligent so that the task can be done with no human interference[8]. As many tasks include payloads added to the drone, the focus will point towards a drone capable of implementing this skill. The theory will be linearly scaled from smaller drones theory, neglecting the forces neglected for smaller drones.

1.3 State of the art

Multirotor drones have been thoroughly studied in several fields, from assembling the drone to studying control theories and disturbance effects. This state of the art will investigate the topics analysed in this thesis giving the reader background knowledge in the fields further analysed. The thesis will look into disturbance estimation in ground effects and wind gust, and it will study control methods and their robustness to compare them.

Ground, Ceiling and Wall Effects

For quadcopters, ground, ceiling and wall effects do affect the motion when near an object. As illustrated in figure 1.3, a quadcopter reacts in different manners while close to an object visualised with the moments and forces presented along with pressure changes.

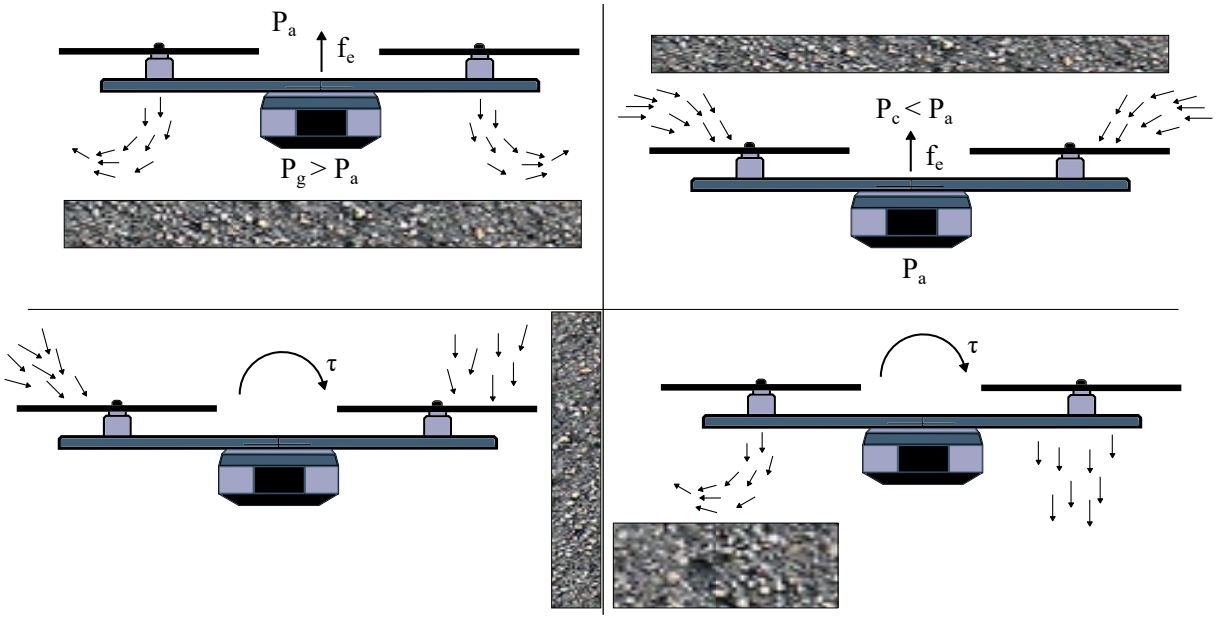


Figure 1.3: Representation of ground, partial ground, ceiling and wall effects

The two top pictures show how the drone will react if close to a floor or ceiling. While the drone gets closer to the ground, less force is needed to gain height as the ground effects counteract the gravitational

force with a factor calculated in [9], [10] and several other studies for helicopters and drones. A study [9] has investigated and proven how quadcopters will gain an increased upward force when close to the floor, as the pressure beneath the quadcopter increases while the wind cannot escape, and circulation of airflow will be created under the central body as seen in figure 1.4. This upward force is gained when close to the ceiling, resulting from lowered air pressure above the drone when approaching. As the wind is let away through the rotors, the Reynolds number increases along with the thrust coefficient as stated in [11] and [12]. Also, the ceiling effect can be factorised with a gaining thrust coefficient.

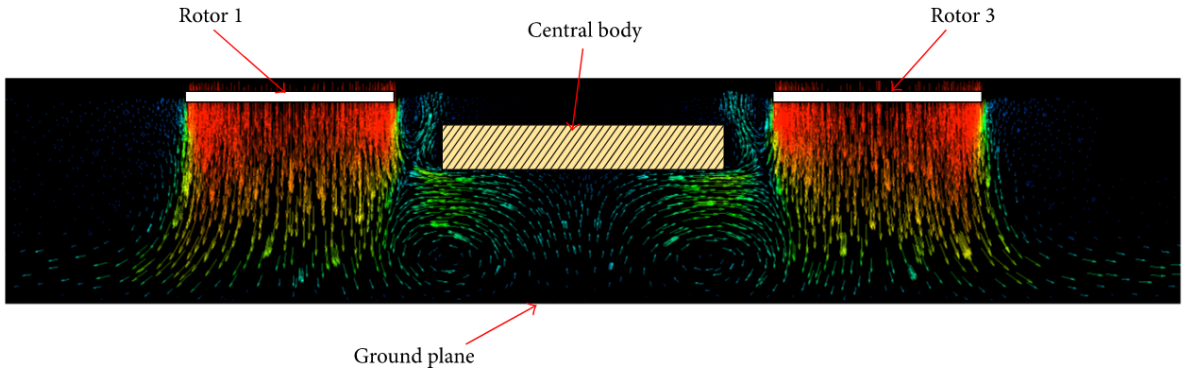
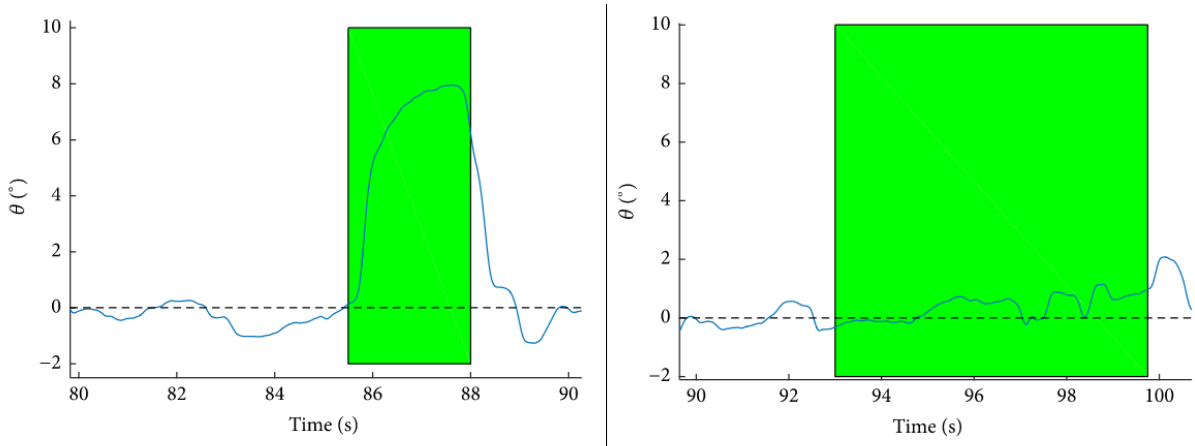


Figure 1.4: Representation of ground effects close to the floor in a CFD [9]

The two bottom pictures in figure 1.3 illustrate how the quadcopter will rotate if giving equal thrust to all rotors and affected by wall effects or partial ground effects from a structure underneath the half of the drone or one rotor. A study of wall effects [13] has tested the effects where the test frame completely immobilised the UAV. This test setup did prove that forces dragging the quadcopter towards the wall and rotating the quadcopter simultaneously had no coupling. Earlier studies have hypothesised that the quadcopter was being pulled towards the wall because it was rotated. This research [13] proves that two forces exist when a drone is close to the wall, both a force pointing into the wall and another one decreasing the thrust coefficient for the rotor/rotors closest to the wall, depending on how the drone is turned.

When a drone only has some part over an object, it will experience this partial ground effect seen in the bottom picture to the right in figure 1.3. The article [9] has studied this effect when hovering with only one rotor over an obstacle and then compared different control strategies to compensate for this type of ground effect. Their study showed that standard PID control did not cover for this effect, and further compensation is needed to counteract the ground effect happening near the ground. Their results can be seen in figure 1.5 where the green box is the time when hovering over an obstacle.



PID controller without compensation

PID controller with compensation

Figure 1.5: Ground effects with one rotor over an obstacle and with standard PID control and PID control with compensation for ground effects [9]

In this study [9], PID with ground effect compensation added was used to control the drone in situations of this effect. However, the review in [10], and [14] has studied many different control strategies and their pros and cons related to their purpose. Table 1.1 elaborates the different strategies and their suitability.

Control methods		
Control strategy	Remarks	Pros and Cons
PID	Provides an average response when using simple mathematics, which is good to get started. It lacks disturbance rejection and parameter uncertainty	Pros: simplicity, easily adjusted, good robustness, fast response. Cons: Limited performance, disturbance rejection, mass changes, unmodeled parameter handling
LQR	Has the potential for disturbance rejection and accurate path tracking but lacks in avoiding obstacles	Pros: Simplicity, disturbance rejection, adaptive and fast convergence. Cons: Robustness, precision, parameter uncertainty
LQG	Handles Gaussian noise and incomplete state information while providing stability	Pros: Handles Gaussian noise, incomplete state information, adaptive, disturbance rejection. Cons: Robustness, precision, simplicity, parameter uncertainty
Sliding mode	Provides stability, robustness and precision as it rejects disturbances, but it increases energy loss with a chattering function	Pros: Robust, adaptive, fast convergence, precision, tracking ability, disturbance rejection. Cons: Simplicity, intelligence, chattering(energy loss)
Adaptive control	Adapts to parameter changes and can self-configure in real-time to centre gravity changes	Adaptive, fast convergence, precision, self-configuration, disturbance rejection. Cons: Robustness, simplicity, parameter uncertainty
MPC	Is a predictive model controller, which can predict the following sequence of the control signals based on current states of the system	Pros: Stability, robustness, disturbance rejection, optimal. Cons: Time consuming, high computing power needed
Integrator Backstepping	Performs great in terms of robustness, fast response, stability, steady-state error and overshoot but is a recursive algorithm	Pros: Adaptive, tracking ability, disturbance rejection, robustness against parameter uncertainty. Cons Robustness, fast convergence, simplicity
Fuzzy logic	Recursive algorithm and difficult tuning approach but provides intelligent control with average results	Pros: Robust, adaptive, intelligent, precision, disturbance rejection. Cons: Parameter uncertainty

Table 1.1: Control strategies for at quadrotor studied in [10] and [14]

Table 1.1 shows a great selection of control methods capable of managing the drone while diverging concerning applications. The study in [14] discussed the advantages of control through the use of several methods gaining hybrid control as no single control law has it all. Several ways of combining these methods could be interesting to investigate towards gaining advantageous control.

1.4 Objectives

This thesis aims to further expand knowledge on quadcopters. The thesis includes a quadcopter model with ground, ceiling, wall effects and wind gusts implemented. Furthermore, different control theories enabling robust control towards these varying effects will be studied. The purpose is to obtain desirable control based on the criteria in the objective, while simulation results from all control laws will be compared to distinguish the desired control. The desired objectives will be categorised as:

- Primary goals
  - Stability at heights where ground and ceiling effects occur
  - Stability when affected by partial ground effects
  - Stability when close to a wall
  - Stability when affected by a sinusoidal wind disturbance
- Secondary goal
  - Control the drone to be able to take off and land
  - Control the drone to be robust against the disturbances
  - Settling at maximum 0.03m steady state error

1.5 Problem Statement

Can a quadcopter be controlled to withstand ground, wall and ceiling effects without colliding with either object and can it further behave stable when exposed to wind disturbances? How will different control laws affect the results, and can they achieve the objectives?

Quadcopter Template 2

Contents

This chapter outlines the design of a chosen quadcopter with additional content and flying manoeuvres. A short enumeration of the vital knowledge about the chosen quadcopter examined in this thesis is listed in this section.

2.1 Quadcopter Frame

The frame used in this thesis should replicate an AscTec Hummingbird drone, while also the specifications of this drone will be used to gain natural drone dynamics, and both disturbances and control can be related to actual drone parameters. The drone specified can be seen in figure 2.1 where the sizes are visualised, and also in the table where they are presented.

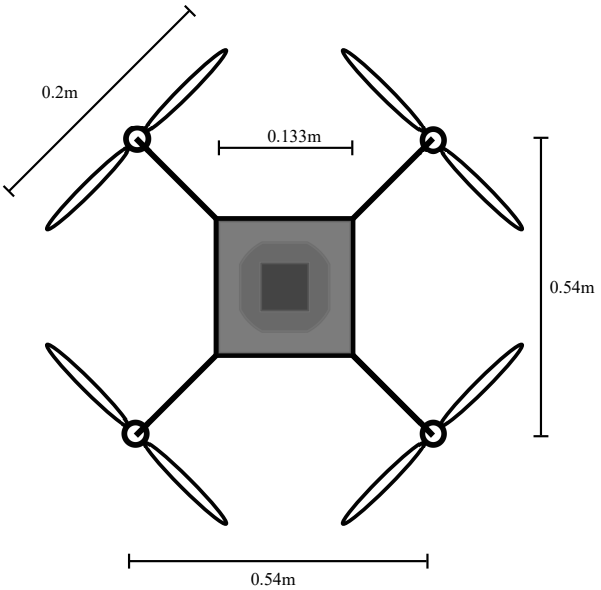


Figure 2.1: Drone illustration with length marked

Parameter	Value
Mass (M)	0.71 Kg
Arm length (l)	0.37 m
Height (h)	0.085 m

Table 2.1

2.2 Brushless DC Motor

A BLDC motor will be chosen to generate power in this thesis, making the propellers create force. This motor is more efficient than brushed motors, and no mechanical wear decreases the life expectancy. It is proportionate to a PMSM using permanent magnets to produce a rotation, where the difference is based on the control structure. PMSM used sinusoidal phase voltages, and BLDC used trapezoidal phase voltages.

This thesis will be based on using an "REB 30" motor invented by MGM Compro. This BLDC can deliver 25-40kW using 63-800 V creating a rotation between 1500-4500rpm[15]. The mathematics of the BLDC motor will be derived in this section with an equation for torque, voltages and power [16]. The torque equation can be obtained from a proportional constant for back-emf and current as expressed in equation 2.1.

$$\tau = k_T(I - I_0) \tag{2.1}$$

τ is the torque, k_T is the proportional constant, while I and I_0 is input current and no-load current. The voltage can then be calculated from the back-emf and resistor losses as seen in equation 2.2.

$$V = IR_m + K_v\omega \tag{2.2}$$

Here V is the voltage drop over the motor, ω is the angular velocity, R_m is motor resistance, and K_v will be used to determine the back-emf from angular velocity. These two equations 2.1 and 2.2 can then be used to derive the power and can be seen in equation 2.3.

$$P = IV = \frac{(\tau + k_T I_0)(k_T I_0 R_m + \tau R_m + k_T K_v \omega)}{k_T^2} \tag{2.3}$$

Estimating that resistor losses are small, this part can be neglected, and equation 2.3 can be further simplified in equation 2.4.

$$P \approx \frac{(\tau + k_T I_0)K_v\omega}{k_T} \tag{2.4}$$

Also, the current in no-load condition can be estimated small, which then also can be discarded, and the final simplified equation can be seen in equation 2.5.

$$P \approx \frac{K_v}{k_T}\tau\omega \tag{2.5}$$

HBC brushless controller (MGM Compro)

To control the motor MGM Compro provides a high-end motor controller called *HBC MEDIUM VOLTAGES series brushless DC motor controller*. This controller is fitted to the motor chosen to focus on and delivers high efficiency and peak current capability along with a complex data monitoring algorithm. It allows the user to observe all currents, in and out power and several more parameters. While this could control the motor and provide a high-end result, the focus on this part is ignored throughout this thesis.

2.3 Basic Drone Movements

The quadcopter will react like a typical drone, with x, y and z movements and roll, pitch and yaw rotations. To illustrate these rotations and movements, figure 2.2 is made, where x, y and z follow a normal coordinate system. The rotations are performed by lowering or increasing the motor's thrust and is illustrated as an increased upward arrow or a decreased arrow. The given motors is titled ω_i where $i = 1, 2, 3, 4$, these announcements set ground for the mathematics in section 3.

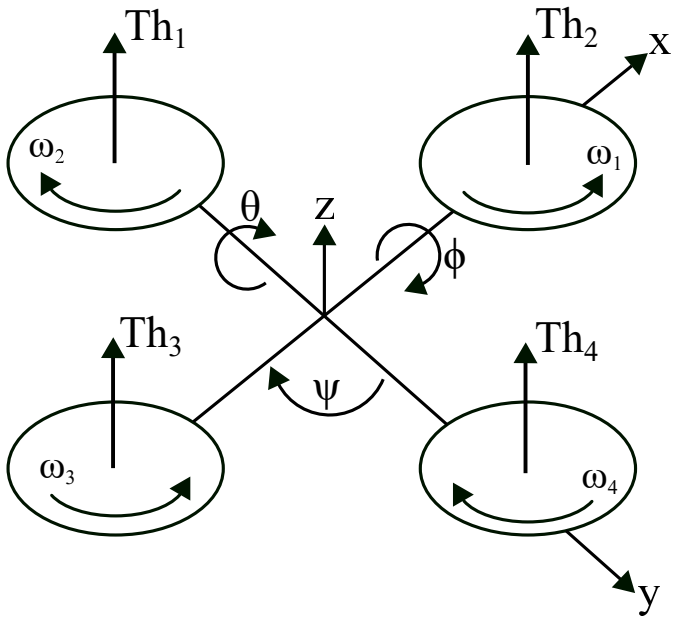


Figure 2.2: Rotations, directions and drone actions

2.4 Sensor Noise Model

While sensor noise, in general, exists in all systems, accurately defining the noise for quadcopter sensors optimises the overall performance of the control. Noise can cause drift of control, offset or instability if too aggressive control is chosen. This section calculates the noise used in the model of this thesis.

Noise

Noise is classified as non-deterministic and shows unpredictable gains in the output result of a sensor. It generally occurs as a Gaussian signal with a mean zero around the actual output. The signal will be modelled as an additive white Gaussian noise(AWGN) through this section, which can be compared with actual noise and often is used to categorise it. The output signal will be described as in equation 2.6

where $x_{measured}$ is the measured signal from sensor output, $x(t)$ is the calculated signal, and $w(t)$ is the added AWGN.

$$x_{measured} = x(t) + w(t) \tag{2.6}$$

To estimate an accurate noise signal, article [17] is used, where experiments have been created, and the variance is found. The variance for roll, pitch and yaw is found to be $0.0015[°]$, for acceleration it is $0.0130[\frac{m}{s^2}]$ and for altitude 2.5815e-05[m]. Compared to other noise models, these variances are relatively low, which relates to an implemented filter. The variation is described as the squared standard deviation σ, which is Gaussian distributed as seen in figure 2.3.

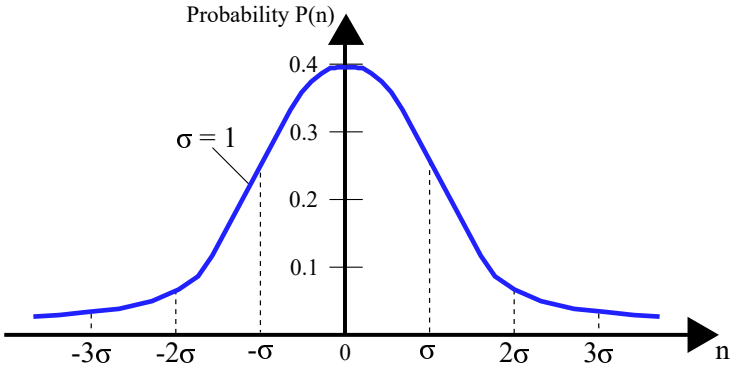


Figure 2.3: White Gaussian noise distribution with a standard deviation of 1

The variance is summarised in table 2.2

Sensor	σ
Roll, Pitch and Yaw	0.0015 [°]
Acceleration	0.013 $\frac{m}{s^2}$
Altitude	2.5815e-5

Table 2.2: Table of variance for sensor output [17]

Mathematics Behind a Quadcopter

3

Contents

The coordinate systems and mathematics behind a quadcopter is established through this chapter using the methods invented by Leonhard Euler and Joseph-Louis Lagrange called "Euler angles" and "Euler Lagrange". The methods will lead to translational and rotational accelerations, which contains the main factors for the model. Further, the mathematics will be linearised to complete with a linear and nonlinear model.

3.1 Coordinate systems

A quadcopter drone's position can be illustrated using two coordinate frames, which can be mapped directly into each other. The frames are called body frame, the frame following the quadcopter's body, and the initial frame, a stationed frame initiated in the body frame when the drone launches. These two frames are used to obtain a model of the quadcopter.

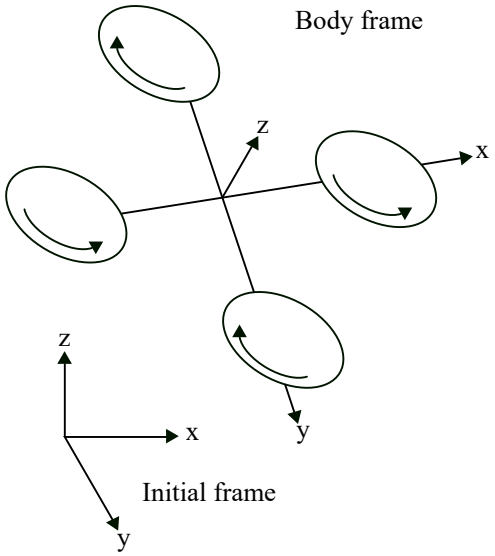


Figure 3.1: Body frame and initial frame

3.2 Inertia

The inertias for such a drone will interact with all calculations throughout all the mathematics. This momentum is increased with the size and weight of the drone and will make use of its form illustrated in figure 2.1. These increased inertias reduce the reactions of the drone while also damping increases as the size and weight rise. The formulas used in this case is based upon a square prism, and the inertia's for x, y, and z can be seen in equation 3.1 [18].

$$\begin{aligned} I_{xx} &= \frac{1}{12}m(l_y^2 + l_z^2) = \frac{1}{12}0.71(0.54^2 + 0.085^2) = 0.0177[kg \cdot m^2] \\ I_{yy} &= \frac{1}{12}m(l_z^2 + l_x^2) = \frac{1}{12}0.71(0.085^2 + 0.54^2) = 0.0177[kg \cdot m^2] \\ I_{zz} &= \frac{1}{12}m(l_x^2 + l_y^2) = \frac{1}{12}0.71(0.54^2 + 0.54^2) = 0.0345[kg \cdot m^2] \end{aligned} \tag{3.1}$$

In these equations, m is the mass of the quadcopter, and l_i is the length, wide and height of the drone. To use these inertias through this section, they will be interpreted into a matrix, where the general inertia matrix can be expressed as in equation 3.2 [18].

$$I_\xi = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}$$

While the drone design is chosen to be symmetric, it will simplify this inertia matrix, as the principal axes of inertia can be used to create a diagonalised matrix shown in equation 3.2 [18].

$$I_\xi = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{3.2}$$

This principal diagonal matrix of inertias will be used when calculating the accelerations of the rotations for the quadcopter.

3.3 Euler Angles

The angles invented by Leonhard Euler represent a rigid body in a 3D Euclidean space and denote the drone's body frame and initial frame locations and rotations. These frames will be established in matrices for linear and rotational positions and combined in a q vector shown in equation 3.3. The rotations will be titled roll(ϕ), pitch(θ) and yaw(ψ), where x, y and z are positions.

$$\xi = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \eta = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}, \quad q = \begin{bmatrix} \xi \\ \eta \end{bmatrix} \tag{3.3}$$

With the positions established, velocity vectors for both linear and rotational frames can also be shown in vectors with V_B expressing the linear velocities for the body and ν expressing the rotational velocities in a rotated vector derived in equation 3.13 and 3.14. These velocity vectors can be seen in equation 3.4.

$$V_B = \begin{bmatrix} v_{x,B} \\ v_{y,B} \\ v_{z,B} \end{bmatrix}, \quad \nu = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.4}$$

With vectors for both positions and velocities settled, a description of rotations in the reference frames described in subsection 3.1 will be presented and illustrated. To illustrate one rotation, a z-axis rotation is presented in figure 3.2, and the mathematics behind it will be shown in three simple equations. The figure shows a rotation of x and y around the z coordinate. The original and the rotated(1) one have two frames. To represent a vector concerning the rotated frame a_1, b_1 and c_1 will be used rotated onto the 1 frame. To calculate the frames, the Pythagoras theorem is used to find the x, y and z components for the vectors of a_1, b_1 and c_1. This can be seen in equation 3.5.

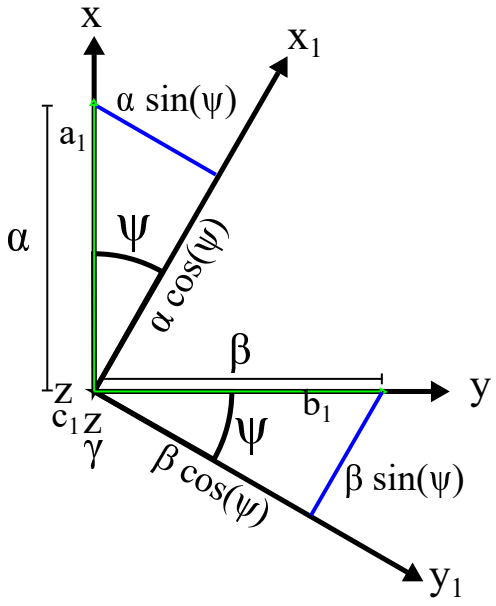


Figure 3.2: z axis rotation of a quadcopter

$$a_1 = \begin{bmatrix} \alpha cos(\psi) \\ -\alpha sin(\psi) \\ 0 \end{bmatrix}, \quad b_1 = \begin{bmatrix} \beta sin(\psi) \\ \beta cos(\psi) \\ 0 \end{bmatrix}, \quad c_1 = \begin{bmatrix} 0 \\ 0 \\ \gamma \end{bmatrix} \tag{3.5}$$

As shown in equation 3.5 the three vectors are calculated, and these three equations will then be multiplied together to obtain one rotation matrix. This can be seen in equation 3.6.

$$= \begin{bmatrix} \alpha cos(\psi) + \beta sin(\psi) \\ -\alpha sin(\psi) + \beta cos(\psi) \\ \gamma \end{bmatrix} \tag{3.6}$$

With the vectors gathered in equation 3.6 the rotation matrix is derived by extracting the length as shown in equation 3.7.

$$= \begin{bmatrix} cos(\psi) & sin(\psi) & 0 \\ -sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} \tag{3.7}$$

The matrix shown in equation 3.7 is then the rotation matrix around the z-axis with length α, β and γ extracted. This matrix will be used for z-axis rotations, and two more rotation matrices are derived in the same way for rotations around x and y. These matrices can be seen in equation 3.8 to 3.10.

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi) & s(\phi) \\ 0 & -s(\phi) & c(\phi) \end{bmatrix} \tag{3.8}$$

$$R_y(\theta) = \begin{bmatrix} c(\theta) & 0 & -s(\theta) \\ 0 & 1 & 0 \\ s(\theta) & 0 & c(\theta) \end{bmatrix} \tag{3.9}$$

$$R_z(\psi) = \begin{bmatrix} c(\psi) & s(\psi) & 0 \\ -s(\psi) & c(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

To compress the matrices, c and s will represent cosine and sine waves. With these rotation matrices stated, a relation between the initial frame and the body frame is obtained, and a complete rotation matrix for all positions can be established as $R_{xyz}(\phi, \theta, \psi)$ and is shown in 3.12.

$$R_{xyz}(\phi, \theta, \psi) = R_x(\phi) \cdot R_y(\theta) \cdot R_z(\psi) \tag{3.11}$$

$$R_{xyz}(\phi, \theta, \psi) = \begin{bmatrix} c(\theta)s(\psi) & c(\theta)s(\psi) & -s(\theta) \\ s(\phi)s(\theta)c(\psi) - c(\phi)s(\psi) & c(\phi)c(\psi) + s(\phi)s(\theta)s(\psi) & s(\phi)c(\theta) \\ s(\phi)s(\psi) + c(\phi)s(\theta)c(\psi) & c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi) & c(\phi)c(\theta) \end{bmatrix} \tag{3.12}$$

Equation 3.12 is an orthogonal rotation matrix, meaning that the transposed is equal to the inverse. The rotation matrices will also be used to derive the time derivatives of ν and η. To obtain the derivative of ν the equations in 3.13 will be used.

$$
\begin{aligned}
\nu = \begin{bmatrix} p \\ q \\ r \end{bmatrix} &= R_x(\phi)R_y(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + R_x(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \\
\nu &= \begin{bmatrix} 1 & 0 & -s(\theta) \\ 0 & c(\phi) & s(\phi)c(\theta) \\ 0 & -s(\phi) & c(\phi)c(\theta) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \\
\nu^{-1} &= \begin{bmatrix} 1 & s(\phi)t(\theta) & c(\phi)t(\theta) \\ 0 & c(\phi) & -s(\phi) \\ 0 & s(\phi)/c(\theta) & c(\phi)/c(\theta) \end{bmatrix}
\end{aligned} \tag{3.13}
$$

To express the transformation from the initial frame to the body frame, W_η is used, and opposite W_η^{-1} will transform the velocities back from the body to the initial frame. This is shown in equation 3.14.

$$
\begin{aligned}
\dot{\eta} = W_\eta^{-1}\nu, \quad \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} &= \begin{bmatrix} 1 & s(\phi)t(\theta) & c(\phi)t(\theta) \\ 0 & c(\phi) & -s(\phi) \\ 0 & s(\phi)/c(\theta) & c(\phi)/c(\theta) \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \\
\nu = W_\eta\dot{\eta}, \quad \begin{bmatrix} p \\ q \\ r \end{bmatrix} &= \begin{bmatrix} 1 & 0 & -s(\theta) \\ 0 & c(\phi) & c(\theta)s(\phi) \\ 0 & -s(\phi) & c(\theta)c(\phi) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}
\end{aligned} \tag{3.14}
$$

As s and c denoted sine and cosine waves, t will denote the tangent. The quadcopter position calculations are settled and can now be represented in both its initial and body frames. Next, the forces and torques will be expressed.

3.4 Forces

To obtain the forces, the equations stated in section 2.2 will be used, and then the power can be written to depend on thrust and air velocity as seen in equation 3.15.

$$
P = Tv_h \tag{3.15}
$$

As the thrust is the one to be found, the two terms of power and air velocity must be calculated. The air velocity can then be estimated to depend on thrust, air density ρ and rotor area A if speed is assumed to be in hover mode, and the equation for v_h can then be found from momentum theory as seen in equation 3.16.

$$
v_h = \sqrt{\frac{T}{2\rho A}} \tag{3.16}
$$

Using the equations in section 2.2 these equations can be rewritten as seen in 3.17

$$
P = \frac{K_v}{K_t}\tau\omega = \frac{K_t k_\tau}{k_T}T\omega = \frac{T^{\frac{3}{2}}}{\sqrt{2\rho A}} \tag{3.17}
$$

Thrust is to be found and can from equation 3.17 be utilised as an expression proportional to angular velocity squared shown in equation 3.18.

$$T = (\frac{K_v k_\tau \sqrt{2\rho A}}{k_T}\omega)^2 = k\omega^2 \tag{3.18}$$

As thrust depends on angular velocity, a constant k will be used to dimension its value. This constant will be based upon earlier studies comparable to the AscTec drone chosen to replicate.

With the equations derived in this section, the force is established as thrust and will determine each motor force. The thrust coefficient k is simulated and tested experimentally on the AscTec Hummingbird in [19], where an averaging over several tests resulted in a constant $k = 5.68 \cdot 10^-8[\frac{N}{\omega^2}]$. The total thrust is expressed in equation 3.19.

$$T_B = \sum_{i=1}^{4} T_i = k \begin{bmatrix} 0 & 0 & \sum \omega_i^2 \end{bmatrix}^T \tag{3.19}$$

Thrust is a positive force working in the quadcopters z-direction, while the drone also will experience a force in the opposite direction. Frictional forces acting according to the velocity will counteract the thrust in x, y and z-direction. These forces are illustrated in equation 3.20.

Frictional forces can be modelled proportional to the linear velocity in ξ and can be made as seen in equation 3.20 [20].

$$F_D = \frac{1}{m} \begin{bmatrix} A_x & 0 & 0 \\ 0 & A_y & 0 \\ 0 & 0 & A_z \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{3.20}$$

This force depends on velocity and drag according to acceleration acting opposite to the thrust. The drag force will have implemented effects aerodynamically and is stated from equation 3.21 to 3.24.

The drag forces can also be calculated through a linear equation by considering the aerodynamically effects as seen in equation 3.21 to 3.24.

$$D_x = (cos(\psi)sin(\theta)cos(\phi) + sin(\psi)sin(\phi))T - \ddot{x}m \tag{3.21}$$
$$D_y = (cos(\psi)sin(\theta)cos(\phi) - cos(\psi)sin(\phi))T - \ddot{y}m \tag{3.22}$$
$$D_z = (cos(\theta)cos(\phi))T - \ddot{z}m - gm \tag{3.23}$$
$$F_D a = \begin{bmatrix} D_x \\ D_y \\ D_z \end{bmatrix} \tag{3.24}$$

3.5 Torques

Torque is produced when the propellers generate different forces in the z-axis, as the propellers are fixed in a housing with equal size to the centre of gravity. Having a propeller in each corner enables the drone to manoeuvre in all directions. To calculate the torques, the derivation is based upon the drag equation from fluid dynamics shown in equation 3.25.

$$F_D = \frac{1}{2}\rho C_D A v^2 \tag{3.25}$$

In equation 3.25 the air density is ρ, again the rotor area is A, v is velocity and C_D is a dimensionless constant. When knowing the drag equation, the torque can be found in equation 3.26, where r is the radius of each rotor and b a constant used to express the torque from the angular velocity.

$$\tau_D = \frac{1}{2} r \rho C_D A v^2 = \frac{1}{2} R \rho C_D A (\omega R)^2 = d\omega^2 \tag{3.26}$$

While the torques are derived only from the drag, inertias are added to the torque equation to realise its mode of action. The inertias will be added and connected to the acceleration shown in equation 3.27.

$$\tau_z = d\omega^2 + I_M \dot{\omega} \tag{3.27}$$

As the quadcopter can be estimated to fly mainly with a constant velocity or hover with zero acceleration, an assumption of $\omega \approx 0$ is guessed and thereby, the inertias can be neglected again. While the propellers are spinning counterclockwise in pairs, the yaw equation will be set to using equation 3.28 with the term $(-1)^{i+1}$ gathering the right sign. With that settled the equation for yaw is shown in equation 3.29, where the constant d is gathered from [21] and equals $1.5 \cdot 10^{-9}[\frac{N}{\omega^2}]$.

$$\tau_z = (-1)^{i+1} d\omega_i^2 \tag{3.28}$$

$$\tau_\psi = d(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2) \tag{3.29}$$

With the torque utilised for yaw, it needs to be gathered for roll and pitch with the positive direction shown in section 2.3. The torques for roll and pitch is calculated in equation 3.30 from standard mechanics, where L is the distance between each propeller and the centre of the quadcopter and k is again the thrust coefficient stated above in equation 3.19.

$$\tau_{\theta,\phi} = \sum RxT = Lk((\omega_2^2 - \omega_4^2)) \tag{3.30}$$

With the torque equations stated, the three equations can be summed up in equation 3.31 to 3.33 and can also be mapped, with equation 3.14, from body to initial frame in equation 3.34.

$$\tau_\phi = Lk((\omega_2^2 - \omega_4^2)) \tag{3.31}$$
$$\tau_\theta = Lk((\omega_3^2 - \omega_1^2)) \tag{3.32}$$
$$\tau_\psi = d(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2) \tag{3.33}$$

$$\tau_B = \begin{bmatrix} \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^T = W_\eta^T \tau_B \tag{3.34}$$

3.6 Euler-Lagrange

With the torques and forces obtained, a dynamical model of the quadcopter can be derived using Euler Lagrange. This method will point out the kinetic and potential energies in a Lagrangian and use it to obtain the double derivatives of the ξ and η. The Lagrangian is shown in equation 3.35.

$$\mathcal{L}(q,\dot{q}) = \mathcal{K}_{trans} + \mathcal{K}_{rot} - \mathcal{P} \tag{3.35}$$

$$= \frac{1}{2}m\dot{\xi}^T\dot{\xi} + \frac{1}{2}\nu^T I\nu - mgz \tag{3.36}$$

From the Lagrangian, the rotational and translational forces can be obtained as seen in equation 3.37.

$$\begin{bmatrix} f \\ \tau \end{bmatrix} = \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}} - \frac{\partial \mathcal{L}}{\partial q} \tag{3.37}$$

q represents the initial frame, and these terms will be treated separately, starting with the translational motion.

3.6.1 Translational Motion

The translational term is described as in equation 3.38.

$$\mathcal{K}_{trans} = \frac{1}{2}m\dot{\xi}^T\dot{\xi} - mg\xi z \tag{3.38}$$

To find its Lagrangian from equation 3.37, equation 3.39 is derived and the expression for f utilised.

$$\begin{aligned} f &= \frac{d}{dt}\frac{\partial \mathcal{K}_{trans}}{\partial \dot{\xi}} - \frac{\partial \mathcal{K}_{trans}}{\partial \xi} \\ &= \frac{d}{dt}\frac{\partial}{\partial \dot{\xi}}(\frac{1}{2}m\dot{\xi}^T\dot{\xi} - mg\xi z) - \frac{\partial}{\partial \xi}(\frac{1}{2}m\dot{\xi}^T\dot{\xi} - mg\xi z) \end{aligned} \tag{3.39}$$

When looking into equation 3.39 there will be partial derivative terms, where it can be seen that taking the partial derivative of $\dot{\xi}$ in the first part will eliminate the minus term, and again in the second term, the derivative of ξ will omit the first part, which will give equation 3.40.

$$f = \frac{d}{dt}(m\dot{\xi}) + mgz = m\ddot{\xi} + mgz \tag{3.40}$$

With the equation shown in 3.40, acceleration can be isolated, and the friction stated in section 3.4 can be added to fulfil the linear accelerations of the drone as shown in equation 3.41.

$$m\ddot{\xi} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R_{xyz}T_B + F_D \tag{3.41}$$

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = -g\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \frac{T}{m}\begin{bmatrix} -sin(\theta) \\ cos(\theta)sin(\phi) \\ cos(\theta)cos(\phi) \end{bmatrix} - \frac{1}{m}\begin{bmatrix} A_x & 0 & 0 \\ 0 & A_y & 0 \\ 0 & 0 & A_z \end{bmatrix}\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{3.42}$$

With the accelerations for translational motion stated, it can be seen that friction only affects the quadcopter while in motion, and when starting to gain height, only the gravitational forces must be overcome.

3.6.2 Rotational Motion

This rotational part is the torques generated to rotate the quadcopter and is also gathered from the Lagrangian implemented in equation 3.37. The rotational energy is shown in equation 3.43 where it is converted into the initial frame by using equation 3.14.

$$\mathcal{K}_{rot} = \frac{1}{2}\nu^T I_\xi \nu \tag{3.43}$$

$$= \frac{1}{2} W_\eta \dot{\eta}^T I_\xi W_\eta \dot{\eta} \tag{3.44}$$

$$= \frac{1}{2}\dot{\eta}^T \mathcal{J} \dot{\eta} \tag{3.45}$$

To reduce equation 3.43 $\mathcal{J}$ is used to represent $W_\eta^T I_\xi W_\eta$ and is shown in equation 3.46.

$$\begin{bmatrix} I_{xx} & 0 & -I_{xx}s(\theta) \\ 0 & I_{yy}c_(\phi)^2 + I_{zz}s(\phi)^2 & (I_{yy} - I_{zz})c(\phi)s(\phi)c(\theta) \\ -I_{xx}s(\theta) & (I_{yy} - I_{zz})c(\phi)s(\phi)c(\theta) & I_{xx}s(\theta)^2 + I_{yy}s(\phi)^2c(\theta)^2 + I_{zz}c(\phi)^2c(\theta)^2 \end{bmatrix} \tag{3.46}$$

With the rotational energy stated, equation 3.37 can be used to utilise τ as seen in equation 3.47.

$$\tau = \frac{d}{dt}\frac{\partial \mathcal{K}_{rot}}{\partial \dot{\eta}} - \frac{\partial \mathcal{K}_{rot}}{\partial \eta} \tag{3.47}$$

Then deriving this term is done through equation 3.48 to 3.51, where a Coriolis term is obtained to satisfy gyroscopic and centrifugal effects in the rotation of the quadcopter.

$$\frac{\partial \mathcal{K}_{rot}}{\partial \dot{\eta}} = \frac{1}{2}(\mathcal{J} + \mathcal{J}^T)\dot{\eta} = \mathcal{J}\dot{\eta} \tag{3.48}$$

$$\tau = \mathcal{J}\ddot{\eta} + (\dot{\mathcal{J}} - \frac{1}{2}\frac{\partial \mathcal{K}_{rot}}{\partial \eta}(\dot{\eta}^T \mathcal{J}))\dot{\eta} \tag{3.49}$$

$$C(\eta, \dot{\eta}) = \dot{\mathcal{J}} - \frac{1}{2}\frac{\partial}{\partial \eta}(\dot{\eta}^T \mathcal{J}) \tag{3.50}$$

$$\tau = \mathcal{J}\ddot{\eta} + C(\eta, \dot{\eta})\dot{\eta} \tag{3.51}$$

The Coriolis term is shown as $C(\eta, \dot{\eta})$ and is further expanded in appendix A. With a derivation of τ shown in equation 3.48 to 3.51, the rotational accelerations of η can be isolated as in equation 3.52.

$$\ddot{\eta} = \mathcal{J}^{-1}(\tau - C(\eta, \dot{\eta})\dot{\eta}) \tag{3.52}$$

3.6.3 Input

The input for the quadcopter will be gathered in a motor mixing algorithm(MMA) which will relate the inputs to the thrust and torques. The inputs are described as u_i and will be the giving currents assumed to equal the rotational speed ω. The MMA is derived using the equations in subsection 3.4 and 3.5 with the result shown in equation 3.53.

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{4k} & 0 & \frac{1}{4Lk} & -\frac{1}{4d} \\ \frac{1}{4k} & \frac{1}{4Lk} & 0 & \frac{1}{4d} \\ \frac{1}{4k} & 0 & -\frac{1}{4Lk} & -\frac{1}{4d} \\ \frac{1}{4k} & -\frac{1}{4Lk} & 0 & \frac{1}{4d} \end{bmatrix} \begin{bmatrix} T \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \tag{3.53}$$

The inputs will be used to operate the drone and create desired trajectories to follow. Each input is related to either rotating the drone or gaining constant height, which means that all four interactions can be performed simultaneously.

With a nonlinear model for an AscTec Hummingbird replication, the next part will generate a linearized model so that both a nonlinear and linear model can be used to control the drone.

3.7 Linearised Model

As there exists a nonlinear model which should imitate the actual performance of a quadcopter minoring an Asctec Hummingbird, it will be used to investigate the performance of the control laws investigated in section 5. The nonlinear model will always be used to evaluate controllers while establishing nonlinear and linear controllers, a linear model of the system has to be created. This section will complete a linear model which will be used to analyse linear control strategies. To create linear controllers, a linear state-space model will be created as $\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu}$ with a set of chosen linearisation points. The states can be seen in equation 3.54 and the inputs is shown in equation 3.53 and again in 3.55.

$$\mathbf{x} = \begin{bmatrix} x & y & z & \phi & \theta & \psi & \dot{x} & \dot{y} & \dot{z} & \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T \tag{3.54}$$

$$\mathbf{u} = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 \end{bmatrix}^T \tag{3.55}$$

To linearise the system, the first order Jacobian will be used, which is stated in equation 3.56

$$\mathbf{f}(\mathbf{x}) \approx \mathbf{f}(\mathbf{x}^*) + \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}-\mathbf{x}^*}(\mathbf{x} - \mathbf{x}^*) + \mathbf{f}(\mathbf{x})_{\mathbf{hot}} \tag{3.56}$$

The system is determined to be linearised in the state where the quadcopter is hovering to simplify the translational dynamics in the system, where the last part in equation 3.42 can be omitted. This selection makes all states zero except the inputs, which will be equal to the inputs needed to overcome gravity and hover as seen in equation 3.57. The equation is calculated independently of the ground effects, which may interfere with hovering at low heights.

$$\frac{\omega_i^2 \cdot 4 \cdot b}{m} = g \tag{3.57}$$

$$u_i = \omega_i = 5338.4241 \tag{3.58}$$

More accuracy could be obtained using all decimals to get a more exact linearisation, but four decimals are assumed usable to work further within this case. The linearisation points can then be described as in table 3.1, and further the state-space model is described in 3.59 and 3.60.

Variable	ϕ	θ	ψ	$\dot{\phi}$	$\dot{\theta}$	$\dot{\psi}$	$\ddot{\phi}$	$\ddot{\theta}$	$\ddot{\psi}$	$\ddot{x}$	$\ddot{y}$	$\ddot{z}$	u_1	u_2	u_3	u_4
Value	0	0	0	0	0	0	0	0	0	0	0	0	5338	5338	5338	5338

Table 3.1: Table of the variables and chosen linearisation points

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 9.464e-11 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -9.464e-11 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.59}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0.0009188 & 0.0009188 & 0.0009188 & 0.0009188 \\ 0 & 0.006266 & 0 & -0.006266 \\ -0.006266 & 0 & 0.006266 & 0 \\ -0.0004642 & 0.0004642 & -0.0004642 & 0.0004642 \end{bmatrix} \tag{3.60}$$

With $\mathbf{A}$ and $\mathbf{B}$ derived in the state-space model, the linear system can be described in equation 3.61 with the output.

$$\begin{aligned} \dot{x} &= \mathbf{Ax} + \mathbf{Bu} \\ y &= \mathbf{Cx} + \mathbf{Du} \end{aligned} \tag{3.61}$$

3.8 Summary

A linear and nonlinear model is created in this chapter to control a quadcopter affected by disturbances. The two models imitate each other when in hover condition, but they deviate when other states are examined. The linear model will be used to form a linear control strategy, where the nonlinear model will be used to validate linear and nonlinear controllers and create nonlinear controllers.

With the model for an AscTec Hummingbird replication, the next part will generate disturbances towards testing different control strategies.

Disturbances and Noise 4

Contents

This chapter will elaborate and calculate the disturbance to be used in this thesis. With knowledge combined from other studies, ground effects and wind estimation will be mathematically obtained and added to the model.

4.1 Ground, Ceiling and Wall Effects

This phenomenon happens when the drone is approaching obstacles that could be occurring from all angles as described in section 1.3. When the quadcopter is affected by these effects, it either gains an upward force for all motors helping the quadcopter rise or deviates the force of one or more rotors, making the drone rotate. The ground effects will appear as either translational disturbances or rotational disturbances, described separately throughout this chapter. Several studies describe these effects with mathematics, depending on the specific event, and these calculations will be used in this thesis to test different control strategies and compare them based on the objectives in section 1.4.

4.1.1 Translational Effects

The translational effects affecting the quadcopter is called the ground effect and the ceiling effect. These disturbances equally interfere with each motor, creating only vertical forces.

Ground and Ceiling Effect

The translational ground effects will be based upon the research done for small-scaled quadcopters in [22], where calculations for ground effect in hover mode and through forwarding flight is obtained for both high speed, low speed and no speed. The obtained ground effects will be assumed equal to the

ceiling effects, as Conyers have assumed in [23], to simplify the model, while validation of this is proven. The research [22] is built upon several earlier studies gathering mathematical and experimental evidence to perform valid ground effect estimation. The most common way to describe ground effects is obtained from Cheeseman and Bennett [24], who in 1955 created a relationship between the height above ground, rotor diameter and speed. This effect is obtained for a helicopter but is proven to work on quadcopters as well, and the formula can be seen in equation 4.1.

$$\frac{F_{IGE}}{F_{OGE}} = \frac{1}{1-(\frac{R}{4z})^2} \quad if \quad \frac{z}{R} > 0.25 \tag{4.1}$$

Article [22] have tried to fit lots of experimental data from other studies with comparison to own tests and obtained a model able to estimate ground effect concerning height, velocity, induced velocity, air density and rotor diameter to specialise the effects further. The model is fitted such that ground effects are neglected when the height exceeds z>4R, depending on the study. Cheesman and Bennet [24] only allowed the force to perform up to z>1.5R where newer studies have shown that it is still measurable at z>4R [25]. Two models are proposed in [22], and the second model will be used in this thesis to estimate the ground effects. The equation for ground effects is seen in equation 4.2 where a second-order term is used as a coefficient of velocity for the drone. V is the drones velocity where vh is the induced velocity calculated $vh = \sqrt{\frac{T_h}{2\rho\pi R^2}}$ wherein ρ is the air density. Equation 4.3 shows the effect when velocity equals 0, and the quadcopter is in hover mode.

$$\frac{T}{T_h} = (0.104\frac{R}{z} - 0.0952)(\frac{V}{vh})^2 - 0.171\frac{R}{z} + 1.02 \tag{4.2}$$

$$\frac{T}{T_h} = -0.171\frac{R}{z} + 1.02 \tag{4.3}$$

In the equation, R is the radius of the rotor, and z is the height above ground. The equation does not take uneven ground surfaces into account, which also applies to the ceiling effects. Equation 4.2 will be used in the model for estimating ground and ceiling effects. The equation for accelerations in x, y and z-direction will be as seen in equation 4.4.

$$m\ddot{\xi} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R_{xyz}T_B\frac{T}{T_h} + F_D \tag{4.4}$$

4.1.2 Rotational Effects

Rotational disturbances are called partial ground effects and wall effects in this thesis and will both lead to rotating the quadcopter because of not equally disturbed motors.

Partial Ground effects

The partial ground effects shown in figure 1.3 represents ground effects affecting only some of the rotors creating a rotation in the quadcopter if not counteracted by control. This term will be utilised as equation 4.2 but only attached to some of the rotors, where it in section 4.1.1 is used equally on the thrust. As this is partially affecting the thrust, the quadcopter would begin rotating depending on the motors disturbance. This disturbance will be simulated, disturbing one motor and two motors in two different experiments. These experiments will be used to understand effects and learn how to counter control them.

Wall Effects

Wall effects influence both dragging the quadcopter towards the wall and also decreasing the thrust of the rotors closest to the wall as described in section 1.3. This means that two independent forces occur when approaching a wall, which in this section will be simplified by using the partial ground effects as negative disturbances to the motors closest to the wall and one constant force dragging the drone towards the wall. The effect of this will be tested as a horizontal force applied to the drone, with different types of strength and at the same time testing for adverse partial ground effects on the quadcopter.

4.1.3 Summary

As the equations can perform in the range of z>0.05m and z<0.85m, "if statements" will be used to equalise the thrust factor to one when the altitude is higher than one meter. When the altitude is beneath 0.05 meters, the factor is assumed to perform as if the altitude is 0.05 meters, the most significant thrust factor. An example of the if statement given for ground effects can be seen in equation 4.5, where the factor now is called T_f.

$$\begin{aligned}
&\textbf{if} \quad z < 0.05 \\
&\quad T_f = 1.3298 \\
&\textbf{elseif} \quad z < 0.85 \\
&\quad T_f = (0.104\frac{R}{z} - 0.0952)(\frac{\sqrt{dx^2 + dy^2}}{vh})^2 - 0.171\frac{R}{z} + 1.02 \\
&\textbf{else} \quad z > 0.8 \\
&\quad T_f = 1
\end{aligned} \tag{4.5}$$

This implementation will be used on both the translational and rotational effects, and an example of this can be seen in figure 4.1 where the ground effect is tested on the mathematical model with simple PD controllers added.

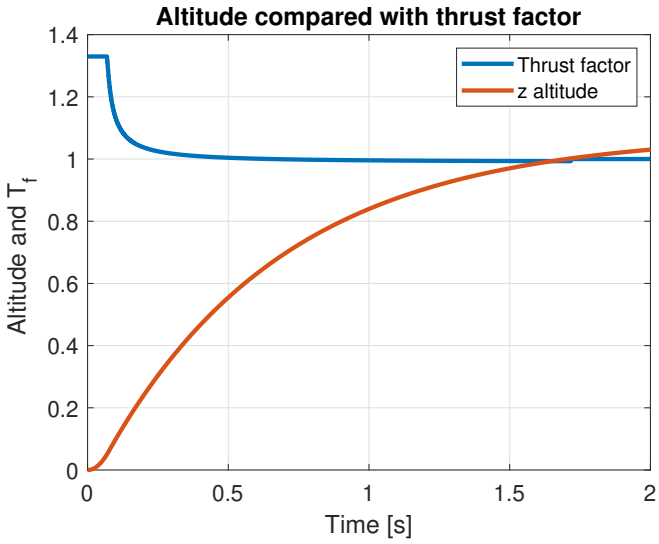


Figure 4.1: Visualising the effect of ground effects

4.2 Noise Model

Noise occurs in all machinery related to the motor, turbulent flow, vibrating panels etc. Because of noise, control has to be designed carefully so that the closed-loop system will not become unstable. The controller must be robust towards both the noise in a system and the previously described disturbances. In section 2.4 an assumed realistic sensor noise is created with different variances for measuring altitude, angles and acceleration. The model contains Gaussian distributed white noise and will be implemented onto the output sensor signals. To visualise the effects of noise implemented onto the output signals, again the same PD controllers will be used as in section 4.1.3, and the results can be seen in figure 4.2.

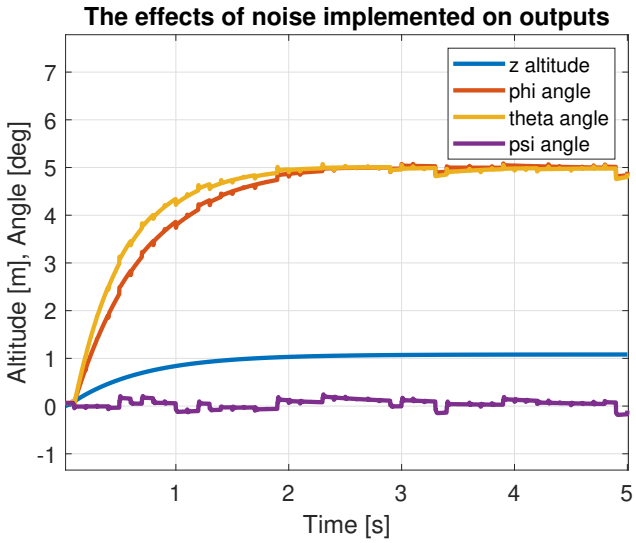


Figure 4.2: Visualising the effects of noise on outputs

As seen in figure 4.2, noise interferes with the output signal creating an oscillating signal. When this signal settles, it oscillates around the actual value, which in the end may cause the controller to react to these errors continually. This will drain the battery and cause limited flight time, which of course, has to be avoided. One of the options could be to estimate the actual signal and thereby straighten out the oscillations gaining a more settled output. Another option could be to decrease the aggression of the controllers, which should lead to not reacting to minor errors. Also, both options could be used simultaneously to gain decreased current consumption.

4.3 Wind Effects

Wind effects can have a significant impact while controlling a drone. Because of the drone's low weight, a moderate breeze can send the drone off course or collide with obstacles. To gain robustness towards minimising these effects, a simple wind disturbance is used in this thesis to test how it can be counteracted by control. This disturbance will be implemented as a sinusoidal force connected with some white noise to create a chattering nonlinear disturbance, disturbing the roll and pitch moment of the quadcopter. With the wind disturbance implemented into the model, the sensor output signal can be seen in figure 4.3, where both the noise and wind effects are implemented.

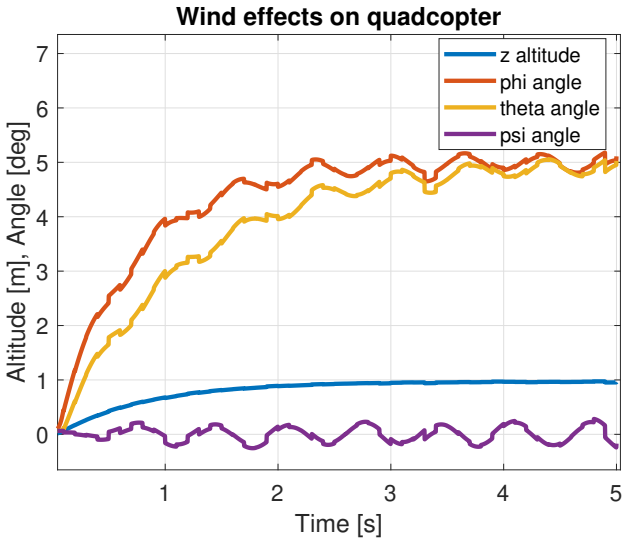


Figure 4.3: Visualising the effect of wind on outputs

As expected, the drone would be moved out of position, which gains the interest of stabilising it in this condition, so that it might fly more steady and collisions with objects can be avoided. Also, many assignments cannot be performed in this condition, such as photographing or dropping cargo.

Control of Quadcopter 5

Contents

This chapter elaborates two different control strategies to counteract the disturbances described in section 4 and the sensor noise in section 2.4. The controllers will be created theoretically in this section.

5.1 Control Strategies

In section 1.3 different control strategies is listed in table 1.1 stating remarks and pros and cons. The first mentioned strategy, *PID*, is a simple linear controller gaining average responses whilst being easily adjusted. This type of control law is called a linear control law and will be used in this thesis as the simplest control strategy to compare the drone's handling against more robust, disturbance rejecting and adaptive control strategies. Other strategies to compare the linear controller will be a Sliding mode control strategy to gather disturbance rejection and precision of the disturbances in section 4. To further realise the control, quadcopter limitations based upon the chosen AscTec Hummingbird will be presented and used throughout this chapter. The limitations will be based upon [26] where tests are performed to find these limitations and [27] where the drone can be purchased.

Limitations	Min	Max
Motor speed	1.100 [rpm]	8.600 [rpm]
Roll and Pitch angle	-85 [°]	85 [°]
Speed	0 [m/s]	15 [m/s]

Table 5.1: Limitations for quadcopter

5.2 Linear Control

The linear strategy will be built upon linear analysis to understand its limitations and achievabilities. In section 3.7 a linear model of the system was created, and it will be used to obtain these controllers. The state-space model created in equation 3.61 will be used to gain transfer functions G_i, to analyse the system. To calculate the transfer functions of the system, equation 5.1 is used to derive the functions from the state-space model. The C matrix determines how many transfer functions are calculated, where four is chosen to control the altitude, roll, pitch and yaw. Two more will be added when control of the x and y position is needed.

$$G = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} \tag{5.1}$$

With a C matrix chosen, sixteen transfer functions will be calculated, but because the x and y-axis are chosen as they are, the torques of roll and pitch will only contain two motor inputs seen in equations 3.31 and 3.32, which means that some of the transfer functions are zero. The system of the transfer function can be seen in equation 5.2.

$$\begin{bmatrix} z \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} G(1,1) & G(1,2) & G(1,3) & G(1,4) \\ G(2,1) & 0 & 0 & G(3,4) \\ 0 & G(3,2) & G(3,3) & 0 \\ G(4,1) & G(4,2) & G(4,3) & G(4,4) \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \tag{5.2}$$

The transfer functions coherent with z, ϕ, θ, and ψ are equal for each command but may occur in negative and positive signs. The functions can be seen in equation 5.3 to 5.6 with their associated closed-loop transfer function expressed only in positive sign. The functions will now be expressed in continuous time functions but will be analysed in both continuous and discrete time.

$$G_z(1,i) = \frac{0.0009188}{s^2} \quad , \quad G_{zCL}(1,i) = \frac{0.0009188}{s^2 + 0.0009188} \tag{5.3}$$

$$G_\phi(2,i) = \frac{0.006266}{s^2} \quad , \quad G_{\phi CL}(2,i) = \frac{0.006266}{s^2 + 0.006266} \tag{5.4}$$

$$G_\theta(3,i) = \frac{0.006266}{s^2} \quad , \quad G_{\theta CL}(3,i) = \frac{0.006266}{s^2 + 0.006266} \tag{5.5}$$

$$G_\psi(4,i) = \frac{0.0004642}{s^2} \quad , \quad G_{\psi CL}(4,i) = \frac{0.0004642}{s^2 + 0.0004642} \tag{5.6}$$

These functions are defined in continuous time Laplace domain but will be executed in discrete time, and thereby the functions have to be discretised. To perform a discretisation, Backward Euler's method will be used, saying that $s = \frac{z-1}{Tz}$, where T is the sampling time. The sampling time contributes to the reaction of control, which can create instability if too low and may drain the battery and computer power if too high. A general sampling time for equipment in quadrotors lies between 50hz and 500hz. An article has looked into differences between sampling rates [28] where they have tried 70Hz, and 100Hz for an MPC controller and 400Hz for a PID control depending on their equipment. 70Hz seemed too low where 100Hz could be used, and 400Hz is undoubtedly qualified. 250Hz is chosen in this thesis as this is assumed to be qualified to perform reasonable control, meaning that T=0.0040s.

5.2.1 Linear Analysis

With transfer functions obtained for the linear plants, controllability will be analysed, starting with the root locus. As all transfer functions have the same structure, one root locus will create a controller. As ϕ and θ equals, the analysis will be based on these transfer functions, and their closed-loop s- and z-plane root locus can be seen in figure 5.1.

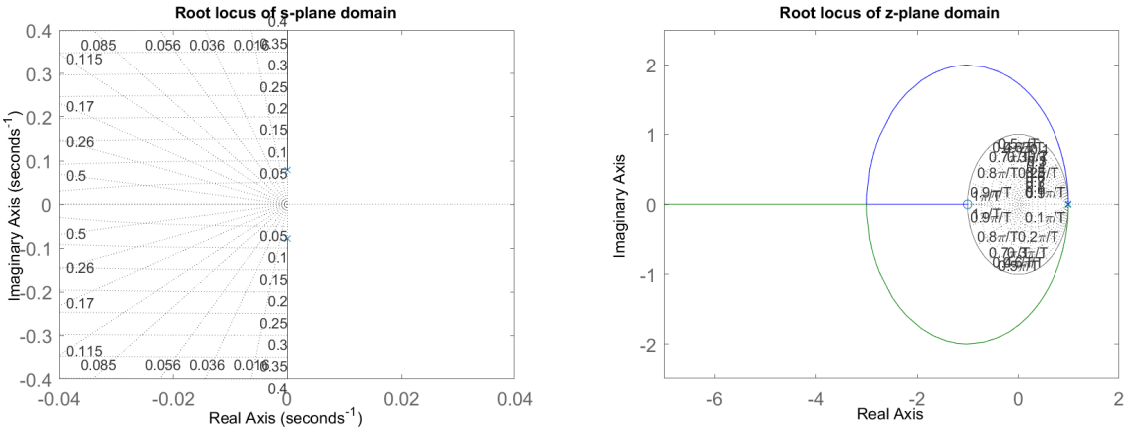


Figure 5.1: Root locus of roll and pitch transfer functions in s- and z-domain

As seen from these figures in 5.1, the poles in the s-domain can be assumed to stay on the y axis, meaning that the system is marginally stable and oscillating, which can be increased or decreased. While observing the z-domain plot, the roots of the closed-loop are located at $1 + 0.0003i$ and $1 - 0.0003i$, which is assumed to lie on the unit circle, meaning it is also marginal stable which it was supposed to be. To create stability in this system, it is essential to create a system with poles in the left half-plane for the continuous-time domain and make the poles in discrete-time behave inside the unit circle. If poles are outside the unit circle or in the right half-plane for the continuous format, the system becomes unstable.

To gain stability and reach the desired goals, different linear controllers can be used. Some of these could be PI, PD, PID, LQR or LQI, which have different advantages. Looking into the transfer functions from 5.3 to 5.6 it is observed that an integrator occurs in the functions, and it is thereby assumed that steady-state errors do not arise in the system, which eliminates the advantages of PI, PID, LQI control. Neglecting the integral term in these controllers also degrades the controller's order, and lower accelerations from actuators are needed, meaning power would be saved. Looking through several papers mentioned earlier, a general PD controller has been used to control drones gaining a simple system control. The proportional-derivative (PD) controller ads a single zero into the system at location $\frac{-K_P}{K_D}$ in continuous time. The PD controller structure can be seen in equation 5.7.

$$G_{PD} = K_P + K_D s = K_D(s + \frac{K_P}{K_D}) \tag{5.7}$$

As this controller structure implements a zero, the purpose is to implement it where stability can be obtained. Implementing it on the left half-plane in the s-domain will make the system break away from zero and connect again on the other side of the zero implemented. In the s-domain, the system is considered stable with every negative zero placement. Looking into the z-domain root locus, instability occurs when the branches break out of the unit circle, which happens if the zero is placed between [-1,0]. If the zero is placed in zero, the branches follow the unit circle in its outskirts and connect in minus one,

from where it breaks to minus infinity and 0. If instead the zero is placed somewhere between [0,1], the branches do not break outside of the unit circle, and the system becomes stable if the gain is kept under 1.60e+5. These figures can be seen in figure 5.2 where a zero at minus one is seen to the left, zero at zero in the middle and zero at one to the right.

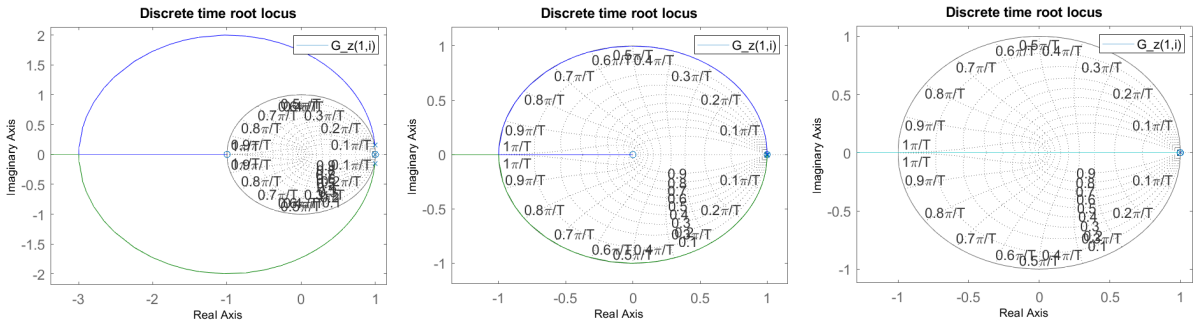


Figure 5.2: Illustration of root locus differences of different zero locations by PD control

From this knowledge about instability, it is calculated that the relationship between $\frac{-K_P}{K_D}$ has to equal -1000 at most to place a pole between [0,1]. Having established this boundary, the controller values can be guessed with knowledge of the terms. The proportional gain affects the sensitivity and susceptibility and determines how fast or slow the controller reacts. This gain increases overshoot and steady-state error, while it might be impossible to reach stability if too low. Otherwise, it might lead to instability. The derivative gain will counteract the proportional gains increase of overshoot and lead to low stability while also increasing settling time if overshoot occurs, which means that the proper relationship must be found. Controllers will not be derived in this section but are instead created in chapter 6, where a hyperparameter optimising strategy is used to find the optimal controllers. Instead, section 6 finds and uses the bounds to create a starting guess in the optimising tactic with suitable hyperparameters.

Next, a nonlinear control strategy will be created to respond to this linear control method and analyse the difference in the case of the chosen disturbances.

5.3 Sliding Mode Control

A Sliding mode controller (SMC) is a nonlinear control strategy that uses a discontinuous control signal to cause the system to slide along a cross-section of the system's normal behaviour. It is a well-known control law with the advantages of being robust against system uncertainties and disturbances, achieving fast convergence, and it can be made simple as mentioned in table 1.1 in section 1.3. The main consequence of using sliding mode as a control law is its chattering function which results in energy loss and can, in the worst case, damage the physical parts of the quadcopter.

The SMC will be designed to be robust against disturbances and guarantee convergence in finite time. As this is obtained, the next step will be to reduce the chattering to decrease the energy loss and possibly other consequences. Based on the model in section 3 the system will be divided into an inner subsystem containing z ϕ, θ and ψ, and an outer subsystem containing x and y. PD controllers will control the external subsystem, as sliding controllers in cascade are challenging to use as they react simultaneously, and a faster inner system is not reached. The sliding mode controller architecture can be seen in 5.3.

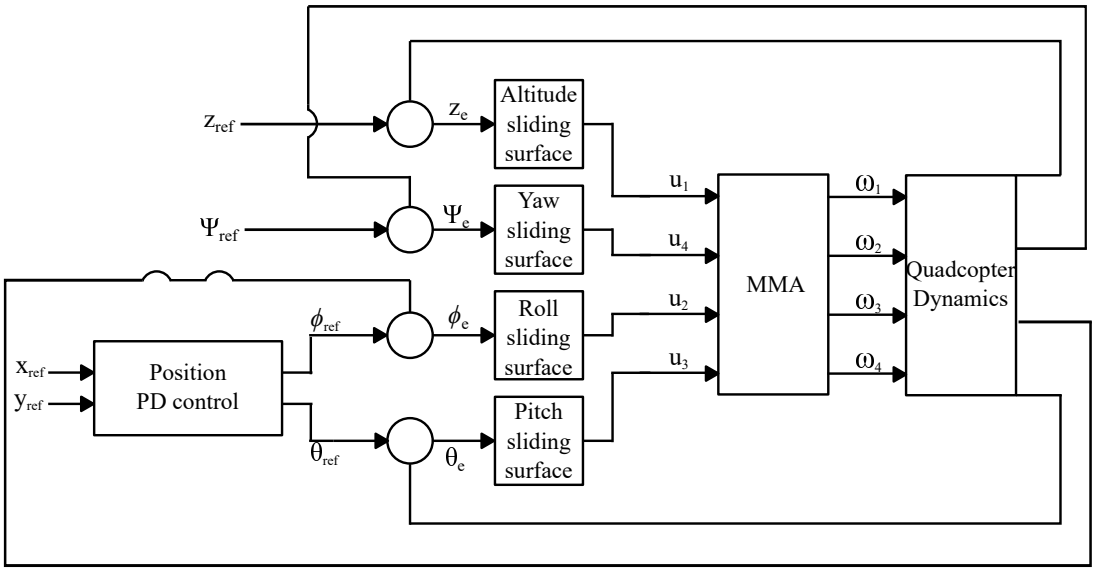


Figure 5.3: Illustration of sliding mode control in a block diagram

5.3.1 Controller Design

The controller design is created from sliding mode control theory to ensure that the states z, ϕ, θ, and ψ converge toward their references [29]. The sliding surfaces for these two DOF's can be seen in equations 5.8 to 5.12 where the goal is to equalise these to zero in finite time.

$$s_1 = c_z(z_{ref} - z) + (\dot{z}_{ref} - \dot{z}), \qquad c_z > 0 \tag{5.8}$$
$$s_2 = c_\phi(\phi_{ref} - \phi) + (\dot{\phi}_{ref} - \dot{\phi}), \qquad c_\phi > 0 \tag{5.9}$$
$$s_3 = c_\theta(\theta_{ref} - \theta) + (\dot{\theta}_{ref} - \dot{\theta}), \qquad c_\theta > 0 \tag{5.10}$$
$$s_4 = c_\psi(\psi_{ref} - \psi) + (\dot{\psi}_{ref} - \dot{\psi}), \qquad c_\psi > 0 \tag{5.11}$$
$$\tag{5.12}$$

To make these sliding surfaces reach zero in finite time and become an invariant set, the control term u will be used in the presence of $f(t, y, \dot{y})$. The errors for this control will be defined: $e_1 = y - r$, which gives the closed-loop dynamics in equation 5.13.

$$\begin{aligned} \dot{s}_i &= ce_1 + e_2 \\ \dot{e}_1 &= e_2 \\ \dot{e}_2 &= \ddot{y} - \ddot{r} \end{aligned} \tag{5.13}$$

In these closed loop dynamics r describes the references and $\ddot{y} = f(t, y, \dot{y}, u)$ with the control law linked describes the actual signal. $\ddot{y}$ can then be divided into a disturbance term f and an input term G so that the controlled term y can be written $\ddot{y} = f + Gu$, which can be used to rewrite the closed loop dynamics:

$$\dot{e}_2 = f + Gu - \ddot{r} \tag{5.14}$$

With the dynamics stated, u will be built to drive the error to zero in finite time and become an invariant set. To know that this is possible, stability has to be proven. This will be done using the Lyapunov direct

method where the Lyapunov candidate is chosen as in equation 5.15 so that the derivative becomes:

$$V_i = \frac{1}{2}s_i^2 \qquad (i = 1, 2, 3, 4) \tag{5.15}$$

$$\dot{V}_i = s_i\dot{s}_i \tag{5.16}$$

The goal of creating stability is to obtain a system that is positive definite for the candidate 5.15 and negative definite for the derivative in equation 5.16 which establishes global stability.

- V(x) is positive definite
- $\dot{V}$(x) is negative definite
- V(x) $\rightarrow \infty$ as ||x|| $\rightarrow \infty$

This will give a system with global finite-time convergence. With the Lyapunov candidate chosen as in equation 5.15 is it obvious that the function is positive definite for all sliding surfaces s_i as V(0)=0 and V(s_i)>0 $\forall$ $s_i \neq 0$. To obtain negative definiteness for the derivative term, the Lyapunov candidate will be further expanded to achieve this goal. First, the candidate will be rewritten to equation 5.17.

$$\dot{V}(s_i) = s_i\dot{s}_i = s_i(f + Gu - \ddot{r} + ce_2) \tag{5.17}$$

To achieve negative definiteness u will be used and designed so the goal can be achieved. u is designed so it replicates some of the dynamics, and the equation for the candidate can be simplified. The control of u is chosen u= $\hat{g}^{-1}(\ddot{r} - ce_2 + \nu)$ which then results in this candidate:

$$\dot{V}(s_i) = s_i\dot{s}_i = s_i(f + G(\hat{g}^{-1}(\ddot{r} - ce_2 + \nu) - \ddot{r} + ce_2) \tag{5.18}$$

For this to be usable, it is assumed that $\hat{g}$=G and that there exists a bound |f|$\leq$ L, where L is the bound. This can be seen in equation 5.19.

$$\dot{V}(s_i) = s_i(f + \nu) = s_if + s_i\nu \leq |s_i|L + s_i\nu = \underline{-\alpha V^{\frac{1}{2}}} \tag{5.19}$$

With this established ν becomes a control term which will be defined:

$$\nu = -\rho sgn(s_i) \quad and \quad sgn(s_i) = \begin{cases} 1 & \text{for } s_i > 0 \\ -1 & \text{for } s_i < 0 \end{cases}, sgn(0) \in [-1, 1] \tag{5.20}$$

Now that ν is defined and u is created concerning negative definiteness for the derivative term of the Lyapunov candidate, this Lyapunov candidate can be used to obtain ρ in equation 5.21.

$$\dot{V}(s_i) \leq |s_i|L + |s_i|\rho = (L - \rho)|s_i| = -\alpha V^{\frac{1}{2}} = -\alpha\sqrt{\frac{1}{2}s_i^2} = -\frac{\alpha}{s_i}|s_i| \Rightarrow \rho = L + \frac{\alpha}{\sqrt{2}} \tag{5.21}$$

From equation 5.21 it can be seen that negative definiteness is accomplished, as $-\alpha\sqrt{\frac{1}{2}s_i^2}$ can only be a negative number or zero, meaning the goal of stability is reached. Now the next step will be to create robustness and decrease the assumed chattering. An example to prove this stability will be made from control of $\ddot{z}$ where equation 3.42 is used, and $\ddot{z}$ can be seen in equation 5.22.

$$\ddot{z} = \frac{T}{m}c(\phi)c(\theta) - g - \frac{1}{m}A_z\dot{z} \tag{5.22}$$

A bounded term d(t) will be added to the equation to guarantee a stable closed-loop in the presence of boundaries. Then the equation can be reduced by letting f=-g-$\frac{1}{m}A_z\dot{z}$+d(t) and G=$\frac{1}{m}$ while T is the input, meaning the equation can be simplified to:

$$\ddot{z} = f + Gu \tag{5.23}$$

This will lead to $\dot{e_2} = f + Gu - \ddot{r}$ which is defined in 5.14 and used through the stability proof, meaning from here stability is obtained.

By reaching stability mathematically, the tuning gains c, and α will be found by using the same optimisation strategy as for the linear controller, which can be found in section 6. The controller obtained can be seen in equation 5.24.

$$u = \hat{g}^{-1}(\ddot{r} - ce_2 - (L + \frac{\alpha}{\sqrt{2}}sgn(s_i)) \tag{5.24}$$

Hyperparameter Optimisation

6

Contents

This chapter explores the possibilities of using hyperparameter optimisation to tune controller values. This type of tuning method defines some hyperparameters which values is further used to optimise the learning algorithm. It is defined as a machine learning method used to optimise the tuning of data within some required constraints.

6.1 Controller Tuning

When acquiring a model with a control method completed to gain control of the system, it is ideal for tuning this method to obtain the best results from controller values. This tuning to optimise control regarding the goals can be performed using different techniques. One option is to integrate a controller onto the quadcopter and test its performance, then regulate the values compared to the results seen. This will be done several times until satisfying controllers are obtained. This method could damage the drone without a test-stand, as inadequate values lead to a crash, whereas a test-stand would be preferred. This method could be time-saving for the experienced control tuner, as it is about guessing right with the knowledge one has acquired. This method will, in most cases with less experienced people, be time-consuming.

Another option requires a model of the system simulating the dynamics and behaviour with the chosen controller values. This method is safer for the quadcopter and cheaper by not necessitating a test-stand but in many cases less accurate due to the unmodelled dynamics. Tests could, in this case, also be running faster, and satisfying controller values could be reached using less time. This method's time consumption

depends on the number of parameters and simulations required, which is a case with many controllers that may be time-consuming, such as this drone.

A third option for optimising the controllers is a machine learning algorithm in the category of Hyperparameter optimisation. This method uses hyperparameters to perform a learning algorithm based on different constraints, weights or learning rates to indicate data patterns. In this case, a surrogate model will approximate the actual system from input to output to save time. These methods used with the surrogate model are assumed to predict the output from chosen controller values quickly and further predict optimal controllers.

Three generally used learning algorithms are the "grid search", "the random search", and the "Bayesian optimisation", which all have some set constraints in which they search for an optimum. To demonstrate the search algorithms figure 6.1 illustrate how they investigate optimum differently. The optimum is visualised as a red dot.

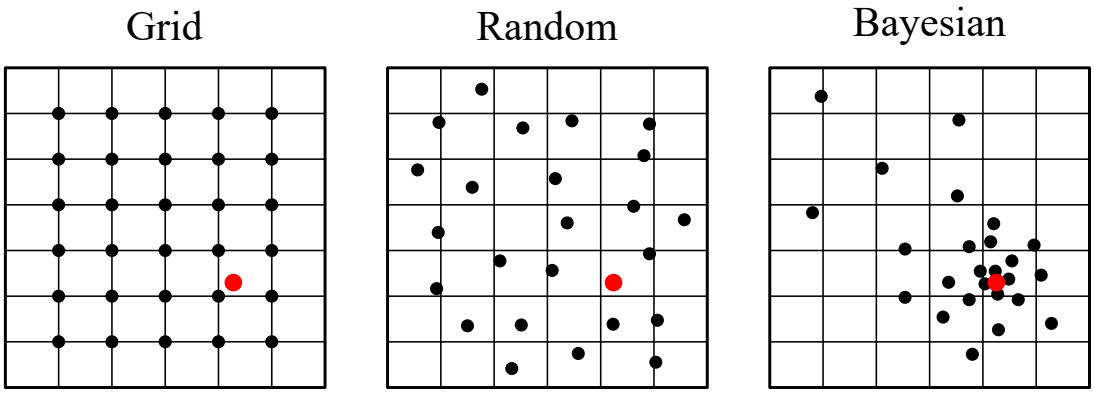


Figure 6.1: Hyperparameter optimisation methods

The *grid search* is the most straightforward search algorithm used to generate candidates within a specified search space as illustrated in figure 6.1 [30]. Here samples distributed with equal space between them is used. It is the most widely used search algorithm because of its simplicity, but it may be challenging to use if one does not have experience narrowing the search space or if several hyperparameters are awaiting tuning. Article [30] describes the method to be preferable when the user has enough experience of choosing hyperparameters, narrowing the search space sufficient. Also, it is described that more than three hyperparameters tuned simultaneously are not intended with this method.

The *random search* instead chooses hyperparameters independently using a probability distribution which comes out with these random candidates seen in figure 6.1. This search algorithm improves the grid search as it can continue until the desired accuracy is obtained or the maximum computational effort is reached. The most significant difference between the grid and random search is that the grid search has a fixed set of hyperparameters uniformly distributed, where random search might be more likely to approach the optimal point if it is not uniformly distributed. Also, grid search finishes searching when the grid assigned is completed. Random search instead can be set to have a greater probability of finding the optimum if more time is used. This is referred to as the Monte Carlo technique [30], which is needed if larger amounts of multi-dimensional datasets are processed. The random search might be more effective than grid search but at the cost of being a computationally intensive method.

The *Bayesian optimisation* outperforms the grid and random search as it aims to find the optimal point from using a minimum number of trials. It can balance exploitation and exploration, which means it

decides a current best on current information or collects more information if needed. The Bayesian technique also has the benefits of not requiring to possess preliminary knowledge of the hyperparameter distribution, and it is computationally efficient as it correlates each trial to become a better candidate requiring fewer attempts.

The Bayesian optimisation method will be used through tuning controllers as it seems to perform best and most costly. It will be described in the next section.

6.2 Bayesian Optimization

Bayesian optimisation (BO) is categorised as a black-box optimisation method as it is only possible to observe outputs based on given inputs but not assess the equations. Instead, it deals with optimising the functions within, based on the outputs. This learning algorithm is built upon the fundamentals of creating a surrogate model that can determine the objective function and declare the next sampling point it uses from an acquisition function. In this thesis, the surrogate model will be non-parametric, where a hypothesis will create the primary objective function, and observational data from here with prior costs will be used to fit posterior data to maximise the probability of reaching the optimum. In the acquisition function Gaussian process (GP) will be used to create the probability function, as it is overwhelmingly the most widely used [30]. This thesis will create a GP acquisition function, establish some hyperparameters and a cost function and use Matlab commands to perform the Bayesian optimisation. This process will be described in this section, and figure 6.2 visualises the progress.

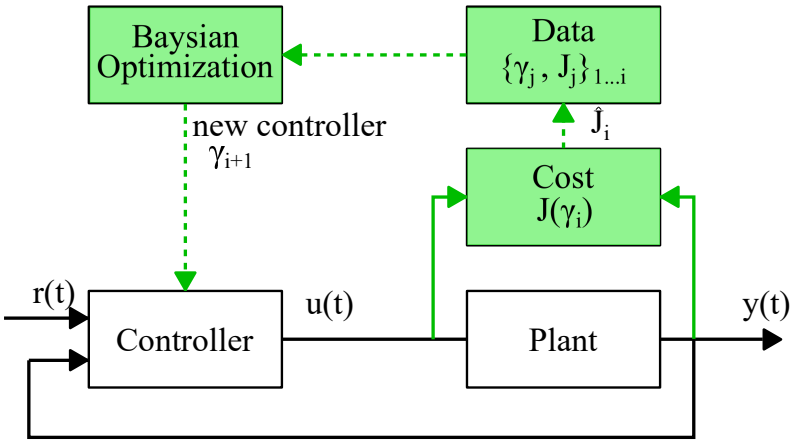


Figure 6.2: Block diagram of the proposed Bayesian optimization (BO) method used to tune controllers. Controllers are parameterised γ_i and evaluated by cost function J in closed loop. By evaluating previous attempts $\gamma_i, \hat{J}_{i_{1...i}}$, BO proposes a new controller γ_{i+1} closer to find the global optimum.

6.2.1 Overview of Bayesian Optimisation

The GP uses multivariate Gaussian distribution to find an infinite number of real-valued variables $x_1, ..., x_n \in \chi$ and it can be described as in equation 6.1.

$$f(\lambda) \sim GP(\mu(\lambda), k(\lambda, \lambda')) \tag{6.1}$$

Here μ is a mean vector, and k is the covariance function, both described in equation 6.2.

$$\mu(\lambda) = E[x] = \begin{bmatrix} \mu_{x_1} \\ \vdots \\ \mu_{x_n} \end{bmatrix}, \quad k(\lambda, \lambda') = E[(x' - \mu(x'))] = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_m) \\ \vdots & \ddots & \vdots \\ k(x_m, x_1) & \dots & k(x_m, x_m) \end{bmatrix} \tag{6.2}$$

These two terms determine the features of the surrogate model where μ regulates the precision and amplitude of the samples and $k(\lambda, \lambda')$ the quality of the model. The covariance function is used to establish similarity between the data and satisfy symmetry given as $k(\lambda, \lambda') = k(\lambda', \lambda)$. The covariance function is also known as the kernel function in this optimization and is mainly used in a squared exponential format $k_s E(x, x') = exp(-\frac{||(x-x')||^2}{2l^2})$, while a frequently used mean values is $\mu(\lambda) = 0$. Gathering a large mean value indicates a greater possibility of approaching optimum, while a larger kernel value refers to a worthy exploration. l in the kernel is used as a length scale parameter which can be used to determine if the correlation between x and x' is strong. If the value of l is large, the correlation between the points becomes strong even for points far from each other, where a small value of l means that only points in the near vicinity have a strong correlation. GP is conceptually difficult to understand and has a drawback for large dimensions or datasets, and determining the hyperparameters highly influences its efficiency. To get the knowledge and understanding behind GP and further on Bayesian optimisation, please refer to the reviews in [31], [30] and [32].

Next, the Acquisition function has to be designed so that its trade-off between exploration over search space and exploitation in current areas culminate into a good result. Here a common choice is called the expected improvement (EI), which establishes the next point based on the hyperparameters in the surrogate model. The acquisition function creates the goal of the BO, which is to maximise the EI by decreasing the mean. In this optimisation, the improvement will be further expanded to an expected improvement plus, which is an expansion trying to avoid the possibility of ending in a local minimum. By adding an exploration ratio, t_σ is added so that an acceptable exploration ratio is reached. Adding this ratio the next guess of x must fulfill the condition $\sigma_f(x) < t_\sigma \sigma(x)$, with $\sigma_f(x)$ being the standard deviation of the posterior costs and σ being the standard deviation of the additional noise added in section 4. Adding this expansion increased the possibility of reaching a global minimum by controlling the trade-off between exploring new points or concentrating on near points. Maximising the expected improvement plus acquisition function can be seen in equation 6.3.

$$EIP(x) = \begin{cases} (f(x^+) - \mu(x))\Phi(Z) + \sigma(x)\phi(Z), & \textit{if } \sigma(x) > 0 \\ 0, & \textit{if } \sigma(x) = 0 \end{cases} \tag{6.3}$$

$f(x^+)$ is the value of the best cost function so far, where Φ is the cumulative distribution and ϕ the probability distribution, where Z is described in equation 6.4

$$Z = \begin{cases} \frac{f(x^+) - \mu(x)}{\sigma(x)} & \textit{if } \sigma(x) > 0 \\ 0 & \textit{if } \sigma(x) = 0 \end{cases} \tag{6.4}$$

From these equations, the goal is to approach zero for the mean μ, which is done by increasing the first term in equation 6.3. Also, it is a goal to not end in a local minimum by using the exploration ratio so that exploration is increased if ending in a local minimum, and further unexplored points will be searched to find a global minimum.

To use this optimisation strategy, the tool MATLAB uses implemented code to generate the optimisation. For both linear and nonlinear controllers, there are two variables to be optimised. K_P and K_D are used

for linear controllers, and c and α are used in the nonlinear. To achieve optimised controller parameters, the BO created will be used and described here.

A cost function is designed with the chosen parameters acquired as costs. This function will be calculated through all simulations and based upon the desired variable's reference and actual values. This can be described in equation 6.5.

$$J = \sum_{i=1}^{N} (|ref_{\phi} - x_{\phi}| + |ref_{\theta} - y_{\theta}| + ...) \tag{6.5}$$

Here N is the number of samples, where the reference minus variable represents the controllers optimised. For simplification, all controllers will not be tuned simultaneously as the Gaussian process regression(GPR) would have difficulty performing, and if one parameter creates instability for one controlled value, the cost will go towards infinity. For the GPR a MATLAB command *fitrpg(X,y,'option')* will be used. This function creates the GPR with the option of choosing the desired kernel function as represented in equation 6.6, where a linear controller is chosen as an example.

$$gprMdl = fitrgp([K_P\ K_D], Cost_i, 'KernelFunction', 'squaredexponential') \tag{6.6}$$

Next, the hyperparameters are chosen within the desired values, where a MATLAB command is again used. Here, the data type must be chosen, and the respective span has to be set. These optimisable variables can be seen in equation 6.7 in the respective MATLAB command.

$$K_p = optimizableVariable(K_p, [0, 10], 'Type', 'real') \tag{6.7}$$
$$K_D = optimizableVariable(K_D, [0, 10], 'Type', 'real') \tag{6.8}$$
$$\tag{6.9}$$

Now all the parameters are set, and so the Bayesian Optimisation can be created. The BO should use the expected improvement plus acquisition function, and the MATLAB command also commands the prediction function with all handles. This command can be seen in equation 6.10.

$$\begin{aligned} bayesObject = &bayesOpt(@(tbl)mdlfun(tbl, gprMdl), vars, 'Verbose', 1, ... \\ &'AcquisitionFunctionName', 'expected - improvement - plus') \end{aligned} \tag{6.10}$$

Within this MATLAB command, the prediction function is called mdlfun(tbl,gprMdl) and will be used to find the next point from the acquisition function. Bayesopt will use this prediction function to create a cost function value based on the new parameters collected as the optimisable variables. These new variables are used in the GPR to predict the cost function value. The prediction function is called in equation 6.11.

$$f = predict(gprMdl, vars) \tag{6.11}$$

Then equation 6.10 minimises the prediction function and optimised controllers parameters will be investigated through these equations in this section 6.2. The MATLAB code used for the Bayesian optimisation can be seen in Appendix B.

The following sections will present the controller parameter defining and tuning for linear and nonlinear control with disturbances attached.

6.3 Tuning of Controllers

With Bayesian optimisation, tuning of both linear and nonlinear controllers will be achieved through the following subsections.

6.3.1 Linear Control

For linear control, PD controllers are selected in section 5.2 to control the drone simply and still be able to reject the proposed disturbances. A representation of controllers combined with the model can be seen in figure 6.3 with the different disturbances implemented as well.

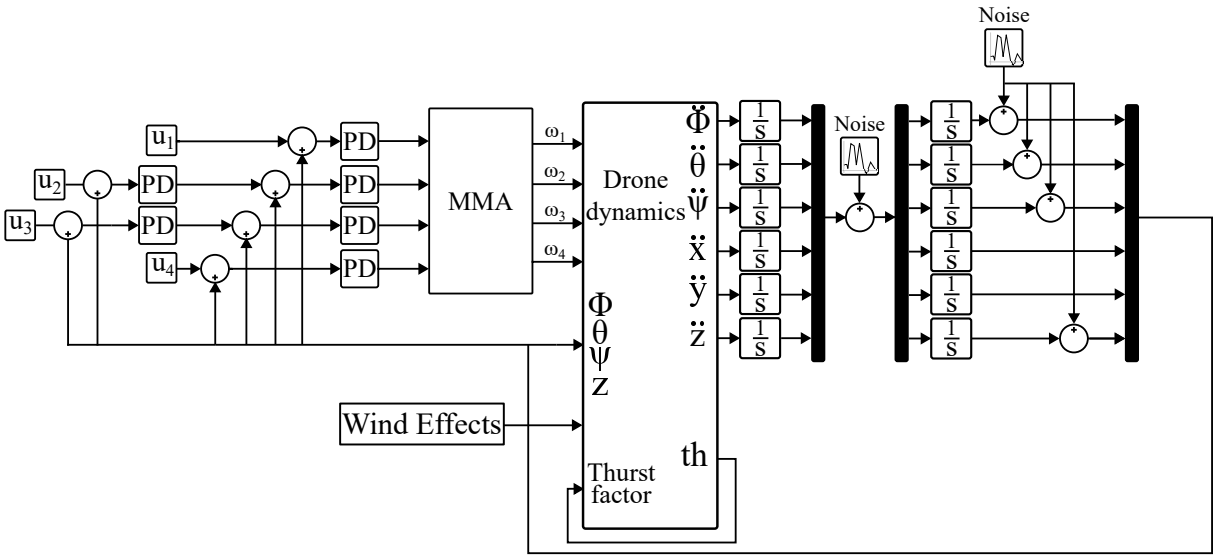


Figure 6.3: A presentation of the system with linear controllers, wind disturbance, noise model and ground, ceiling or wall effects attached

With the model accomplished and controllers connected, tuning of the six PD controllers will be done using Bayesian optimisation. Only the angle and altitude controller will be found as a start to narrow the span of controller values investigated. When these can be used to control the angles of the drone, position controllers are to be optimised.

As a linear control theory is made in section 5.2, some large boundaries of controller values are found and used in the Bayesian script. With the large boundary of $\frac{K_p}{K_D} = 1000$, the hyperparameter boundary will be set to [0,1000] for both K_p and K_D. Creating a stable starting guess with this boundary is difficult, so the costs will not go towards infinity. Because of that, another Bayesian optimisation strategy will be used to find some stable points which can be used as starting guesses for the proposed BO. The new strategy will narrow the boundary spectrum. This strategy is shown in appendix B where the model will be given a set of random numbers within the boundary. These simulations with random numbers for the controllers will be used in the BO to look through a large space towards stable controllers. The BO with random numbers will be run 150 times to visualise usable boundaries for optimal controllers.

As stable controllers are found, the script in appendix B will be used with the starting guess of the found stable controllers from the code in appendix C.

As these scripts have been run, a set of optimal PD controllers is found from the model with disturbances. The controller values can be seen in table 6.1.

Controllers	K_P	K_D
x	4.63	5.28
y	5.04	4.72
z	218.36	173.97
ϕ	5.36	5.08
θ	9.12	7.14
ψ	8.7229	2.7491

Table 6.1: PD controller values found by using Bayesian optimisation

First, the control of the angles is tested with both a roll and pitch change while moving up in altitude. This is presented in figure 6.4.

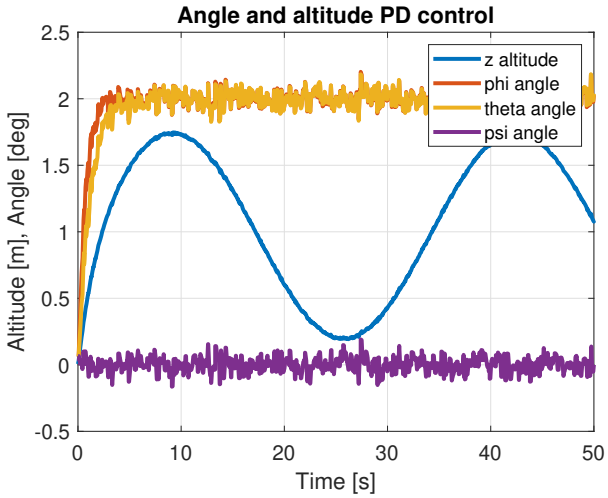


Figure 6.4: Angle control with PD controllers and a changing thrust factor

The noise can easily be seen from the output signals, while also the thrust factor seems to be overcome when near the ground as a smooth sine wave can be maintained. Figure 6.5 shows how the thrust increases when near ground, but while it also chatters, which disturb the control, meaning it has increased robustness towards a changing thrust factor.

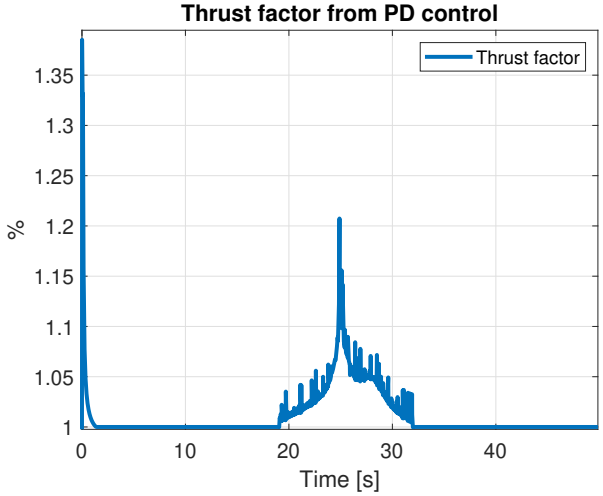


Figure 6.5: Thrust factor for PD control

Further, the control will be omitted to wind disturbances to visualise the robustness of angle control. The effect from wind effects can be seen in figure 6.6.

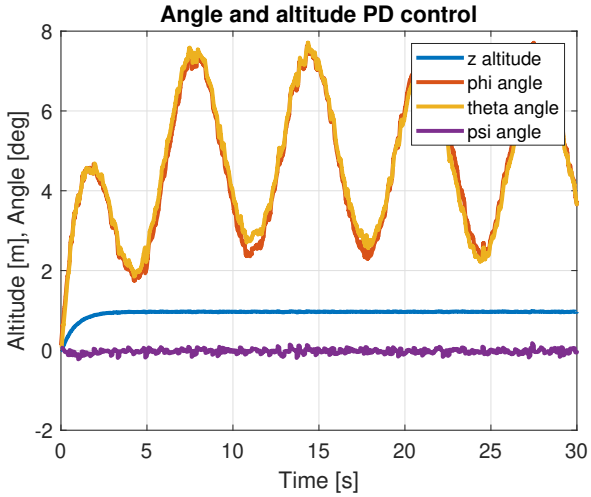


Figure 6.6: Angle control with PD controllers and a sinusoidal wind disturbance

From figure 6.6 it can be seen how the control does not counteract the wind, and the angles are pushed back and forth as the wind interacts with the quadcopter.

While control of angles and altitude is established for linear controllers, also position control is to be made. The results of position control with linear controllers can be seen in figure 6.7. The same strategy will be used here, whereas the controllers shown in table 6.1 provide the tuned controllers from BO.

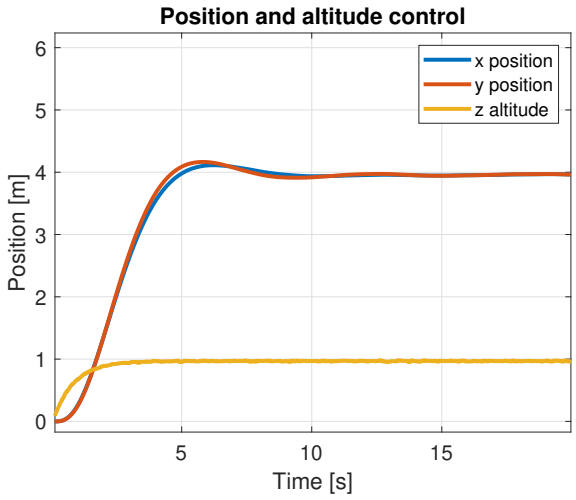


Figure 6.7: Position and altitude control with PD control

Stability in position control has been achieved, but the performance can be further optimised, which will be looked further into in section 7.3.

6.3.2 Nonlinear Control

The structure of the nonlinear control theory can be seen in figure 6.8. The nonlinear control strategy looks like the PD control structure in figure 6.3, but with some minor corrections in the representation

of the system. Also, when attaching the nonlinear control structure, the references become threefold as they cover position, velocity and acceleration.

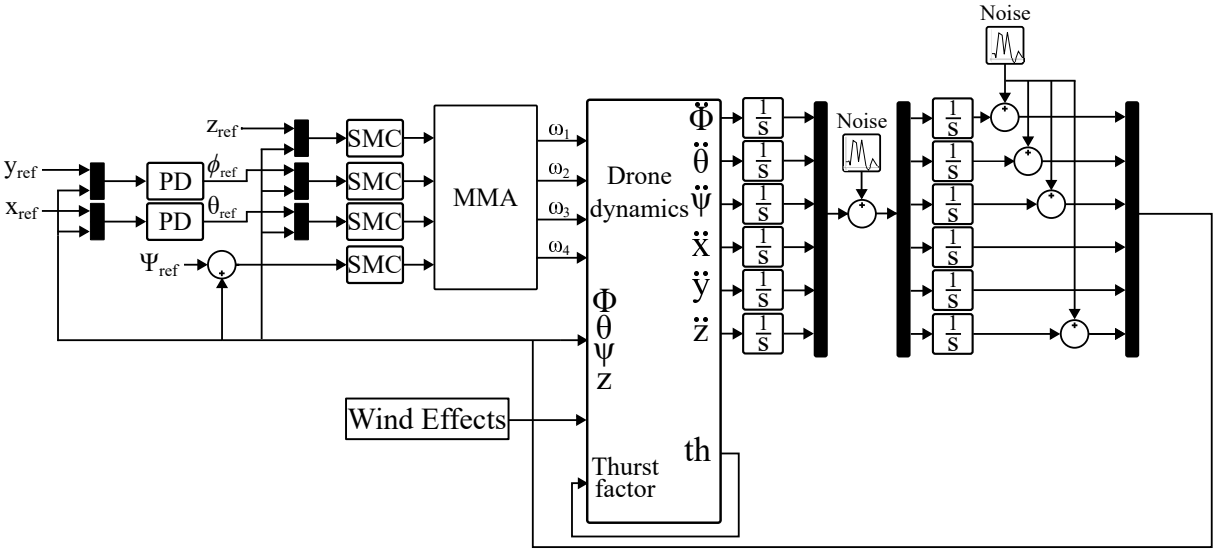


Figure 6.8: A presentation of the system with nonlinear controllers, wind disturbance, noise model and ground, ceiling or wall effects attached

As assumed in section 5.3, the controller applies a chattering performance with the sgn function. The purpose is to smooth the output of the control law. In figure 6.9, the sgn function changes the output instantaneously. The goal is to create a continuous curved change between minus one and one, and because of that, the discontinuous sgn function will be replaced with a continuous approximation to avoid this effect and attain a more smooth control signal. To avoid this discontinuous change, a sigmoid function presented in equation 6.12 will replace the sgn function in further tuning, and the parameter ϵ will be included as a tuning parameter.

$$B_{sf} = \frac{s_i}{|s_i| + \epsilon_{sf}} \tag{6.12}$$

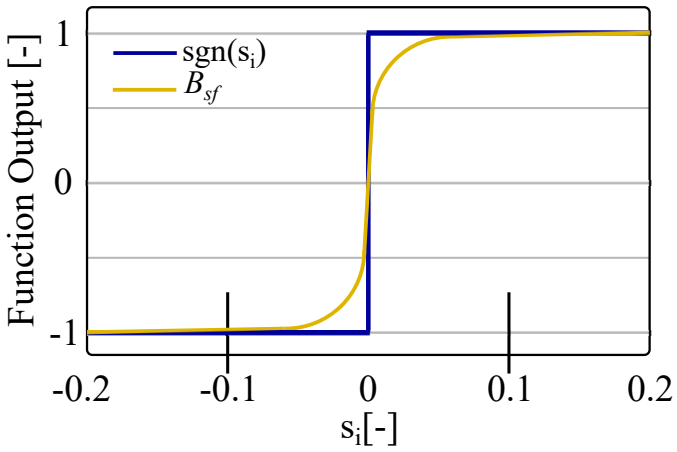


Figure 6.9: Illustration of output signal going from -0.1 s_i to 0.1 s_i with respectively a sgn function and a sigmoid function

By implementing the sigmoid function, the SMC controllers can be found with the same technique as linear controllers. Again the inner loop is tuned first, after which the outer loop controllers are optimised. The coefficients optimised can be seen in table 6.2 with the respective optimised values.

Controllers	c	α	L	ϵ
x	0.43	3.7	5.59	4.87
y	0.39	0.26	0.37	0.96
ϕ	4.17	0.52	4.44	40.00
θ	8.49	3.07	1.98	42.10
ψ	5.48	4.97	4.42	47.92

Table 6.2: SMC controller values found using Bayesian optimisation

With the tuned controller values for the sliding mode controllers, the same tests have been created to find optimised controllers for the SMC and compare them with linear control performance. First, the inner loop controllers are found, where control of angles and altitude can be seen in figure 6.10, and compared to the same test in section 6.3.1.

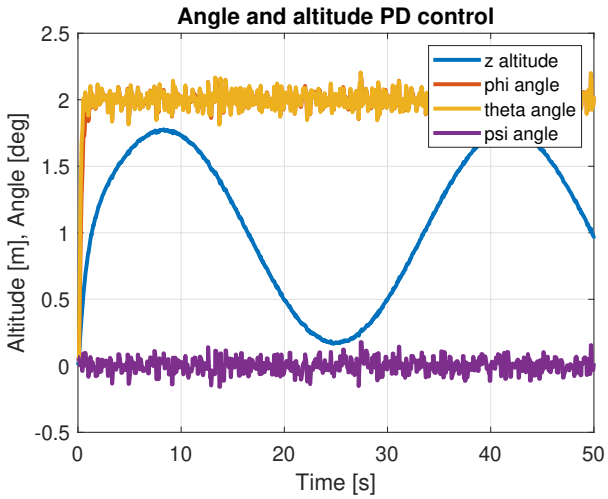


Figure 6.10: Angle and altitude control with changing thrust factor for SMC

The response looks more or less the same, but a faster response time and settling time on angle control is achieved with the SMC control. Further, the tests with the same wind disturbance are created with the optimised SMC control and can be seen in figure 6.11.

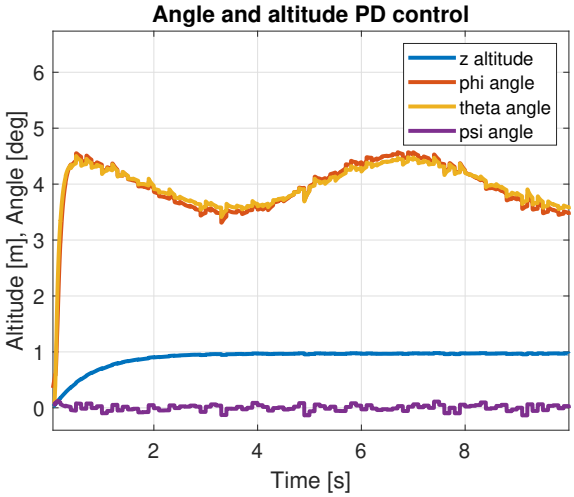


Figure 6.11: Angle and altitude control with a wind disturbance acting on SMC

Comparing figure 6.6 and figure 6.11 a clear difference can be found in rejecting the disturbance from wind. The SMC is more robust against wind disturbance as the sinusoidal curve is smaller in amplitude than from PD control. With the angle and altitude control established for nonlinear control, the position control is further created and can be seen in figure 6.12.

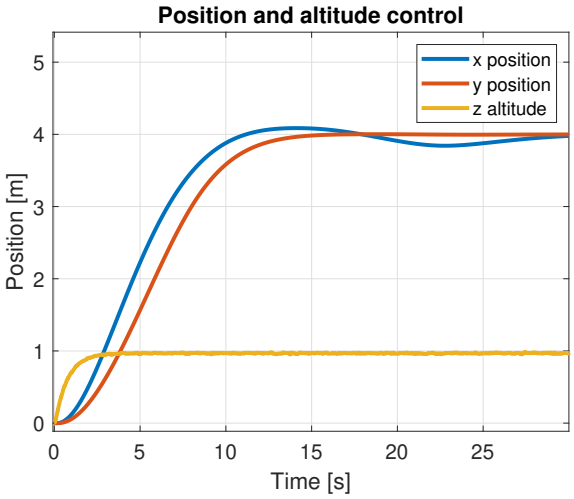


Figure 6.12: Position and altitude control for SMC

Here, the position control is slightly slower than the PD control, which is not desired, and will be further tuned in section 7.3.

6.4 Summary

Position and angle control is achieved both by the linear and nonlinear controller in section 6.3. Some results are visualised in plots through the section where the stability of controllers is proven. With this achieved, gaining robustness will be further studied in the next section, where some additional initiatives will be included in the model.

Improvement Initiatives 7

Contents

This chapter establishes some further initiatives towards gaining better performance and faster response. The initiatives will be described and created in this chapter. Tests will be shown in the next chapter with the elected initiatives implemented.

7.1 Initiatives

With the disturbances created in section 4 and integrated into the model, some issues occur in the results of the optimal controllers found in section 6.2. Stability is proven, but the angle control is chattering because of the implemented sensor noise. This negatively affects the controllers, and better results are assumed if a filter reduces noise before using control. A typical filter used to estimate the actual outputs without noise is the Kalman Filter, which creates an estimate between the modelled output and the observed output. Correct implementation of this would lead to diminished chattering from sensor noise and increased control performance.

When looking into the angle control and position control of respectively linear and nonlinear control, the ground effects disturbance did not seem to impact significantly. In angle control, the sinusoidal effects from wind did influence the angle position with the disturbance seen in figure 4.3. This effect was illustrated in section 6.3. This means that disturbance rejection was not completed, and further initiatives could be added for this rejection. For obtaining more desired results, the sliding mode control created will be expanded to an integral sliding mode control(ISMC). This method is assumed to derive the sliding mode towards zero from time zero.

These two initiatives are presented in this section and further used in the results in section 8.

7.2 Extended Kalman Filter

The Kalman filter generally tries to estimate the states from already known linear dynamics combined with a series of noisy measurements. This filter can be used when trying to eliminate the chattering outputs achieved from the model, but the filter can also be used for many more tasks. If requested, this filter type can achieve sensor fusion, where some sensors can be combined to obtain the most precise results. It can also estimate disturbances as wind or obstacles to prepare the drone before closing in on a disturbance. In this thesis, the filter will estimate the actual output from the noisy output seen in the control, as the other disturbances seem to be diminished by control.

The classic Kalman filter operates with linear systems and cannot be used on a quadcopter model as it is nonlinear. Because of that, it is expanded to a so-called Extended Kalman filter (EKF), where knowledge will be gathered from [33]. The classic filter will use Gaussian distribution to estimate the signal from a linear function, but while the function becomes nonlinear, the Gaussian distribution will become not Gaussian. When using the extension to the linear filter, it instead tries to locally linearise the system around a current mean, using the linear model created in section 3. The EKF is used to find an optimal state estimated from two Gaussian distributions of predicted and measured states. The control of the filter determines which measurement input is trusted most, and the optimal state estimate would by the tuning be placed close to the most trusted output. This can be visualised in figure 7.1 where the measurement is trusted more than the predicted states, and in between them, the optimal state estimate is placed closest to the measurement.

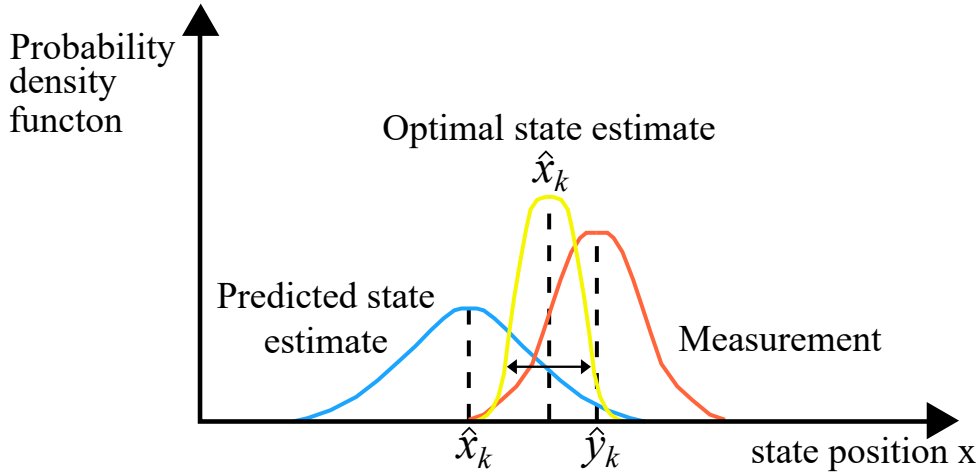


Figure 7.1: Illustration of the optimal state estimate in the EKF

As described, the linear and nonlinear model will be used to estimate an optimal sensor output in this thesis, and from there, the predicted states is shown in equation 7.1.

$$\begin{aligned} x_k &= f(x_{k-1}, u_k) + \omega_k \\ y_k &= g(x_k) + \upsilon_k \end{aligned} \tag{7.1}$$

From equation 7.1 two external terms ω_k and υ_k are implemented to describe process noise and measurement noise in a stochastic variable which is to be comparable with the ones in 2.4. Both these variables are implemented as Additive white Gaussian Noise (AWGN) with a normal distribution with mean zero and the variance created in section 7.1. These terms for process noise and measurement noise is implemented as $\omega_i \sim (N(0, \sigma_i)$ and $\upsilon_i \sim N(0, \sigma_i)$. Further, the linear model has to be corrected in

every step, meaning it uses varying parameters. The linear model will obtain an initial prediction of state error covariance and find a correction matrix. These matrices are shown in equation 7.2.

$$\mathbf{G_k} = \frac{\partial f(\hat{x}_k, u_k)}{\partial x}\bigg|_{\hat{x}_{k-1}, u_k}$$
$$\mathbf{H_k} = \frac{\partial g(\hat{x}_k)}{\partial x}\bigg|_{\hat{x}} \tag{7.2}$$

The EKF uses a prediction state and a correction state to find the optimal estimate. These states can be seen in equation 7.4 where the predicted value $\hat{x}_k^-$ is calculated through the nonlinear model using the initial corrected state. Then in the prediction, P_k^- is the error covariance of the prior estimate using Q to represent the uncertainty coming from noise to correlate with the actual noise.

The correction step uses a Kalman gain K_k to determine how the observation should be trusted compared to the motion. A matrix R is implemented and used to determine whether the measurement should be trusted more or the prediction or something in between by controlling the Kalman gain. This matrix is also further used to tune the filter. To prove how the R matrix can be used, only the measurement equation 7.3 is created with R=0, meaning that the measurement should be trusted entirely. If instead R goes towards infinity, the prediction should be trusted.

$$R = 0 \Rightarrow K_k = P_k^- H_k^T (H_k P_k^- H_k^T + \underline{0})^{-1} = P_k^- H_k^T (H_k^{-1} P_k^{-1} H_k^{T^{-1}}) = H_k^{-1} \Rightarrow K_k = H_k^{-1}$$
$$\hat{x}_k = \hat{x}_k^- + H_k^{-1}(y_{k+1} - H_k \hat{x}_k^-) \Rightarrow \hat{x}_k = y_{k+1} \tag{7.3}$$

The two matrices which determine the purpose of the filter is then the Q matrix and the R matrix, with the R matrix's use proven in equation 7.3 and the Q matrix variance described in section 2.4. With these controllable matrices, the Kalman gain can be calculated and used to find the correction of the state estimate $\hat{x}_k$. These calculations in equation 7.4 will then repeat over time and are the mathematics behind the Extended Kalman Filter.

Prediction:
$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_k)$$
$$P_k^- = G_k P_k G_k^T + Q$$
Correction: (7.4)
$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R)^{-1}$$
$$\hat{x}_k = \hat{x}_k^- + K_k(y_{k+1} - H_k \hat{x}_k^-)$$
$$P_k = (I - K_k H_k) P_k^-$$

The equations stated for the filter have to be made usable for the model, which proceeds in discrete time. To achieve this, the equations will be discretised by the use of Forward Euler's method, and the discretised nonlinear part can be seen in equation 7.5.

$$\hat{x}_k^- = \hat{x}_{k-1}^- + T_s f(x_k, u_k) \tag{7.5}$$

This means that the sampling time T_s is taken into account, and the EKF's equations will be as in equation 7.6.

$$\begin{aligned}\dot{x} \approx \frac{x_{k-1} - x_k}{T_s} &= Gx_k + Bu_k \\ x_{k+1} - x_k &= T_sG_k + T_sBu_k \\ x_{k+1} &= (I + TsG)x_k + T_sBu_k \\ G_{d,k} &= I + T_sG \end{aligned} \tag{7.6}$$

The Filter is created, and its use has to be fitted to eliminate the noise of output sensor signals. Here the matrices Q and R again are in focus. As described earlier, the Q matrix describes the noise variances and includes those found where the R matrix determines which signal to trust. These matrices to control are described in equation 7.7 where a gain g is added to the R matrix to adjust which state to trust. These equations will be fitted to create the optimal estimate between the measured noisy signals and the predicted calculated outputs.

$$Q = \begin{bmatrix} \sigma^2 & & \\ & \ddots & \\ & & \sigma^2 \end{bmatrix} \quad , \quad R = \begin{bmatrix} \sigma^2 g_1 & & \\ & \ddots & \\ & & \sigma^2 g_{12} \end{bmatrix} \tag{7.7}$$

7.3 Sliding Mode Tuning

The sliding mode control proposed in section 5 is a classical Quasi-Sliding mode controller eliminating the chattering and with convergence in infinite time. Further tuning of this control can be added towards being more robust and fast [29].

To optimise the robustness of this control law an integral term can be added so the control becomes and integral sliding mode controller ISMC while also the reaching phase is eliminated. In this format the desired loop dynamics are from t=0, where the same sliding variable is used:

$$s_i = e_2 + ce_1 = 0 \qquad \forall t \geq 0 \tag{7.8}$$

Now instead, the control law will be recreated as equation 7.9 which leads to the derivative of the sliding variable in equation 7.10. Again $\hat{g}$ is assumed to equal g.

$$u = \hat{g}^{-1}(\ddot{r} + u_1 + u_2 - ce_2) \tag{7.9}$$

$$\dot{s}_i = \dot{e}_2 + ce_2 = f(t, y, \dot{y}) + gu - \ddot{r} + ce_2 = f(t, y, \dot{y}) + g\hat{g}^{-1}(\ddot{r} + u_1 + u_2 - ce_2) - \ddot{r}ce_2$$

$$\dot{s}_i = f(t, y, \dot{y}) + u_1 + u_2 \tag{7.10}$$

Now in the ISMC the design task is to design u_1 and u_2 so that u_1 compensates the disturbances $f(t, y, \dot{y})$ enforcing the sliding from t=0 and u_2 forces the sliding variable to equal zero as time goes towards infinity. Design task u_1 will be done first, and next design task u_2 will be created.

7.3.1 Design of $\mathbf{u}_1$

First, an auxiliary sliding variable S will be created with the original sliding variable s implemented as seen in equation 7.11

$$S = s - z \quad with \quad \dot{z} = u_2 \tag{7.11}$$

As in the SMC $|f(t, y, \dot{y})| \leq L$ and the auxiliary sliding variables, dynamics can be calculated:

$$\dot{S} = \dot{s} - \dot{z} = \dot{s} - u_2 = f(t, y, \dot{y}) + u_1 + u_2 - u_2 = f(t, y, \dot{y}) + u_1 \leq L + u_1 \tag{7.12}$$

Choosing $u_1 = -\rho B_{sf}(S)$ with B_{sf} being the sigmoid function used in section 6.3.2 then:

$$\dot{S} = f(t, y, \dot{y}) + u_1 = f(t, y, \dot{y}) - \rho B_{sf}(S) \leq L - \rho B_{sf}(S) \tag{7.13}$$

With this determined the stability of Lyapunov should be guaranteed again with ($V = S^2/2$ and $\dot{V} = -\alpha V^{1/2}$) while also equivalent control is obtained if $\dot{S} = 0 \Rightarrow u_{1,eq} = -f(t, y, \dot{y})$. With u_1 created to compensate the disturbances, u_2 will be designed.

7.3.2 Design of u_2

For the design of control parameter two the equivalent control mentioned above will be used and substituted into equation 7.10 so the sliding variable will become:

$$\dot{s}_i = f(t, y, \dot{y}) + u_1 + u_2 = f(t, y, \dot{y}) + u_{1,eq} + u_2 = f(t, y, \dot{y}) - f(t, y, \dot{y}) + u_2 = u_2 \tag{7.14}$$

This means that the sliding dynamics do not depend on the disturbances and can be driven towards zero. The control law for u_2 is then described in equation 7.15.

$$u_2 = -k s_i \tag{7.15}$$

With these additions to the control, sliding control is obtained from initial time, proven by equation refeq:sProve with the initial condition $b = s(0)$ then the auxiliary sliding is achieved $\forall t \geq 0$.

$$S = s - z, \quad \dot{z} = u_2 \qquad \Rightarrow \qquad S = s - \int_0^t u_2(t)dt + b \quad = s + k \int_0^t s(t)dt + b \tag{7.16}$$

7.3.3 Summarised Equations for ISMC

ISMC is created in this section to obtain better robustness and faster response. The equations and control law for this control method is summarised in equations 7.17 to 7.21.

$$\dot{e}_1 = e_2 \tag{7.17}$$

$$\dot{e}_2 = f(t, y, \dot{y}) + gu - \ddot{r} \tag{7.18}$$

$$g = \frac{1}{m} \tag{7.19}$$

$$f(t, y, \dot{y}) = -g - \frac{1}{m} A_z \dot{z} + d(t) \tag{7.20}$$

$$u = \hat{g}^{-1}(\ddot{r} - ce_2 - ks_i - \rho B_{sf}(S)) \tag{7.21}$$

Results

8

Contents

This chapter presents the results obtained from the optimisation strategy with the use of linear and nonlinear controllers.

8.1 Overview

This chapter will present the results essential to compare the different control strategies (PD, SMC, ISMC) and their performance. The different results will include disturbance rejection against gusts of wind and the thrust factor in separate cases. The signals created in this chapter have been through the created EKF calculated in section 7.2 wherefrom more smooth lines will appear. This can be seen in figure 7.1 where the signals before and after an EKF is presented.

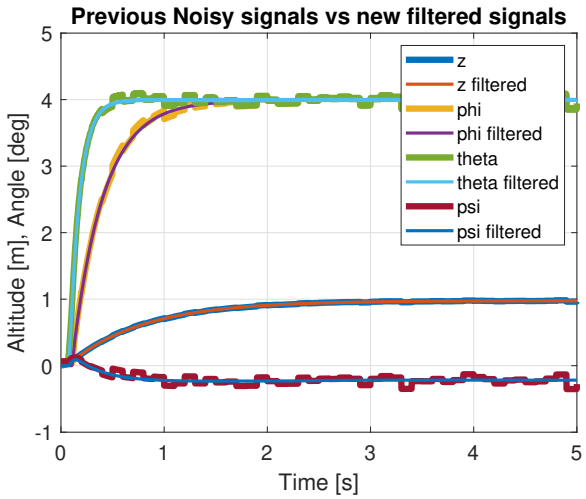


Figure 8.1: Noisy signals vs signals filtered by the EKF in section 7.2

The results created in this section will be analysed and discussed after each result. The EKF counteracts the noise created in the system and will not be tested further. All results are affected by the EKF.

8.2 Comparison of Control with Ground, Wall, Ceiling and Wind Effects

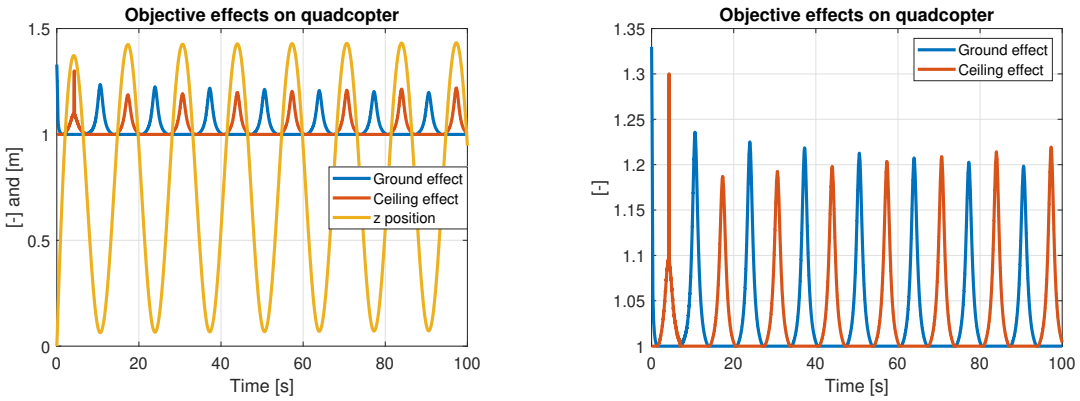
In section 6 results affected by ground effects is presented, where both linear and nonlinear controllers were stable. This section will create several further tests to visualise the rejection of ground, wall and ceiling effects on a quadcopter. The tests created will be:

- Test with ground and ceiling effects
- Test with partial ground effects on two rotors
- Test with wall effects on two rotors
- Test with ground effects and wind disturbance

Ground effects are shown for PD and SMC angular control in section 6, where the angular and vertical position has no greater reaction to this. In this chapter, tests will only be performed for position control in z, x and y, and the positions will be the reference. To further visualise the differences, a figure with the responding errors will be shown beside the positions. To demonstrate the effects acting on the quadcopter, figure 8.2 is made, where z position along with ground and ceiling effect is shown in figure (a) and the zoomed effects in figure (b). This figure presents how the factor changes along with the altitude representing the action counteracted by control. A change in equation 4.2 is made to obtain usable ceiling effects for the drone. Here equation 4.2 is changed to equation 8.1 where a ceiling is placed at 1.5 meters.

$$\frac{T}{T_h} = (0.104\frac{R}{(1.5-z)} - 0.0952)(\frac{V}{vh})^2 - 0.171\frac{R}{(1.5-z)} + 1.02 \tag{8.1}$$

$$\tag{8.2}$$



(a) (b)

Figure 8.2: Ground and ceiling effects on a quadcopter with PD control

As expected, the thrust factor changes along with altitude represented in figure (a). The ceiling is placed at 1.5 meters and the ground at 0 meters. It can be seen from figure 8.2 (b) that the thrust factor for the ceiling effect goes up along with time as the ground effect goes down, meaning that the PD control used to visualise these effects does not keep the desired position. With this change in thrust summarised, the results can be created.

8.2.1 Test with Ground and Ceiling Effects

With the knowledge of the ground and ceiling effects and a representation of them in figure 8.2, a comparison between the three control theories will be simulated and shown for ground and ceiling effects here. The goal is to follow the altitude reference so that a collision with the ground or ceiling can be avoided, but the positions in x and y coordinates will also be represented. The reference for altitude will be a sine wave with amplitude 0.7m and mean 0.775m which is shifted 2π radians to begin in 0. This means it should reach a distance of 0.025m to the ceiling and 0.075m to the ground to follow the reference. Both x and y will be set to 4 meters, so the drone will be moved a distance to determine how prominent position control is along with these effects. The results can be seen in figure 8.3.

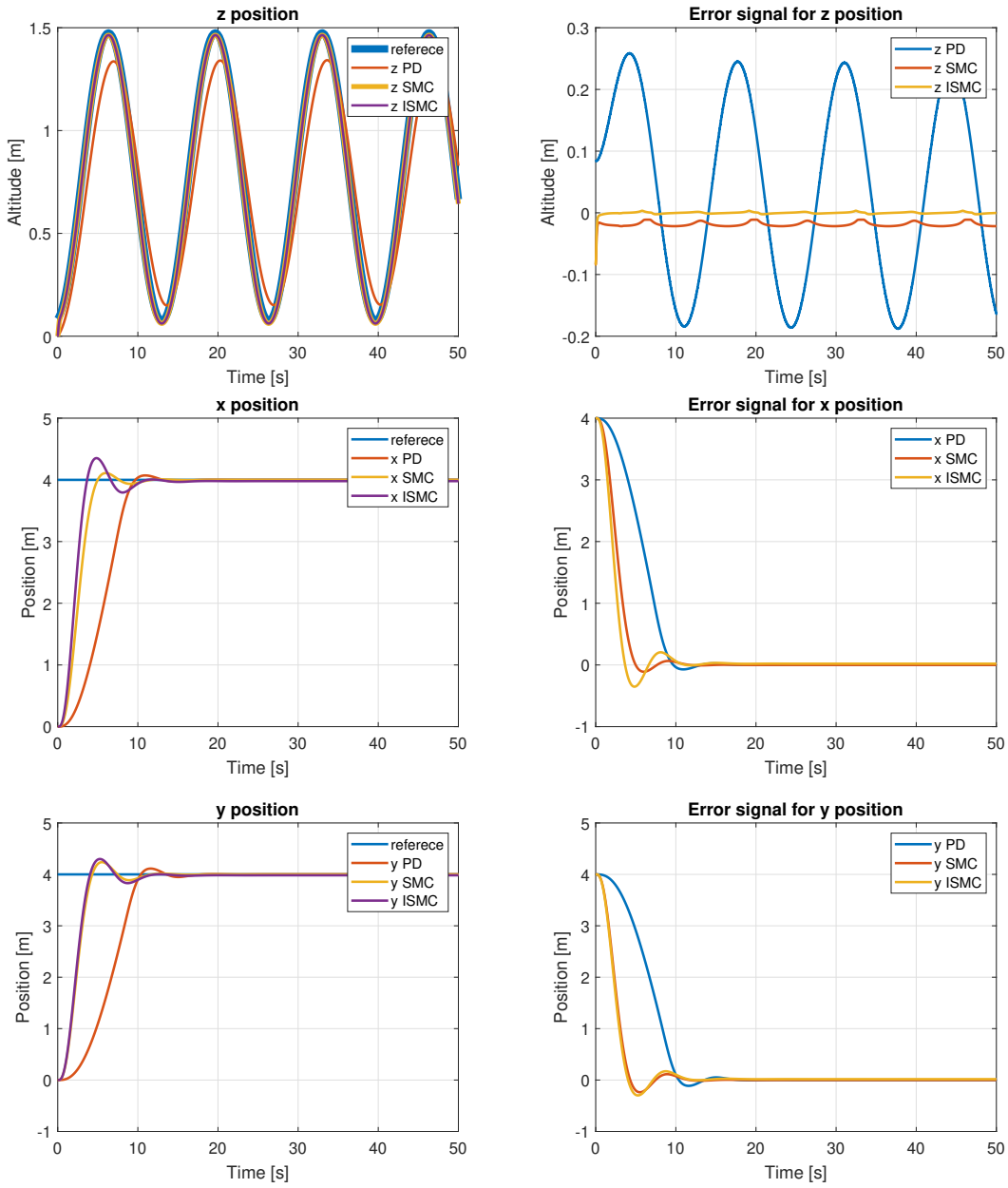


Figure 8.3: Ground and ceiling effects on quadcopter, with a change in position. Ground is placed at 0 meters and ceiling is placed at 1.5 meters.

Review of Ground and Ceiling Results

The results show a clear difference between the sliding and linear controllers when following the sinusoidal reference in altitude. The linear control law does not reach the desired reference, but it will neither collide with the ground or ceiling. The linear control lags the reference and never catches up, showing a significant error in the z position. Comparing the SMC and ISMC, the difference can be seen in steady-state error, where SMC does generate a slight error of 0.02m while also oscillating higher when affected by ground or ceiling effects. Both nonlinear control laws keep the drone from reaching the ground and ceiling and will be categorised as usable for tests.

While observing the x and y positions, the linear PD control law acts acceptable along with the nonlinear controllers, where it gains the lowest overshoot but highest settling time. All control laws can be used to

reach a desired position along the horizontal axis, with approximately zero meters in steady-state error and under ten seconds in settling time. Overshoot, settling time, and steady-state errors for all controllers can be seen in table 8.1, where the z position for PD control is categorised as too weak to be shown.

	Overshoot	Settling time	Steady state error
PD z	x	x	x
PD x	1.3%	9.2s	0.12%
PD y	1.8%	10s	0.1%
SMC z	0%	0.2s	0.02m
SMC x	2%	4.7s	0.3%
SMC y	5.1%	6.2s	0.15%
ISMC z	0%	0.15s	0m
ISMC x	6.25%	6.1s	0%
ISMC y	5.5%	6.3s	0%

Table 8.1

8.2.2 Test with Partial Ground Effects

In this test, the quadcopter is exposed to partial ground effects affecting two of its rotors to have an increased thrust, meaning that the quadcopter would try to rotate in x and y-direction as shown in figure 1.3. The results will be shown for all positions, where the crucial results will be how the x and y positions behave. Every time the drone approaches the ground, the drone will rotate, and control has to try to counteract this rotation. The results for partial ground effects can be seen in figure 8.4.

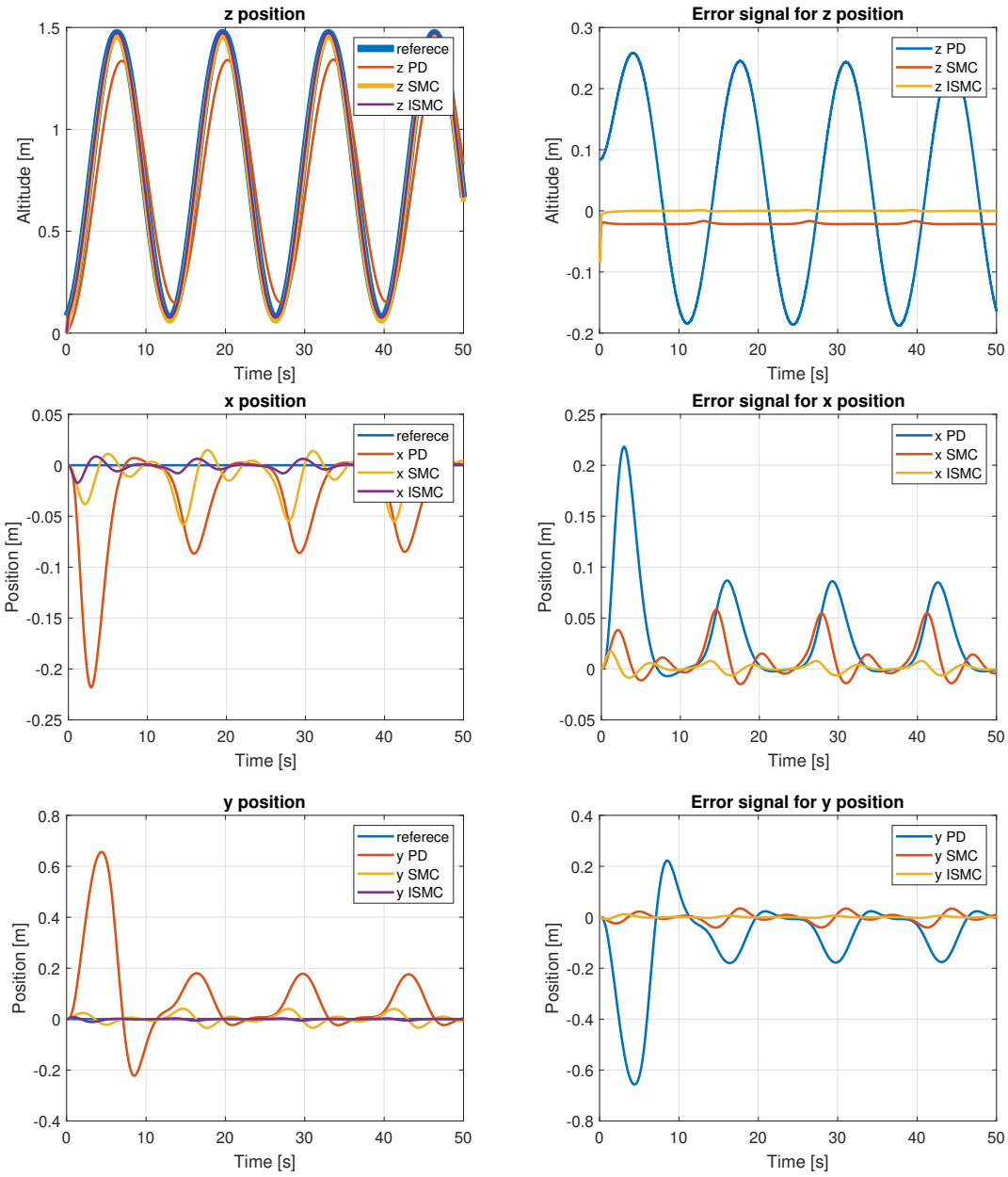


Figure 8.4: Partial ground effects affecting two rotors positively as illustrated in figure 8.2

Review of Partial Ground Results

For the partial ground results, the z position equals the previous case with the ground and ceiling effect results, meaning that the nonlinear controllers perform best and reach the position while PD control does not keep up.

The essential results for partial ground effect are the x and y position affected every time ground effects occur. The consequence of partial ground effects is observed for the horizontal position, which gets pushed off the zero references and tries to get back on track. While comparing the altitude with the x and y position, it is observed that the drone keeps its position when approaching the ground but first gets off course when elevating from the ground. As it elevates, the horizontal position gets a boost of thrust that is not controlled, thereby pushing the drone away from its reference as two rotors experience a greater thrust.

All controllers keep the drone oscillating around its references for x and y, where the linear control law achieves the most significant error followed by SMC and with ISMC performing best regarding counteracting the effects. As the signals do not settle, overshoot, settling time, and steady-state error will not be described for these signals. Instead, the oscillation error and most significant error will be presented in table 8.2. PD control drifts 0.64m away from the reference at the first meeting with the ground effects followed by an oscillating error of 0.18m when affected further. This error is large and requires ample space to test on a drone, where the nonlinear controllers keep the errors under 0.06m, gaining certain robustness. Only ISMC control law is categorised as usable for professional use in this test, as its highest error reaches 0.01m and the goal is below 0.03m

	Oscillation	Largest error
PD x	0.08m	0.22m
PD y	0.18m	0.64m
SMC x	0.06m	0.06m
SMC y	0.05m	0.05m
ISMC x	0.008m	0.01m
ISMC y	0.005m	0.009m

Table 8.2

8.2.3 Tests with Wall Effects

This test is performed to affect two rotors in the opposite direction of the partial ground effects. The drone should reach a certain height and move a distance in the x position while staying at zero in the y position. A wall is, in this case, placed at 1.5m in x position, and the drone will be moved 1.4m. The goal here is to observe whether the drone can perform close to the wall without colliding. Control in all directions is of interest in this test to visualise performance.

The equation for ground effects is changed so that it affects the x position and is implemented so that it has a negative effect on the thrust dragging the drone towards the wall, which can be seen in equation 8.3, and the results are shown in figure 8.5.

$$\frac{1}{\frac{T}{T_h}} = (0.104\frac{R}{(1.5-x)} - 0.0952)(\frac{V}{vh})^2 - 0.171\frac{R}{(1.5-x)} + 1.02 \tag{8.3}$$

$$\tag{8.4}$$

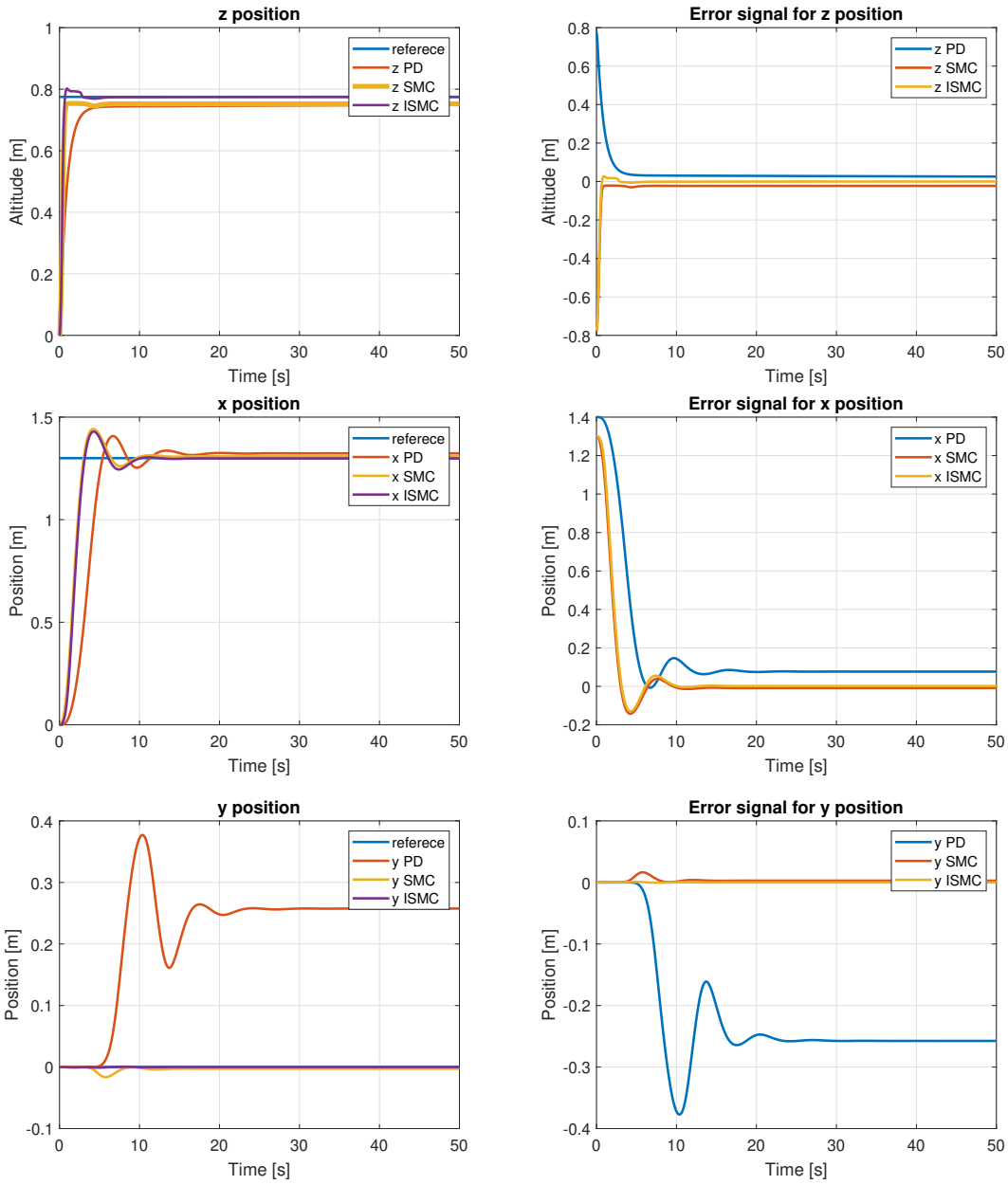


Figure 8.5: Wall effects affecting two rotors negatively with the opposite effect of the factor shown in figure 8.2

Review of Wall Effects

The wall effects should only affect the x and y position, where the altitude seems to have no challenge reaching a distance of 0.775m. This test can observe how fast the drone reaches a certain height and categorise the test with step information. This test shows again that the linear control law is slower than the sliding controls but performs acceptable in z and x position, only with a small, steady-state error in both coordinates. Instead, in the y position, the PD control is highly affected by the wall, and the drone is dragged off course to reach a sizeable steady-state error of 0.25m. This will be an inoperable outcome of controller performance. Instead, the nonlinear controllers will perform great in this test where settling time and steady-state error is almost equal, as seen in table 8.3. The challenge for these controllers will be the overshoot when nearing the wall, as a closer distance than 0.15m can result in a collision with the wall, where the reference is placed at 1.35m for SMC and ISMC. On the upper hand, the PD controller

has a reference of 1.4m which means it can try closing in on the wall, but the steady-state error sets it to perform equally with the nonlinear controllers. When looking at the error for x position, the PD control law does not overshoot the reference, meaning that collision with the wall is avoided using PD control. The ISMC control law performs best in this test when comparing steady-state error, and also it keeps the drone steady in y position when nearing the wall, where SMC has a small oscillation. Overshoot, settling time, and steady-state errors will be compared in table 8.3.

	Overshoot	Settling time	Steady state error
PD z	0%	3.5s	0.04m
PD x	0%	9.2s	0.9m
PD y	0.37m	20s	0.25m
SMC z	0%	0.13s	0.015m
SMC x	8.8%	6.1s	0.004m
SMC y	0m	6.2s	0.003m
ISMC z	1.2%	0.15s	0m
ISMC x	8.7%	6s	0m
ISMC y	0m	6.3s	0m

Table 8.3

8.2.4 Comparison of Control with Wind Disturbances

This section features the effects of a sinusoidal wind disturbance. Section 6 shows results of PD and SMC controllers simulated with a sinusoidal wind disturbance and reacting to it. The results showed significant impacts on both control laws but with more stability using SMC. The case here will include a four-meter movement in the x and y position while trying to reach the sinusoidal reference for altitude. While choosing these references, ground effects are turned on for all propellers, while the drone now is exposed to a sinusoidal wind disturbance. This test should illustrate how difficult it can be when exposed to several disturbances. The tests can be seen in figure 8.6.

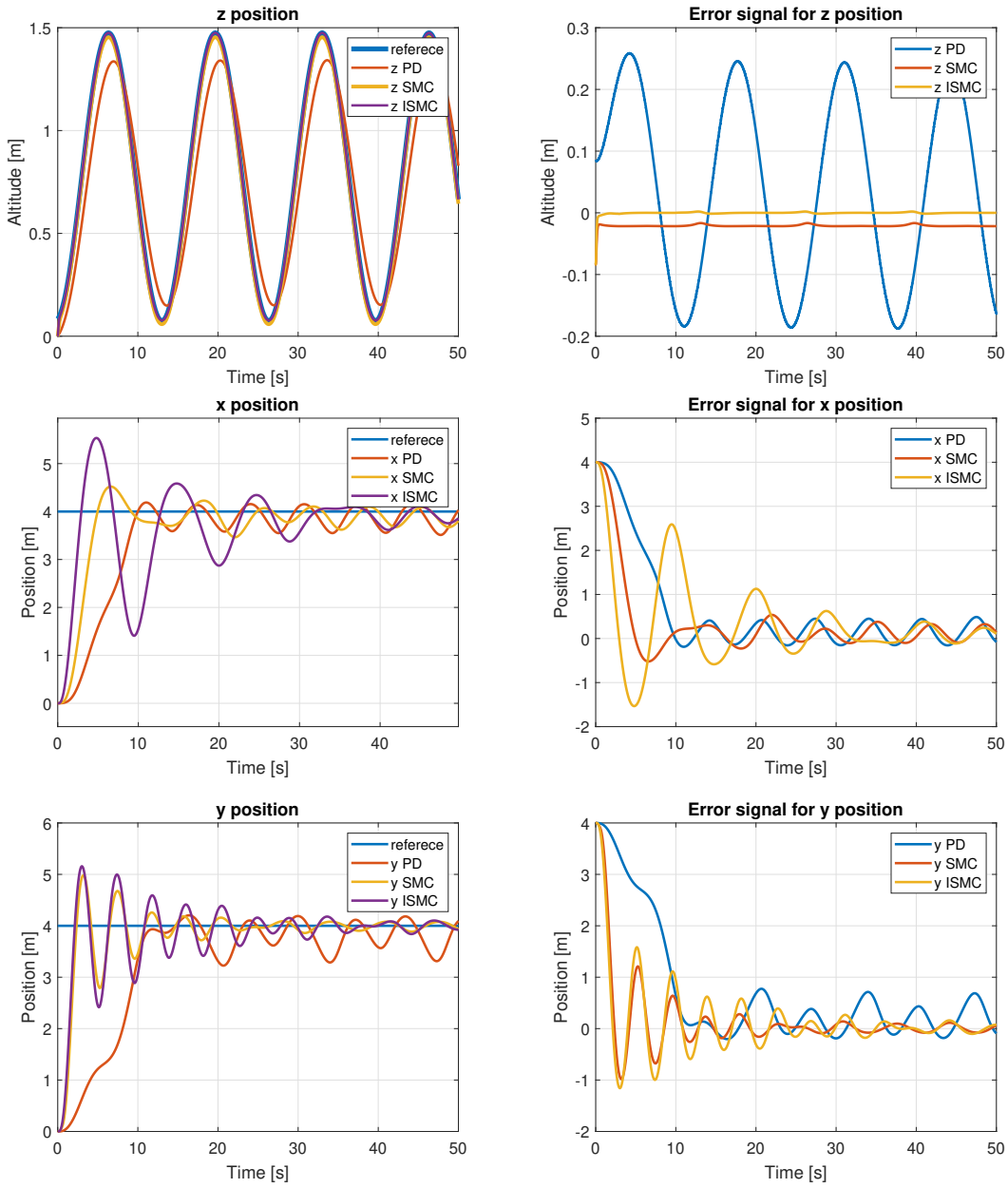


Figure 8.6: Wind effects affecting the quadcopter, while exposed to ground effect and trying to follow a sinusoidal reference in altitude

This test is the most requiring performance test, which results in oscillations from all controllers. This test diverges from the previous tests as the linear PD control performs similar to the nonlinear controllers. Every control method has its advantages in this test, as the PD controller generates the best largest error but worst oscillation and slowest rise time, while SMC performs faster with a little larger error but smaller settled oscillations. Last, the ISMC performs worst regarding largest errors but settles with the lowest oscillations and has the fastest rise time. Again, the z position behaves as in the previous test with a sinusoidal altitude reference and will not be further discussed. All controllers are categorised unusable in this case, where the performances can be seen in table 8.4.

	Settled oscillations	Largest error	Rise time
PD x	0.3m	0.3m	9.6s
PD y	0.83m	0.83m	10.2s
SMC x	0.27m	0.41m	4.7s
SMC y	0.16m	1.12m	3.2s
ISMC x	0.24m	1.37m	3.1s
ISMC y	0.17m	1.43m	3.2s

Table 8.4

Discussion 9

Contents

This chapter discusses the results simulated in chapter 8 and elaborates on the strengths and weaknesses of each control theory. Every test will be discussed in separate sections throughout this chapter.

This thesis illustrates how a quadcopter reacts when exposed to ground, ceiling, wall, wind disturbances and noise and how different control laws affect the results. The report is created with knowledge from earlier studies merging the manufacturers' data with studied data to create a realistic drone model. Further data is gathered from several papers and studies to create natural disturbances.

To test the controller reactions when the drone is exposed to the analysed disturbances, three control strategies have been created to exhibit their actions on the drone, while an extended Kalman filter is used to diminish the noise in the system. The controllers are illustrated theoretically and further tuned through a Bayesian optimisation algorithm that provides a global minimum. This tuning method may provide overshoot as it finds the global minimum within a specific period, where overshoot is generated. This section will then debate the results shown in section 8 and the respective controllers handling of the drone.

Ground and Ceiling Effects

From the results in section 8.2.1, SMC and ISMC showed minor errors when following the altitude reference, where also robustness against ground and ceiling effects occurs. The controllers will only oscillate below one centimetre when affected by these effects. The PD controller showed a slow response and could not keep up with the reference resulting in significant errors in position as it both lags in time and amplitude. This result displays how standard linear control may be easy to implement and work but may lead to unwanted results, which occurs multiple times in the results section. ISMC provides the best altitude tracking with zero steady-state error and the most negligible oscillations when the ground and ceiling are affecting the drone.

All controllers provide stable control with a difference in overshoot, steady-state error and settling time when looking at the horizontal positions. The SMC controller performed best for the position control, providing the fastest settling time while also achieving relative low overshoot and steady-state errors. Only SMC and ISMC are chosen acceptable regarding this test as they achieve the goals in section 1.4, where PD control displays a too large error in altitude control.

Partial Ground Effects

The partial ground effects shown in section 8.2.2 illustrated a reaction to the x and y position when exposed to a disturbance. In this test, the controllers performed identical to previous results in altitude and only x and y position control is further discussed.

When exposing a quadcopter to an uneven disturbance, a position reaction occurs, showcased in this test. All control laws generate stable performance, where the drone always gets back on track after the partial ground effect is affected. The essential outcome from this test concerns how robust the x and y positions are to the exposure effect. The linear controller reacts with the most considerable oscillation from this effect, with an error of 0.64m. This indicates that PD control is the worst in this test and is considered incompetent. However, also the SMC controllers do not keep the drone within a desired distance from the reference, as it oscillates up to 0.06 meters from the reference, and the goal stated in section 1.4 is a maximum of 0.03m from the reference. So as all controllers oscillate when exposed to this disturbance, only the ISMC controller is chosen acceptable with a maximum of 0.01m error in total. Also, the ISMC demonstrate great robustness against this effect.

Wall Effects

This test should affect the quadcopter reversed of partial ground effects, meaning it should lower the thrust from two rotors instead of raising it. This test is performed to see if the drone could close in on a wall where the effect further drags the drone towards the wall without colliding. This test is only interesting considering the x and y coordinates as these are the only affected parts.

All controllers provide stable control in this test, while the linear control has the least resilience to the exposure effect. The x position has a steady-state error of 0.09m, where it in the y position goes up to 0.25m, meaning it again does not meet the requirements. The SMC and ISMC perform almost equal where a small change exists in overshoot for ISMC in z position, but ISMC provides zero steady-state error through all positions. Also, a minor oscillation occurs for SMC control in y position when reaching the wall, giving the ISMC the advantages in this test. For wall effects, SMC and ISMC are acceptable for a quadcopter as they stay within the objectives.

Wind Disturbance

This test included ground effects along with a sinusoidal wind disturbance to expose the controller for extreme disturbances. As described in section 8.2.4, the z position behaves as in earlier results and will not further be discussed.

All controllers did experience difficulties when counteracting the effects from these disturbances, which resulted in significant and fast oscillations. Settled oscillations, largest errors and rise time can be found in table 8.4 for the three controllers, which could determine which one performs best. The results show that no controlling law can manage to achieve the goals from 1.4, categorising all theories unacceptable concerning the objectives. The SMC controller did react best comparing the results in table 8.4, where it has average errors while still behaving fast and with few oscillations. In the case of these effects, either a more robust control algorithm has to be made to behave acceptably, or some estimations of the disturbances could help the drone react before being exposed to these disturbances.

Conclusion 10

Can a quadcopter be controlled to withstand ground, ceiling and wall effects without colliding with either object and can it further behave stable when exposed to wind disturbances? How will different control laws affect the results, and can they achieve the objectives?

While the number of commercial drones expands every year, the understanding and knowledge need to be intensified. Drones seem to be used in numerous fields, accomplishing countless various tasks. To achieve the user's goals for every assignment, a great understanding of the subject and its contents must be studied. Further, how the drone reacts to the challenges it is exposed to should be examined.

This thesis investigates a drone exposed to ground, ceiling, wall, and wind effects, following references desired and affected by noise. These effects seem crucial to counteract if drones should advance their interest in business. The drone used to model in this thesis is an AscTec Hummingbird, where reliable results should be achieved because the model is built based on a real drone. Sensor noise is achieved from earlier studies investigating the noise occurring in a drone approximately the same size. All effects were also accomplished with knowledge from several earlier studies investigating and testing the responses of these effects. All this theory from earlier studies culminates in a model demonstrating the exposed effects while presumed to act realistic and usable if tried on a quadcopter with minor changes.

The noise implemented in the system is almost eliminated by an implementation of an Extended Kalman filter represented in figure 8.1, which smooths out the signal by comparing a predicted state with the measured, finding an optimal state estimation.

Three control theories are investigated and used to control the drone in a test, where the effects are acting on the quadcopter. In this thesis, the three different control strategies are PD, SMC and ISMC. These have been created and compared to visualise the differences between standard linear control and more advanced nonlinear control. These control theories are theoretically explained in sections 5 and 7.3, from where tuning is needed. To obtain optimal performance regarding control, a hyperparameter optimisation algorithm called Bayesian optimisation is created in section 6 and used to find global minimums for all controllers. This algorithm delivered usable controllers ready for tests.

In section 8 four tests are simulated comparing these controller responses to the respective effects. From the tests simulated, it can be concluded that the nonlinear controllers perform best regarding ground, ceiling and wall effects. Here ISCM obtains the lowest steady-state error for all tests while including the quickest responses. It also performed most robust against the effects. However, using the Bayesian strategy for tuning controllers creates a fast rise time while simultaneously generating overshoot for the ISMC. This overshoot could be minimised by choosing a different tuning strategy or self-tuning after the optimisation strategy, as the lowest error over time may be achieved with the algorithm. However,

overshoot comes with being optimal in the Bayesian Optimisation strategy.

The PD controller is categorised as unsatisfactory, as it fails to fulfil the requirements set in the objectives in section 1.4. All controllers failed to behave satisfactorily regarding the wind disturbance presented in figure 8.6, as significant errors are achieved, and oscillations occur through the test span. Instead, the ISMC showed promising results for ground, partial ground, ceiling and wall effects strongly followed by the SMC. The performance could be further optimised by self-tuning the control so overshoot can be minimised. Otherwise, stability could be achieved when exposed to wind by combining the control methods or expanding to a more advanced strategy. Otherwise, an estimation of the wind disturbance could help the quadcopter counteract the disturbance.

All controllers were capable of take-off and land. SMC and ISMC did reach stability for ground, ceiling, partial ground, and wall effects, while PD control oscillated when affected to partial ground effects but did also reach stability for ground, ceiling and wall effects. No control law achieved stability when exposed to wind disturbances.

This thesis can conclude how crucial robust control is when using drones exposed to different disturbances. The thesis tests several objective effects, while further could be tested with more advanced and robust control theories. It is concluded that nonlinear control is more robust than linear control and that linear control did not live up to the objectives in this thesis, making nonlinear control the only usable method case with these objectives and goals.

Bibliography

[1] A. R. Earls, "drone (uav)," *TechTarget*, 2019.

[2] S. E. S. A. Research, "European atm mater plan: Roadmap for the safe integration of drones into all classes of airspace)." `https://www.sesarju.eu/masterplan`, 2018.

[3] A. Chapman, "Australian drone magazine," *Australian Drone magazine*, 2016.

[4] T. S. Pedersen, "Model and control of a large scaled quadcopter," *Aalborg University, Department of Energy-technology*, 2021.

[5] M. A. Boon, A. P. Drijfhout, and S. Tesfamichael, "Comparison of a fixed-wing and multi-rotor uav for environmental mapping applications: A case study," *ResearchGate*, 2017.

[6] M. Satell, "Ultimate list of drone stats for 2021," *Philly By Air*, 2021.

[7] T. I. A. og Erhvervsfremme, "Flere dronevirksomheder i danmark - status, potientialer, lovgivning og marked," *Teknologisk Institut*, 2018.

[8] T. I. A. og Erhvervsfremme, "Droner nu og i fremtiden, resultatkontrakt: Drones-as-a-tool," *Teknologisk Institut*, 2019.

[9] P. Sanchez-Cuevas, G. Heredia, and A. Ollero, "Characterization of the aerodynamic ground effect and its influence in multirotor control," *Hindawi*, 2017.

[10] A. Matus-Vargas, G. Rodriguez-Gomez, and J. Martinez-Carranza, "Ground effect on rotorcraft unmanned aerial vehicles: a review," *Springer*, 2020.

[11] J. Cai, "Changes in propeller performance due to ground and partial ground proximity," *University of Dayton, Ohio*, 2020.

[12] S. Gao, C. D. Franco, D. Carter, D. Quinn, and N. Bezzo, "Exploiting ground and ceiling effects on autonomous uav motion planning," *IEEE*, 2019.

[13] D. C. Robinson, H. Chung, and K. Ryan, "Computational investigation of micro rotorcraft near-wall hovering aerodynamics," *IEEE*, 2014.

[14] A. Zulu and S. John, "A review of control algorithms for autonomous quadrotors," *SCIRP*, 2014.

[15] M. COMPRO, "Ulm electric motor reb-30." `https://www.aeroexpo.online/prod/mgm-compro/product-171210-35928.html`, 2021.

[16] D. Muzar and E. Lanteigne, "Experimental characterization of brushless dc motors and propellers for flight application," *CSME*, 2016.

[17] D. R. Pedersen, D. Ársælsson, and J. F. Ankersen, "Hovering control and ground effect compensation of a quadcopter," *AAU*, 2021.

[18] J. L. Meriam, L. G. Kraige, and J. N. Bolton, *Engineering Mechanics Dynamics*. Wiley, Eight Edition.

[19] T. Hinzmann, "Adaptive rate control for multirotor uavs," *ETH*, 2014.

[20] X. Xiang, Z. Wang, Z. Mo, G. Chen, E. Blasch, and K. Pham, "Wind field estimation through autonomous quadcopter avionics," *Research Gate*, 2016.

[21] W. Neeley, "Design and development of a high-performance quadrotor control architecture based on feedback linearization," *Engineering ETDs*, 2016.

[22] X. Kan, J. Thomas, H. Teng, H. G. Tanner, V. Kumar, and K. Karydis, "Analysis of ground effect for small-scale uavs in forward flight," *IEEE*, 2019.

[23] S. A. Conyers, "Empirical evaluation of ground, ceiling, and wall effect for small-scale rotorcraft," *Digital Commons @ DU*, 2019.

[24] I. C. Cheeseman and W. E. Bennett, "The effect of the ground on a helicopter rotor in forward flight," *R. & M. No. 3021*, 1955.

[25] P. W. Sui, N. Chan, S. Lee, and Z. Kong, "Mitigating ground effect on mini quadcopters with model reference adaptive control," *Springer*, 2019.

[26] W. Neeley, "Design and development of a high-performance quadrotor control architecture based on feedback linearization," *ENGINEERING ETDS*, 2016.

[27] Productz, "Asctec hummingbird." `https://productz.com/en/asctec-hummingbird/p/xKp1`, 2021.

[28] I. Grujic and R. Nilsson, "Model-based development and evaluation of control for complex multi-domain systems: Attitude control for a quadrotor uav," *Aarhus University*, 2016.

[29] J.-J. E. Slotine and W. Li, *Applied Nonlinear Control*. Prentice Hall, Eight Edition.

[30] T. Yu and H. Zhu, "Hyper-parameter optimization: A review of algorithms and applications," *Researchgate*, 2020.

[31] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *IEEE*, 2016.

[32] E. Brochu, V. M. Cora, and N. de Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv:1012.2599v1 [cs.LG]*, 2010.

[33] R. Isermann, "Fault-diagnosis systems, an introduction from fault detection to fault tolerance," *Springer*, 206.

Coriolis term

Coriolis term can be represented as a matrix of 3x3. This term exists when an object is in motion within a reference frame and with respect to an inertial frame. Then it is about Newton's laws of motion for an object in a non-accelerating frame. When this object is transformed into a rotating frame, the laws of Coriolis and centrifugal accelerations appear.

Coriolis matrix can be seen in equation A.1.

$$\begin{bmatrix} C11 & C12 & C13 \\ C21 & C22 & C23 \\ C31 & C32 & C33 \end{bmatrix} \tag{A.1}$$

From here, the different terms can be derived in the following equations:

$$\begin{aligned}
C11 &= 0 \\
C12 &= (I_{yy} - I_{zz})(\dot{\theta}c(\phi)s(\phi) + \dot{\psi}s(\phi)^2c(\theta)) + (I_{zz} - I_{yy})\dot{\psi}c(\phi)^2c(\theta) - I_{xx}\dot{\psi}c(\theta) \\
C13 &= (I_{zz} - I_{yy})\dot{\psi}c(\phi)s(\phi)c(\theta)^2 \\
C21 &= (I_{zz} - I_{yy})(\dot{\theta}c(\phi)s(\phi) + \dot{\psi}s(\phi)^2c(\theta) + (I_{yy} - I_{zz})\dot{\psi}c(\phi)^2c(\theta) + I_{xx}\dot{\psi}c(\theta) \\
C22 &= (I_{zz} - I_{yy})\dot{\phi}c(\phi)s(\phi) \\
C23 &= -I_{xx}\dot{\psi}s(\theta)c(\theta) + I_{yy}\dot{\psi}s(\phi)^2c(\theta)s(\theta) + I_{zz}\dot{\psi}c(\phi)^2s(\theta)c(\theta) \\
C31 &= (I_{yy} - I_{zz})\dot{\psi}c(\phi)s(\phi)c(\theta)^2 - I_{xx}\dot{\theta}c(\theta) \\
C32 &= (I_{zz} - I_{yy})(\dot{\theta}c(\phi)s(\phi)s(\theta) + \dot{\phi}s(\phi)^2c(\theta)) + (I_{yy} - I_{zz})\dot{\phi}c(\phi)^2c(\theta) \\
&\quad + I_{xx}\dot{\psi}c(\theta)s(\theta) - I_{yy}\dot{\psi}s(\phi)^2c(\theta)s(\theta) - I_{zz}\dot{\psi}c(\phi)^2c(\theta)s(\theta) \\
C33 &= (I_{yy} - I_{zz})\dot{\phi}c(\phi)s(\phi)c(\theta)^2 - I_{yy}\dot{\theta}s(\phi)^2c(\theta)s(\theta) - I_{zz}\dot{\theta}c(\phi)^2c(\theta)s(\theta) + I_{xx}\dot{\theta}c(\theta)s(\theta)
\end{aligned}$$

Bayesian Optimisation 1

B

The code used to generate optimised controllers using a Bayesian optimisation algorithm is presented in this appendix. The example shown in figure B.1 illustrates how the code is built with Matlab commands and used to obtain usable results. In this example, two linear PD controllers are tested with a P and D constant for the given four parameters. This example can be rewritten for the nonlinear controllers by replacing the four PD parameters with the nonlinear parameters. This algorithm finds a global minimum for every time it runs a test, which it then continuously uses in the next run, meaning that few tests can be run with this algorithm if a good starting guess is chosen.

```
%Bayesian optimisation by: Tinus Stengaard Pedersen
clc; clear all; close all;
%% 20 runs with a start guess for PD values of two controllers
Samples = 20;

Prange = 1000;
Drange = 1000;
X_best=[g1 g2 g3 g4];
for i = 1:Samples

    P_rolle = X_best(1,1);
    D_rolle = X_best(1,2);
    P_pitche = X_best(1,3);
    D_pitche = X_best(1,4);

    % PD
    eps(i,:) = [P_roll D_roll P_pitch D_pitch ];
    sim('Testsmcangles',10);
    Cost(:,:) = [ans.cost.Data(:,:)]; %Raw Cost values

    for ii = 1:length(Cost(:,1,1))
        Jcost(ii) = Cost(ii,1) + Cost(ii,2); %Weighted cost
    end

    Jc(i,1) = sum(Jcost); %cost

%Guassion random process model:
gprMdl = fitrgp(eps,Jc,'KernelFunction','squaredexponential');

%Hyperparameters:
P_roll = optimizableVariable('Proll',[0,5],'Type','real');
D_roll = optimizableVariable('Droll',[0,5],'Type','real');
P_pitch = optimizableVariable('Ppitch',[0,5],'Type','real');
D_pitch = optimizableVariable('Dpitch',[0,50],'Type','real');

% Vars
vars = [P_roll D_roll P_pitch D_pitch];

bayesObject = bayesopt(@(tbl)mdlfun(tbl,gprMdl),vars,'Verbose',1,...
    'AcquisitionFunctionName','expected-improvement-plus',...
    'PlotFcn',@plotMinObjective);
% Best prediction


X_best = [bayesObject.XAtMinEstimatedObjective.Proll
          bayesObject.XAtMinEstimatedObjective.Droll
          bayesObject.XAtMinEstimatedObjective.Ppitch
          bayesObject.XAtMinEstimatedObjective.Dpitch]
end

function f = mdlfun(tbl,gprMdl) %Re-evaluating the surrogate model
P_roll = tbl.Proll;
D_roll = tbl.Droll;
P_pitch = tbl.Ppitch;
D_pitch = tbl.Dpitch;

vars = [P_roll D_roll P_pitch D_pitch];

f = predict(gprMdl,vars);
end
```

Figure B.1: Matlab code for running Bayesian optimisation on linear PD controllers

Bayesian Optimisation 2

C

Here the Bayesian optimisation algorithm searches a considerable span of value to look for global minimums throughout the chosen range. This algorithm can provide stable values to use as start guesses for the BO in appendix B. There will be no starting guesses in this example, and the algorithm will not use the last test run to look for a more prominent next run. Instead, rand is used to generate random values for the controllers, which is run several times until a satisfying starting guess is provided. The rand command gives a value between [0-1] for every run, multiplied by the range set. This algorithm can be seen in C.1 again with the build-in Matlab commands.

```
%Bayesian optimisation by: Tinus Stengaard Pedersen
clc; clear all; close all;
%% 20 runs with a start guess for PD values of two controllers
Samples = 20;

Prange = 1000;
Drange = 1000;

for i = 1:Samples

    P_roll = 0 + rand*Prange;
    D_roll = 0 + rand*Drange;

    P_pitch = rand*Prange;
    D_pitch = rand*Drange;

    % PD
    eps(i,:) = [P_roll D_roll P_pitch D_pitch ];
    sim('Testsmcangles',10);
    Cost(:,:) = [ans.cost.Data(:,:)]; %Raw Cost values

    for ii = 1:length(Cost(:,1,1))
        Jcost(ii) = Cost(ii,1) + Cost(ii,2); %Weighted cost
    end

    Jc(i,1) = sum(Jcost); %cost
end
%Guassion random process model:
gprMdl = fitrgp(eps,Jc,'KernelFunction','squaredexponential');

%Hyperparameters:
P_roll = optimizableVariable('Proll',[0,5],'Type','real');
D_roll = optimizableVariable('Droll',[0,5],'Type','real');
P_pitch = optimizableVariable('Ppitch',[0,5],'Type','real');
D_pitch = optimizableVariable('Dpitch',[0,50],'Type','real');

% Vars
vars = [P_roll D_roll P_pitch D_pitch];

bayesObject = bayesopt(@(tbl)mdlfun(tbl,gprMdl),vars,'Verbose',1,...
    'AcquisitionFunctionName','expected-improvement-plus',...
    'PlotFcn',@plotMinObjective);


% Best prediction
X_best = [bayesObject.XAtMinEstimatedObjective.Proll
          bayesObject.XAtMinEstimatedObjective.Droll
          bayesObject.XAtMinEstimatedObjective.Ppitch
          bayesObject.XAtMinEstimatedObjective.Dpitch]


function f = mdlfun(tbl,gprMdl) %Re-evaluating the surrogate model
P_roll = tbl.Proll;
D_roll = tbl.Droll;
P_pitch = tbl.Ppitch;
D_pitch = tbl.Dpitch;

vars = [P_roll D_roll P_pitch D_pitch];

f = predict(gprMdl,vars);
end
```

Figure C.1: Matlab code for running Bayesian optimisation on linear PD controllers