# System identification and optimal control of a small-scale unmanned helicopter

**MC Terblanche**

**21057567**

Dissertation submitted in fulfilment of the requirements for the degree *Magister in Computer and Electronic Engineering* at the Potchefstroom Campus of the North-West University

Supervisor: Dr KR Uren

Co supervisor Prof G van Schoor

May 2014

# Declaration

I, Marthinus Christoffel Terblanche, hereby declare that the dissertation entitled "**System identification and optimal control of a small-scale unmanned helicopter**" is my own original work and has not already been submitted to any other university or institution for examination.

_____.

Marthinus Christoffel Terblanche

Student number: 21057567

Signed on the 27th day of April 2014 at Johannesburg.

# Acknowledgements

I would like to thank the love of my life, Marli, for keeping me from work, but also giving me the time to complete this chapter of my life. I greatly appreciate the feedback and inspiration from my editor and sister. I would like to thank my parents for their guidance throughout my life. Thanks to my fellow students for always being willing to help. Thank you to my study leaders Dr. Kenny Uren and Prof. George van Schoor, for their inputs and advice throughout the study.

I would like thank the NRF and THRIP for their financial support, without which this study would not have been possible[1].

[1] This work is based on the research supported in part by the National Research Foundation of South Africa (UID: 80020); The grant holder acknowledges that opinions, findings and conclusions or recommendations expressed in any publication generated by the NRF supported research are that of the authors, and that the NRF accepts no liability whatsoever in this regard.

# Abstract

The use of rotary winged unmanned aerial vehicles in military and civilian applications is rapidly increasing. The primary objective of this study is to develop an automatic flight control system for a radio controlled (RC) helicopter. There is a need for a simple, easy to use methodology to develop automatic flight controllers for first-flight. In order to make the work accessible to new research groups without physical helicopter platforms, a simulation environment is created for validation.

The size 30 RC helicopter in AeroSIMRC is treated as the final target platform. A grey box, time-domain system identification method is used to estimate a linear state space model that operates around hover. Identifying the unknown parameters in the model is highly dependent on the initial guess values and the input data. The model is divided into subsystems to make estimation possible.

A cascaded controller approach is followed. The helicopter's fast angular dynamics are separated from the slower translational dynamics. A linear quadratic regulator is used to control the helicopter's attitude dynamics. An optimised PID outer-loop generates attitude commands from a given inertial position trajectory. The PID controllers are optimised using a simplex search method. An observer estimates the unmeasured states such as blade flapping. The controller is developed in Simulink®, and a plug-in written for AeroSIMRC enables Simulink® to control the simulator through a UDP interface to validate the model and controller.

The identified state space model is able to accurately model the flight data from the simulator. The controllers perform well, keeping the helicopter stable even in the presence of considerable disturbances. The attitude controller's performance is validated using an aeronautical design standard (ADS-33E-PRF) for handling qualities. The trajectory tracking is validated in a series of simulator flight tests. The linear controller is able to sustain stable flight in constant winds of up to 60% of the helicopter's maximum airspeed.

*Keywords: helicopter, optimal linear control, LQR, system identification, simulation*

# Contents

# List of figures

# List of tables

# Nomenclature

## Subscripts

| | |
|---|---|
| *long* | Longitudinal |
| *lat* | Lateral |
| *col* | Collective |
| *ped* | Pedals |
| *BF* | Body-fixed frame |
| *SF* | Spatial-fixed frame |
| *EF* | Earth-fixed frame |

## Symbols

| Symbol | Unit | Description/Quantity |
|---|---|---|
| $u$ | m/s | Longitudinal velocity (part of **v**) |
| $v$ | m/s | Lateral velocity (part of **v**) |
| $w$ | m/s | Vertical or heave velocity (part of **v**) |
| $p$ | rad/s | Roll rate in the BF (part of **ω**) |
| $q$ | rad/s | Pitch rate in the BF (part of **ω**) |
| $r$ | rad/s | Yaw rate in the BF (part of **ω**) |
| $\phi$ | rad | Roll angle (part of **Θ**) |
| $\theta$ | rad | Pitch angle (part of **Θ**) |
| $\psi$ | rad | Yaw angle (part of **Θ**) |
| $a$ | rad | Longitudinal angle of TPP |
| $b$ | rad | Lateral angle of TPP |
| $\delta_{long}$ | - | Longitudinal cyclic pitch control input |
| $\delta_{lat}$ | - | Lateral cyclic pitch control input |
| $\delta_{col}$ | - | Collective pitch control input |
| $\delta_{ped}$ | - | Pedal control input |
| $\tau_f$ | s | Rotor time constant |
| $g$ | $m/s^2$ | Gravitational acceleration |
| $X$ | N | Force component acting on the CG along the *x*-axis (part of **F**) |
| $Y$ | N | Force component acting on the CG along the *y*-axis (part of **F**) |
| $Z$ | N | Force component acting on the CG along the *z*-axis (part of **F**) |

| | | |
|---|---|---|
| $L$ | N.m | Torque along the $x$-axis (part of $\boldsymbol{\tau}$) |
| $M$ | N.m | Torque along the $y$-axis (part of $\boldsymbol{\tau}$) |
| $N$ | N.m | Torque along the $z$-axis (part of $\boldsymbol{\tau}$) |
| $x$ | m | Northern position in the inertial reference frame (part of **x)** |
| $y$ | m | Eastern position in the inertial reference frame (part of **x)** |
| $z$ | m | Downwards position in the inertial reference frame (part of **x)** |
| $m$ | kg | Mass |
| $\gamma$ | - | Rotor lock number |
| $\Omega$ | r/min | Rotor speed |
| $\boldsymbol{\omega}$ | rad/s | Angular velocity vector in the BF |
| $\boldsymbol{\Theta}$ | rad | Euler angle vector |
| $\mathbf{I}$ | $kg.m^2$ | Inertia matrix |
| $\mathbf{R}$ | - | Position rotation matrix |
| $\mathbf{P}$ | - | Velocity rotation matrix |
| $\mathbf{F}$ | N | Vector of force acting on the CG |
| $\boldsymbol{\tau}$ | N.m | Vector of torque acting on the CG |
| $\mathbf{v}$ | m/s | Velocity vector |
| $\mathbf{H}$ | N.m.s | Angular momentum vector |
| $\mathbf{J}$ | - | Cost function matrix |
| $\mathbf{Q}$ | - | State weighting matrix |
| $\mathbf{R}$ | - | Input weighing matrix |

## Abbreviations

| | |
|---|---|
| ACAH | Attitude Command Attitude Hold |
| ADS | Aeronautical Design Standards |
| AFCS | Automatic Flight Control System |
| ALFUS | Autonomy Levels for Unmanned Systems |
| ANN | Artificial Neural Network |
| AQ | Attitude Quickness |
| BF | Body-Fixed frame |
| CG | Centre of Gravity |

| | |
|---|---|
| CIFER | Comprehensive Identification from FrEquency Responses |
| COTS | Commercial Off The Shelf |
| DLL | Dynamic Link Library |
| DOF | Degrees Of Freedom |
| EF | Earth-Fixed Frame |
| FCS | Flight Control System |
| FFT | Fast Fourier Transform |
| GM | Gain Margin |
| GNC | Guidance, Navigation, and Control |
| HITL | Hardware-In-The-Loop |
| IARC | International Aerial Robotics Competition |
| ITAE | Integral of Time and Absolute Error |
| LQ | Linear Quadratic |
| LQG | Linear Quadratic Gaussian |
| LQR | Linear Quadratic Control |
| MIMO | Multi-Input Multi-Output |
| MMAC | Multiple Model Adaptive Control |
| MOSCA | MOdeling for flight Simulation and Control Analysis |
| NRMSE | Normalised Root Mean Square Error |
| PCA | Principle Component Analysis |
| PCH | Pseudo Control Hedging |
| PEM | Prediction-Error Method |
| PID | Proportional, Integral, and Derivative |
| PIO | Pilot Induced Oscillation |
| PM | Phase Margin |
| RBE | Rigid Body Equations |

| | |
|---|---|
| RC | Radio Control |
| RCDH | Rate Command Direction Hold |
| RGA | Relative Gain Array |
| ROI | Return On Investment |
| RSD | Relative Standard Deviation |
| SAS | Stability Augmentation System |
| SF | Spatial Frame |
| SIDPAC | System IDentification Programs for AirCraft |
| SISO | Single-Input Single-Output |
| TPP | Tip Path Plane |
| THRIP | Technology and Human Resource Innovation Project |
| UAS | Unmanned Aerial Systems |
| UAV | Unmanned Aerial Vehicle |
| UDP | User Datagram Protocol |
| VTOL | Vertical Take-Off and Landing Capability |
| VTUAV | Vertical Take-Off Unmanned Aerial Vehicle |

# Chapter 1: Introduction

*This chapter starts with background on the relevance of the research. A problem statement is derived from the basic introduction to unmanned aerial systems, automatic flight control, and modelling. In addition, the objectives needed to solve the research problem are given, as well as the methodology followed to reach these objectives.*

## 1.1 Background

The past decade has seen a dramatic increase in military and civilian use of Unmanned Aerial Vehicles (UAVs) [1]. The UAV sector has the fastest growing rate in the international aerospace industry [2]. The US Department of Defence investment in Unmanned Aerial Systems (UAS) has grown from $284 million in 2000 to $4.1 billion in 2011, illustrating just how fast the industry is growing [3]. A UAV's main task is to perform dull, dangerous, and dirty tasks where human pilots have difficulty [4]. Figure 1.1 shows some examples of different unmanned helicopter sizes.

**a) Northrop Grumman Fire Scout (1,430 kg) [5]** **b) Yamaha RMAX (93 kg) [6]** **c) Stanford's XCell Tempest (5 kg) [7]**

**Figure 1.1: Military, civilian and research unmanned helicopters.**

Denel is a South African aerospace and defence company that already has a range of fixed wing UAVs. Denel Aviation has decided to expand their research to Rotary Winged UAVs (RWUAVs). In support of this initiative, Denel has initiated a collaborative research project with South African universities through the Technology and Human Resource Innovation Project (THRIP) fund for the development of a RWUAV Automatic Flight Control System (AFCS). The present study forms part of the first set of research on UAV Guidance, Navigation, and Control (GNC) at the university. Currently all work is being done in simulation until hardware has been completed. A logical place to start work on a helicopter is with a hover controller. This hover controller can then be expanded to a trajectory tracking controller. Once this point has been reached, the control and navigation can be improved to expand the flight envelope and improve the trajectory generation capabilities. The methodology is tested in a Simulink®/AeroSIMRC simulation environment, providing a fast and inexpensive way to validate controllers.

## 1.2 Unmanned Aerial System architecture

A typical UAS architecture is shown in Figure 1.2. It is necessary to give some background on UASs to demonstrate where this study fits in. The three main components of an UAS are Guidance, Navigation, and Control.

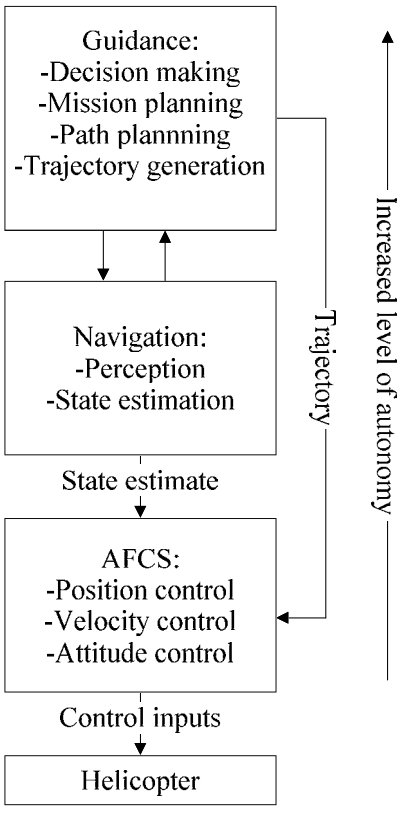



**Figure 1.2: Unmanned Aerial System Guidance, Navigation and Control. Adapted from [9]**

The AFCS has a flight trajectory as input and must manipulate the helicopter controls in order to stay on the trajectory. Navigation is concerned with estimating the current state of the vehicle and its environment. This state information is provided to the controller for feedback. The guidance system determines the flight plan of the helicopter. It considers mission goals and how to reach them. The guidance system will generate the trajectory for the controller to follow. This study is abstracted from guidance and navigation. It is assumed that the helicopter position and orientation are known. It is also assumed that mission goals have been determined, decisions made, and a trajectory to reach these goals has been generated. The flight control system needs to determine the appropriate helicopter states to follow the given trajectory. Guidance and navigation are separate from AFCS design and form their own research fields. Integrating these fields into a functional UAS is seen as an important goal for the research group.

According to a report by the National Institute of Standards and Technology (NIST) on the Autonomy Levels for Unmanned Systems (ALFUS) framework, the AFCS is placed at the 2nd lowest level [8]. Human piloted aircraft are at level 0 of ALFUS, since they have no autonomy. Each higher level of GNC builds upon the lower levels. The controller presented here is designed for ALFUS level 1, which is Automatic Flight Control.

The next level for the AFCS would be to make this controller adaptive to changes in the aircraft and its environment. It should then be improved up to a point where it can handle complex 3D trajectories. When this goal is reached, the flight controller becomes less important and guidance and navigation should become the focus.

Guidance and navigation have in fact become the focus for many of the mature research groups [9]. This trend can also be seen in the objectives of competitions that are designed to push the state-of-the-art, such as the International Aerial Robotics Competition (IARC) [10]. It is still necessary for a new research group to first reach this 1$^{st}$ level of autonomy before it can move on to the next levels.

## 1.3 Helicopter control

Helicopters pose a unique control-engineering problem. They are Multi-Input Multi-Output (MIMO) underactuated systems that are open loop unstable, have significant coupling, exhibit nonlinear dynamics, and are very difficult to model accurately [11], [12], [13], [14]. Still, considering all these challenges, Proportional-Integral-Derivative (PID) is the most commonly used control technique [9]. Typically, such a controller is designed by decoupling each axis as much as possible and designing a separate controller for each of these Single-Input Single-Output (SISO) systems. The highly acclaimed APM:Copter open-source autopilot uses this technique [15]. This

decoupled SISO approach is not only popular in unmanned helicopters, but also used in many full-scale helicopters that have a PID-based Stability Augmentation System (SAS) to help improve the aircraft's handling. Nonlinear and intelligent techniques have seen increased popularity in academic research, since they provide a way to improve performance. The problem with these systems is that they are difficult to implement outside of a simulation environment and have consequently not seen much use in real systems [14].

The helicopter's cross coupled nature can be addressed by a linear full-state feedback controller, without sacrificing too much of the simplicity associated with PID control. It is always preferred to use the simplest controller capable of meeting the design criteria. With the start of a new UAV research project, the primary focus is not on improving performance, or pushing the flight envelope, but ensuring safe and transparent control. LQR control is an optimal control technique that is ideally suited to this need, because it has a clear design procedure with guaranteed stability. It handles MIMO systems well and has an intuitive way of defining controller performance criteria. Optimal control has been used extensively in the aerospace industry, and is a proven methodology. All the advantages of LQR control can only be gained if the controller is derived from an accurate model of the helicopter.

## 1.4 System identification

Without a helicopter to weigh and to measure, it is not possible to obtain the necessary mechanical and aerodynamic parameters needed for accurate first principle modelling. When hardware is available for measuring, it is still a considerable problem, as the experimental setup is expensive and in many cases, wind tunnel data is needed [16]. It is more common to see first principle models being developed for nonlinear simulators. When a model is needed for control design, it is often advantageous to have a simpler structure. Achieving a high degree of accuracy from first principle modelling becomes more difficult as the process increases in complexity. System identification can be the ideal solution to these problems.

System identification gives the engineer a "first hand" experience with the aircraft. It facilitates an understanding of essential concepts and the dominant dynamics. Thanks to the amount of time spent working with the flight data, it is possible to gain a better understanding from system identification than from first principle modelling [17]. Grey box modelling uses all the knowledge gained from first principle modelling, but has the more data driven design process of system identification. Black box modelling does not provide any information on how the system that is being modelled works. On the other hand, with grey box modelling, the helicopter's physical characteristics are

identified. Such a model has a structure determined by first principle equations, where the coefficients of the differential equations are then estimated by some form of optimisation algorithm attempting to minimise the error between the model and the flight data. This also has advantages later on in the controller design. Physical insight can then be used to simplify the control problem. Grey box system identification leads to a model that is not only transparent, but also verifiable.

## 1.5 Problem statement

This research study aims to develop an optimal controller able to stabilise a simulated RC helicopter and autonomously follow specified trajectories. The performance of the controller is highly dependent on the accuracy of the model used. A sufficiently accurate model is needed for controller design. The control system should be validated in a Radio Controlled (RC) helicopter simulator.

Part of the problem is to provide new research groups with an inexpensive, easy to use methodology to set up an RWUAV test environment with a first-flight controller. This is specifically relevant to researchers that do not have the infrastructure for hardware validation.

## 1.6 Objectives

The following objectives support the problem statement:

- Decide on a suitable controller that is stable for hover and low speed flight, can account for the multivariable dynamics, and is simple enough for a first-flight implementation.
- Use an appropriate modelling technique to model the helicopter, specifically for controller design purposes.
- Design a trajectory following controller and verify it using the developed model.
- Develop an interface to enable controllers designed in Simulink®/Matlab® to control a Commercial Of The Shelf (COTS) helicopter simulator.
- Validate the controller in the simulation environment.

## 1.7 Methodology and chapter layout

The methodology also forms the structure for the chapter layout. A flow diagram of the methodology followed is shown in Figure 1.3. Next, the diagram is explained and where each part of the methodology fits into the dissertation is discussed.

First, a literature study is conducted in Chapter 2, to decide on a suitable controller architecture. Once the controller has been chosen, it is possible to identify a modelling technique that will match the controller requirements.

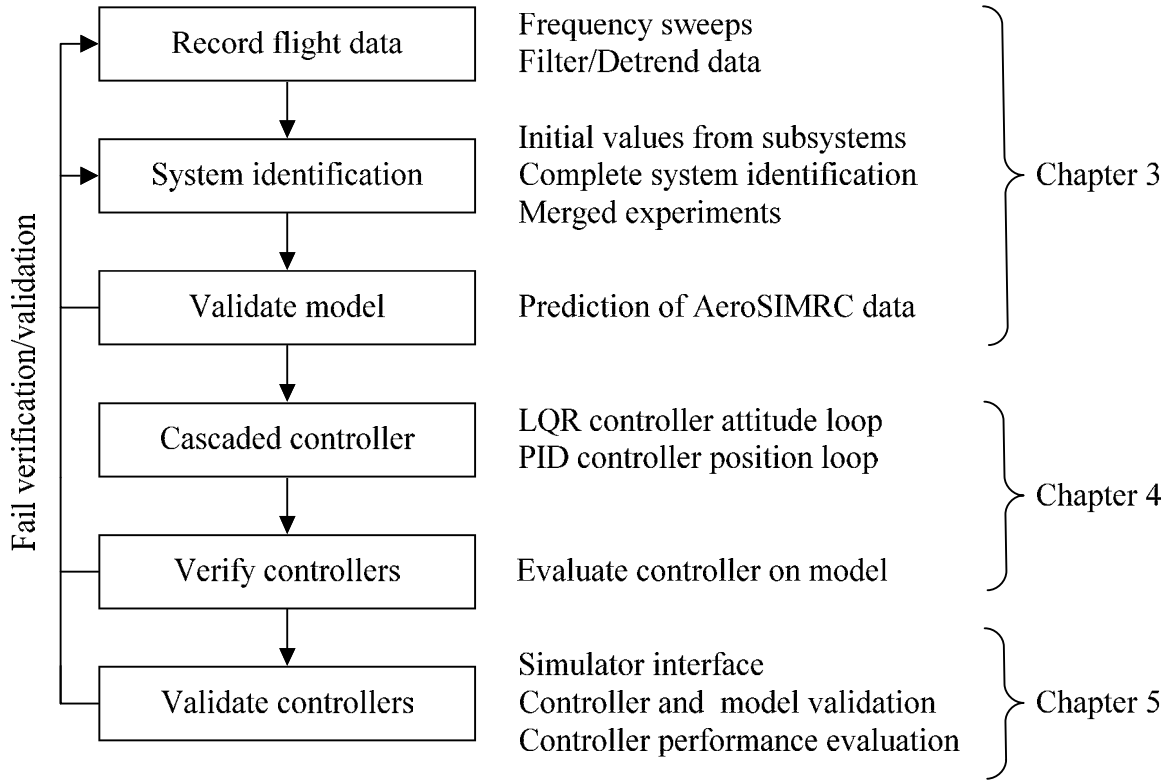



**Figure 1.3: Methodology followed to develop the flight controller**

Optimal linear quadratic control will be used for the faster helicopter attitude dynamics, and optimised PID controllers for the slower translational trajectory control. The LQR control methodology uses a state space model to calculate feedback gains. It is decided that system identification is the best option to obtain an accurate model of the helicopter.

Chapter 3 addresses modelling the helicopter. Helicopter flight data is needed to begin system identification. AeroSIMRC is used as the platform. Data is recorded for human piloted flight, with special frequency sweep inputs. The complete state space model is broken down into smaller subsystems to make identification possible. The modelling methodology is **verified** on a known state space model. The model is **validated** using a set of simulator flight data not used for estimation. The model's ability to predict the helicopter response is used as a performance metric. A User Datagram Protocol (UDP) interface is developed to communicate helicopter states and control inputs between Simulink® and AeroSIMRC.

In Chapter 4, the identified state space model is used to design a cascaded controller. The helicopter attitude and altitude is controlled with an LQR controller. The controller is **verified** on the state space model. Optimal PID controllers are developed to control the translation of the helicopter. The PID controllers provide reference inputs to the LQR controller. The PID controllers are optimised with a simplex search method. The complete control system is tested in Simulink® on the state space model.

The system model and controllers are validated in Chapter 5. The COTS simulator is seen as the final application of the control system. The controller is **validated** to determine if it can successfully control the helicopter and follow the specified trajectories. This would normally be done with practical flight tests. This study did not have access to a helicopter and the simulator is seen as the practical application. Chapter 6 gives a conclusion to the study and recommendations for future work.

# Chapter 2: Literature overview

*This study has three primary areas of interest: system identification, control, and simulation. These areas of interest are applied to an RC helicopter. To understand the approach followed in each section, some background on RC helicopters is first needed. The chapter will present a broad overview of previous research on each of the areas. From the overview, it should be apparent why system identification was chosen, why optimal control is used, and why a simulator is necessary.*

## 2.1 Flying a helicopter

It is important to first get an overview of helicopter flight and control before delving deeper into the technical aspects. This section explains the mechanics needed to fly an RC helicopter. The equations of motion, reference frames, and stability derivatives will be explained in Chapter 3.

A change in vertical speed is called the heave velocity. This is controlled with a lever next to the pilot, the collective pitch lever. When the helicopter turns, changing the direction the pilot is facing, it is called yawing. In a manned helicopter, this motion is controlled by foot pedals. The direction the helicopter is facing is its heading. The pilot controls the helicopter's roll and pitch with the cyclic pitch stick. Pushing forward on the cyclic pitches the helicopter forward, and causes longitudinal motion. Moving the cyclic to the right will cause a roll to the right, which is lateral motion [18].

With an RC helicopter, the pilot is on the ground and uses a remote control to fly the helicopter. Figure 2.1 shows how each gimbal on the pilot's remote affects the helicopters motion. The left gimbal is used to control collective pitch and heading. The right gimbal is used to control cyclic

pitch. The figure shows the on-axis responses; cross coupling will cause some significantly smaller off-axis responses.

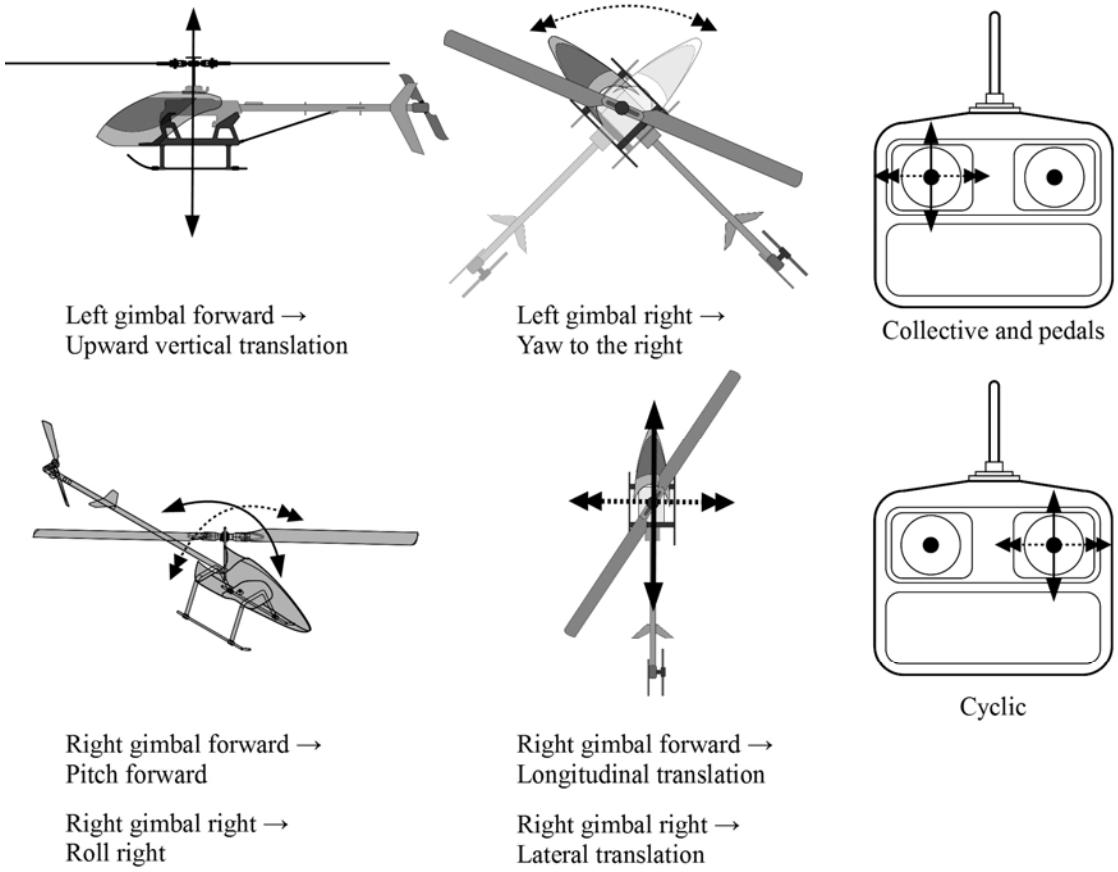



**Figure 2.1: RC helicopter control inputs and subsequent motion**

The main rotor blades work on the same principle as an aircraft wing, the difference being that the helicopter does not need to move forward to create lift, since the spinning blades can do this even when the helicopter is stationary. This Vertical Take-Off and Landing Capability (VTOL) is a helicopter's biggest advantage over fixed wing aircraft. This unique feature is also the limiting factor that keep helicopters from flying as fast or as efficient as their fixed wing counterparts [12].

The rotor thrust is increased by changing the pitch of the main rotor blades. The collective pitch, as the name implies, is the pitch that collectively changes for all the rotor blades. The pitch angle of each rotor blade is called feathering. A change in collective pitch will increase the airflow through the main rotor, which increases the thrust. The main rotor also provides the primary steering mechanism for the helicopter. By changing the pitch of the blades on one side of the rotor, the lift can be controlled here and this can be used to control the roll or pitch angles of the helicopter. This is called cyclic pitch, as the rotor blades' pitch need to be changed in a once-per-revolution fashion. If the pilot wants to roll to the right, he would like maximum rotor pitch on the left of the helicopter and a minimum on the right, with no change to the front and back. This will tilt the Tip Path Plane

(TPP) and cause the helicopter to roll. The TPP is the imaginary disc created by the motion of the rotor blades, and its tilting is called flapping. The swash plate is used to facilitate this cyclic change in rotor pitch. The cyclic control allows the pilot to change the direction of the thrust created by the main rotor blades [19].

The flybar is a damping mechanism that makes flying the helicopter easier. RC helicopters are extremely agile and this makes it difficult for a pilot to control them. Cyclic pitch changes are applied to the flybar that provides a lagged rate feedback response. The pitch changes are then applied to the rotor through mechanical linkages from the stabiliser. This makes stable controlled flight of small-scale aircraft possible by slowing down the response of the main rotor, thereby giving the pilot more time to react. In a full size helicopter, this is generally not needed because of the big main rotor and much larger inertia. The flybar can be replaced by an electronic unit that serves the same purpose. These flybarless units have gyros and accelerometers that measure the attitude rates and augment the pilot inputs to provide the necessary damping [20].

The tail rotor serves two purposes: first, it is needed to counteract the torque created by the main rotor, and second, it is needed to control the heading. The tail rotor has collective pitch to change its thrust, but no cyclic pitch. Most RC helicopters have a yaw rate gyro to help stabilise the heading. The vertical stabiliser serves the purpose of keeping the helicopter pointed in the direction that it is flying, much like a weathercock in the wind [20]. The helicopter equations of motion and reference frames will be discussed in Chapter 3.

## 2.2 Helicopter modelling

This section will give an overview of how small-scale helicopters are modelled. The focus of this section is not on helicopter theory, but rather on the methodology of helicopter modelling. The two primary methodologies followed in modelling small-scale helicopters are system identification, and first principle modelling.

### 2.2.1 First principle modelling

The first principle modelling approach uses the laws of physics to describe the helicopter mathematically. Whereas system identification uses parameter estimation and curve fitting to flight data. First principle modelling is popular in rotor and helicopter design. Flight tests are not needed to develop the model, but are needed for validation. Padfield and Munzinger both created helicopter simulations through a first principle approach [13], [21]. The first principle approach can be supplemented with system identification for areas where the equations become too complex. This

approach was followed by Munzinger [21]. For further insight into helicopter dynamics, reference can be made to Prouty [19], Padfield [13], Seddon [18] and Bramwell [12].

Budiyono develops a first principle helicopter model. He first derives a state space model with the stability and control derivatives to be calculated. The helicopter data is used to determine the derivatives from helicopter theory [22]. This produces a linear model similar to the one that will be derived in Chapter 3. SimModHeli is a simulation developed for testing small-scale helicopter flight control systems [23]. This simulator provides an accurate nonlinear model of the helicopter. The challenge with such a simulator is the amount of data that is needed from the helicopter. A high fidelity simulator such as SimModHeli can be a valuable resource to test control systems before flight tests. The nonlinear models used in simulators are often linearised in a specific trim condition to extract a linear model for control system design [24].

NASA developed a model called the Minimum-Complexity Helicopter Math Model (MCHMM). This model strives to model all the dominant characteristics of a helicopter, while trying to keep it as simple as possible. This keeps computation time low, as well as providing a better engineering understanding of the helicopter dynamics [25]. Mnich also emphasises the importance of understanding the plant being modelled. When developing a model, it is important that the model is only as complex as necessary. Complex models will take more time and money, and will also hamper the understanding of the basic principles [25]. The complexity of first principle models are often unnecessary for flight controller development, and are mostly needed for helicopter design and simulation.

### 2.2.2 System identification

System identification methods have been very successful in helicopter modelling [9]. Frequency domain methods have been applied to the US army's ARH-70, reducing the development time of flight control systems significantly [26]. The same techniques were applied to the MQ-8B Fire Scout Vertical Take-Off Unmanned Aerial Vehicle (VTUAV). This large unmanned helicopter weighs 1300 kg. System identification was a key component in this craft's successful development [27]. Mettler et al. were one of the first to use system identification techniques on small-scale rotorcraft [28]. They identified the dynamics of a Yamaha RMAX at Georgia Tech and were able to use the model to improve the flight controller performance using this model. La Civita developed a new method for model identification which he calls MOdeling for flight Simulation and Control Analysis (MOSCA) [29]. This model is developed from first principles and system identification is used to get linear models at different operating points. The linear models are then used to develop a robust full-envelope $H^{\infty}$ flight controller. Shim uses time domain system identification to model a

Yamaha R50 at Berkeley. The model is used to develop a μ-synthesis controller [16]. Yuan follows a similar approach to Shim, using time domain techniques to identify a model for a Hirobo Eagle. He adds value to the work by presenting a method to make estimation without initial values easier. By using the physical interpretation of the variables, constraints can be put on their values. By dividing the model into subsystems and first identifying the uncoupled systems, the estimation algorithm has less trouble to converge [30].

Frequency domain identification using NASA's Comprehensive Identification from FrEquency Responses (CIFER®) software is commonly found in the literature. CIFER® is a package developed by NASA specifically for full size helicopter system identification in the frequency domain. CIFER® is a frequency response analysis tool that fits linear equations to data [28]. CIFER® is an expensive product that is out of reach to most new research groups. More detail on the system identification process is given in the next section.

### 2.2.3 Model complexity

The three main flight modes are hover, forward flight, and vertical flight. A linear model cannot accurately represent all of these modes. In this study, the helicopter will be considered in hover mode. To simplify the modelling process, several assumptions can be made for a helicopter in hover

- Servo dynamics are much faster than the rotor time constant.
- Limited airflow over the fuselage makes its aerodynamic contribution minor.
- Only small deviations are made from trim flight.
- Constant rotor RPM is maintained.
- Operation is Out-of-Ground-Effect (OGE).
- Uniform rotor inflow is maintained.
- The helicopter body is rigid.

Padfield describes three levels of complexity for rotor mathematical modelling [13]. Level 3 modelling uses finite element methods and computational flow dynamics. This complexity is necessary when designing a new rotor blade or helicopter. Level 2 provides some complexity over level 1, which will typically be necessary for high bandwidth control such as active control of vibration or rotor flapping. Level 1 modelling is adequate for understanding the principles of helicopter flight and for developing low bandwidth control. This study will focus on level 1 modelling.

## 2.3 System identification process

System identification is no different from other modelling methods in that it is an iterative process. The system identification process is shown in Figure 2.2. Before estimation can begin, there are some decisions that need to be made: the model structure, the estimation algorithm, the experimental design, and a measure of fit. Once these steps have been completed, the model can be estimated and then validated. If the model fails validation, the designer will return to any one of the previous steps. The goal of the model is to represent the helicopter's dominant dynamics. The model will be used for the design of a linear optimal controller. A state space model of the helicopter is preferred for MIMO controller design [31]. Transfer function models can be cumbersome in MIMO systems. Each of the blocks from the flow diagram in Figure 2.2 will be discussed next.

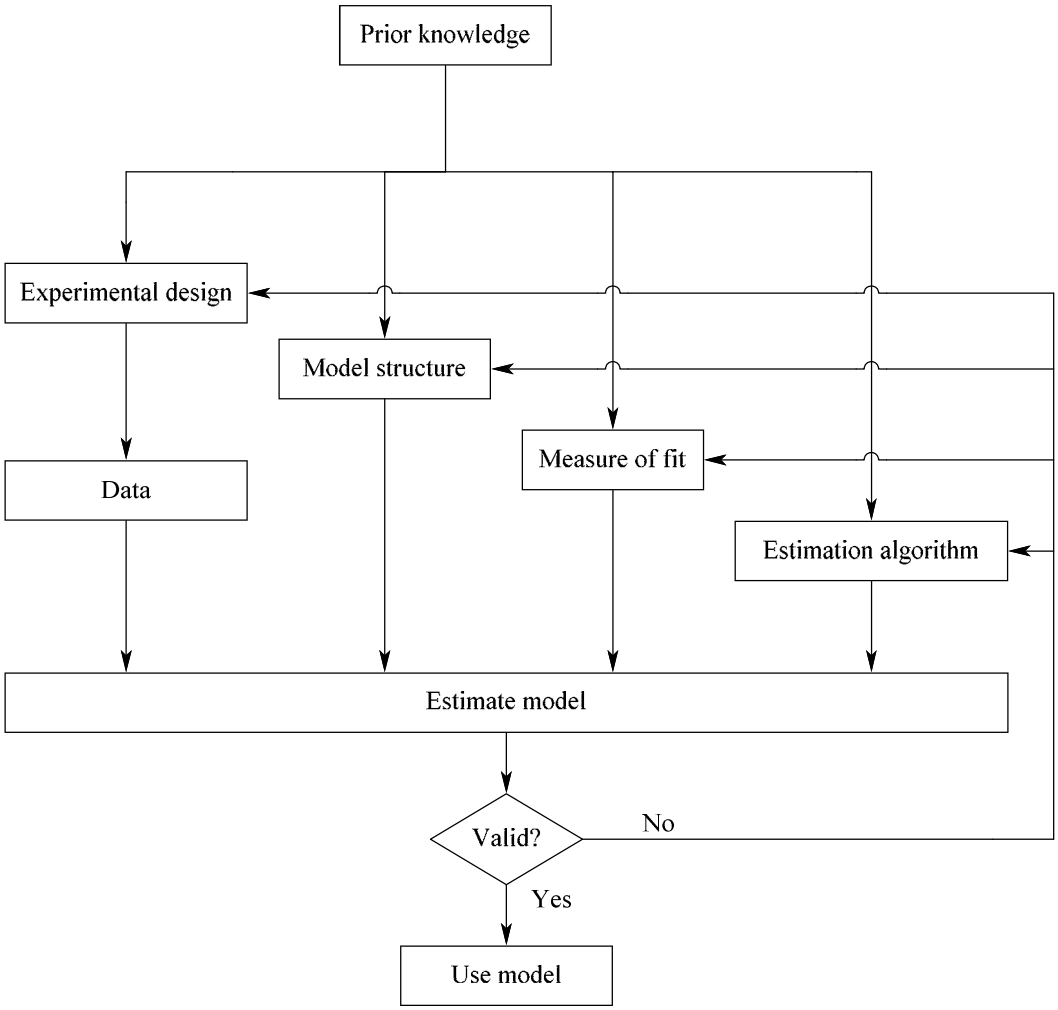



**Figure 2.2: Flow diagram of the system identification process. Adapted from [32] and [33].**

The optimal controller that is developed in Chapter 4 is a modern control technique that uses a state space model. Since the control is done in the time domain, it is desirable to model in the time

domain. Modelling in the time domain is more intuitive than frequency domain methods. The model is however validated in the frequency domain once the controller is complete. The System Identification Toolbox in Matlab® is available, and it has time domain parameter estimation algorithms that can be used. This approach is not as common as the frequency domain, but successful examples can be found [34], [35]. Using the System Identification Toolbox in Matlab®, rather than CIFER®, will make the methodology used in this research accessible to a considerably wider audience.

### 2.3.1 Model structure

Using knowledge of helicopter dynamics, Mettler was able to develop a grey box state space model [17]. This differs from a black box model where the state space entries have no physical meaning. In grey box modelling, a set of differential equations describing the helicopter is first derived. The coefficients of these equations are then used as the variables that the optimisation algorithm controls. This model is a state space model with stability and control derivatives that describe the helicopter. It is specifically developed for small-scale helicopters, and is widely used in the literature. It has been proven in control system design, which makes it ideal for this study [9]. It is sufficient to capture the dominant dynamics of the helicopter, while not having too many parameters to estimate. This last attribute is crucial for system identification, as it will improve the optimisation algorithms.

Mettler adapted the structure from the full scale helicopter model developed by Tischler [36]. It is a hybrid model containing both the normal rigid body equations, as well as the rotor tip path plane (TPP) equations that govern the rotor fuselage interaction. To account for the active yaw damping system most RC helicopters are fitted with, the model is augmented with a yaw rate feedback state [17].

### 2.3.2 Experimental design

High quality flight data is extremely important in system identification. Input-output data is used in the estimation process and if this data does not contain sufficient information on the helicopter dynamics, the model will be inaccurate. The quality of the information contained within the data will depend on the flight tests. This makes the input signal given to the helicopter one of the most important aspects in the modelling process [14]. Persistent excitation refers to a signal that will be able to excite all of a system’s properties. Such a signal will give all of the information needed to identify the model.

Frequency sweeps, or chirp signals, on each control input can be used to excite the aircraft. By changing the amplitude of the inputs, it can be ensured that the aircraft does not deviate too much from its linear operating region [14]. Any external disturbances need to be minimised to avoid identifying effects not present in the model structure.

### 2.3.3 Estimation algorithms

Yuan and Katupitiya use NASA's System IDentification Programs for AirCraft (SIDPAC) toolbox to estimate model parameters of a small size 60 Hirobo Eagle RC helicopter [30]. Mettler uses CIFER® for estimation of the larger Yamaha RMAX as well as the small hobby XCell [17]. Shim uses a Prediction-Error Method (PEM) algorithm to model the Yamaha RMAX [16]. Each one of these studies resulted in accurate models of the helicopters, and successful controller design using the models. In this study, a PEM algorithm is used for parameter estimation. The PEM and N4SID algorithms in Matlab® form the basis of the System Identification Toolbox's time domain estimation algorithms [37]. The PEM algorithm attempts to find a set of parameters that will minimise the difference between the measured output and the predicted model output.

When using the PEM tool, it is very important that the parameter initial values be selected correctly. The algorithm can easily become stuck in local minima that results in the parameters not converging to the correct values [32]. Physical knowledge of the plant should be used to make informed estimates for the initial values. As the estimation process is continued, these initial values will get closer and closer to the real values. Another way to help the estimator cope with the complex problem of minimising the error of a function with so many parameters is to limit the number of parameters being estimated at a time. This can be done by setting certain parameters to be "free" for estimation while others are fixed. Once a good estimate of a parameter is found, it can be fixed and another can be freed [30]. This process will be explained in the methodology (Chapter 3).

### 2.3.4 Model validation

A different data set should be used for validation than for identification. This ensures the model is a good representation of the helicopter dynamics and is able to give a general prediction [37]. Throughout the identification process, the Normalised Root Mean Square Error (NRMSE) can be used to evaluate the identified model accuracy,

$$NRMSE = 100\left(1 - \frac{\|\mathbf{y} - \hat{\mathbf{y}}\|}{\|\mathbf{y} - mean(\hat{\mathbf{y}})\|}\right). \tag{2.1}$$

The NRMSE gives a measure of how well the model is able to predict the validation data. The vector **y** contains the validation data and $\hat{\mathbf{y}}$ the estimated values [37]. With a good understanding of the helicopter dynamics, it can be seen that if an identified model shows very high NRMSE values, but oscillates around the validation data, it is not a realistic model. This problem is often encountered as the PEM optimisation algorithm tries to minimise the error.

The residual is the difference between the model's predicted output and the validation data [32]. This is a good measure of the modelling error. The residual plot shows those parts of the data that cannot be explained by the model. The two main residual analysis tools used are the whiteness test and the independence test. The whiteness test indicates whether the residuals are auto-correlated or not. Ideally, one would want the residuals not to show auto-correlation and essentially be only the random noise that cannot be modelled. When a model or process has no disturbance component, it is called an output error model. In this case, the residuals will have autocorrelation and show recognisable trends [37]. The errors cannot be truly random because there is no noise/disturbance to generate these errors. The independence test gives an indication of how well the outputs of the model correlate with the inputs, and that the outputs should not be dependent on past values of the output. If the system has feedback, there will be correlation [32].

The estimated model can either be given a set of initial values and a series of inputs to compare its response to the validation data, or the input-output data from the validation set can be used and only a finite number of steps can be predicted by the model. The first case is better for validation, but is not practical for unstable models. The model's intended use is for controller design, and thus the model only needs to be able to predict the system's response within the dominating time constant of the system [37].

In any model where parameters are estimated, it is important to analyse the certainty of these estimations. The System Identification Toolbox uses the covariance matrix for this. The covariance matrix is a measure of a parameter's uncertainty. The parameter's uncertainty is an indication of how confident the estimation algorithm is about the parameter's value. This uncertainty can then be used to calculate the standard deviation of the estimated parameters [32].

## 2.4 Automatic flight controllers

The flight control system differentiates an autonomous helicopter from a manned helicopter. Manned helicopters often have Stability Augmentation Systems (SAS), but the difference is that these FCSs only augment the pilot's inputs. The controller has limited authority and can be overruled by the pilot at any time. In an autonomous system, there is no pilot and the FCS has full

control over the aircraft. The SAS normally controls only attitude and altitude, whereas the flight controller in a UAV typically does trajectory tracking [20].

The three main categories for automatic flight control on helicopters are Intelligent, Model-Based Nonlinear, and Linear controllers [9]. In Figure 2.3 each of these categories is expanded to show the popular controller architectures used in helicopter flight control systems. Next, a brief overview will be given of work done in each of these fields.

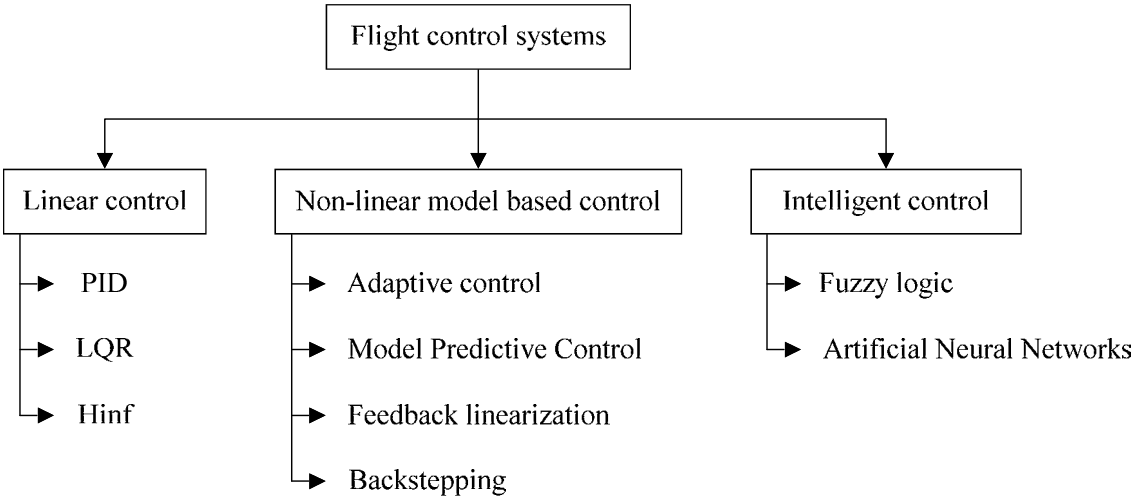



**Figure 2.3: Three primary control categories seen in AFCS used in RWUAVs. Modified from [9]**

Intelligent controllers are primarily used in research environments, due to the difficulty in proving stability for commercial use. Johnson et al. developed an ANN controller using online and offline training data which guaranteed stability [38]. The controller uses approximate model inversion and Pseudo Control Hedging (PCH). The controller provides increased robustness due to the ability to adapt to model and environmental changes. It was tested on Georgia Tech's GTMax helicopter as the main control system for research on guidance and navigation [39]. A controller that learns how to perform aerobatic manoeuvres from a small set of human pilot demonstrations was developed at Stanford University. The controller was able to perform a complex set of manoeuvres previously not possible under automatic control. It performs the manoeuvres better than the pilot by extracting an optimal trajectory from the demonstrations [40]. A fuzzy logic controller that is capable of navigation in the case of a tail rotor failure was developed by Garcia and Valavanis [41]. The controller was tested in X-plane and proved capable of waypoint navigation without authority over heading. The vehicle reached rotational speeds of up to 560 deg/s during the actuator lockout experiments. The vehicle became unstable beyond 600 deg/s [41].

Raptis developed a nonlinear backstepping controller that was tested in simulation. The controller is proven to be stable and performs well compared to a PID controller [42]. This approach is

promising due to the relative simplicity it has compared to other nonlinear controllers. Koo and Sastry first split the system into an inner and outer loop. They then moved on to design a nonlinear controller based on the differential flatness of the outer system [43].

Within linear control, the most popular techniques being used on RUAS are PID, $H^{\infty}$ and LQR type controllers. All of these can be used in a gain scheduling system to accommodate different operating conditions. Their simplicity and ease of implementation make linear techniques the most popular in practice [14]. Linear systems can guaranty stability for a specific operating region and bounded disturbances.

PID is the most popular control system and has been proven repeatedly. Mettler [44] designs a full aerobatic controller using gain scheduled PI controllers. The manoeuvres were first performed by an expert pilot and then approximated by piecewise linear functions that are used as reference trajectories. The controller was able to successfully orchestrate Split-S and Immelman manoeuvres [45]. PID is used in the hierarchical control system at Berkeley. The research group at Berkeley later went on to develop a nonlinear approximate linearisation controller, a fuzzy logic controller, and a μ-synthesis controller [16][34][46]. The very successful open source autopilot APM:Copter uses PID control in its AFCS [47].

$H^{\infty}$ has been proven to work very well with unmanned helicopters. $H^{\infty}$ is well suited to the noisy and uncertain conditions typical of these platforms [14]. La Civita develops a modelling technique called MOdeling for flight Simulation and Control Analysis (MOSCA). He then goes on to develop a robust loop shaping $H^{\infty}$ controller for Carnegie Mellon University's Yamaha RMAX. Several linear controllers are gain scheduled to achieve full-envelope control. The controllers are validated in flight tests [29].

Weilenman compares $H^{\infty}$ to $H^{2}$ and LQR performance during helicopter hover. It is concluded that $H^{\infty}$ provides substantially better performance than the two competitors do. He then goes on to say that $H^{\infty}$ should be used if this performance is required, but it is not possible without substantial knowledge of the plant disturbances. If performance is satisfactory, LQR should be used. $H_2$ methods should be avoided as they require the same knowledge of the plant but will not provide a measurable improvement [48]. Kim used time domain system identification to model a Raptor E620 and then develop an $H^{\infty}$ controller. The controller is demonstrated to have better performance and robustness than a PD controller [49].

Gavrilets and Mettler proved that LQR control could be used in aerobatic flights. The reference state trajectories were derived from expert pilot examples. The smooth transition between trim flight and the start of a manoeuvre was key to their success and it was in part made possible by

initialising the integrator gains to the correct values. It was also important that the trim controller be able to start the helicopter in the correct position for the aerobatic manoeuvre [44], [45]. A linear tracking controller is designed by Raptis. It consists of two control loops, one for the heave-yaw motion, and one for the longitudinal-lateral motion. Reference states are calculated with a backstepping technique that uses a linear state space model. The controller uses an output feedback LQR design to stabilise the error dynamics [50]. LQR is popular to use as a baseline controller for more advanced control techniques or intelligent systems [51].

Even though nonlinear controllers should theoretically outperform linear controllers significantly, experimental results have not shown such a big margin between the two. This is possibly due to difficulties in implementing these nonlinear techniques [9]. Nonlinear controllers need to be further tested, since they can increase insight regarding helicopter dynamics [14].

The controller choice is influenced by more than just performance criteria. The simplest solution is to use four SISO PID controllers for the inner-loop and another set of SISO PID controllers for the outer loop. It has been shown that this approach is adequate, but cannot remove the cross coupling in the attitude dynamics. Therefore, a multivariable LQR controller is more suited for this system. It does not have much added complexity and can handle the complex dynamics better [14].

LQR provides a straightforward way to implement a MIMO controller if a good model of the system is available. LQR will theoretically always give a stabilising gain. It has an infinite Gain Margin (GM) and a Phase Margin (PM) of greater than 60° [52]. An added advantage of using LQR in a research environment is that it can be expanded to a gain-scheduling controller or time varying LQR [53], [54]. Time varying LQR has been proved to achieve better performance and should be considered if the steady state performance is unsatisfactory [55]. Further details regarding the design of an LQR controller are given in Chapter 4.

## 2.5 Simulation environment

Simulations are an important part of the design of an AFCS. They provide an affordable and safe way to test control algorithms. Flight tests are dangerous if not performed correctly. For example, in 2009 a farmer in Korea was killed by a crop dusting Yamaha RMAX commercial RWUAV. This shows how dangerous even small-scale commercial UAVs can be [56]. In this study, the simulation is not just a verification step before flight tests, since the simulator is the final application. This makes the choice of simulator even more important.

### 2.5.1 Role of simulators

In his doctoral thesis, La Civita states that the primary reason for the relatively slow development of UAV control systems and autonomy is not the lack of knowledge in aviation or control systems, but the time it takes to integrate such a system to aviation standards [29]. Munzinger stresses the importance of a flight simulator in the development of modern controllers for helicopters. He develops a simulator from first principles in his doctorate at Georgia Tech [21]. Proctor et al. stated that the US military is considering using game simulators for training. This is because of the cost and time savings involved with a COTS game [57].

Ribeiro and Oliveira developed a test environment for autopilot design. The controllers are developed in Matlab®/Simulink® and X-plane is used to model the aircraft flight dynamics. A stationary model aircraft is connected to Matlab® via a microcontroller and serial interface. This lab facility enables students and control engineers to develop flight controllers in Matlab®, test the controllers in X-plane, and observe the hardware response [58].

### 2.5.2 Simulator comparison

Many factors need to be considered when choosing a simulator. The four deciding factors for this study are: development time, budget, available models, and interface capabilities. Professional engineering simulators provide high fidelity models, but are too expensive. Game simulators are inexpensive and have gained popularity as simulation tools [59]. The focus of this study is not the development of a high fidelity simulator. A minimum amount of time should be spent on the simulator development, which limits the use of simulators where a helicopter model still needs to be developed.

The next factor that needs to be considered is the models included in the simulator. A retrofitted RC helicopter is the most likely platform to be used in the laboratory. This is also the deciding factor in the choice of a simulator, as very few simulators have RC models. The ability to interface to an external program and access to the helicopter's states are crucial. Without these capabilities, there is no data to work with when modelling. Furthermore, without the ability to override the helicopter controls, the developed controller cannot be validated.

Two popular professional tools for simulating rotary winged aircraft are Rotorlib and HeliSIM. RTDynamics' RotorLib is a high-fidelity simulation tool that can be used either in the Matlab® environment or as a library in C [60]. Presagis' HeliSIM is a software package used to design and simulate helicopters. It comes with graphical interface and has high-fidelity models [61].

Game simulators are primarily used by enthusiasts to practise flying, whereas engineering flight simulators are developed as tools for engineers to develop and test new aircraft systems. Even though games are not primarily developed for engineering, they still boast accurate flight models. These simulators focus more on the handling qualities of the aircraft than the underlying physics. Four game simulators are commonly used in the literature: AeroSIMRC, Microsoft Flight Simulator (MSFS), FlightGear and X-plane. Except for MSFS, an interface to the other three are included in the ArduPilot software, an indication of their popularity among the UAV hobby community [47].

Popular hobby RC simulators such as Phoenix and AeroFly do not have interface capabilities and are excluded from the comparison. Phoenix is the most widely used RC helicopter simulator with extremely realistic models, and if it had the possibility to interface with an external program, it would have been the first choice for the present research.

MSFS does not contain any RC helicopter models, which is the primary interest here. FlightGear is free and open-source simulation software. It has been used for many fixed wing simulations in the amateur community and academia [62], [63]. It is difficult to determine how well the rotary winged models perform, as they have not been used as often. However, FlightGear is often used as graphical front-end for other Flight Dynamics Models (FDMs) [64]. FlightGear itself does not have any RC helicopter models.

X-Plane is a commercial product that differs from other flight simulators in the way it determines FDMs. X-Plane uses blade element theory instead of relying on flight data stored in lookup tables. X-Plane has been used by Velocity, NASA, Scaled Composites and Carter Aviation to name only a few. It includes a massive environment and a list of features not available in any other “game” simulators [65]. Failures and complex environmental situations can be simulated in X-Plane. For example, Valavanis and Garcia tested their fuzzy controller in the presence of a tail rotor failure using X-Plane [41]. It is also used by Valavanis to simulate RC helicopters and design controllers for these helicopters [14].

X-plane is designed to model full-size aircraft and helicopters. Therefore, researchers who used X-plane for their RC simulations had custom built and tuned models. This is a possible indication that the RC helicopter model included in X-plane is insufficient. Roy stresses that the biggest problem with X-Plane is that access to the inner-loop AFCS is not available [59]. No access to internal control is not problematic, since a retrofitted RC helicopter has the same limitation.

AeroSIMRC is a popular game type RC aircraft and helicopter simulator. It is well validated by expert RC helicopter pilots and has many models to choose from. Helicopter parameters can be accessed and controls can be overridden through a plug-in interface. AeroSIMRC has not been used

in many academic papers and the models are not easily adapted. A plug-in interface is provided, but a plug-in for external access to the helicopter states and control inputs still needs to be developed [66].

Roy's method is used to evaluate the simulators named so far [59]. In a case study on different applications and different simulators, Roy found that X-Plane had cost the company a large amount of money due to the time needed to integrate it, as well as the insufficient accuracy of game based simulators. Another problem that took time was building a model that accurately represented the company's aircraft. In this study, the problem is solved by choosing a model that is already included in the software.

Roy goes on to say that using dedicated simulation software would have been a more viable solution. The Return On Investment (ROI) for this project is not as clearly definable as in Roy's case studies. The project is research based and will have no immediate monetary value. It will not be sold and will not generate an income. Consequently, ROI will not be considered in the choice of simulator. Commercial software like Presagis' Helisim and RTDynamics' Rotorcraft libraries are well out of reach for the scale of the current study.

**Table 2.1: Comparison of simulator software**

| **Software** | **RC helicopter model** | **Interface capabilities** | **Cost** | **Development time needed** | **Model validity** | **Support/User community** | **Score** |
|---|---|---|---|---|---|---|---|
| **Weight** | **Yes/No** | **Yes/No** | **0.3** | **0.3** | **0.3** | **0.1** | |
| **AeroSIMRC** | Y | Y | 4 | 4 | 4 | 2 | 79% |
| **X-Plane** | Y | Y | 4 | 1 | 3 | 4 | 63% |
| **FlightGear** | N | Y | 5 | 4 | 2 | 4 | X |
| **MSFS** | N | Y | 4 | 3 | 2 | 4 | X |
| **Phoenix** | Y | N | 4 | NA | 5 | 4 | X |
| **Professional** | NA | NA | 0 | 0 | 5 | 5 | X |

The comparison of different simulator options presented here has shown that AeroSIMRC is the best option for the present research. It offers all the features that are required from a simulator, and also comes at a very low price. The simulator has been found to be sufficiently validated by its users. Roy stresses the fact that test pilots are a good form of validation. The difference with RC helicopters is that there are many more pilots, since RC helicopters are so widely available. Last but not least, the author found the RC model helicopter in X-plane to be unrealistic and this swayed the choice from X-plane to AeroSIMRC.

## 2.6 Critical literature evaluation

It has been found that system identification is more suited for modelling the helicopter than first principles. A high degree of accuracy with low complexity is required for controller development. These normally contradictory goals have been proven to be attainable with curve fitting and estimation methods where not all the underlying physics need to be modelled [17]. First principle models require knowledge about the helicopter that is not attainable without physical hardware. Grey box system identification provides a particularly attractive solution. Grey box models provide information on the physical platform being modelled. This information can be used for validation, as well as to gain insight into the model parameters. These insights can be used to improve the controller performance, which would not have been possible with a black box model [67]. Time domain system identification is intuitive and can be validated with frequency domain analysis [16]. Parameter estimation can be greatly improved by limiting the number of parameters that need to be estimated at a time. The estimation algorithms are highly dependent on the initial values, and the estimation results can be improved considerably by providing better initial values [30].

Linear controllers are the most widely used controllers in helicopters [9]. Linear controllers are simpler to develop and easier to maintain than nonlinear and intelligent controllers [14]. A simple, reliable, and robust controller is preferred for the first-flight of a helicopter. LQR controllers are the simplest solution that addresses the MIMO nature of the helicopter [68].

The importance of a simulation environment has been discussed. Several COTS simulators were compared, and it was found that AeroSIMRC provides the best solution for this research problem. AeroSIMRC can interface externally and provides accurate helicopter models that need no further development. The only extra development required is the interface to Simulink®.

## 2.7 Conclusions

This chapter has provided the background and choices made to complete the objectives given in Chapter 1. The system identification procedure and important aspects of the process have been discussed. The model structure proposed by Mettler is ideal for this study [17]. The different controller options were evaluated and it was determined that a steady state LQR controller will be developed [14], [68]. AeroSIMRC will be used for validation [66].

# Chapter 3: Modelling

*This chapter consists of four closely linked sections. First, the helicopter state space model is derived, providing the stability derivatives that can be estimated from flight data. Second, the methodology followed for system identification is discussed. Third, to acquire flight data and to validate the model, a test environment is required. Fourth, the results from the applied methodology are given and the validity of the model is inspected.*

## 3.1 Helicopter modelling

This section aims to arrive at the state space model needed for estimation. The helicopter reference frames and 6 Degree Of Freedom (DOF) Rigid Body Equations (RBE) are first covered. The RBEs are then linearised and simplified to find the stability derivatives.

A superscript is added to the relevant variable to distinguish between the different reference frames, vectors are printed in bold, and transformations are given in subscripts. The conventions are given in Figure 3.1.

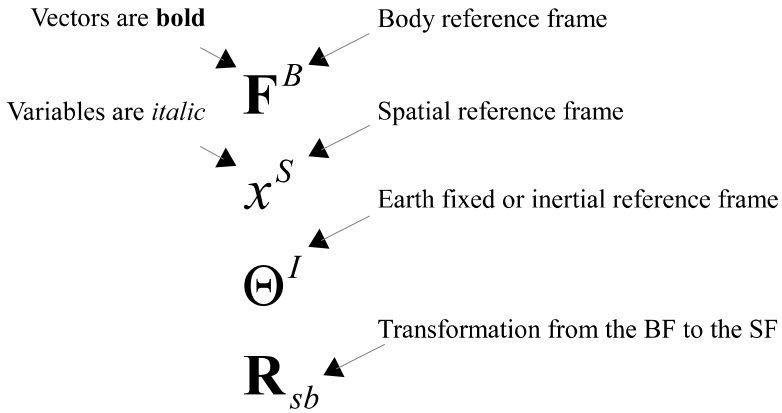



**Figure 3.1: Notation standards followed throughout the chapter**

### 3.1.1 Reference frames

Several reference frames are used to simplify the description of the helicopter motion. All coordinate systems are right-handed and orthogonal.

#### 3.1.1.1 Body-Fixed frame (BF)

The BF conventions are shown in Figure 3.2. The origin of this frame is centred in the Centre of Gravity (CG) of the helicopter and it rotates with the helicopter. The *x*-axis points forward out of the helicopter's nose. The *z*-axis points downward through the bottom of the helicopter. The *y*-axis points to the right when sitting in the helicopter and facing forward. Rotation is defined as positive in a clockwise direction. This frame is what the helicopter pilot would experience in flight. Rotation around the *x*, *y*, and *z*-axis is called roll, pitch, and yaw respectively.

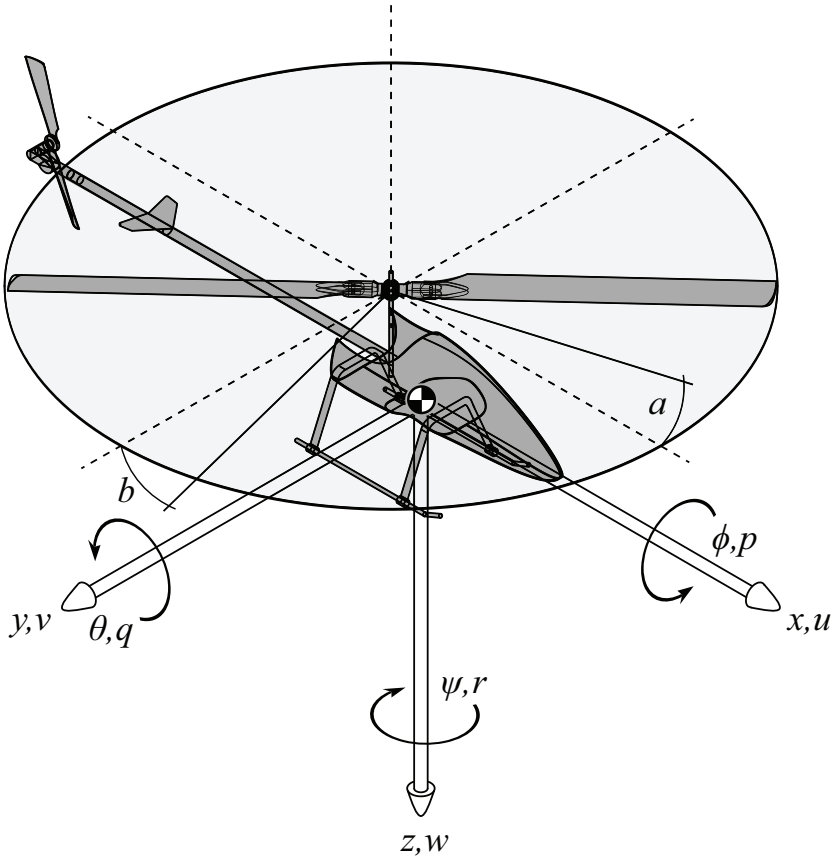



**Figure 3.2: Body reference frame conventions used in this study**

#### 3.1.1.2 Earth-Fixed frame (EF)

The relation between the BF and the EF is shown in Figure 3.3. This reference frame indicates the position of the helicopter. In this specific case, it is also the inertial reference frame. The earth is

seen as flat, stationary and infinite in area. The *x*-axis is flat on the earth and its orientation can be chosen arbitrarily, but for convenience, it is chosen to be due north. The *y*-axis is perpendicular to the *x*-axis and flat on the earth, pointing due east. The *z*-axis is perpendicular to these two and points downwards. This frame can be seen as the reference frame an RC helicopter pilot would see when standing stationary on the runway.

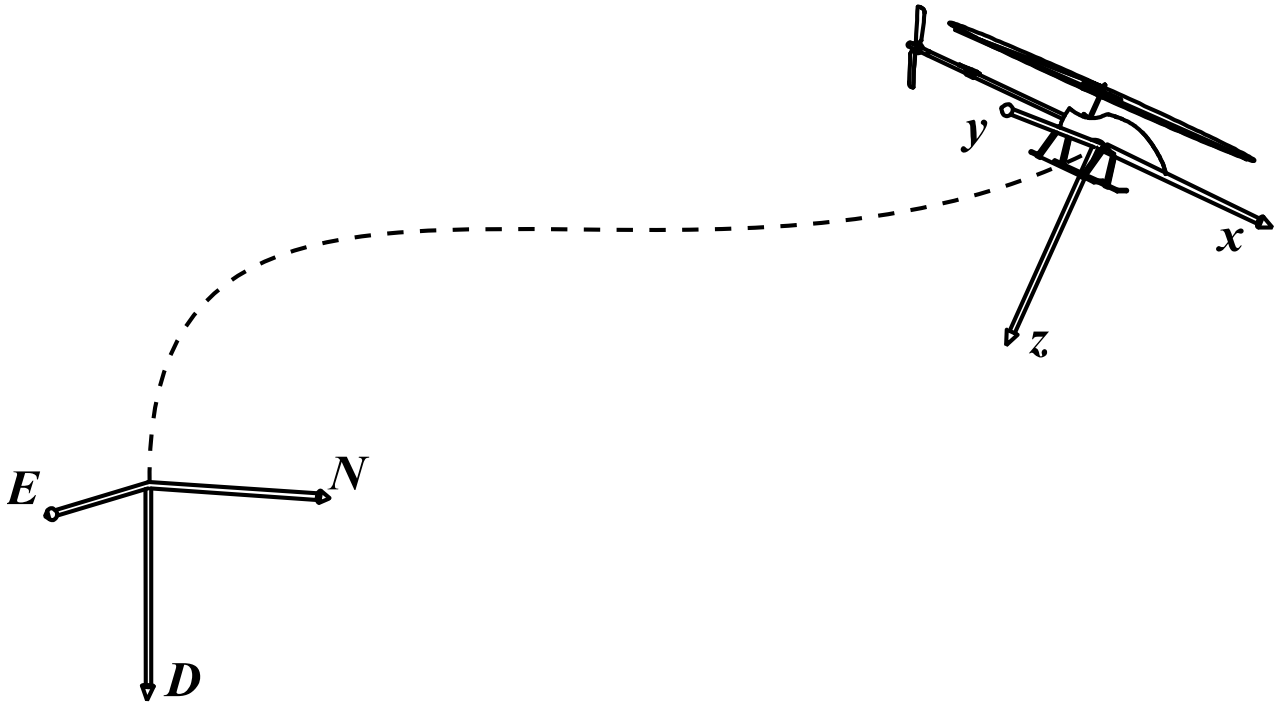



**Figure 3.3: Earth fixed reference frame**

### 3.1.1.3 Spatial Frame (SF)

The origin of this reference frame is centred on the CG of the helicopter, the same as the BF. Contrary to the BF, the SF's orientation stays the same as the EF orientation. This implies that the origin will move with the helicopter and the orientation will keep pointing in one direction, even if the helicopter's attitude changes. This can be compared to a camera mounted on the helicopter that keeps pointing to a point on the horizon, even when the helicopter changes direction.

### 3.1.1.4 Rotation matrix

The rotation matrix is used to transform between coordinate systems. In particular, it is used to change between the BF and the SF that has the same orientation as the EF. The derivations are made from Bak's notes [69], but can be found in any standard textbook on aircraft stability analysis or matrix algebra [70].

The three basic rotations around each axis is given by

$$\mathbf{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}, \tag{3.1}$$

$$\mathbf{R}_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}, \tag{3.2}$$

$$\mathbf{R}_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.3}$$

The standard rotation matrix in aerospace is the z-y-x or 3-2-1 rotations. This can be achieved with a matrix multiplication

$$\mathbf{R} = \mathbf{R}_x(\phi)\mathbf{R}_y(\theta)\mathbf{R}_z(\psi). \tag{3.4}$$

The order of the rotations is important, and a different sequence will not necessarily give the same results. Rotation from the SF to the BF is given by

$$\mathbf{R}_{bs}(\boldsymbol{\Theta}) = \begin{bmatrix} c\theta c\psi & c\theta s\psi & -s\theta \\ s\phi s\theta c\psi - c\phi s\psi & s\phi s\theta s\psi + c\phi c\psi & s\phi c\theta \\ c\phi s\theta c\psi + s\phi s\psi & c\phi s\theta s\psi - s\phi c\psi & c\phi c\theta \end{bmatrix}, \tag{3.5}$$

where $\boldsymbol{\Theta}$ is the Euler angles

$$\boldsymbol{\Theta} = \begin{bmatrix} \theta & \phi & \psi \end{bmatrix}^T. \tag{3.6}$$

Rotation in the opposite direction (from BF to SF) is given by

$$\begin{aligned} \mathbf{R}_{sb}(\boldsymbol{\Theta}) &= \mathbf{R}_{bs}^T(\boldsymbol{\Theta}) \\ &= \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}. \end{aligned} \tag{3.7}$$

To transform the position vector $\mathbf{x}^B$ from the BF to the SF is a simple matrix multiplication

$$\mathbf{x}^S = \mathbf{R}_{sb}(\boldsymbol{\Theta}) \cdot \mathbf{x}^B. \tag{3.8}$$

#### 3.1.1.5 Attitude description

Euler angles will be used to describe the helicopter's body in the EF. Euler angles contain discontinuities that could cause a gimbal lock. This only happens in extreme manoeuvres where the pitch is ±90°. The model developed here is only valid for hover, and should never have such extreme attitude angles. Quaternions could be used to solve the problem for modelling aerobatic manoeuvres [71]. Quaternions have the disadvantage of being less intuitive to use, which is an unnecessary complication in this case.

#### 3.1.1.6 Euler rates

Transformation of the attitude angles from the SF to the BF is given by

$$\begin{aligned}\mathbf{P}_{sb}(\boldsymbol{\Theta}) &= \mathbf{P}_{bs}^{-1}(\boldsymbol{\Theta}),\\ &= \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & \dfrac{s\phi}{c\theta} & \dfrac{c\phi}{c\theta} \end{bmatrix}.\end{aligned} \tag{3.9}$$

The Euler rates can then be defined as

$$\begin{aligned}\dot{\boldsymbol{\Theta}} &= \mathbf{P}_{sb}(\boldsymbol{\Theta}) \cdot \boldsymbol{\omega},\\ \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} &= \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & \dfrac{s\phi}{c\theta} & \dfrac{c\phi}{c\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix},\end{aligned} \tag{3.10}$$

with $\boldsymbol{\omega}$ the angular velocity projected onto the BF, and $\dot{\boldsymbol{\Theta}}$ the angular velocity projected onto the SF.

### 3.1.2 Equations of motion

The helicopter's rotation and translation is modelled as a 6 DOF rigid body. These governing equations can be used to describe any rigid body movement. These derivations are mainly based on Nelson [70].

The equations of motion are derived in the BF, since in this reference frame the moments and products of inertia stay constant [14]. However, Newton's second law only applies to objects in an inertial reference frame. This is why the equations given in the previous section are so important, since they allow us to convert between these reference frames.

The vector **F**, in (3.11), is made up of the three orthogonal forces shown in Figure 3.2. The total external force applied to the helicopter is described by

$$\mathbf{F}^B = \begin{bmatrix} X & Y & Z \end{bmatrix}^T. \tag{3.11}$$

The total torque or moment applied to the helicopter's CG is

$$\boldsymbol{\tau}^B = \begin{bmatrix} L & M & N \end{bmatrix}^T. \tag{3.12}$$

The translational velocity is

$$\mathbf{v}^B = \begin{bmatrix} u & v & w \end{bmatrix}^T, \tag{3.13}$$

and the rotational velocity is

$$\boldsymbol{\omega}^B = \begin{bmatrix} p & q & r \end{bmatrix}^T. \tag{3.14}$$

Newton's second law is given by

$$\mathbf{F} = m\frac{d^I\mathbf{v}}{dt}, \tag{3.15}$$

and

$$\boldsymbol{\tau} = \frac{d^I\mathbf{H}}{dt}, \tag{3.16}$$

where **H** is the angular momentum. The notation $\frac{d^I}{dt}$ and $\frac{d^B}{dt}$ is used to indicate the time derivative with respect to the inertial and body reference frames. This will in general be different for a rotating body, since an observer in the body reference frame cannot observe the rate of change in rotation of its own reference frame. The two Newton equations can only be used in an inertial reference frame; therefore, it is needed to relate the BF velocities to the EF.

Observing a rotating reference frame from an inertial reference frame gives

$$\frac{d^I\mathbf{v}}{dt} = \frac{d^B\mathbf{v}}{dt} + \boldsymbol{\omega} \times \mathbf{v}^B. \tag{3.17}$$

This is necessary, as previously stated, since $\frac{d\mathbf{v}}{dt}$ in the BF will not be the same in the inertial reference frame.

Substituting (3.17) in (3.15), and expanding the cross product results in the following set of equations

$$\begin{aligned} X - mg\sin\theta &= m(\dot{u} + qw - rv), \\ Y + mg\cos\theta\sin\phi &= m(\dot{v} + ru - pw), \\ Z + mg\cos\theta\cos\phi &= m(\dot{w} + pv - qu). \end{aligned} \tag{3.18}$$

Newton's second law for rotational motion states that the torque is equal to the change in angular momentum

$$\boldsymbol{\tau} = \frac{d^I \mathbf{H}}{dt}. \tag{3.19}$$

As with the translational motion, it is necessary to work in the inertial reference frame where

$$\frac{d^I \mathbf{H}}{dt} = \frac{d^B \mathbf{H}}{dt} + \boldsymbol{\omega}^B \times \mathbf{H}, \tag{3.20}$$

with angular momentum given by

$$\mathbf{H}^B = \mathbf{I}\boldsymbol{\omega}^B. \tag{3.21}$$

The complete inertia matrix is given by

$$\mathbf{I} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}. \tag{3.22}$$

The BF is aligned with the helicopter so that (3.22) can be written as

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}. \tag{3.23}$$

The moments of inertia are given by

$$\begin{aligned} \mathbf{I}_{xx} &= \sum (y_m^2 + z_m^2) dm, \\ \mathbf{I}_{yy} &= \sum (x_m^2 + z_m^2) dm, \\ \mathbf{I}_{zz} &= \sum (x_m^2 + y_m^2) dm, \end{aligned} \tag{3.24}$$

where $x_m$, $z_m$, $y_m$ are the distances of elementary masses from the CG [72].

From (3.21) the $\frac{d\mathbf{H}}{dt}$ vector consists of

$$\begin{aligned} \dot{h}_x &= I_{xx}\dot{p}, \\ \dot{h}_y &= I_{yy}\dot{q}, \\ \dot{h}_z &= I_{zz}\dot{r}. \end{aligned} \tag{3.25}$$

After substituting (3.21) into (3.20), the equations for the moments acting on a rigid body's CG from (3.19) are used

$$\boldsymbol{\tau}^B = \mathbf{I}\cdot\dot{\boldsymbol{\omega}} + \boldsymbol{\omega}\times(\mathbf{I}\cdot\boldsymbol{\omega}), \tag{3.26}$$

with the vector components of $\boldsymbol{\tau}$ given by

$$\begin{aligned} L &= I_{xx}\dot{p} + qr(I_{zz} - I_{yy}), \\ M &= I_{yy}\dot{q} + pr(I_{xx} - I_{zz}), \\ N &= I_{zz}\dot{r} + pq(I_{yy} - I_{xx}). \end{aligned} \tag{3.27}$$

The helicopter velocity in the inertial reference frame is given by

$$\begin{bmatrix} v_x^I \\ v_y^I \\ v_z^I \end{bmatrix} = \mathbf{R}(\boldsymbol{\Theta}) \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \tag{3.28}$$

using the rotation matrix equation to transform from the BF to the EF. Now, integrating the inertial velocity vector will provide the helicopter position. The helicopter orientation is given by $\boldsymbol{\Theta}$. The angular velocity is transformed from the BF to the SF to give the Euler rates using (3.10).

With these equations, the helicopter's motion in 6 DOF can be described. The next required step is to find the forces and moments acting on the helicopter that will cause these rotations and translations.

### 3.1.3 Stability derivatives

The equations of motion can be linearised using small perturbation theory. The forces and moments in the equations of motion are then each expanded as a Taylor series. This will give us the linear equations of motion necessary to develop a model for the helicopter movement. Improving the fidelity of this model, the TPP degrees of freedom, and the active yaw damping system are also added. These equations are then written in state space form that is ready to be identified from flight data.

In small perturbation theory, every variable in the equations of motion can be seen as the variable's equilibrium or trim value plus some small perturbation from that value

$$\mathbf{x} = \mathbf{x}_0 + \delta\mathbf{x}, \tag{3.29}$$

where $\mathbf{x}_0$ is the state in trim, or equilibrium, and $\delta\mathbf{x}$ is a small perturbation/disturbance from that state. It is assumed that the aerodynamic effects are linear within this small movement. This is in fact generally true and it is extensively used by flight dynamists to evaluate aircraft motion [13], [19].

Using small angle approximations, the following equations are simplified to

$$\begin{aligned} \cos\delta\theta &\cong 1, \\ \sin\delta\theta &\cong \delta\theta, \\ \text{and } \delta x\delta y &\cong 0. \end{aligned} \tag{3.30}$$

The following states are set to zero to simplify the analysis of hover even further,

$$u_0 = v_0 = w_0 = p_0 = q_0 = r_0 = \theta_0 = \phi_0 = 0. \tag{3.31}$$

Using this approximation, the rigid body equations of (3.18) and (3.27) can be simplified as follows

$$\begin{aligned} X_0 + \delta X - mg\delta\theta &= m\delta\dot{u}, \\ Y_0 + \delta Y + mg\delta\phi &= m\delta\dot{v}, \\ Z_0 + \delta Z + mg &= m\delta\dot{w}, \\ L_0 + \delta L &= I_{xx}\delta\dot{p}, \\ M_0 + \delta M &= I_{yy}\delta\dot{q}, \\ N_0 + \delta N &= I_{zz}\delta\dot{r}, \\ \delta\dot{\theta} &= \delta q, \\ \delta\dot{\phi} &= \delta p, \\ \delta\dot{\psi} &= \delta r. \end{aligned} \tag{3.32}$$

With the helicopter in hover trim, only the $Z$ force needs to compensate for gravity. All other forces and moments are zero.

Dropping the trim forces further simplifies the equations to

$$\begin{aligned}
\delta\dot{u} &= \frac{\delta X}{m} - g\delta\theta, \\
\delta\dot{v} &= \frac{\delta Y}{m} + g\delta\phi, \\
\delta\dot{w} &= \frac{\delta Z}{m}, \\
\delta\dot{p} &= \frac{\delta L}{I_{xx}}, \\
\delta\dot{q} &= \frac{\delta M}{I_{yy}}, \\
\delta\dot{r} &= \frac{\delta N}{I_{zz}}.
\end{aligned} \tag{3.33}$$

In order to linearise the set of equations, the forces and moments that exist in trim are expanded with a Taylor series. A first order approximation is considered. The force *X* in (3.33) for example, can be expanded as a Taylor series. This series is the sum of the all the forces' *X* contribution

$$X = X_0 + \frac{\partial X}{\partial u}\delta u + \frac{\partial^2 X}{\partial u^2}\delta u^2 + \frac{\partial X}{\partial w}\delta w + \frac{\partial^2 X}{\partial w^2}\delta w^2 + \ldots \tag{3.34}$$

For example, the *X* force produced by a change in heave velocity is $\frac{\partial X}{\partial w}\delta w$. For a shorter mathematical notation the derivatives are written as

$$\frac{\partial X}{\partial u} = X_u. \tag{3.35}$$

The partial derivatives are written in semi-normalised form as

$$X_u = \frac{X_u}{m} \text{ and } L_p = \frac{L_p}{I_{xx}}. \tag{3.36}$$

This Taylor series expansion of *X* can now be substituted back into the equations of motion, using the new notation. With everything now in small disturbance notation, the δ is omitted, which gives

$$\dot{u} = (X_u u + X_v v + X_w w + X_p + X_q + X_r + \ldots + X_{\delta_{lat}}\delta_{lat} + \ldots) - g\theta. \tag{3.37}$$

Each of the six equations of motion can now be rewritten in terms of the stability derivatives. However, in practice some derivatives have a negligible effect on the helicopter motion and can be omitted.

Mettler et al. [17] identify which of these derivatives are important and should be used for modelling a small-scale helicopter. The standard quasi-steady 6 DOF model is augmented with equations for the rotor tip path plane and the yaw rate gyro to arrive at the 11 state hybrid model.

The rotor tip-path plane can be described by the first order flapping equations

$$\begin{aligned} \dot{a} &= -q - \frac{a}{\tau_a} + A_b b + ... + A_{lon}\delta_{lon}, \\ \dot{b} &= -p - \frac{b}{\tau_a} + B_a a + ... + B_{lat}\delta_{lat}, \end{aligned} \tag{3.38}$$

with $a$ and $b$ the TPP angles, $A_b$ a cross coupling term, and $A_{lon}$ the input gain control derivative.

The rotor time constant is

$$\tau_f = \frac{16}{\gamma\Omega}, \tag{3.39}$$

where $\Omega$ is the rotor angular velocity and $\gamma$ is the rotor lock number. This can of course be calculated from first principles, but it will be estimated from flight data here.

In quasi-steady 6 DOF models such as explained in Padfield [13], the moments are directly related to the angular velocity by the stability derivatives, $L_p$, $M_q$ and $N_r$. The interaction between the pilot input and the fuselage motion in the hybrid model is not direct. Lateral input is applied to the rotor, which causes a flapping angle $b$, which is related to the moment $L$ by the stability derivative $L_b$. The moment causes an angular acceleration $\dot{p}$ which can then be integrated to find the velocity and then the attitude angle.

A simplified model for the yaw dynamics is given by

$$\frac{r}{\delta_{ped}} = \frac{N_{ped}}{s - N_r}, \tag{3.40}$$

with $N_r$ the yaw damping and $N_{ped}$ the sensitivity to the pedal input. The yaw rate feedback provided by the gyro can be modelled as a low pass filter

$$\frac{r_{fb}}{r} = \frac{K_r}{s + K_{r_{fb}}}. \tag{3.41}$$

The constraint

$$K_{r_{fb}} = -2N_r, \tag{3.42}$$

is added to allow identification. Only the yaw rate and pilot input are measured, and not the rate feedback, which makes (3.42) necessary [17].

A summary of the hybrid model equations with only important parameters included, is shown in Table 3.1.

**Table 3.1: Hybrid helicopter model governing equations**

Translational velocities:

$$\dot{u} = -g\theta + X_u u + ... + X_a a$$
$$\dot{v} = -g\phi + Y_v v + ... + Y_b b$$
$$\dot{w} = Z_u u + Z_w w + ... + Z_a a + Z_b b + Z_{col}\delta_{col}$$

Angular rates:

$$\dot{p} = L_u u + L_v v + ... + L_b b$$
$$\dot{q} = M_u u + M_v v + ... + M_a a$$
$$\dot{r} = N_r r + ... + N_{ped}(\delta_{ped} - r_{fb})$$

Attitude angles:

$$\dot{\phi} = p$$
$$\dot{\theta} = r$$

Flapping angles:

$$\dot{a} = -q - \frac{a}{\tau_a} + A_b b + ... + A_{lon}\delta_{lon}$$
$$\dot{b} = -p - \frac{b}{\tau_a} + B_a a + ... + B_{lat}\delta_{lat}$$

Yaw rate gyro:

$$\dot{r}_{fb} = -K_{r_{fb}} r_{fb} + K_r r$$

It is convenient to have the helicopter equations in state space form for identification, where

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \tag{3.43}$$

with state vector

$$\mathbf{x} = [u \quad v \quad p \quad q \quad \Phi \quad \Theta \quad a \quad b \quad w \quad r \quad r_{fb}]^T, \tag{3.44}$$

and input vector

$$\mathbf{u} = \begin{bmatrix} \delta_{lat} & \delta_{lon} & \delta_{col} & \delta_{ped} \end{bmatrix}. \tag{3.45}$$

The state matrix

$$\mathbf{A} = \left( \frac{\partial \mathbf{F}}{\partial \mathbf{x}} \right)_{\mathbf{x}=\mathbf{x}_0}, \tag{3.46}$$

contains the stability derivatives and is given by

$$\mathbf{A} = \begin{bmatrix} X_u & 0 & 0 & 0 & 0 & -g & X_a & 0 & 0 & 0 & 0 \\ 0 & Y_v & 0 & 0 & g & 0 & 0 & Y_b & 0 & 0 & 0 \\ L_u & L_v & 0 & 0 & 0 & 0 & L_a & L_b & 0 & 0 & 0 \\ M_u & M_v & 0 & 0 & 0 & 0 & M_a & M_b & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1/\tau_a & A_b & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & B_a & -1/\tau_a & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & Z_a & Z_b & Z_w & Z_r & 0 \\ 0 & 0 & N_p & 0 & 0 & 0 & 0 & 0 & N_w & N_r & N_{rf} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_r & K_{rf} \end{bmatrix}. \tag{3.47}$$

The input matrix

$$\mathbf{B} = \left( \frac{\partial \mathbf{F}}{\partial \mathbf{u}} \right)_{\mathbf{x}=\mathbf{x}_0}, \tag{3.48}$$

contains the control derivatives and is given by

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ A_{lon} & A_{lat} & 0 & 0 \\ B_{lon} & B_{lat} & 0 & 0 \\ 0 & 0 & Z_{col} & 0 \\ 0 & 0 & N_{col} & N_{ped} \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.49}$$

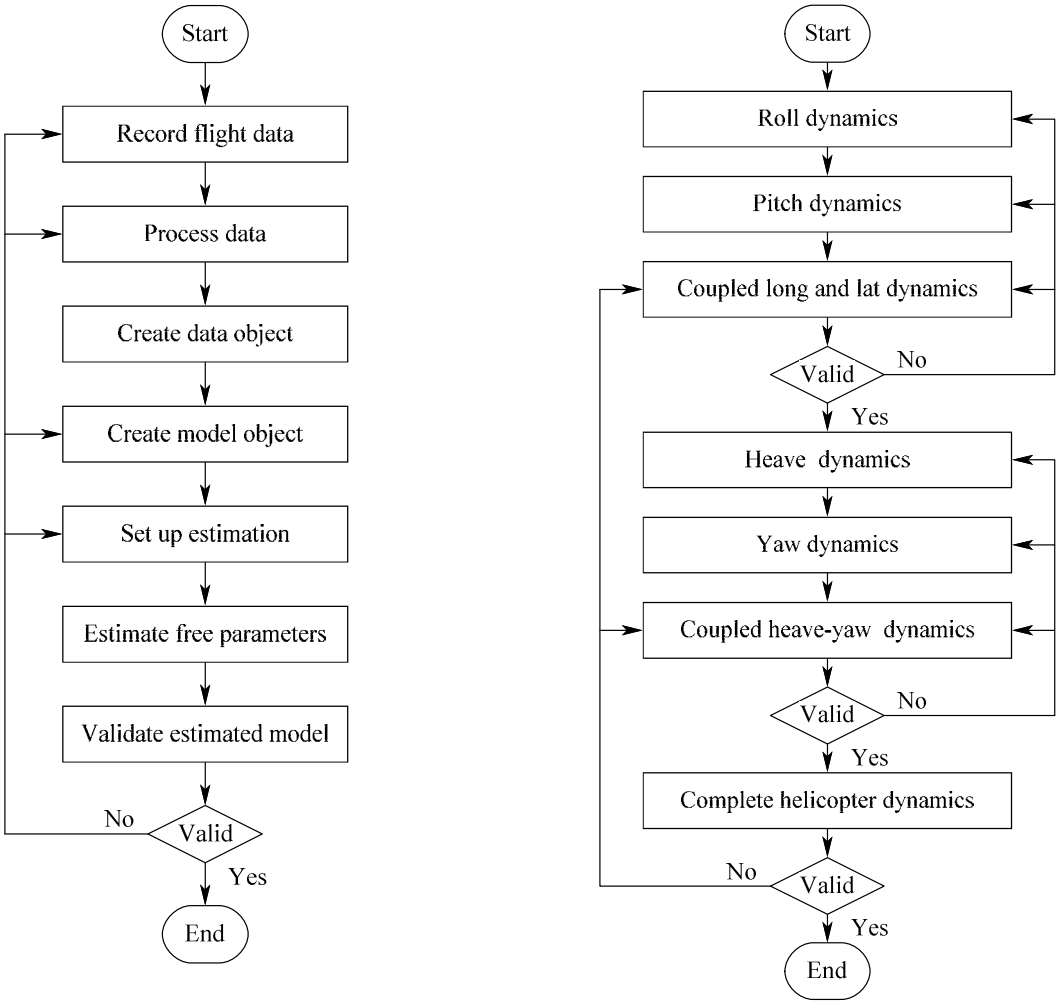
These stability and control derivatives can be estimated through system identification or calculated from first principles. System identification will be used in this study.

The stability derivatives can be used to evaluate the performance of the aircraft and provide information on how the craft responds to inputs and disturbances. A brief summary of the important parameters and the dynamics they dictate are included in Appendix A.

The model used here does not include stabiliser bar dynamics. AeroSIMRC does not explicitly model the stabiliser bar. Its dynamics are lumped with the rotor and should be identified together with the rotor. The heading angle $\psi$ is not included in the state space model because it does not influence the helicopter dynamics. None of the stability derivatives are dependent on the heading.

### 3.1.4 Conclusion

The equations necessary to describe the helicopter motion have been derived. The equations were linearised and written in state space form. This state space model van now be used in the next section on system identification.




**a) Subsystem identification workflow in Matlab®** **b) Subsystem identification methodology**

**Figure 3.4: System identification workflow for subsystems and complete model**

## 3.2 System identification

The process followed to identify and validate the helicopter will be described here. The model is broken down into subsystems for easier identification. Once the complete model has been identified reasonably well, a set of multiple experiments is used to generalise the model. The model is then validated with a different data set.

The methodology followed here is based on the research done by Yuan [30], Mettler [17] and Shim [16]. Whereas Yuan used NASA's SIDPAC toolbox, Mettler used CIFER® and Shim used PEM for parameter estimation.

Figure 3.4(a) shows the workflow for identifying a subsystem. It follows the standard system identification process discussed in Chapter 2. Once the estimation algorithm and model structure have been chosen they are not changed often. The most frequent changes are to the flight data recording and data processing.

Figure 3.4(b) shows the process followed when identifying the different sub-systems. As can be seen from the flow diagram, it is an iterative process. The subsystems follow each other so the values from the previous estimation can be used as initial values in the next estimation. If these initial values do not provide adequate results, the previous subsystem must be estimated once again.

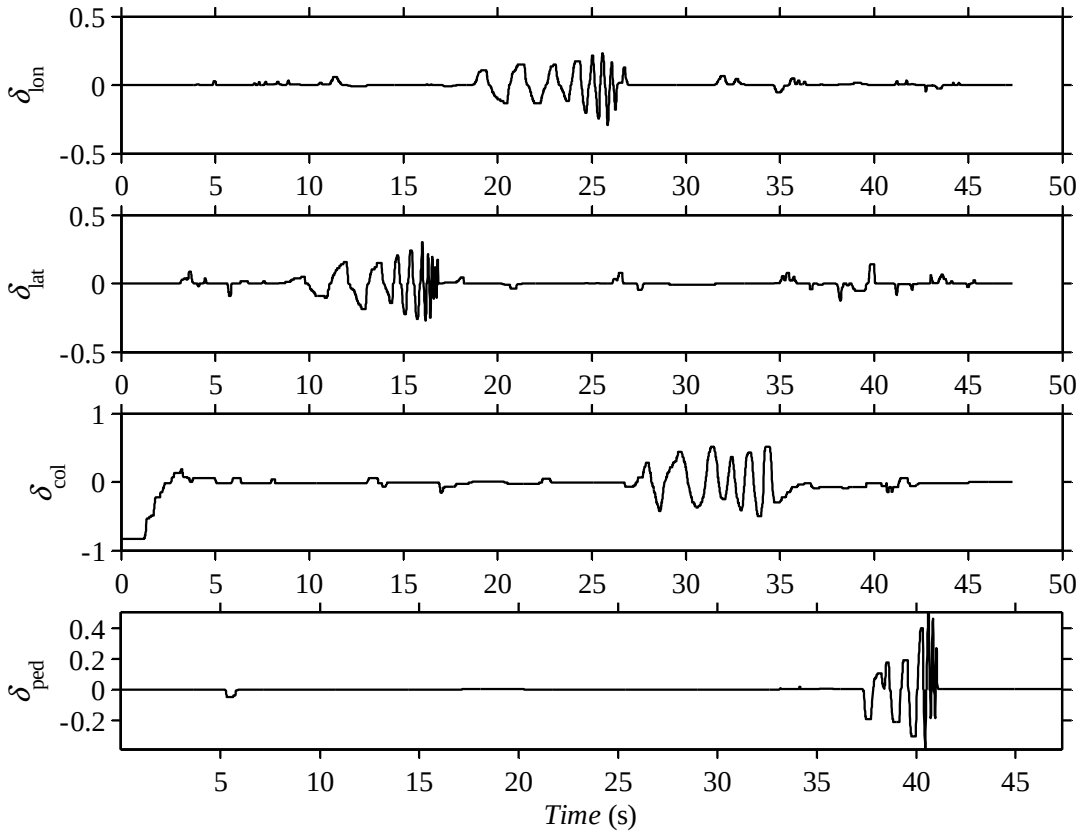



**Figure 3.5: Typical frequency sweep input data**

A set of frequency sweep flight data is shown in Figure 3.5. This is a typical set that would be used to estimate the complete helicopter dynamics and has an excitation signal on each axis. Similar sets of data were used to estimate each subsystem, in those cases only exciting the applicable axis. The collective input starts at -1 when the helicopter is on the ground and is increased until trimmed hover flight. Stabilising inputs are given on the inputs that are not being excited.

### 3.2.1 Identification methodology

The subsystems in Figure 3.4(b) are now explained and the procedure for estimation is given. Each subsystem's state space model is given, as well as the parameters that need to be estimated and the input-output data that should be used.

**1. Roll dynamics**

$$\begin{bmatrix} \dot{v} \\ \dot{p} \\ \dot{\Phi} \\ \dot{b} \end{bmatrix} = \begin{bmatrix} Y_v & 0 & g & g \\ L_v & 0 & 0 & L_b \\ 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & -1/\tau_a \end{bmatrix} \begin{bmatrix} v \\ p \\ \Phi \\ b \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ B_{lat} \end{bmatrix} \delta_{lat} \tag{3.50}$$

The PEM estimator had no difficulty identifying the lateral translational dynamics lumped with the roll dynamics. The outputs $v$ and $p$ are used, with the input $\delta_{lat}$ to estimate the parameters $Y_v$, $L_v$, $L_b$ and $B_{lat}$. The rotor time constant $\tau_a$ is also estimated, but it does not converge consistently and is fixed to a value determined through trial and error in the coupled longitudinal and lateral identification step. The initial values for all the parameters are the values identified by Yuan for the Eagle RC helicopter [30]. These values should be closer to the size 30 helicopter of the simulator than the big Yamaha RMAX's values. The rotor time constant initial value is calculated with

$$\tau_a = \frac{5}{\Omega_R / 60}, \tag{3.51}$$

where $\Omega_R$ is the rotor speed, taken as 1800 r/min. The parameter $g$ is the gravitational constant and is fixed to 9.81 m/s.

**2. Pitch dynamics**

$$\begin{bmatrix} \dot{u} \\ \dot{q} \\ \dot{\Theta} \\ \dot{a} \end{bmatrix} = \begin{bmatrix} X_u & 0 & -g & -g \\ M_u & 0 & 0 & M_a \\ 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & -1/\tau_a \end{bmatrix} \begin{bmatrix} u \\ q \\ \Theta \\ a \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ A_{lon} \end{bmatrix} \delta_{lon} \tag{3.52}$$

The pitch dynamics are similar to the roll dynamics and the estimator was able to identify the longitudinal translational dynamics lumped with the pitch dynamics. The outputs $u$ and $q$ are used with the input $\delta_{lon}$. The parameters that are free for estimation are $X_u$, $M_u$, $M_a$ and $A_{lon}$. The rotor time constant is refined from the previously estimated value. All other free parameters are once again started from the Eagle RC's values.

**3. Coupled longitudinal and lateral dynamics**

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{p} \\ \dot{q} \\ \dot{\Phi} \\ \dot{\Theta} \\ \dot{a} \\ \dot{b} \end{bmatrix} = \begin{bmatrix} X_u & 0 & 0 & 0 & 0 & -g & -g & 0 \\ 0 & Y_v & 0 & 0 & g & 0 & 0 & g \\ L_u & L_v & 0 & 0 & 0 & 0 & L_a & L_b \\ M_u & M_v & 0 & 0 & 0 & 0 & M_a & M_b \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1/\tau_a & A_b \\ 0 & 0 & -1 & 0 & 0 & 0 & B_a & -1/\tau_a \end{bmatrix} \begin{bmatrix} u \\ v \\ p \\ q \\ \Phi \\ \Theta \\ a \\ b \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ A_{lon} & A_{lat} \\ B_{lon} & B_{lat} \end{bmatrix} \begin{bmatrix} \delta_{lon} \\ \delta_{lat} \end{bmatrix} \tag{3.53}$$

Equations (3.50) and (3.52) are combined to form the coupled longitudinal and lateral state space model. All the inputs and outputs that were used in the roll and pitch identification steps are now used. The cross coupling parameters $L_a$, $M_b$, $L_u$, $M_v$, $A_b$, $B_a$, $A_{lat}$ and $B_{lon}$ are added. The initial values for these new parameters are set to those of the Eagle RC and the rest of the parameters start from their previously estimated values. The system easily became unstable during the estimation process if $\tau_a$ was either too big or too small. An appropriate value for $\tau_a$ was chosen iteratively and not estimated with the rest of the parameters.

**4. Heave dynamics**

$$\dot{w} = Z_a a + Z_b b + Z_w w + Z_r r + Z_{col}\delta_{col} \tag{3.54}$$

Only a single input and output is used in this step, namely $\delta_{col}$ and $w$. The cross coupling terms $Z_a$ and $Z_b$, as well as the yaw rate response $Z_w$, are left out of the estimation. This step is only concerned with identifying the heave dynamics.

**5. Yaw dynamics**

$$\begin{bmatrix} \dot{r} \\ \dot{r}_{fb} \end{bmatrix} = \begin{bmatrix} N_r & N_{rf} \\ K_r & K_{rf} \end{bmatrix} \begin{bmatrix} r \\ r_{fb} \end{bmatrix} + \begin{bmatrix} N_{ped} \\ 0 \end{bmatrix} \delta_{ped} \tag{3.55}$$

$$N_{rf} = -N_{ped} \tag{3.56}$$

$$K_{rf} = 2N_r \tag{3.57}$$

Equations (3.56) and (3.57) are needed, since the rate feedback, $r_{\text{fb}}$, is not measured. The yaw feedback system is modelled as a low pass filter. Equation (3.57) states that the poles of the feedback system are twice as fast as the helicopter airframe's yaw response [28]. These constraints were enforced in the estimation process. The unknown parameters $N_{\text{rf}}$, $K_{\text{rf}}$ and $N_{\text{ped}}$ are estimated using the input $\delta_{\text{ped}}$ and the output $r$. The yaw rate feedback gyro equipped to the helicopter in the simulator creates a well-behaved system that is easy to identify.

### 6. Coupled heave and yaw dynamics

$$\begin{bmatrix} w \\ r \\ r_{fb} \end{bmatrix} = \begin{bmatrix} Z_w & Z_r & 0 \\ N_w & N_r & N_{rf} \\ 0 & Z_r & K_{rf} \end{bmatrix} \begin{bmatrix} w \\ r \\ r_{fb} \end{bmatrix} + \begin{bmatrix} Z_{col} & 0 \\ N_{col} & N_{ped} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{col} \\ \delta_{ped} \end{bmatrix} \tag{3.58}$$

As the model was iteratively refined, it became unnecessary to estimate the heave and yaw parameters separately. It was only necessary to determine the initial values for further estimation of the coupled system. The coupled heave and yaw model could estimate all the parameters simultaneously. The outputs are $w$, $r$ and the inputs are $\delta_{\text{col}}$ and $\delta_{\text{ped}}$.

### 7. Complete dynamics

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{p} \\ \dot{q} \\ \dot{\Phi} \\ \dot{\Theta} \\ \dot{a} \\ \dot{b} \\ \dot{w} \\ \dot{r} \\ \dot{r}_{fb} \end{bmatrix} = \begin{bmatrix} X_u & 0 & 0 & 0 & 0 & -g & -g & 0 & 0 & X_r & 0 \\ 0 & Y_v & 0 & 0 & g & 0 & 0 & g & 0 & Y_r & 0 \\ L_u & L_v & 0 & 0 & 0 & 0 & L_a & L_b & 0 & 0 & 0 \\ M_u & M_v & 0 & 0 & 0 & 0 & M_a & M_b & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1/\tau_a & A_b & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & B_a & -1/\tau_a & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & Z_a & Z_b & Z_w & Z_r & 0 \\ 0 & 0 & N_p & N_q & 0 & 0 & 0 & 0 & N_w & N_r & N_{rf} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_r & K_{rf} \end{bmatrix} \begin{bmatrix} u \\ v \\ p \\ q \\ \Phi \\ \Theta \\ a \\ b \\ w \\ r \\ r_{fb} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ A_{lon} & A_{lat} & 0 & 0 \\ B_{lon} & B_{lat} & 0 & 0 \\ 0 & 0 & Z_{col} & 0 \\ 0 & 0 & N_{col} & N_{ped} \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{lon} \\ \delta_{lat} \\ \delta_{col} \\ \delta_{ped} \end{bmatrix} \tag{3.59}$$

The complete model is formed by combining the coupled heave and yaw model with the coupled longitudinal and lateral model.

Three parameters, $X_{\text{r}}$, $Y_{\text{r}}$ and $N_{\text{q}}$ were added to the original state space model. These parameters were added after unmodelled cross coupling from the yaw rate to the translational velocity was

observed. This is the final estimation and all the parameters from the previous steps are estimated from their predetermined values. $N_q$, $X_r$, $Y_r$, $Z_a$ and $Z_b$ are new parameters with initial values of zero. All six of the outputs from the simulator and all four of the inputs are used.

When estimating so many parameters simultaneously, the PEM algorithm has difficulty in finding a stable model. This makes the initial values determined in the previous steps so important. If these values are not close to the true plant parameters, the estimator will not be able to find a stable model. Since system identification is an iterative process, it was often necessary to go back to the coupled longitudinal-lateral and coupled heave-yaw steps to find new parameter initial values.

### 3.2.2 Verification of methodology

To verify the proposed *methodology*, it was applied to the known state space model of Shim [16]. The model is excited with doublet inputs. This process made it possible to see the effect of using different initial values. Identifying the known state space model is relatively simple. All the dynamics that are present in the known model can be modelled by the state space model that is being used for identification. There are no disturbances and the final values of the parameters are known, making the choice of the initial conditions a test, rather than a problem. The identified model showed a near perfect match, with a NRMSE value of 98%. Once the identification process is verified, it can be applied to the more difficult problem of identifying the true plant.

### 3.2.3 Merged experiments

It is difficult to create one experiment where all of the dynamics are sufficiently excited, so using multiple experiments makes this task easier. No two experiments are the same, because human pilots fly the aircraft. By repeating several experiments multiple times, one merged experiment is produced that contains better data for identification.

The identified hover model should be a representative of the helicopter parameters and not of parameters that give a good fit value to a single experiment. This makes using multiple experiment data even more attractive, since not only will the merged experiment include a much wider range of excitation signals, it will also prevent over-training the parameters and simply blindly fitting them to one experiment. This will result in a better average hover model. In a model estimated from multi-experiment data, Matlab® assigns a weighted mean to the parameters, using the covariance matrix. Parameters are estimated from the experiments where they have the lowest uncertainty.

It is important that the data used in multiple experiments are from the same operating point. To help choose data sets in the identification process, a graphical method is used. By first estimating the

model on all of the experiments, it is assumed that the majority of the experiments are in the correct operating condition. This assumption is not unusual, as one would have assumed that the helicopter is in hover if the parameters were estimated from a single data set. The estimated model's fit to different sets of estimation data can be plotted on a bar graph for visual comparison. This allows the identification of experiments that possibly contain nonlinear characteristics. These experiments can be removed from the estimation data set, and replaced with more suitable linear flight data. More on this process in Appendix A.

### 3.2.4 Validation methodology

The model is continuously validated as the identification process is followed. At each step, the model's predicted output is compared to the validation data. It is clear from Figure 3.4 that when a coupled system's performance is not satisfactory, the parameters in the previous decoupled system are estimated again. If this does not solve the problem, it is safe to assume that the flight data being used is the problem. Either the flight can be recorded again, or a different range of data from the experiment can be used.

To validate the variance of the parameters in the model, each parameter is perturbed from its nominal value at random, within one standard deviation. These models with random variables can then be compared to decide if the variance in the output, caused by the change in parameter values, is acceptable.

To verify the internal structure of the model, the parameters are constrained to physically meaningful values. Other simpler methods of verification are also used. A 3D animation of the helicopter is seen as the simulations are run, which makes it easier to visualise the helicopter and notice discrepancies.

## 3.3 Simulink®/AeroSIMRC interface

In this section the development of the interface between Simulink® and the simulator is given. The interface uses a UDP connection to exchange data between the two programs. The interface enables validation of the model and controllers in a software environment.

Figure 3.6 shows the interfaces between Simulink® and AeroSIMRC. AeroSIMRC's has an interface through which a plug-in can access its internal variables and override some of these variables. The plug-in is developed in the form of a Dynamic Link Library (DLL) that is called once every program cycle. AeroSIMRC sends a data structure of internal variables to the plug-in, and a structure with instructions is received back from the plug-in.

To make the data from AeroSIMRC available for system identification in Matlab®, a plug-in was developed that can save the internal variables of the simulator to a standard ASCII text file. The file can then be parsed by Matlab® for the identification process. This is specifically for experiments where the pilot is controlling the helicopter and no feedback from Matlab® is necessary.

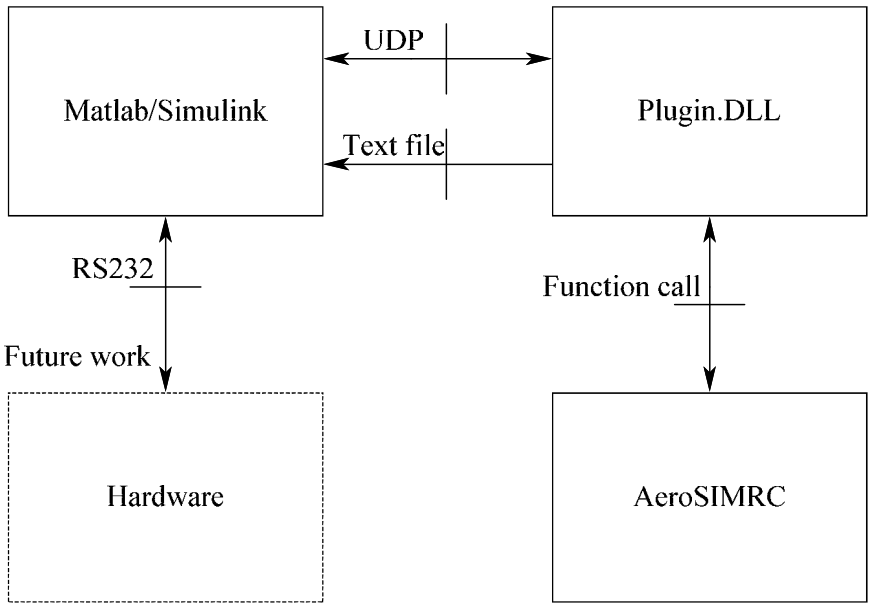



**Figure 3.6: System architecture of test environment**

For real time control, the plug-in needs to receive the helicopter states, calculate control inputs, and reply with the new inputs to the simulator. This process is repeated with every program cycle that AeroSIMRC calls the plug-in DLL.

It is considerably more convenient to develop control systems in Matlab® and Simulink® than to start with C or another low level programming language. This is especially true in the early development phases. Therefore, it was decided to interface the plug-in with Simulink® and use it only to relay data between AeroSIMRC and Simulink®.

Establishing communication between Simulink® and the plug-in DLL is discussed next. The two most common solutions to communication between Simulink® and other applications on the same system are using sockets or shared memory. Shared memory is ideal for applications that need to use a common block of data and access it fast and often. Synchronisation can become problematic if more than two programs access the data, but in this case, no more than two will be used.

Sockets, specifically network sockets, have the advantage that synchronisation is easy to deal with and multiple clients can access the data if it is multicast. Network sockets are easy to use, maintain and scale. Another advantage is the processes that are communicating can be on different computers and even different operating systems. The bandwidth of the system is below 1 kHz with data

packets of less than 1 KB. These slow and small transmissions are ideally suited to socket communication. It is common practice to use network sockets for control system testing between Matlab® and an external application [58], [41], [73], [64].

X-plane comes with a UDP output option, which makes its data available for any program listening. A similar approach will be followed here with AeroSIMRC. Simulink® and Matlab® have built in functions to handle Serial, UDP, and TCP communication. UDP is a low overhead, error correctionless transmission protocol, ideal for situations where real-time operation is important. Packets are not guaranteed to arrive in the correct order (or even arrive at all), but there will never be a delay waiting for packets. UDP sockets can be used for easy communication between applications. Packet loss will be minimal, if not zero, if it is set up to run in local loopback mode. The basic program flow is shown in Figure 3.7.

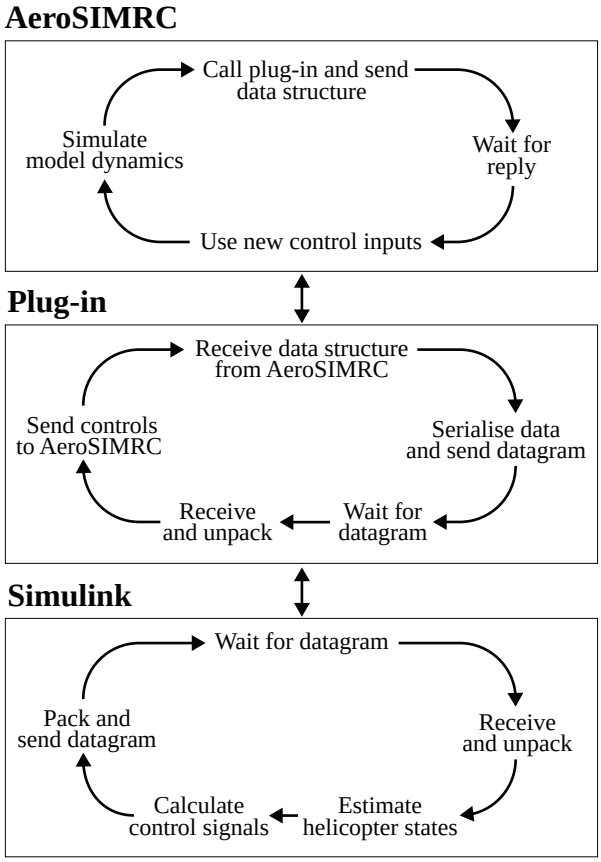



**Figure 3.7: Program flow for simulator interface**

Using a game based simulator comes with certain disadvantages that have to be managed. AeroSIMRC does not have a fixed rate at which the program cycle calls the plug-in. If the hardware it is running on cannot render fast enough, it will start to increase the integration time step between

frames to try to maintain real time operation. The plug-in is called once every frame and this creates a peculiar problem, since a fixed sampling rate is needed in Simulink®.

AeroSIMRC has a variable for the *model's* maximum time step. This ensures that, even if the hardware is very slow, the model does not become unstable, instead slowing down the frame rate and ensuring that the *model's* maximum time step is not exceeded.

At a frame rate of 120 Hz, or a time step of 8.3 ms, to keep a model that has a maximum time step of 2 ms real time, AeroSIMRC will have to do 8.3/2 = 4 integration steps per frame. If the hardware cannot handle calculating the model dynamics 4 times per frame, that is to say, in 8.3ms, the simulation will slow down. A workaround for this problem is to record a video during simulation which forces the simulation to keep a fixed sampling rate. The model's maximum integration time step can then be set as low as possible. Real time execution will be lost, but the accuracy of the simulation will increase.

A Simulink® block for the AeroSIMRC interface is created, making it possible to simply remove the previously used state space model and replace this with the new AeroSIMRC model block. This enables rapid testing of control algorithms on the simulator.

## 3.4 Modelling results

This section includes the results from the modelling process. The identified stability derivatives, the model fit to validation data, and a further discussion of the validation results are given.

### 3.4.1 Identified model

Table 3.2 contains the final stability derivatives and their uncertainty expressed as the standard deviation estimated from a merged experiment set. To give a more useful value to the uncertainty, the Relative Standard Deviation (RSD) is used. The RSD is calculated with

$$\%RSD = \frac{s}{\bar{x}} \times 100, \tag{3.60}$$

where $s$ is the standard deviation and $\bar{x}$ is the mean value of the estimation. A low RSD indicates a small uncertainty of an estimated parameter. The stability derivatives relating to translational velocity show extremely high uncertainty. This was experienced in the identification process where these parameters did not converge well. Parameters with such high uncertainties could arguably be left out of the model. The same problem with the velocity derivatives were observed by Valavanis [14] and Mettler [17].

The parameters that have been added to the model to account for cross coupling observed in the data also show a high uncertainty, and their usefulness is questionable. The rotational and heave-yaw parameters show very low uncertainty. This is a good indication that the model correctly accounts for these dynamics. Parameters that could not be estimated freely in experiments, and were iteratively determined, have no information pertaining to their uncertainty. Estimation seems to show a very high uncertainty, this is due to the insignificant parameters being left free for estimation, and the more important parameters being set to fixed values. The parameters with no information on uncertainty were determined iteratively from separate experiments and not estimated again once good fit values are obtained. It can be seen in Appendix B, that these fixed parameters produce excellent results on multiple experiments.

**Table 3.2: Stability derivatives with uncertainty of estimates**

| | **Value** | **SD** | **RSD %** | | **Value** | **SD** | **RSD %** | | **Value** | **SD** | **RSD %** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $L_u$ | -0.0603 | 0.047 | 79.06 | $X_u$ | -0.0211 | 0.091 | 431.75 | $A_b$ | 0 | - | - |
| $L_v$ | 0.1689 | 0.260 | 154.41 | $X_r$ | -0.8741 | 0.682 | 78.12 | $B_a$ | 0 | - | - |
| $L_a$ | 0 | - | - | $X_a$ | $-g$ | - | - | $A_{lon}$ | -2.991 | 0.101 | 3.37 |
| $L_b$ | 5642 | 74.5 | 1.32 | $Y_v$ | -0.1059 | 0.165 | 155.96 | $A_{lat}$ | -0.003 | - | - |
| $M_u$ | -0.1697 | 0.292 | 172.18 | $Y_b$ | $g$ | - | - | $B_{lon}$ | 2.925 | - | - |
| $M_v$ | -0.3889 | 0.931 | 239.44 | $Y_r$ | 0.1358 | 0.168 | 124.37 | $B_{lat}$ | 0 | - | - |
| $M_a$ | 7365 | 73.98 | 1.00 | $Z_a$ | 300 | - | - | $1/\tau_a$ | 111 | - | - |
| $M_b$ | 0 | - | - | $Z_b$ | -300 | - | - | | | | |
| $N_p$ | -1.955 | 0.149 | 7.64 | $Z_w$ | -3.536 | 0.034 | 0.98 | | | | |
| $N_q$ | -0.004 | 0.111 | 2775 | $Z_r$ | 0.2659 | 0.096 | 36.17 | | | | |
| $N_w$ | -2.952 | 0.142 | 4.83 | $Z_{col}$ | -19.37 | 0.168 | 0.86 | | | | |
| $N_r$ | -14 | 0.1016 | 0.72 | | | | | | | | |
| $N_{rf}$ | 166.4 | 0.597 | 0.35 | | | | | | | | |
| $N_{col}$ | -15.05 | 0.2007 | 1.33 | | | | | | | | |
| $N_{ped}$ | 166.4 | 0.597 | 0.35 | | | | | | | | |
| $K_r$ | -1.992 | 0.0098 | 0.49 | | | | | | | | |
| $K_{rf}$ | -28.7 | 0.2032 | 0.70 | | | | | | | | |

Equations (3.61) and (3.62) give the same results as Table 3.2, in state space form

$$\mathbf{A} = \begin{bmatrix} -0.0211 & 0 & 0 & 0 & 0 & -g & -g & 0 & 0 & -0.8741 & 0 \\ 0 & -0.1059 & 0 & 0 & g & 0 & 0 & g & 0 & 0.1358 & 0 \\ -0.06033 & 0.1689 & 0 & 0 & 0 & 0 & 0 & 5642 & 0 & 0 & 0 \\ -0.1697 & -0.3889 & 0 & 0 & 0 & 0 & 7365 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -111 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & -111 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 300 & -300 & -3.536 & 0.2659 & 0 \\ 0 & 0 & -1.955 & -0.004 & 0 & 0 & 0 & 0 & -2.952 & -14 & 166.4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.992 & -28.7 \end{bmatrix}, \quad (3.61)$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -2.991 & -0.003 & 0 & 0 \\ 0 & 2.925 & 0 & 0 \\ 0 & 0 & -19.37 & 0 \\ 0 & 0 & -15.05 & 166.4 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad (3.62)$$

The estimated model parameters show some interesting differences between the simulator and real helicopters identified in other studies. A rotor time constant, $\tau_f$, of 0.009 s is very small. Even though this is a small agile helicopter with a stiff rotor, this seems too fast to be physically possible.

This simulator is designed as a training tool and not as a helicopter analysis tool, which could be the reason for the big discrepancy. The roll and pitch rates show very good NRMSE values with low uncertainty, so this value has very likely been estimated correctly. The reason for the difference is that the simulator is designed to have such a stiff rotor.

It can be expected that a small agile rotorcraft will have a bigger on-axis rotor spring coefficient than for example the Yamaha RMAX, but the identified value is significantly larger than that of the small Eagle RC helicopter. This makes control easier for the pilot, as the control inputs map more directly to the roll and pitch rate outputs.

$B_a$, $A_b$, $L_a$ and $M_b$ have been fixed to zero in the estimation process, since these rotor cross couplings could not be observed in the flight data. Either they are not present in the simulator, or they are too small to be significant. However, the cross coupling from heave and collective to yaw rate was very prominent. It was also pointed out in the methodology that several parameters had to be added for unmodelled effects. The fact that some of the off-axis derivatives are zero once again point to a simplification of the helicopter to make flying easier for a RC hobby pilot in training.

It is commonly observed that the longitudinal and lateral velocity derivatives are unreliable and show a high degree of uncertainty, which was also the case for the identified values of the simulator [14], [17]. It was challenging to find consistent values for these variables throughout several experiments. This is also due to the difficulty the pilot had in maintaining a stable hover throughout the flight experiments.

Some key hover parameters identified by Valavanis for a custom Raptor 90 model in X-plane, a Hirobo 60 Eagle used by Yuan, and the big Yamaha RMAX used by Mettler are given in Table 3.3. These can be compared to the 30 Raptor from AeroSIMRC that was identified in this study.

**Table 3.3: Difference in key estimated parameters**

| Derivative | 30 Raptor AeroSIMRC | 90 Raptor X-plane [14] | Hirobo 60 Eagle [30] | Yamaha RMAX [28] |
|---|---|---|---|---|
| $L_b$ | 5642 | 1172.4817 | 331.24 | 142.5 |
| $M_a$ | 7365 | 307.571 | 199.36 | 67.74 |
| $L_a$ | 0 | 0 | 36.688 | 22.14 |
| $M_b$ | 0 | 0 | -41.970 | -7.366 |
| $A_b$ | 0 | 0.7713 | -6.7924 | 0 |
| $B_a$ | 0 | 0.6168 | -1.7674 | 0.5543 |
| $X_u$ | -0.0211 | -0.03996 | -2.7594 | -0.09865 |
| $Y_v$ | -0.1059 | -0.05989 | -0.2345 | -0.2289 |
| $Z_a$ | 300 | 0 | 24.106 | -28.85 |
| $Z_b$ | -300 | 0 | 35.511 | -121.2 |
| $Z_w$ | -3.536 | -2.055 | -1.0540 | -0.5024 |
| $\tau_f$ | 0.009 | 0.032 | 0.1064 | 0.3753 |
| $A_{lon}$ | -2.991 | 4.059 | -0.4556 | -0.3824 |
| $B_{lon}$ | 0 | -0.01017 | -0.2322 | 0.03773 |
| $A_{lat}$ | -0.003 | -0.01610 | -0.0560 | 0.05685 |
| $B_{lat}$ | 2.925 | 4.085 | 0.4228 | 0.4448 |
| $Z_{col}$ | -19.37 | -13.11 | -7.0887 | 40.23 |

All of the models have smaller rotor spring derivatives and larger time constants, which will make them less responsive to pilot inputs. The estimated model is closest to the X-plane model, showing that the simulators have much higher rotor spring coefficients and smaller cross coupling than the real world helicopters. This supports the previous hypothesis that the simulators have been made easier to fly.

### 3.4.2 Model accuracy

The model's performance in predicting a manoeuvre is shown in this section. The data used is a separate set of validation data that is not used to estimate the helicopter model. The model tends to follow the translational velocities well, but these are still the worst of all eight measurable outputs. These velocities also show the highest uncertainty. Velocity is more difficult to estimate than angular rates. A possible explanation could be that there is not sufficient information contained in the flight data, since long translational movements tend to shift the helicopter from hover to cruise flight. This validation set shows the helicopter performing several frequency sweeps in hover. Hover is considered anything below 3 m/s. In many other experiments, the heave velocity showed the worst fit because of the complex airflow through the rotor. The model cannot represent the nonlinear dynamics present in the simulator. It is clear from Figure 3.8 that the model has slight offsets from the validation data, but it does not seem to miss any crucial dynamics. This is good for control design and these offsets should not pose a problem in a feedback controller.

The rotor dynamics dominate the helicopter. The angular rates have very little uncertainty and show very good fit values. The simulator shows less cross coupling than what is seen in the literature on other helicopters. Another reason for the good tracking of the angular rates is the simulator's extremely stiff rotor, giving it an approximately linear relation between the inputs and the pitch and roll rate outputs. The yaw rate is relatively easy to model when yaw inputs are given, because of the rate gyro. When no yaw inputs are given, and only the vertical stabiliser and cross coupling from the heave channel influence the yaw rate, the model's prediction deteriorates. This is not clearly visible in Figure 3.8, but can be observed in other data sets.

The roll and pitch angles are of less importance than the angular rates and translational velocities. These angles are simply the integrated form of another state variable. The angular rates showed a good fit, and logically so does the attitude angles. The results given illustrate the model's ability to give a general representation of the helicopter in hover.

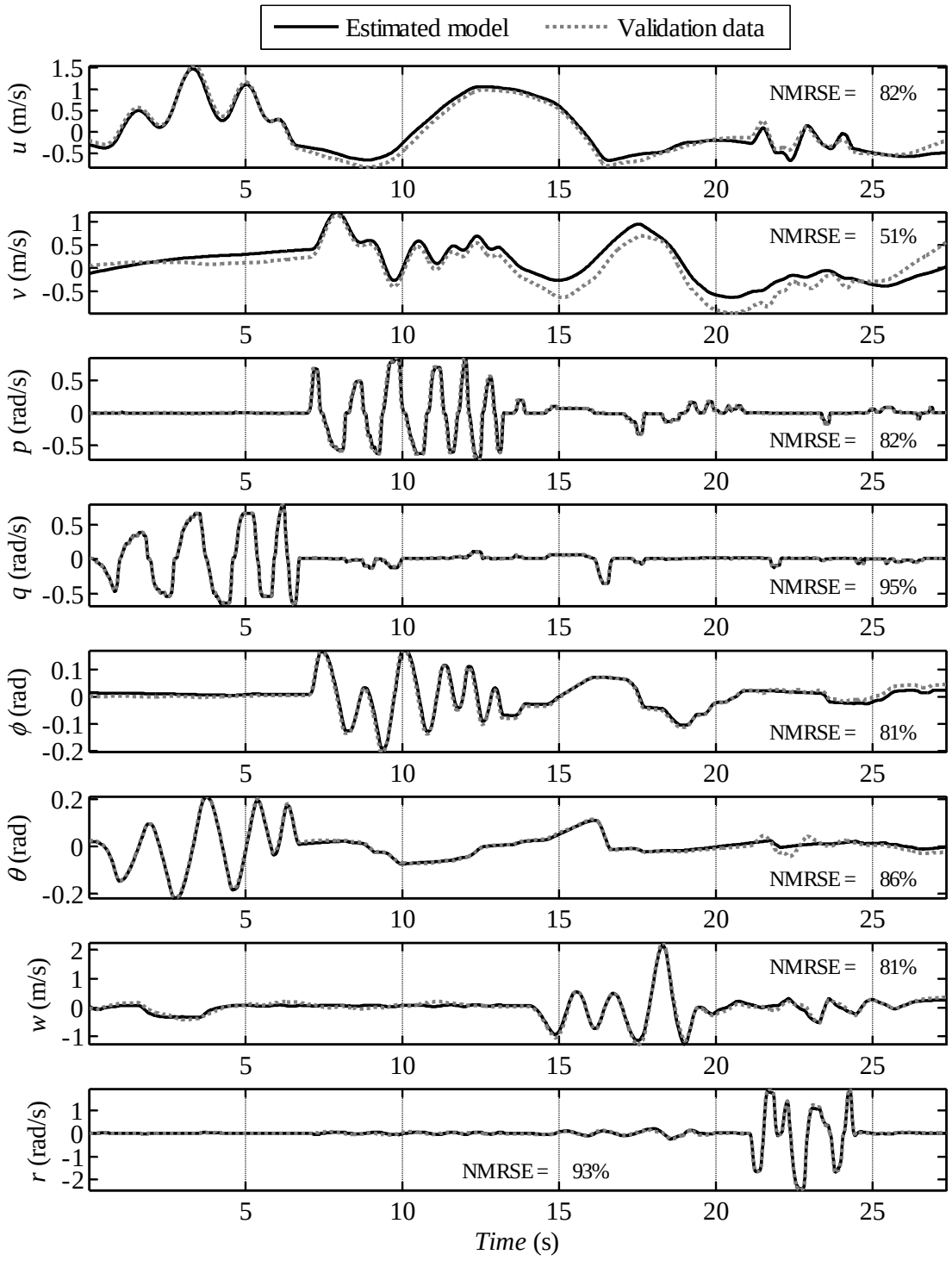



**Figure 3.8: Frequency sweep validation data**

### 3.4.3 Model validation

The first goal of model validation should be to validate the internal structure of the model [74]. The model structure used has been verified and validated several times in other studies on helicopter identification, but it has never been used to identify a model for AeroSIMRC. AeroSIMRC's internal working is essentially a black box. It is clear *what* the simulator is simulating, but

information on *how* it is doing so is not available. This makes it important to validate that the model's internal structure is capable of modelling the helicopters in AeroSIMRC. It seems it would prove the inaccuracy of AeroSIMRC rather than the model, if this structure were not sufficient. A parameter confirmation test is used to validate the structure. Here each estimated parameter is compared to what can physically be expected. All the stability derivatives have the correct sign and even though the sizes are significantly different, the differences can be explained.

A residual analysis shows very little correlation between the inputs and the errors, as well as between the outputs and past values of the outputs. These tests are applicable to output error models. This indicates that the model captures the important dynamics of the helicopter well enough and that the model structure is able to predict the plant response. An important factor to consider here is that the model output was compared to data that was not used to estimate the model; this means the model is general enough to represent the helicopter in hover. The linear model will however fail to predict two types of nonlinear behaviour, namely aggressive manoeuvres and nonlinearities in the helicopter hardware. With aggressive manoeuvres the physical laws are nonlinear. With hardware there are inherent nonlinearities in the form of saturations, such as servo rates, and maximum actuator movement [17].

The NRMSE results from the previous section are the primary proof of the model validity. The model is able to predict the validation flight data with sufficient accuracy. The model validation will be even further proven in Chapter 5, where the controller is evaluated.

## 3.5 Conclusion

A methodology for successfully identifying helicopter stability derivatives was developed by using proven model structures from the literature and adding a systematic description of how each model subset can be identified. The method is verified on a known model, it is then applied to a helicopter in AeroSIMRC, and the identified model is validated using a separate set of flight data. This model can now be used to design a controller in the next chapter.

# Chapter 4: Controller development

*In this chapter, a cascaded approach to controller development is followed. This results in a fast inner loop and a slower outer loop. The LQR controller for the inner loop is developed first. Four decoupled SISO PID controllers are then used for the outer loop. They provide reference signals to the LQR controller. The outer loop is optimised using a simplex search method. Verification results are presented where the state space model follows a 3D figure 8 reference trajectory with a constant yaw rate.*

## 4.1 Cascaded controller development

The helicopter's translational motion is considerably slower than its angular motion. This makes it possible to use a cascaded controller, with the slower outer loop controlling translation, and the faster inner loop controlling rotation [75]. This configuration is shown in Figure 4.1. The inertial position reference trajectory ($x$, $y$, $z$, $\psi$) to the outer loop would typically be provided by a guidance controller. The outer loop then provides a roll, pitch, heave velocity, and yaw rate ($\phi$, $\theta$, $w$, $r$) reference to the inner loop. A full-state feedback optimal control law is developed for the inner loop. This decouples the helicopter dynamics, making it possible to control the translational motion with four SISO PID controllers.

Optimal controllers are not designed with the same objectives as classical controllers: rise time, settling time and steady state errors are not explicitly specified. Reference tracking is not easily evaluated with step responses. A good overall metric to use is the Integral of Time and Absolute Error (ITAE). This will measure the performance of the controller over the entire range of a trajectory being tracked, with an emphasis on achieving a small steady state error, rather than a

quick response. In this application, it is more important that the helicopter keeps to the trajectory, rather than reaching a set point in minimal time.

Once it has been decided what state variables are paired to specific inputs, the control structure can be represented as shown in Figure 4.1. Castillo uses the Relative Gain Array (RGA) and diagonal dominance to show that the best pairing is: lateral cyclic with roll angle, longitudinal cyclic with pitch angle, collective pitch with heave velocity, and pedals with yaw rate [68].

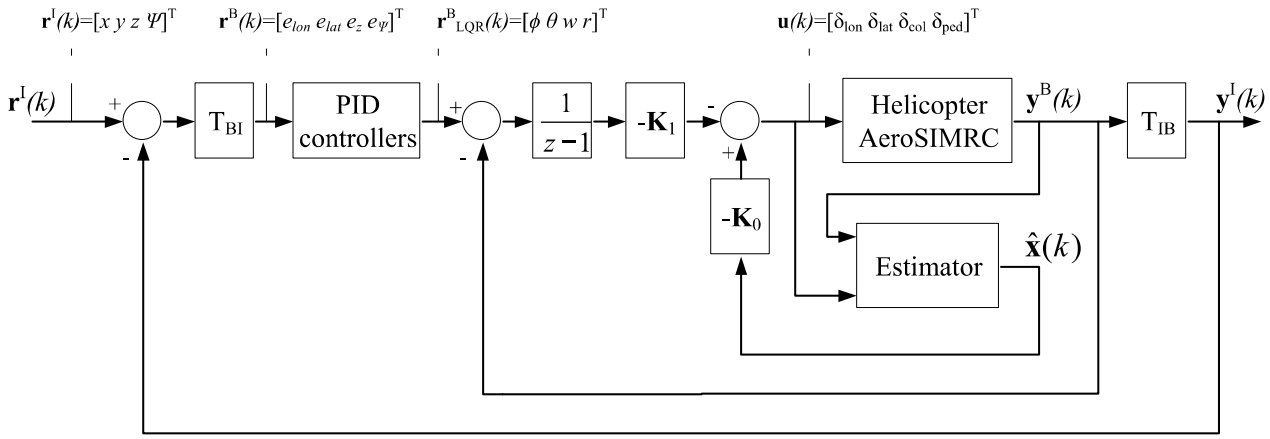



**Figure 4.1: Controller overview ($T_{IB}$: transform from Body reference frame to Inertial reference frame)**

The inner loop controller should have a fast, decoupled response, and reject disturbances such as wind. The inner loop should be designed so that the outer loop can control the translational dynamics as if there were no coupling. It should be able to operate optimally in its designed operating condition, but should also be able to keep the system stable in different operating conditions. The simplest controller capable of these tasks should be used. The methodology followed here is proposed by Valavanis et al. [68].

## 4.1.1 Inner loop

The continuous time model is converted to discrete time with a zero order hold and a sampling rate of 50 Hz. This is sufficiently fast for the helicopter dynamics in hover. The system is tested in simulation, so it is possible to make the sampling rate much faster than what would have been the case on the physical platform. Low cost autopilot systems typically operate at between 50 Hz-100 Hz [76].

### 4.1.1.1 Regulator

The system to be controlled is represented by the state space model

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k),\\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k).\end{aligned} \tag{4.1}$$

Assuming full-state feedback, the control law is given by

$$\mathbf{u}(k) = -\mathbf{K}\mathbf{x}(k). \tag{4.2}$$

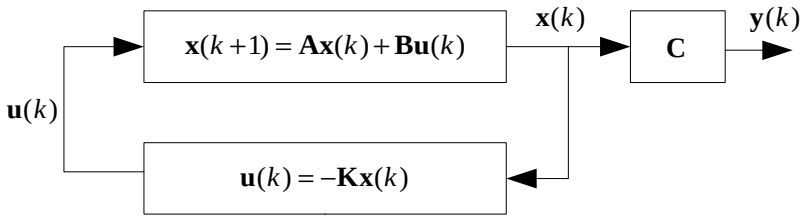



**Figure 4.2: Linear Quadratic Regulator without a reference signal. States are driven to zero.**

The closed loop system in Figure 4.2 is given by

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) - \mathbf{B}\mathbf{K}\mathbf{x}(k),\\ \mathbf{x}(k+1) &= (\mathbf{A} - \mathbf{B}\mathbf{K})\mathbf{x}(k).\end{aligned} \tag{4.3}$$

The feedback law is designed to minimise the quadratic cost function

$$\mathbf{J} = \sum_{k=0}^{\infty} (\mathbf{x}(k)^T \mathbf{Q}\mathbf{x}(k) + \mathbf{u}(k)^T \mathbf{R}\mathbf{u}(k)). \tag{4.4}$$

**Q** must be positive-definite and **R** must be positive semi-definite. The relative values of **Q** and **R** determine the amount of energy the control system can expend to keep the states zero. The gain **K** that minimises the cost function is

$$\mathbf{K} = (\mathbf{R} + \mathbf{B}^T\mathbf{P}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}\mathbf{A}, \tag{4.5}$$

with **P** the only unknown. **P** is the solution to the Discrete time Algebraic Ricatti Equation (DARE)

$$\mathbf{P} = \mathbf{Q} + \mathbf{A}^T(\mathbf{P} - \mathbf{P}\mathbf{B}(\mathbf{R} + \mathbf{B}^T\mathbf{P}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P})\mathbf{A}. \tag{4.6}$$

The solution to the DARE can be found iteratively. The performance of the controller is now dependant on the choice of the cost function. Bryson's rule,

$$\mathbf{Q}_{ii} = \frac{1}{(\text{maximum value of } x_i^2)}, \tag{4.7}$$

$$\mathbf{R}_{jj} = \frac{1}{(\text{maximum value of } u_j^2)}, \tag{4.8}$$

is used as a first iteration to find stable gains. Only the diagonal wieghts are used. Even though all the coefficients in **Q** and **R** can be specified, those off the diagonal become difficult to interpret and

the added complexity is unnecessary. The maximum values of a system identification dataset were used in (4.7) and (4.8) and is given in Table 4.1. Exact values are not important here.

**Table 4.1: Maximum state values for LQR design**

| State | Maximum | Unit | State | Maximum | Unit |
|---|---|---|---|---|---|
| $u$ | 5 | m/s | $a$ | 0 | rad |
| $v$ | 5 | m/s | $b$ | 0 | rad |
| $p$ | 1 | rad/s | $w$ | 2.5 | m/s |
| $q$ | 1 | rad/s | $r$ | 2.5 | rad/s |
| $\phi$ | 0.3 | rad | $r_{fb}$ | 0 | - |
| $\theta$ | 0.3 | rad | $\delta$ | 1 | - |

The cost function can then be changed iteratively to reach the design criteria. It is important to ensure that the actuator physical limits and rate limits are not reached, as the LQR algorithm does not explicitly account for this.

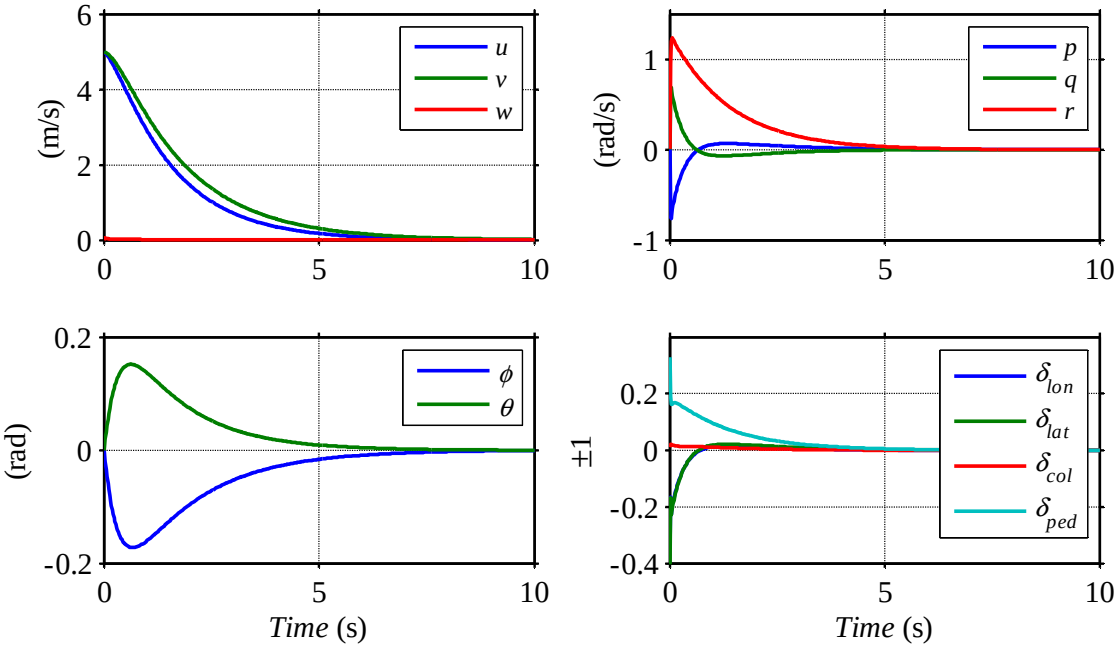



**Figure 4.3: LQR controller first iteration using Bryson's rule to calculate cost function weights**

As can be seen from Figure 4.3, even in the first iteration, the controller performs well. The state space model is given initial conditions of 5 m/s longitudinal and lateral velocity. The inputs are not saturated and the helicopter is stabilised within less than 10 s.

#### 4.1.1.2 Reference tracking with integral action

The inner loop controller should be able to track reference signals for roll, pitch, heave velocity, and yaw rate. Error states are added to the state space model to enable zero steady state error tracking. This can cause a problem if sensors have constant biases. The angular states will then be pushed to zero, although the helicopter will be at an angle, which will cause translational movement. This problem can be solved at the state estimator by estimating each sensor's bias [53].

Integrators ensure that there is zero steady state error [77]. The state vector is augmented with the integral of the error for which the derivative is the tracking error

$$\begin{aligned} \mathbf{x}_I(k+1) &= \mathbf{w}(k) = \mathbf{y}(k) - \mathbf{r}(k), \\ \mathbf{x}_I(k+1) &= \sum_{k=0}^{N} \mathbf{w}(k). \end{aligned} \tag{4.9}$$

The augmented system is

$$\begin{bmatrix} \mathbf{x}(k+1) \\ \mathbf{x}_I(k+1) \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}(k) \\ \mathbf{x}_I(k) \end{bmatrix} + \begin{bmatrix} \mathbf{B} \\ \mathbf{0} \end{bmatrix} \mathbf{u} - \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix} \mathbf{r}. \tag{4.10}$$

The control law is of the form

$$\begin{aligned} \mathbf{u}(k) &= -\begin{bmatrix} \mathbf{K}_a \end{bmatrix} \begin{bmatrix} \mathbf{x}(k) \\ \mathbf{x}_I(k) \end{bmatrix}, \\ \mathbf{K}_a &= \begin{bmatrix} \mathbf{K}_0 & \mathbf{K}_I \end{bmatrix}. \end{aligned} \tag{4.11}$$

For easier notation, the augmented system is now described as

$$\begin{aligned} \mathbf{x}_a(k+1) &= \mathbf{A}_a \mathbf{x}_a(k) + \mathbf{B}_a \mathbf{u}_a(k), \\ \mathbf{y}_a(k) &= \mathbf{C} \mathbf{x}_a(k). \end{aligned} \tag{4.12}$$

The control law is

$$\mathbf{u}_a(k) = -\mathbf{K}_a \mathbf{x}_a(k). \tag{4.13}$$

The gain $\mathbf{K}_a$, can be calculated in the same way as the previous section, where the only difference is the four extra states. A summary of the final weightings are given on the next page in Table 4.2. It can be seen that a heavy penalty is placed on controller input, to avoid overshoot.

**Table 4.2: LQR cost function final weights**

| State | Weight | State | Weight | State | Weight |
|---|---|---|---|---|---|
| $u$ | 0.04 | $w$ | 0.16 | $\delta_{lon}$ | 600 |
| $v$ | 0.04 | $r$ | 0.16 | $\delta_{lat}$ | 600 |
| $p$ | 1 | $r_{fb}$ | 0 | $\delta_{col}$ | 10000 |
| $q$ | 1 | $w_\phi$ | 1 | $\delta_{ped}$ | 10000 |
| $\phi$ | 888 | $w_\theta$ | 1 | | |
| $\theta$ | 888 | $w_w$ | 1 | | |
| $a$ | 0 | $w_r$ | 1 | | |
| $b$ | 0 | | | | |

Up until now, full state feedback has been assumed. To account for states that are not measured (e.g. rotor flapping), an estimator is needed.

#### 4.1.1.3 Estimator

The estimator is described by

$$\hat{\mathbf{x}}(k+1) = \mathbf{A}\hat{\mathbf{x}}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{L}(\mathbf{y}(k) - \hat{\mathbf{y}}(k)), \tag{4.14}$$

$$\hat{\mathbf{y}}(k) = \mathbf{C}\hat{\mathbf{x}}(k), \tag{4.15}$$

where ($\mathbf{y}(k)$ - $\hat{\mathbf{y}}(k)$) is a feedback term to correct for differences in the plant and the model. $\hat{\mathbf{x}}$ is the estimated state vector. Using this new state vector, the control signal becomes

$$\mathbf{u}(k) = -\mathbf{K}\hat{\mathbf{x}}(k). \tag{4.16}$$

The difference between the estimator and the actual states are given by

$$\begin{aligned} \mathbf{e}(k+1) &= \mathbf{x}(k+1) - \hat{\mathbf{x}}(k+1) \\ &= (\mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k)) - (\mathbf{A}\hat{\mathbf{x}}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{L}(\mathbf{C}\mathbf{x}(k) - \mathbf{C}\hat{\mathbf{x}}(k))). \end{aligned} \tag{4.17}$$

The estimator is then

$$\mathbf{e}(k+1) = (\mathbf{A} - \mathbf{L}\mathbf{C})\mathbf{e}(k). \tag{4.18}$$

The estimator is stable if **A**-**LC** has negative eigenvalues. This will ensure that $\hat{\mathbf{x}}$ converges to **x** as $t$ converges to infinity. With the state feedback controller, the dynamics were determined by **A**-**BK**. Here the gain **L** can be calculated in much the same way. The poles of the estimator will determine how fast the estimator responds. By choosing faster poles, the estimator will follow the true plant closer, but it will be more susceptible to noise. As a rule of thumb, the estimator poles should be 10 times faster than the plant's slowest pole.

According to the separation principle, if the estimator and the regulator are asymptotically stable, then the closed loop of the combined system is also stable [31]. The closed loop system dynamics are given by

$$\begin{aligned} \begin{bmatrix} \mathbf{x}_a(k+1) \\ \mathbf{e}(k+1) \end{bmatrix} &= \begin{bmatrix} \mathbf{A}_a - \mathbf{B}_a\mathbf{K}_a & \mathbf{B}_a\mathbf{K}_a \\ \mathbf{0} & \mathbf{A}-\mathbf{LC} \end{bmatrix} \begin{bmatrix} \mathbf{x}_a(k) \\ \mathbf{e}(k) \end{bmatrix} + \begin{bmatrix} \mathbf{B}_a \\ \mathbf{0} \end{bmatrix} \mathbf{r}(k), \\ \mathbf{y}(k+1) &= \begin{bmatrix} \mathbf{C}_a & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_a(k) \\ \mathbf{e}(k) \end{bmatrix} + [\mathbf{0}]\mathbf{r}(k), \end{aligned} \tag{4.19}$$

with the reference signal

$$\mathbf{r}(k) = \begin{bmatrix} \phi_{ref} & \theta_{ref} & w_{ref} & r_{ref} \end{bmatrix}^T. \tag{4.20}$$

The simulator does not model sensor noise typically seen in physical helicopters, which eliminates the need for a Kalman filter. The compensator with reference tracking is shown in Figure 4.4. Figure 4.1 shows how this controller fits into the overall control system.

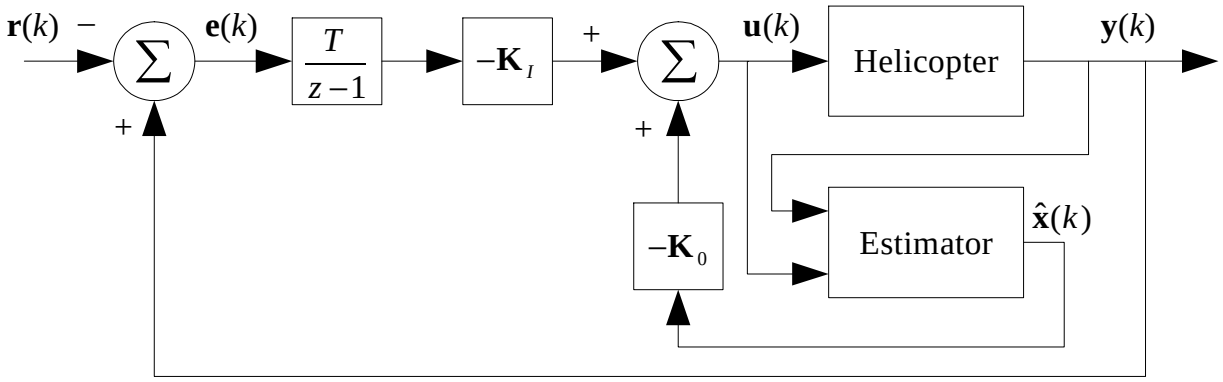



**Figure 4.4: Regulator with estimator and integral reference tracking [31]**

#### 4.1.1.4 Verification

The LQR controller is tested on the state space model to verify the correct operation. Figure 4.5 on the next page, shows the LQR controller tracking a reference signal with no steady state error. The controller will be further evaluated in Chapter 5. The controller is using the estimated states for full state feedback. Adding integrals introduces the possibility of integral windup. An anti-windup system is therefore added to avoid this problem [31]. The figure shows that the controller uses very small input values and the inputs do not oscillate. There is no overshoot in the roll step response, and the controller is able to eliminate any steady state errors.

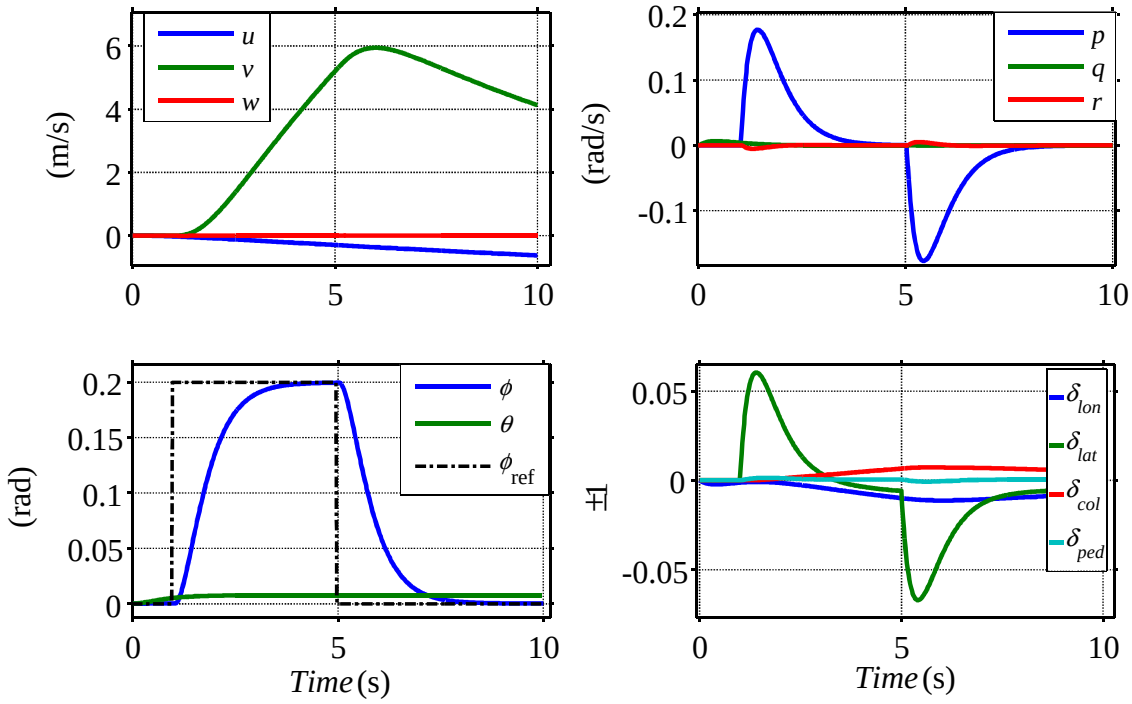



**Figure 4.5: Roll step response of LQR controller with integral states**

### 4.1.2 Outer loop

To fly a trajectory specified in the inertial reference frame, the LQR controller needs reference states for the trajectory. A simple solution is to use a PID controller to convert the *x*, *y*, *z*, and yaw trajectory (given in the inertial reference frame) to the appropriate roll, pitch, heave velocity, and yaw rates. The control architecture used is shown in Figure 4.1.

The inner loop controller functions in the BF, which makes it necessary for the inertial trajectory errors to be converted from the EF to the BF. Once the *x*, *y*, and *z* errors have been converted to body-fixed frame longitudinal and lateral errors, the PID controllers can generate the appropriate roll and pitch references for the inner loop. Without this correction, the helicopter would not be able to change heading and still track a trajectory. If the heading were changed 180° from north, an error in the *x* direction would cause a negative pitch, which would move the helicopter forward when in fact it should go backward. This would increase the error in the inertial frame and the system would become unstable.

The controller is designed for hover and slow flight, which implies that the roll and pitch angles will be small. However, the yaw angle can have any arbitrary value. AeroSIMRC specifies this value between $-\pi$ and $\pi$. This causes a discontinuity that needs to be addressed for the controller to function properly. In the state space simulation, the integrators that determine the angles from the

angular rates do not have this discontinuity. The discontinuity is added to the Simulink® simulation to improve the resemblance to the simulator environment.

#### 4.1.2.1 PID controllers optimisation

The PID controllers are iteratively tuned until the system is stable. Thereafter the gains are optimised using a simplex search method. The objective function that needs to be minimised is the Integral Time Absolute Error (ITAE),

$$ITAE = \sum_{k=0}^{N} k\,|e(k)|, \tag{4.21}$$

where $e$ is the difference between the tracking signal and the output. A PID loop is created for each of the LQR inputs. The controllers are tuned separately until the terminating condition of the objective function is reached. Figure 4.6 illustrates the optimisation workflow.

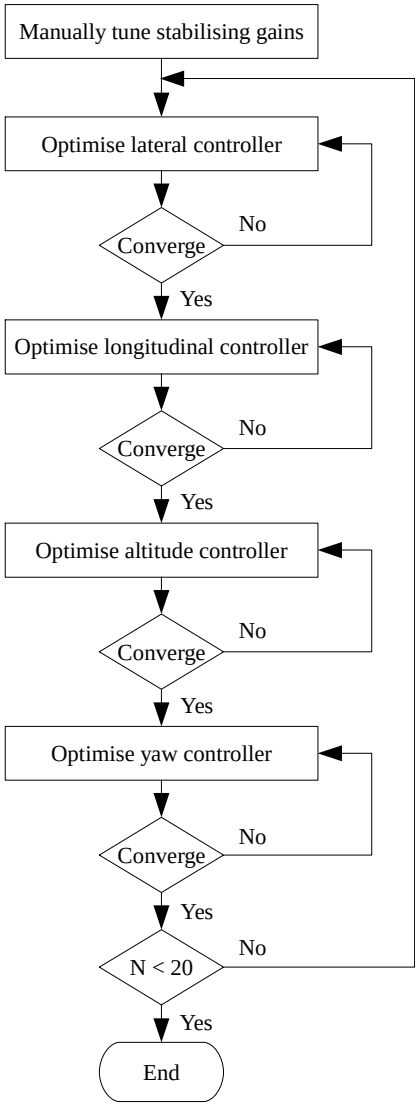



**Figure 4.6: Iterative trajectory controller PID optimisation flow diagram**

The controllers are optimised individually, but they influence each other. This makes it necessary to repeat the process several times, as the gains converge.

The optimal gains are calculated for a specific trajectory, which means that these gains are not necessarily optimal for other trajectories. A 3D figure 8 with the helicopter turning at a constant yaw rate was selected as the trajectory to use. The reference trajectory is given by

$$\left.\begin{aligned} x_{ref}(t) &= 30\sin\left(2\pi\left(\frac{1}{60}\right)\mathrm{t}\right) \\ y_{ref}(t) &= 30\sin\left(2\pi\left(\frac{1}{120}\right)\mathrm{t}+\frac{\pi}{2}\right)-30 \\ z_{ref}(t) &= 10\sin\left(2\pi\left(\frac{1}{60}\right)\mathrm{t}+\frac{\pi}{2}\right)-10 \\ \psi_{ref}(t) &= \frac{2\pi}{60}t \end{aligned}\right\} \text{for } 0 \le t \le 120. \qquad (4.22)$$

It was found that a smooth trajectory works better than one with sharp turns for the optimisation. When optimising for a very aggressive trajectory with fast changes in direction, the gains tend to be very high. In a complete UAS, the guidance controller would decide if sharp turns are necessary. The optimised gains are shown in Table 4.3. The lateral and longitudinal integral gains are very small, possibly justifying the use of only PD controllers.

**Table 4.3: Optimised PID gains for figure 8 manoeuvre from (4.22)**

| **Controller** | *Kp* | *Ki* | *Kd* | *N* |
|---|---|---|---|---|
| Longitudinal PID | 0.1063 | 0.0006 | 0.1243 | 19.8855 |
| Lateral PID | 0.0987 | 0.0006 | 0.1956 | 81.5702 |
| Heave velocity PID | 10.09 | 0.2 | 8.0198 | 50.9698 |
| Yaw rate PD | 3.6 | - | 1 | - |

The controllers were set up in parallel form and the forward Euler method was used on all integrators and differentiators. The trajectory generating outer loop can be replaced with a number of other controllers for better performance. If the inner loop is linearised, an LQR controller could be used for the outer loop as well as the inner loop. Once a higher-level guidance controller is available, the outer loop will most likely change to a velocity controller rather than a position controller. Other controllers such as Model Predictive Control (MPC) could possibly improve performance. However, MPC will sacrifice simplicity, which is a major objective in this study.

#### 4.1.2.2 Verification

The PID controllers are tested on the state space model to verify the correct operation. The results for the trajectory given in (4.22) are shown in Figure 4.7 and Figure 4.8 (on the next page). The helicopter follows the reference trajectory closely. The attitude angles change as the helicopter's heading changes, proving that the controllers is able to keep to the reference with an arbitrary heading. The helicopter starts from a stationary position. As it catches up with the reference trajectory, the attitude angles become smoother. In the next chapter, this same trajectory will be flown with the helicopter always facing the direction it is flying, which is more economical and practical. The PID controllers will be validated on the target platform in Chapter 5.

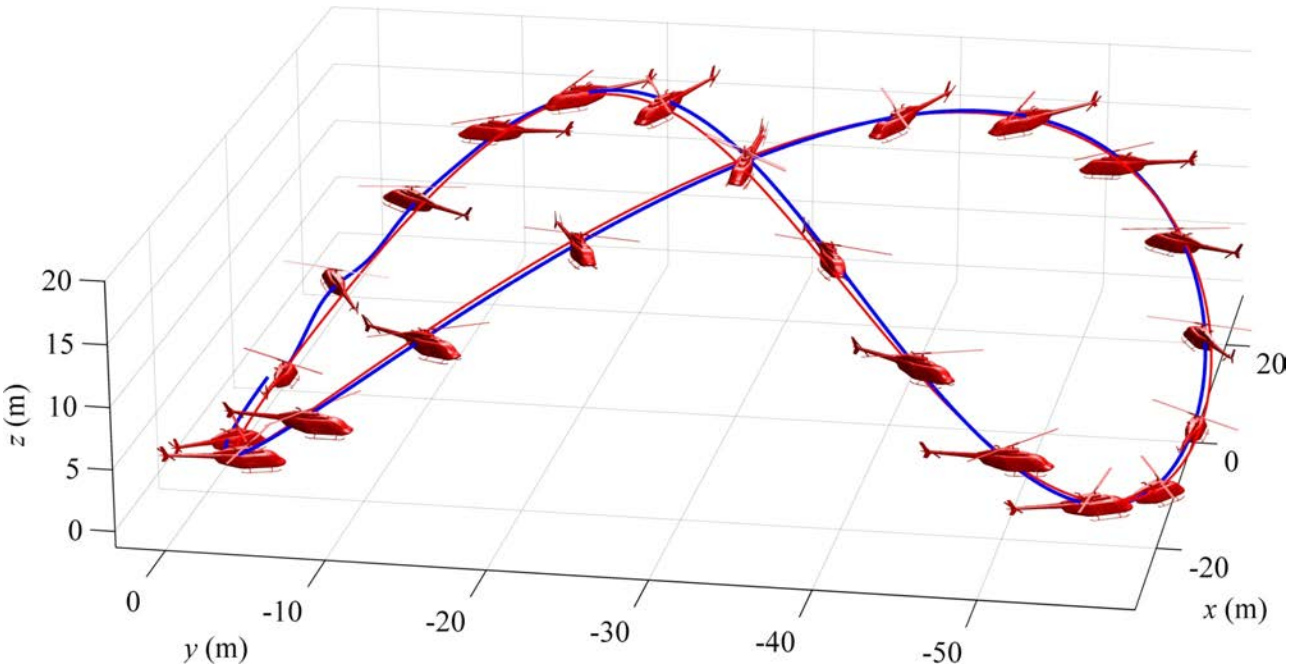



**Figure 4.7: 3D view of the figure 8 trajectory with a 720° yaw rotation. (Red line: Reference. Blue line: State space model)**

## 4.2 Conclusion

The methodology presented provides a clear way to develop a flight controller capable of trajectory tracking from a state space model. The attitude dynamics are stabilised and decoupled with the LQR controller. The outer loop SISO controllers are able to control the helicopter and follow a given inertial trajectory. The interaction between the inner and outer loops makes it necessary to iteratively develop the controller, as changes to the inner loop will affect the outer loop. The PID controllers' performance is limited by the performance of the inner loop. The state space model experiments are the first part of the verification, which will be continued in the next chapter. In the following chapter, the controllers will be validated on the target platform.

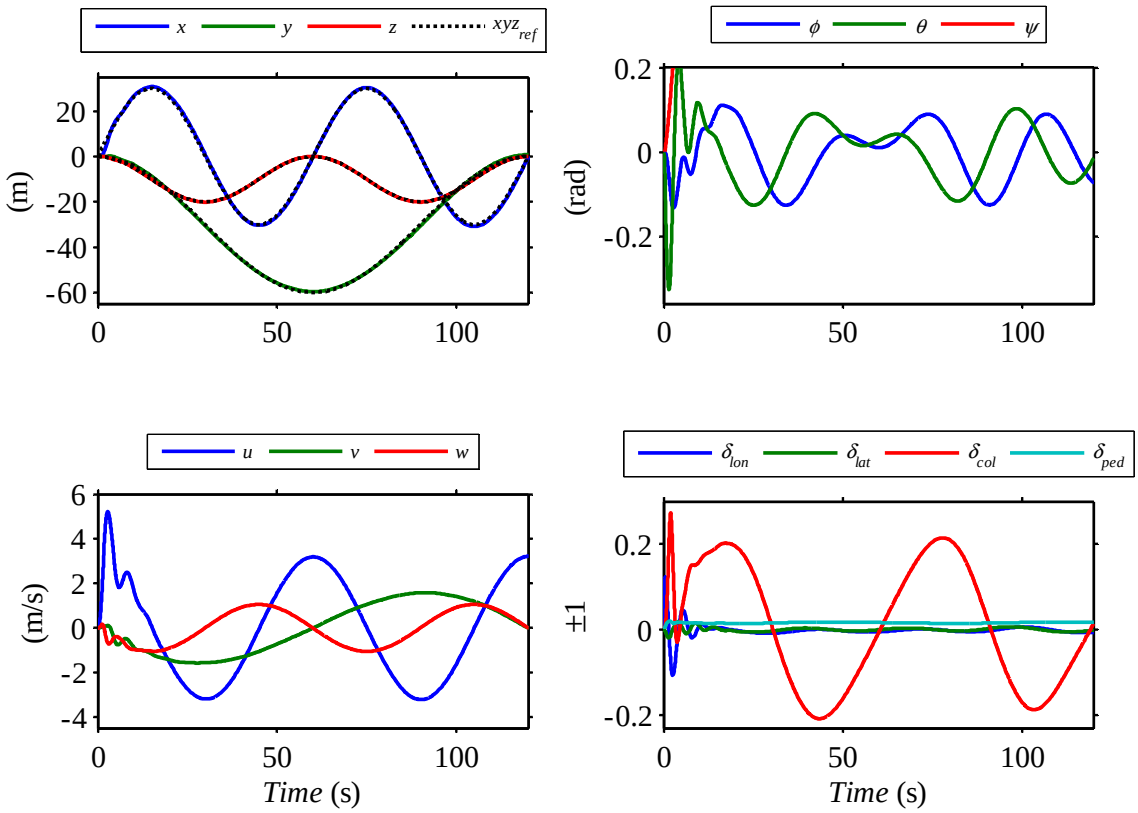



**Figure 4.8: Inertial position, attitude and velocity plots from figure 8 reference trajectory**

# Chapter 5: Controller evaluation

*This chapter serves three purposes: Verification, validation and evaluation. It is first verified that the controller is able to control the model adequately. The model response is compared to the flight-data from AeroSIMRC to validate the model. The performance of the inner loop controller is then evaluated using the Aeronautical Design Standards (ADS-33E-PRF)* [78]. *Using this standard validates the performance of the attitude control system. The outer loop is validated through a series of trajectory tracking experiments. Throughout this chapter, "system model" will refer to the controller and helicopter model as a whole, in contrast to "flight-data", which refers to the data gathered from AeroSIMRC in experimental flights.*

## 5.1 Role of the inner and outer loops

A pilot controls the helicopter's position by changing the helicopter attitude, and consequently changing the direction of the helicopter's thrust. This same principle applies to the control system. In Chapter 4, the controller development was separated into two subsystems based on the different time constants of the helicopter's rotational and translational motion. The controller architecture is shown in Figure 5.1.

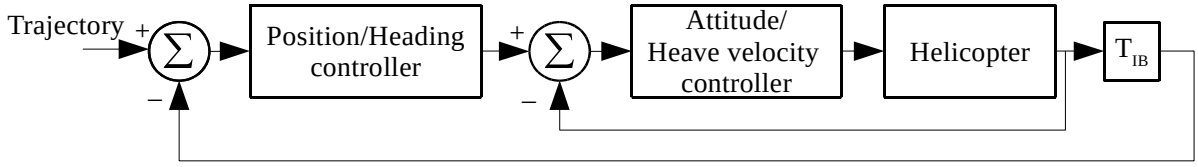



**Figure 5.1: Cascaded controller architecture**

The faster inner loop controls the helicopter's attitude as well as heave velocity. The heave velocity does not form part of the fast attitude dynamics but is convenient to have in the inner loop because

of the strong relation between collective pitch and heave velocity. The inner loop's main task is to make the airframe's capabilities available to the outer loop. It dictates the outer loop performance. Just as the inner loop cannot surpass the airframe's inherent limitations, the inner loop will have its own limitations that the outer loop cannot surpass. This emphasises the importance of the lower level controllers. Each level's performance is dependent on the previous level.

The outer loop controls the helicopter trajectory. An inertial trajectory is transformed into an attitude reference for the inner loop. The outer loop is much slower and easier to control due to the attitude controller that has made the airframe a well-behaved system. In this study, the trajectory provided to the outer loop is manually generated beforehand. In a fully autonomous UAS, the trajectory would be generated by a high-level guidance controller.

## 5.2 Flying qualities specification

Aircraft flying qualities are used to assess an aircraft's ability to safely complete a given mission. Pilots rate an aircraft based on its handling qualities, which are a measure of how difficult it is to control the aircraft. Flying qualities generally refer to both handling quality and more quantitative performance metrics. Flying qualities have been applied to Remotely Piloted Vehicles (RPVs) as well as autonomous UAVs [27]. Even though there is no pilot in an autonomous UAV, the manned aircraft specifications still provides a measure of how "good" an aircraft is flying. Research is being done on setting up a flying qualities standard for UAVs but it will focus mostly on RPVs as they are by far the most commonly used [79]. Until such a standard is available the manned specifications can be used. In general, these specifications will be more conservative due the limitations of the pilot. The flight controller in this study is a first iteration solution, there is no backup control system. With this in mind, handling qualities become even more important as a safety pilot will surely be needed for backup once the controller is used on hardware.

The ADS-33 is the industry standard for rotorcraft evaluation [78]. The ADS-33 specifications will not be used "as is", but give a good indication of which qualities in a rotorcraft flight control system are important. Most modern helicopters have some form of Stability Augmentation System (SAS) that helps the pilot fly the helicopter. The SAS commonly consists of Attitude Command Attitude Hold (ACAH) or Rate Command Direction Hold (RCDH) modes. This makes it appropriate to use these manned specifications as a guideline for evaluating the inner loop controller developed in this study. A key factor to be noted is that Pilot Induced Oscillation (PIO) is a major problem with manned aircraft, and many of the stability margins are set to avoid this. This is not a problem in

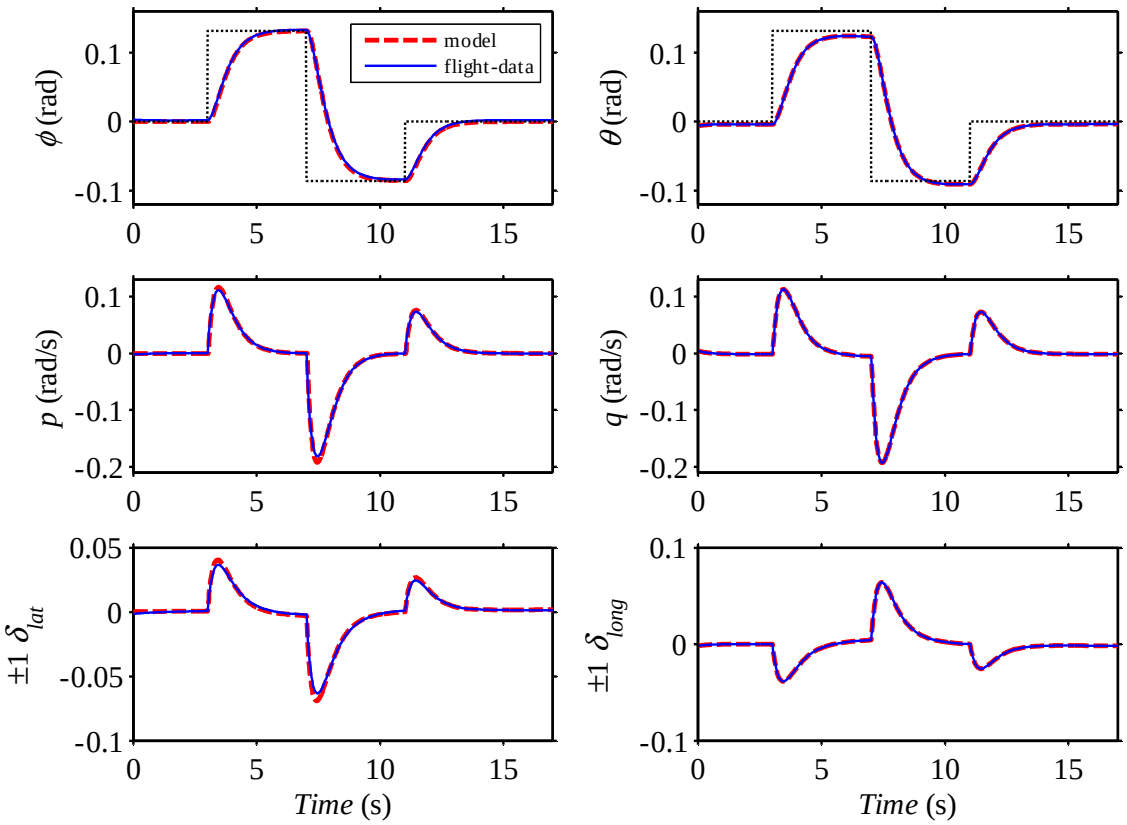
unmanned aircraft, and will be appropriately treated. Several researchers have used the ADS-33 to evaluate and design controllers for UAVs [27].

ADS-33 performance is divided into 3 levels. Level 1 indicates excellent performance with no work load placed on the pilot. Level 2 indicates moderate performance and requires pilot compensation. Level 3 indicates major deficiencies. The rotorcraft is barely controllable and the pilot is under a very high workload.

## 5.3 Attitude controller performance

The agreement between the system model and flight-data is first assessed in the time domain and then in the frequency domain. The frequency domain information is used to perform a stability analysis. Several experiments are done to determine the controller performance. Throughout most of the experiments, the model response is compared to flight-data. The yaw rate and heave velocity are not evaluated here. The performance of the yaw and heave controllers are rather evaluated with the outer loop as they form part of the inertial position trajectory and not the attitude dynamics.

### 5.3.1 Flight test validation




**a) Roll doublet test.** **b) Pitch doublet test.**

**Figure 5.2: Time domain validation of model in two flight tests.**

The system model response is compared to flight data for validation. The helicopter is trimmed at hover. A doublet input is given to avoid large unwanted translational velocities caused by a pure step input. Figure 5.2 shows that the model and the flight-data closely match. The system model validation builds further confidence in the state space model developed in Chapter 3. This experiment also serves as validation for the test environment. Proving that the controller developed in Simulink® can be used to directly control the helicopter in AeroSIMRC.

The frequency response of the model is calculated analytically from the state space model of the inner loop. Since it is a MIMO system it is described by a matrix of transfer functions

$$\mathbf{T}(s)_{attitude} = \begin{bmatrix} \frac{u}{\phi_{ref}} & \frac{u}{\theta_{ref}} & \frac{u}{w_{ref}} & \frac{u}{r_{ref}} \\ \frac{v}{\phi_{ref}} & \frac{v}{\theta_{ref}} & \frac{v}{w_{ref}} & \frac{v}{r_{ref}} \\ \frac{p}{\phi_{ref}} & \frac{p}{\theta_{ref}} & \frac{p}{w_{ref}} & \frac{p}{r_{ref}} \\ \frac{q}{\phi_{ref}} & \frac{q}{\theta_{ref}} & \frac{q}{w_{ref}} & \frac{q}{r_{ref}} \\ \frac{\phi}{\phi_{ref}} & \frac{\phi}{\theta_{ref}} & \frac{\phi}{w_{ref}} & \frac{\phi}{r_{ref}} \\ \frac{\theta}{\phi_{ref}} & \frac{\theta}{\theta_{ref}} & \frac{\theta}{w_{ref}} & \frac{\theta}{r_{ref}} \\ \frac{w}{\phi_{ref}} & \frac{w}{\theta_{ref}} & \frac{w}{w_{ref}} & \frac{w}{r_{ref}} \\ \frac{r}{\phi_{ref}} & \frac{r}{\theta_{ref}} & \frac{r}{w_{ref}} & \frac{r}{r_{ref}} \end{bmatrix}, \tag{5.1}$$

relating each input-output pair. The larger font in (5.1) indicate the on-axis responses. The on-axis responses are the transfer functions with the same reference variable and output variable. The off-axis response occurs when a reference input causes a disturbance on a different variable.

In order to identify the frequency response from flight data, a chirp signal is given as reference, with the other channels used only for stabilisation. This is much easier than in Chapter 3, since the controller can perform the sweeps and keep the helicopter stable, where the pilot previously had to do it. The closed loop transfer function

$$T(s)_{\phi/\phi_{ref}} = \frac{\mathrm{FFT}(\phi)}{\mathrm{FFT}(\phi_{ref})}, \tag{5.2}$$

is calculated from the input-output data pair using the Fast Fourier Transform (FFT).

The same problems that were present when the airframe dynamics had to be identified are present here. It is difficult to get low frequency data from the aircraft since low frequency input causes big translational movements. Figure 5.3 shows that the frequency response magnitude closely matches the model between 0.1 rad/s and 10 rad/s, this is the important part of the spectrum. The flight data response has noise at the higher frequency partly due to the chirp signal not sufficiently exciting the system at those frequencies, and partly due to noise in the simulator.

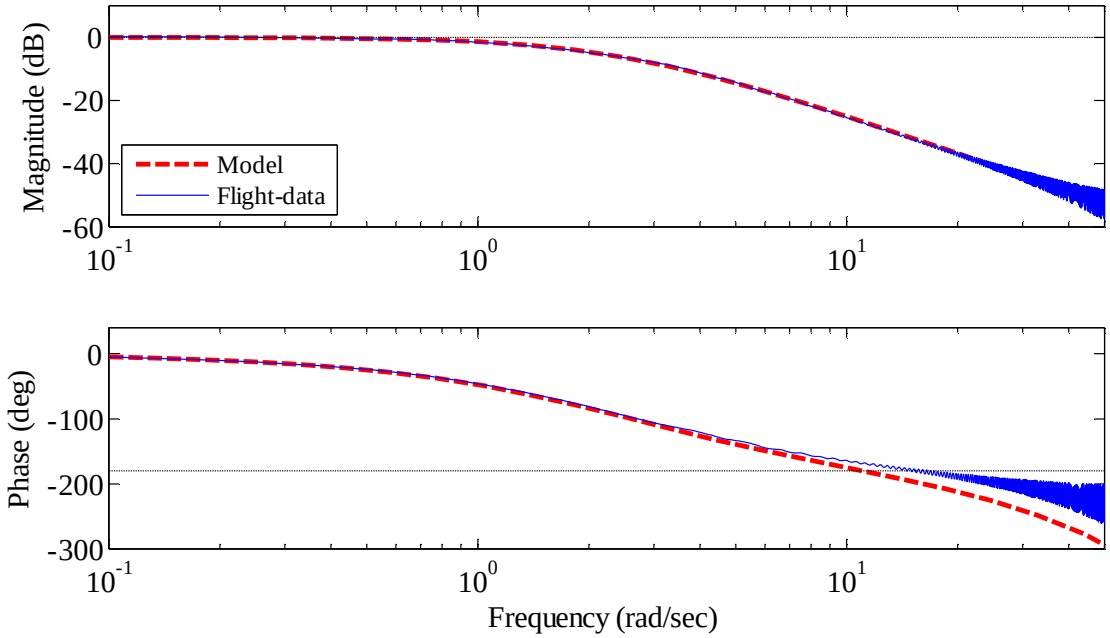
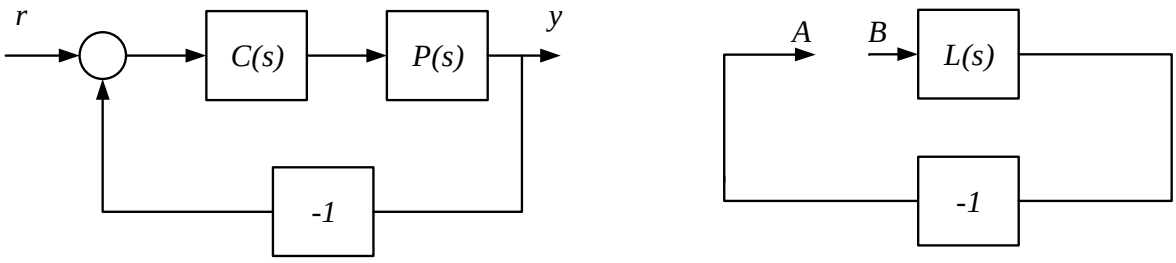



**Figure 5.3: Roll closed loop transfer function from model and flight data.**

### 5.3.2 Small amplitude attitude changes

The closed loop transfer function was used to validate that the model accurately represents the real system. In this section, the stability of the system is assessed using the Nyquist stability criterion. It provides information on the measure of stability, unlike Lyapunov based methods that only indicate whether the system is stable or not [80]. The guidelines given in ADS-33 are used. The open loop transfer function is found by opening the closed loop system at AB as shown in Figure 5.4, the input is measured at A and the output at B. The open loop transfer function is given by

$$L(s) = P(s)C(s). \tag{5.3}$$




**a) A simple closed loop feedback system.**

**b) The open loop transfer function from broken loop.**

**Figure 5.4: Open and closed loop configurations. From [80].**

The loop transfer function is used for the stability analysis. It is difficult to operate an unstable system in open loop, fortunately the loop gain can be calculated from the closed loop transfer function. If the closed loop transfer function is given by

$$H_{\phi_{ref},\phi}(s) = \frac{L_\phi(s)}{1+L_\phi(s)}, \tag{5.4}$$

the open loop transfer function is

$$L_\phi(s) = \frac{H_{\phi_{ref},\phi}(s)}{1-H_{\phi_{ref},\phi}(s)}. \tag{5.5}$$

In classic control theory the phase crossover frequency, $\omega_{180}$, is the smallest frequency at which the loop transfer function phase angle crosses -180°. The gain crossover frequency, $\omega_{gc}$, is the smallest frequency where magnitude of the open loop transfer function crosses 0 dB.

The gain margin is defined as

$$GM = \frac{1}{\left|L(j\omega_{180})\right|}, \tag{5.6}$$

and the phase margin is defined as

$$PM = \pi + \arg L(j\omega_{gc}). \tag{5.7}$$

The gain margin is the amount by which the gain can be increased before the system will become marginally stable. The phase margin is the amount of phase lag that can be allowed before the feedback loop reaches marginal stability. The bandwidth is defined as the frequency at which the magnitude of the transfer function reaches -3 dB.

This same notion of determining a system's stability is used in the ADS-33 to evaluate the handling qualities of rotorcraft. It is now possible to assess the loop gain using the criteria for stability given in the ADS-33 section on small attitude changes and stability.

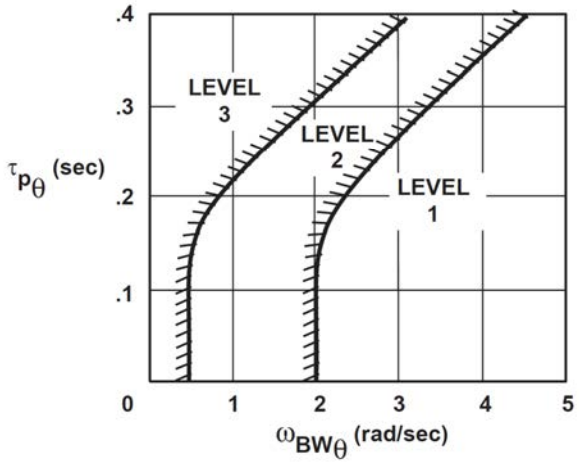



**Figure 5.5: ADS-33 requirements for small-amplitude roll changes. From [78]**

The flying qualities bandwidth is divided into two definitions, one for phase, and one for gain bandwidth. The phase bandwidth, $\omega_{phase}$, is the frequency at which the phase angle is 45° higher than -180°. The gain bandwidth, $\omega_{gain}$, is the frequency at which the magnitude is 6 dB higher than the magnitude at the phase crossover frequency of -180°. The flying qualities bandwidth is then the smaller of the two, and the system is either phase or gain limited. The phase delay

$$\tau_p = \frac{\Delta\Phi_{2\omega_{180}}}{57.3(2\omega_{180})}, \tag{5.8}$$

is used with the bandwidth to evaluate the small amplitude performance as shown in Figure 5.5.

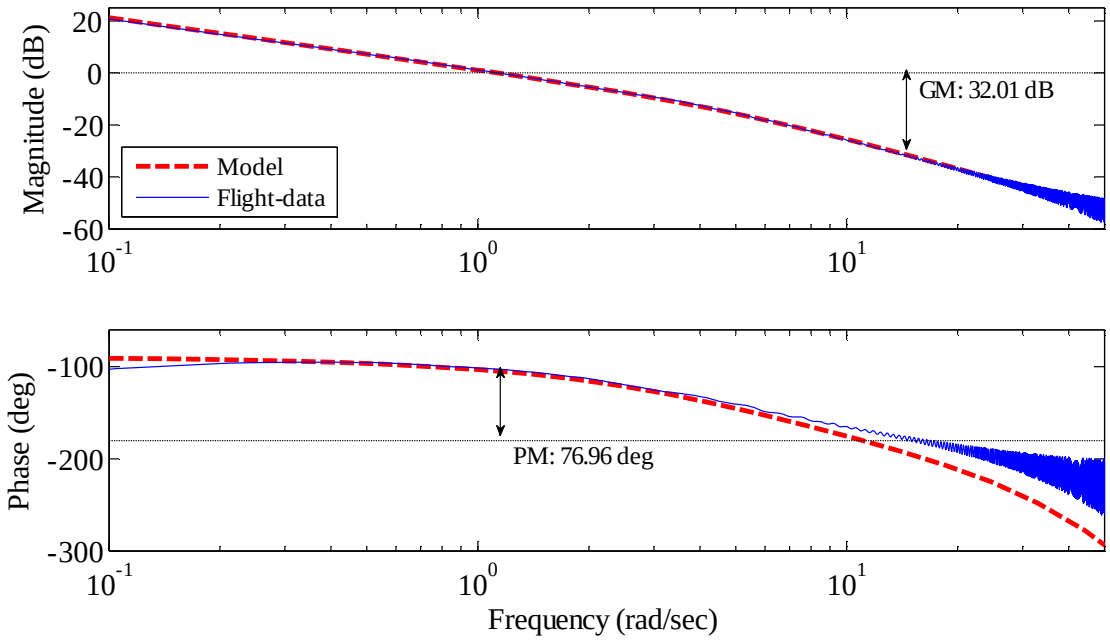



**Figure 5.6: Roll angle open loop transfer function.**

The difference between the phase angle at two times the crossover frequency and at the crossover frequency is given by $\Delta\Phi_{2\omega_{180}}$. A bode plot with the definitions from the ADS-33 documentation are included in Appendix C. Roll and pitch bandwidth and phase delay are calculated similarly, but each have their own requirements for level 1 handling qualities.

The loop gain frequency response plot is given in Figure 5.6 (previous page). The phase bandwidth is smaller than the gain bandwidth, indicating a phase limited system. This is preferred in manned systems as it reduces the risk of PIO. The bandwidth is important, as it will determine the system's ability to reject disturbances. Disturbance rejection will be evaluated later on. Without adequate bandwidth, the helicopter will have trouble performing small precision tasks that require fast tracking movements.

Table 5.1 shows a summary of the results obtained from the roll frequency response experiments. The roll and pitch responses are well within the level 1 requirement for handling quality shown in Figure 5.5 (previous page). This good performance can be attributed to the small size and agility of the helicopter, and the controller's ability to make use of these advantages. The MIL-F-9490 standard uses the classical definitions of phase and gain margin to evaluate the handling qualities. The controller in this study meets the minimum requirements set for the PM and GM.

**Table 5.1: Small attitude change results**

| | **Roll** | | | | |
|---|---|---|---|---|---|
| | *GM* (dB) | *PM* (deg) | $\omega_{phase}$ (rad/sec) | $\omega_{gain}$ (rad/sec) | $\tau_{p\theta}$ (sec) |
| Model | 27.6 | 75.4 | 3.81 | 7.65 | 0.03 |
| Flight data | 32 | 77 | 4.29 | 10.05 | 0.015 |
| MIL-F-9490 | 6 | 45 | - | - | - |
| | **Pitch** | | | | |
| | *GM* (dB) | *PM* (deg) | $\omega_{phase}$ (rad/sec) | $\omega_{gain}$ (rad/sec) | $\tau_{p\phi}$ (sec) |
| Model | 25.1 | 75.6 | 3.95 | 8.13 | 0.027 |
| Flight data | 27.8 | 76 | 4.3 | 8.04 | 0.025 |
| MIL-F-9490 | 6 | 45 | - | - | - |

### 5.3.3 Moderate amplitude attitude changes

The criterion used in ADS-33 for moderate amplitude attitude changes is Attitude Quickness (AQ), given by

$$\mathrm{AQ}_{roll} = \frac{p_{pk}}{\Delta\phi_{\min}}. \tag{5.9}$$

The minimum attitude change, $\Delta\phi_{min}$, is defined as the minimum value of the step input's first oscillation. A critically damped controller, such as the one presented here, will have the same maximum step value, $\Delta\phi_{pk}$, as the minimum attitude change value. The peak angular rate attained is $p_{pk}$.

AQ is experimentally determined by giving a moderate amplitude (±20°) step input to the roll and pitch axis. A higher AQ is better, indicating a more agile aircraft. The ADS-33 requirement for level 1 handling qualities is shown in Figure 5.7.

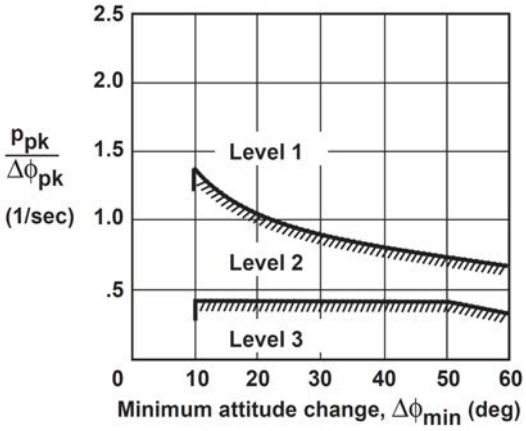



**Figure 5.7: Requirements for moderate amplitude roll attitude changes. From [78]**

Figure 5.8 shows the result of a 20° step input on the roll attitude. The same experiment was conducted for the pitch axes, but only the numerical results are presented here for brevity. The results for moderate amplitude attitude changes are compared to the level 1 handling requirements and summarised in

Table 5.2.

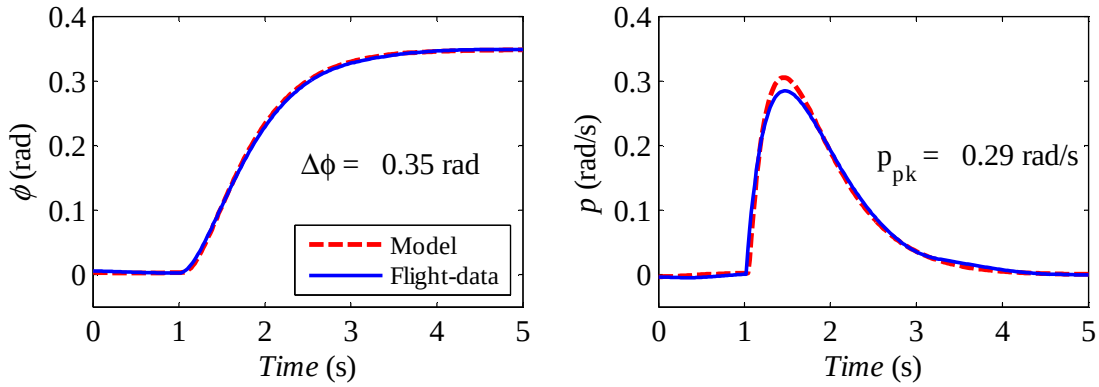



**Figure 5.8: Roll step input to determine the attitude quickness.**

The controller falls within the level 2 category, very close to level 1. This can be expected as the controller was designed to track much smaller input signals, and was not optimised for step inputs. The model and flight data show good comparison.

**Table 5.2: Attitude quickness results compared to ADS-33 level 1 requirement.**

| | **Roll:** $\frac{p_{pk}}{\Delta\phi}$(1/s) | **Pitch:** $\frac{q_{pk}}{\Delta\theta}$(1/s) |
|---|---|---|
| Model | 0.94 | 1.15 |
| Flight data | 0.83 | 1.15 |
| Level 1: 20° step | 1.2 | 1.75 |

### 5.3.4 Large amplitude attitude changes

The large amplitude response is used to determine the maximum attitude angles and rates. Step inputs are given to the pitch and roll axis. The size of the reference signal is limited to the maximum step that does not saturate the actuators. The results are given in Table 5.3.

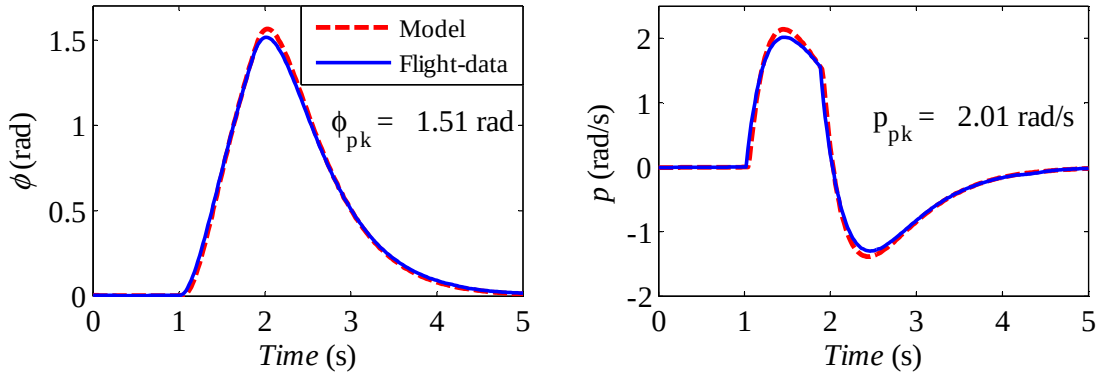



**Figure 5.9: Maximum roll angles and rates.**

The controller was able to achieve a roll angle of 85° and a roll rate of 115 deg/s. In the flight test, the helicopter is turned on its side and has nearly no thrust to keep it flying. It quickly loses altitude until it is returned to its upright position. The controller is able to achieve the level 1 requirements for aggressive agility and surpass the angular rates quite far, achieving more than double the rates in both pitch and roll.

**Table 5.3: Large attitude results compared to ADS-33 level 1 aggressive agility requirements.**

| | **Angular rates (rad/s)** | | **Attitude Angles (rad)** | |
|---|---|---|---|---|
| | $p_{pk}$ | $q_{pk}$ | $\phi_{pk}$ | $\theta_{pk}$ |
| Model | 2.14 | 2.02 | 1.57 | 1.27 |
| Flight data | 2.01 | 1.91 | 1.51 | 1.2 |

| | | | | |
|---|---|---|---|---|
| ADS-33 Level 1 | 0.87 | 0.52 | 1.04 | 0.52 |

The small RC helicopter is much more agile than a manned helicopter. This test would be difficult to implement on a real helicopter, for risk of damaging the structure and equipment on the helicopter at such high attitude rates. The controller performs better in this test than the moderate attitude changes test since the step input is not limited to 20°. The step input is much larger, and the helicopter does not need to reach this value before turning back to normal orientation.

## 5.4 Trajectory controller performance

The first step is to validate that the model adequately represents the flight data. This is done in the frequency domain and the time domain. The frequency domain analysis provides a good way to compare the complete model with the flight data. In order to determine if the controller performs adequately, several test trajectories are flown in different operating conditions. The ADS-33 does not give performance criteria for trajectory tracking since it is designed for manned aircraft. The performance evaluation in this section is more qualitative than the previous section since it is difficult to add meaningful performance criteria to the trajectory tracking. Error metrics are only useful when compared to other controllers or different designs. For this reason, the focus is on determining the disturbance rejection capabilities and getting an overall view of the controller' ability to follow trajectories.

### 5.4.1 Flight test validation

The system model is first compared to the flight data to see if it is a good representation of reality. This is separate from the controller performance, which is evaluated in the sections after the model has been proven sufficient. A circular reference trajectory is given by

$$\left.\begin{aligned} x_{ref}(t) &= 15\sin\left(2\pi\left(\frac{1}{60}\right)\mathrm{t}\right) \\ y_{ref}(t) &= 15\sin\left(2\pi\left(\frac{1}{60}\right)\mathrm{t}+\frac{\pi}{2}\right)-15 \\ z_{ref}(t) &= -0.5t \\ \psi_{ref}(t) &= 12\left(\frac{\pi}{180}\right)t \end{aligned}\right\} \text{for } 0 \le t \le 120. \tag{5.10}$$

The constant yaw rate tests the helicopter's ability to keep tracking the reference signal with any heading.

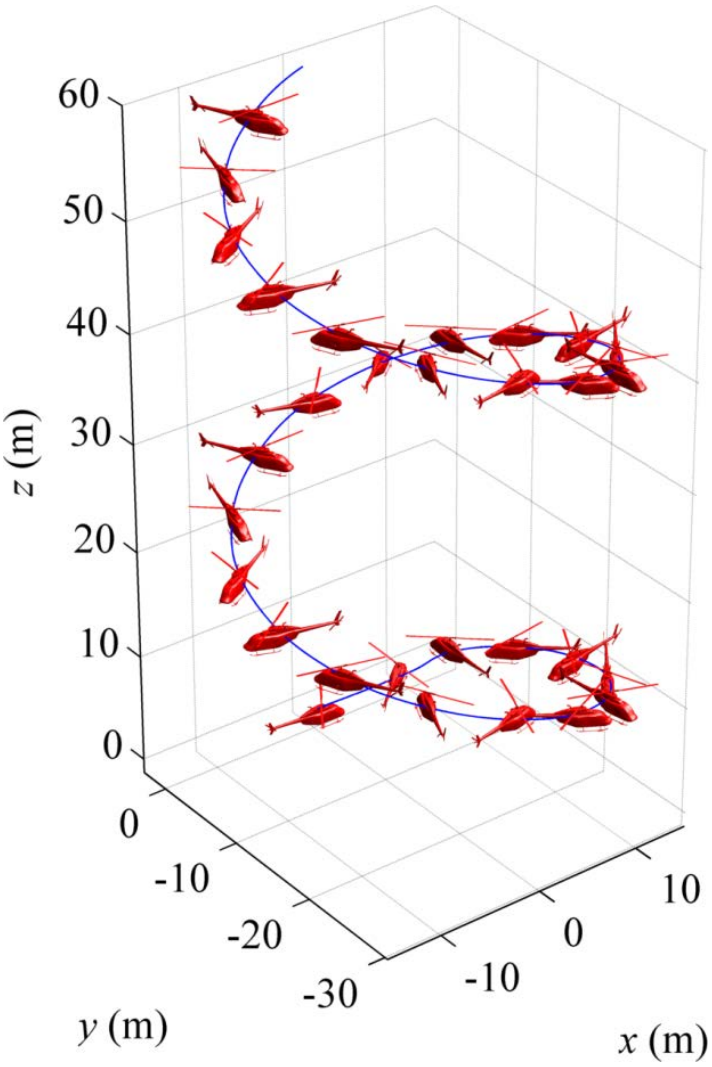



**Figure 5.10: 3D view of circular trajectory flown in 120 seconds.**

Figure 5.10 shows a 3D view of the trajectory flown to validate the model with flight test data. The trajectory tracking system model and flight data show a very good comparison. The helicopters in the figures are not scaled proportionally to the rest of the image, their purpose is to indicate attitude throughout the trajectory. The helicopters are spaced equally in time, not position.

Figure 5.11 (next page) shows the time domain performance for the first 30 seconds of this circular trajectory. The smooth trajectory is more suited to the helicopter than sharp corners. The performance to sharp corners will be evaluated later. There is a very good agreement between the system model and flight data, with an average NRMSE of 99.5% for all the channels.

The controller is able to track to the reference trajectory. The biggest deviation is at the start of the trajectory, when the helicopter is still accelerating from the stationary position. The state space model was created for hover, but the system model is able to simulate the flight data even for forward flight.

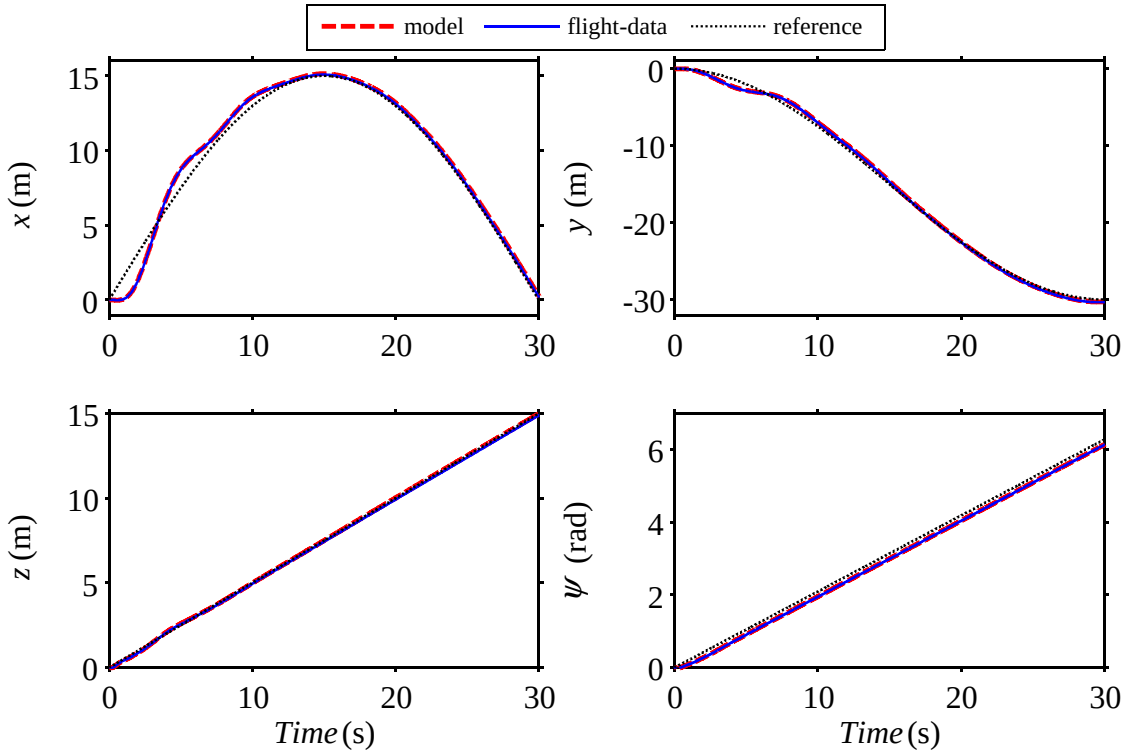



**Figure 5.11: First 30 seconds of a 120 second spiral trajectory with reference (5.10)**

The state space model validation (Chapter 3) used a limited prediction horizon to evaluate the model's performance, this was necessary since the model is unstable. With the controller in the loop, the system model is evaluated with an infinite horizon prediction. The comparison of flight data to the system model proves the system's ability to simulate the helicopter and controller accurately.

The outer loop frequency response verified the time domain results, but did not provide new information on the controller performance. The transfer functions showed very good agreement between the model and the flight data. The bode plots are included in Appendix C. The chirp signal did not work as well with the translational dynamics as with the angular dynamics. The results were sensitive to changes in the chirp amplitude and frequency range. A better measure of performance will be found in the time domain evaluation in the following section. From here on in this chapter, the flight data will be shown compared to the reference signal. The system model is validated and there is no need to clutter results with the model response.

### 5.4.2 Reference tracking

The most important measure of performance for the outer loop is the ability to track reference signals, as this is the controller's intended task. This primary task will be tested in different conditions.

#### 5.4.2.1 Figure 8 manoeuvre

Figure 5.12 is a 3D plot of the helicopter path and the reference trajectory. The figure 8 manoeuvre was conducted with the helicopter keeping its heading in the direction of forward flight, and not the specified heading in (5.11).

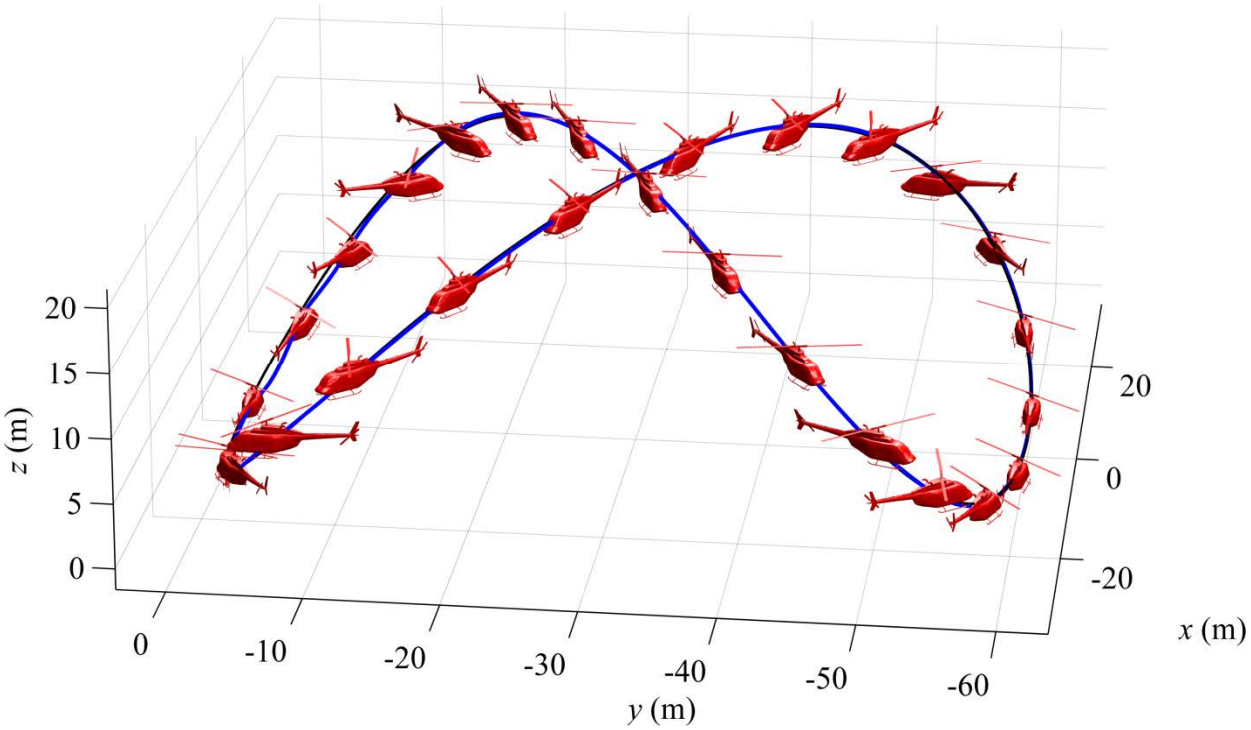



**Figure 5.12: 3D figure 8 manoeuvre flight in forward flight mode. (Reference: Black. Flight: Blue)**

The figure 8 manoeuvre reference trajectory is given by

$$\left.\begin{aligned} x_{ref}(t) &= 30\sin\left(2\pi\left(\frac{1}{60}\right)\mathrm{t}\right) \\ y_{ref}(t) &= 30\sin\left(2\pi\left(\frac{1}{120}\right)\mathrm{t}+\frac{\pi}{2}\right)-30 \\ z_{ref}(t) &= 10\sin\left(2\pi\left(\frac{1}{60}\right)\mathrm{t}+\frac{\pi}{2}\right)-10 \\ \psi_{ref}(t) &= 0 \end{aligned}\right\} \text{for } 0 \le t \le 120. \qquad (5.11)$$

The controller calculates the heading online by using the angle between its current position and the position of the reference trajectory. The same manoeuvre is completed with a constant heading; the results are shown in Figure 5.13.

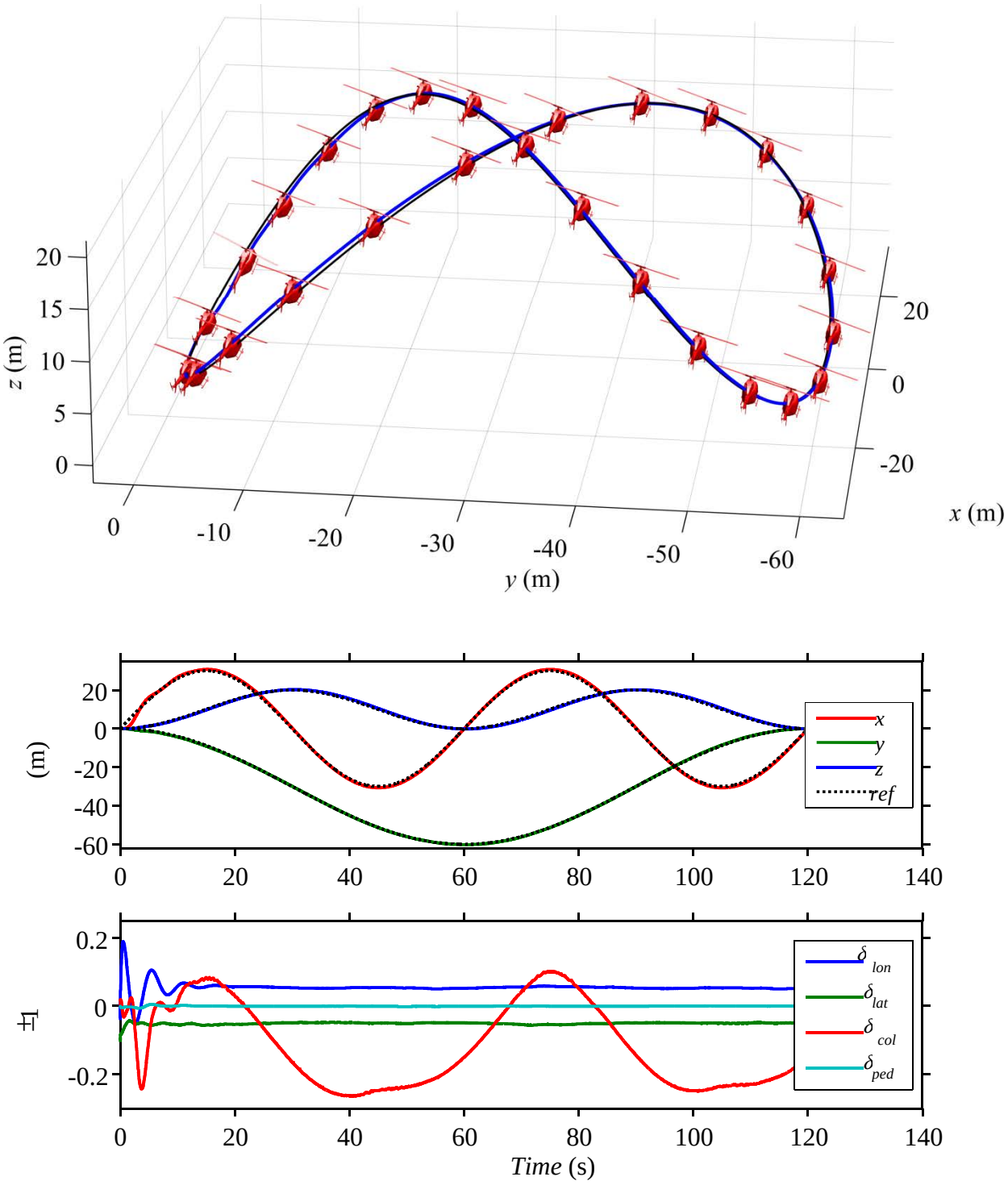



**Figure 5.13: 3D figure 8 trajectory with constant heading**

The inputs are kept small, reaching only 25% of their maximum values. It should be noted that the control inputs are smooth and do not contain oscillations. The helicopter reaches velocities of up to 5 m/s. This proves the controller is able to follow a trajectory that has much faster velocities than what is considered as hover. The importance of a good guidance controller can once again be seen here. The rate at which the trajectory is followed is determined by the guidance controller. Large rate changes will cause large attitude changes, which in most cases are not desirable.

It is important that even though the controller performance could be improved with more aggressive gains, one of the goals were to have smooth control inputs, not to oscillate around reference trajectories in an attempt to keep error metrics low. A constant collective pitch is more important than a small NRMSE, the controller is allowed to lose or gain a few meters of altitude. However, this is not the case when coming in for a landing. An advanced controller would have the ability to schedule these gains accordingly.

#### 5.4.2.2 Square manoeuvre

The next trajectory is a square manoeuvre with a reference heading explicitly included. The reference trajectory is given by

$$\begin{aligned}
&x_{ref}(t) = 10\sin\left(2\pi\left(\frac{1}{50}\right)\mathrm{t}+\frac{3\pi}{2}\right)+10 && 0 \le t \le 25\\
&x_{ref}(t) = 20 && 25 \le t \le 50\\
&x_{ref}(t) = 10\sin\left(2\pi\left(\frac{1}{50}\right)\mathrm{t}+\frac{\pi}{2}\right)+10, && 50 \le t \le 75\\
&y_{ref}(t) = 10\sin\left(2\pi\left(\frac{1}{50}\right)\mathrm{t}+\frac{\pi}{2}\right)+10, && 25 \le t \le 50\\
&y_{ref}(t) = 20, && 50 \le t \le 75\\
&y_{ref}(t) = 10\sin\left(2\pi\left(\frac{1}{50}\right)\mathrm{t}+\frac{3\pi}{2}\right)+10, && 75 \le t \le 100\\
&z_{ref}(t) = 0, && 0 \le t \le 100\\
&\psi_{ref}(t) = \psi_{ref} + \frac{\pi}{2}, && 25 \text{ s intervals.}
\end{aligned} \tag{5.12}$$

The controller's ability to fly a trajectory with a heading specified is shown here, the heading is not calculated online. The helicopter turns at each corner of the square to fly in its optimal configuration, pointing forward. It can be seen that this is not an ideal trajectory, the sharp bends are difficult to navigate, and it would be much better if these corners were rounded. This kind of task will be performed by the guidance controller. This is why it is important to determine the low-level

controller's operational limits. These limits will be used by the guidance controller to create a flight path.

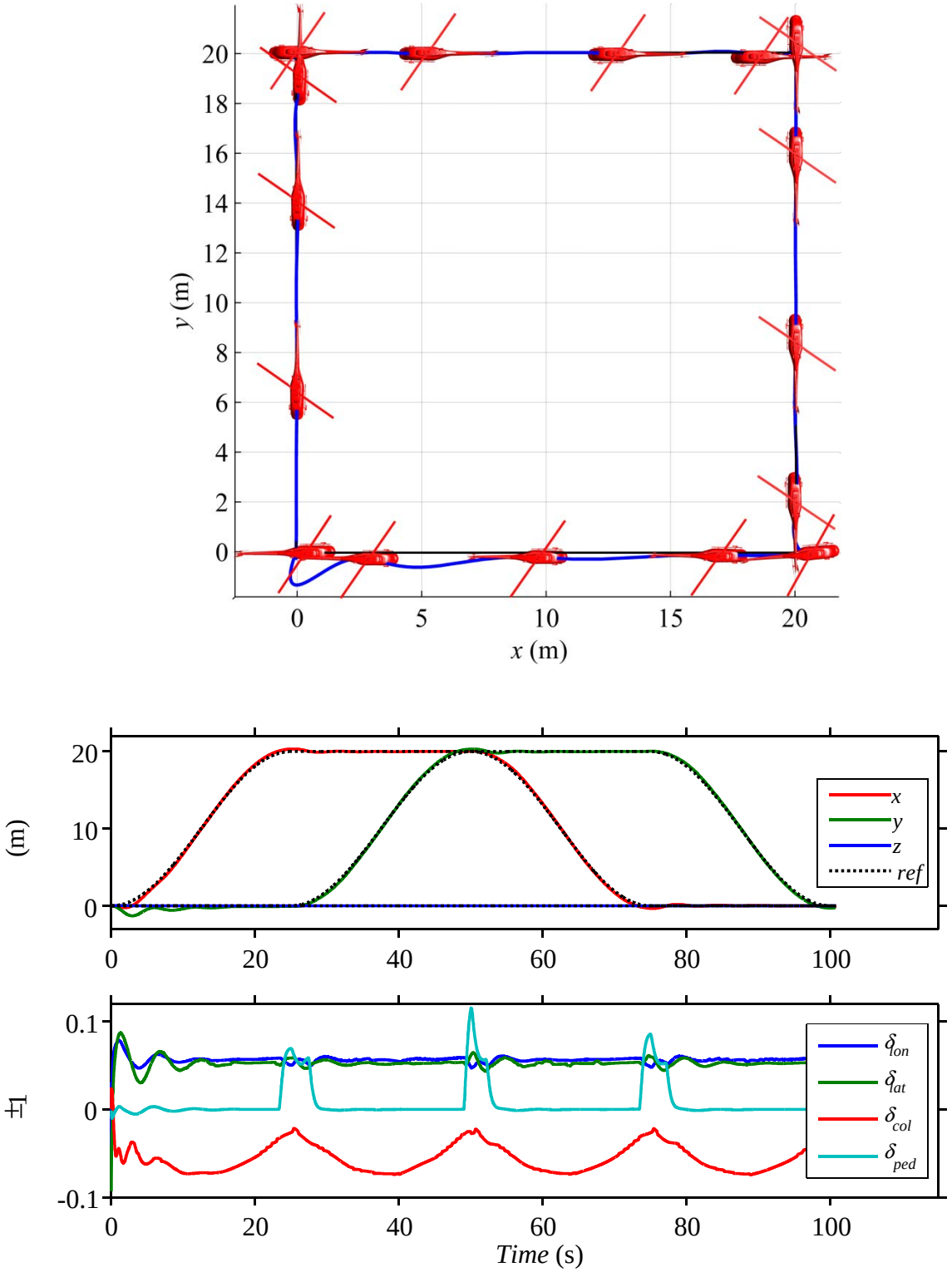



**Figure 5.14: Square trajectory flight data compared to reference signal.**

When the simulation is initialised, the helicopter is not in perfect hover trim. The helicopter shows some lateral translation as it settles into trimmed flight. The trajectory shown in Figure 5.14 is more

difficult than the smooth figure 8 manoeuvre, even though it is essentially 2 dimensional, with the altitude kept constant. The helicopter flies each side of the square, turning at the corners. Sharp corners are difficult, as the helicopter's velocity must change abruptly. The reference trajectory is given as smooth functions, to minimise overshoot at the changes in direction. If an outer loop velocity controller were to be used, the velocity at the corners could be set very low. This can be done indirectly with the position controller by providing a velocity reference and integrating to position references.

### 5.4.3 Reference tracking with disturbances

Disturbance rejection is a good measure of a controller's performance. In this section, disturbances will be seen as output disturbances, such as wind and as abrupt changes to the helicopter during flight. Evaluating the performance to changes in the model is a problem in the experimental setup. AeroSIMRC provides an interface to change basic model parameters in-flight. The user can set a percentage change, but the exact magnitude and units are not shown. It is possible to make measured changes to the models in the simulator, but the test setup with Simulink® in the loop did not allow changes to be made in-flight.

Figure 5.15 shows the same figure 8 trajectory that has been previously used. Figure 5.16 indicates when the disturbances enter the system and how the controller compensates for these changes.

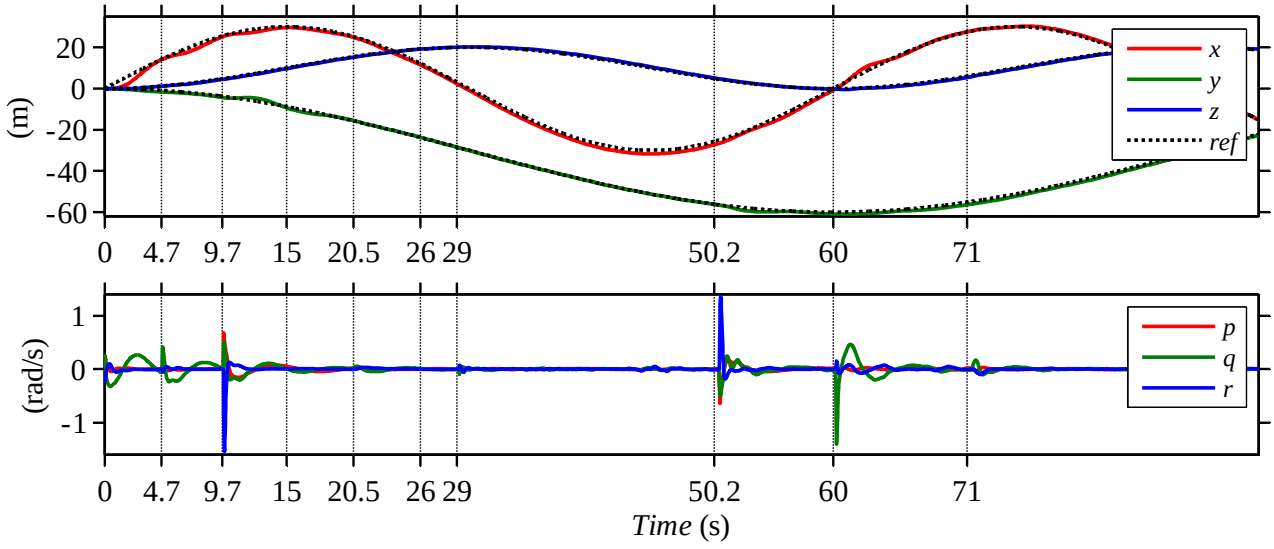



**Figure 5.15: Figure 8 manoeuvre with model changes.**

On a physical system, changes in the model could be caused by picking up a load, changing its weight and shifting its CG. Loss in control surface and motor efficacy are general disturbances to test the robustness in the case of unknown changes in the helicopter. It is important to test the controller's ability to work under different trim conditions. RC pilots trim their aircraft during flight

and before flight. These trim values can slowly change over time. In the tests performed here, the values are changed from their minimum to maximum, illustrating the worst-case scenario.

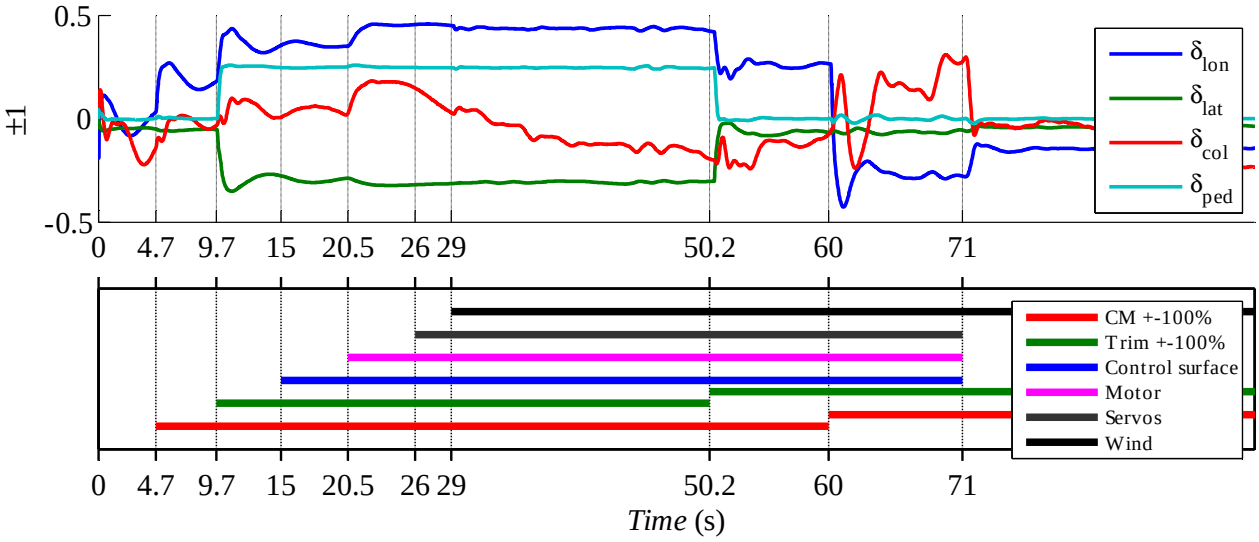



**Figure 5.16: Model changes, helicopter attitude rates due to changes, and controller inputs to compensate for changes.**

From Figure 5.16 it can be seen that the largest disturbance, at 9.7 seconds, is a change in trim condition. This step-like disturbance is quickly compensated for. The average longitudinal cyclic pitch is driven to 50% of the maximum value. Saturating inputs is always unwanted as this causes the helicopter to enter a nonlinear region where the controller is not proven stable. At 50.2 seconds, the trim values are changed from the minimum to the maximum values, again causing a large disturbance. Even though the helicopter experiences large angular rates, position tracking does not suffer much. The difference in time scale between attitude dynamics and translation act as a filter for attitude disturbances. The model disturbances evaluated here are the maximum values available in the simulator. The controller was able to keep the helicopter stable while following the reference trajectory. The evaluations done in this subsection prove the controllers validity when model changes are introduced in the simulator environment.

### 5.4.4 Wind disturbance rejection

A constant wind with high frequency components is simulated. The controller's ability to follow a trajectory and reject these environmental disturbances is shown in Figure 5.17. The trajectory given in (5.10) is once again used. The constant wind provides a low frequency disturbance. The turbulence added to the wind provides higher frequency disturbances. The controller performs the worst for winds from the side, this can be expected since the aerodynamic drag area is much larger than from the front. Side winds also have a greater influence on the tail rotor, changing the inflow rate.

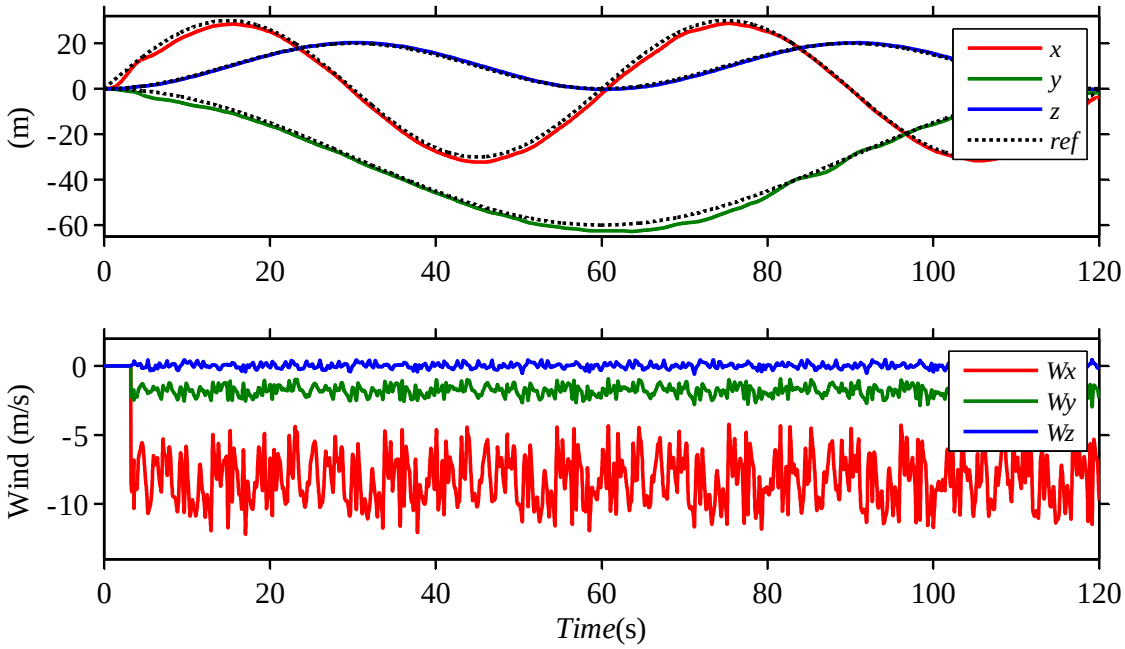



**Figure 5.17: 3D figure 8 trajectory in wind.**

In Figure 5.17, the helicopter is shown to closely track the reference signal, with the integrators compensating for the constant wind and the sufficient bandwidth that is available makes it possible to reject turbulence on the wind. Flying in winds of up to 10 m/s is akin to cruise-flight of this same speed. This is far from the controller's original objective of only hover and low speed flight. The ability of the controller to reject step disturbances, such as a wind gust has been evaluated, results are included in Appendix C. The controller was able to recover from a 20 m/s gust, that acted on the helicopter for 6 seconds.

## 5.5 Conclusion

The system model is validated using frequency and time domain analysis. The controllers are proven to work well in flight tests and the performance of the attitude controller is level 1, or close to level 1 for the tests performed. The trajectory tracking controller is able to control the helicopter in significant winds of up to 60% of the helicopter's maximum forward velocity. Even though the controller and model are developed for hover and slow forward flight, validation results show that the helicopter in AeroSIMRC is able to follow trajectories far out of hover. The controller is able to reject both low frequency disturbances, such as trim changes, and higher frequency disturbances, such as turbulence, without deviating too far from the flight path.

With all this said, there is still room for improvement on the controller. The best way to achieve this is to use a global optimisation strategy. It has been shown that when using a set of common optimisation goals, different types of linear controllers all have similar performance [81]. This

implies that the controller architecture should be chosen on more than just error metrics. Simplicity, maintenance, and development time are important factors to consider.

# Chapter 6: Conclusions and recommendations

*This final chapter ties all the conclusions made throughout the dissertation together. Limitations of the study are given and recommendations for future research are made.*

## 6.1 Conclusion

### 6.1.1 Modelling

The grey box system identification approach followed proved to be successful. The physical insight provided by grey box modelling showed where the simulator has the greatest differences compared to similar physical helicopters. It was clear that the rotor head in the simulator is extremely stiff and consequently the attitude rates are closely approximated by linear functions of the control inputs. The high rotor stiffness dominated the dynamics of the helicopter and is the most important parameter to estimate. Such big discrepancies bring into question the simulator's validity as an engineering simulation tool. The simulator showed significantly less cross coupling than results from comparable physical helicopters [30]. A black box method would not have been able to provide this information. Breaking down the state space model into smaller subsystems enabled consistent estimation of the unknown parameters. Placing physical constraints on the parameters in the subsystems ensured that the estimation algorithm could not simply increase parameters indefinitely, resulting in a system that oscillates closely around the target value. Problems were still experienced with this aspect of the model. The estimated model response had to be visually inspected for abnormally high frequency content. Merging multiple experiments for estimation data

provided a way to avoid overtraining the model. The results obtained from the identification were very dependent on the estimation flight data, once again emphasising the importance of data in system identification. Using multiple data sets for identification helped to get a more general model, but good data sets still had to be selected visually. This problem is partly caused by the measure of fit used. These error metrics do not penalise high frequency noise. The final model was validated using different sets of data from common helicopter manoeuvres. Confidence in the model validity was increased even more by the controller results. The controller was designed using the model and proven to fly in the simulator, with the simulator response closely matching the model.

### 6.1.2 Controller

The methodology used to develop the controller worked well. The weight selection of the cost function for the LQR controller design was an intuitive way to change the controller performance. The inner loop controller performed well according to the performance criteria provided by the ADS-33. Using a cascaded design with separate loops for the different time scales present in the helicopter was successful. Optimisation of the outer loop PID gains using the simplex search method showed good results. Using only the ITAE as performance criteria is not ideal. The controller gains are made too large in an attempt to achieve better tracking, but this sacrifices stability. Multiple objective optimisation should be used with objectives for gain and phase margins. This will provide more control over the desired performance, not just a minimum ITAE. All of these optimisations and design techniques depend on the quality of the model. This should emphasise the importance of the model.

The linear controllers were proven to work well on the nonlinear helicopter in the simulator. The controllers were capable of stable flight very far from the designed operating region. This once again sheds some doubt on how nonlinear the plant really is. The flight test data and model data from several trajectories were compared and showed a very good match. The trajectory controller's frequency response was compared for the flight data and model. Even though there was a good match, there is still some uncertainty as to why the phase and gain plots show strange characteristics. This area needs further investigation. Injecting low frequency chirp signals are difficult, due to the large translational velocities that are produced. It is also not straightforward to choose the correct amplitude and frequency combination, and the results are heavily dependent on these parameters.

Throughout the controller design, it became apparent that the most limiting aspect of the methodology followed here is clear design goals. Even when following an optimal design methodology, the problem is still to determine appropriate goals for the optimisation process. This

trend can also be seen in the literature, since very few studies have definite goals. Most studies aim to improve performance, or achieve autonomous flight, with very little to define how well the aircraft should perform. Using the ADS-33 is an attempt to move towards fixed design goals. In this study, it was only used as an evaluation guideline, and performance could have been improved by using it as a design guideline. Until such a standard exists for unmanned aircraft, the manned standards can be used.

### 6.1.3 Test environment

The test environment has been proven as a great asset to a new research group that does not own a physical hardware platform. It worked well to use a 3rd party simulator. Using a COTS gaming simulator saved time in the development phase and required no expert knowledge in helicopter simulation. Validation in a simulation is difficult if the simulator is created by the party that needs to validate its models. Using the 3rd party COTS simulator provided an environment closer to the real world, where the researcher does not have access to the mathematical equations governing the test bed. In this study, the target platform was the helicopter in the simulator, and therefore the models could be validated on the simulator. This does not make it possible to say that the models will be valid for the real world. The simulator would need to be validated using the target hardware platform.

The interface between Simulink® and AeroSIMRC served its purpose well. However, game based simulators have disadvantages. For example, the simulation cannot be executed faster than real-time. Problems were also experienced with automatically configuring the simulation, with user input necessary to start each simulation.

The controller can be used to test other guidance and navigation algorithms in simulation. In order to use the controller on a physical helicopter, another set of validation tests will be needed.

## 6.2 Future work

### 6.2.1 Modelling

There is room for improvement in the automation of the modelling process. The current methodology often requires human input. By defining a better metric for the model accuracy, this process can become less involved. This metric should be able to prioritise the helicopter outputs, putting more weight on the data that has a greater influence on the helicopter dynamics. A possible solution here is Principle Component Analysis (PCA) [82]. Using PCA will help to prioritise the important signals according to their variance, making it possible to compare different sets of data

with a single metric. This does not take into account high frequency oscillations in the estimated models, which have been a major problem. Combining frequency domain estimation with time domain estimation could help solve this problem. Having a way to determine if a particular set of flight data is suited for estimation will make using multiple data sets easier, eliminating the need for visual inspection. A possible solution is to use frequency-amplitude graphs to evaluate the signals [33]. These graphs indicate if the excitation signals used in identification will excite unwanted nonlinear behaviour.

### 6.2.2 Controller

As stated earlier, setting global design goals and optimising the controllers with these goals as the objective functions will greatly improve the performance. The LQR controller's cost function weights and the PID controller's gains will then be the variables that are adjusted. Performance should be specified in accordance with ADS-33 or a similar RWUAV standard, if one exists.

Estimating multiple models for different flight conditions will allow the controller to be gain scheduled, opening the full flight envelope. This can be the first step in expanding the controller. The LQR controller can be improved by changing it from a steady state controller to a time varying LQR controller. Stability will need to be carefully monitored, but the performance for different operating conditions can be improved [55]. The controller should eventually be expanded to allow full 3D flight. Quaternions will need to be used instead of Euler angles to avoid gimbal locks.

According to the survey by Kendoul, adaptive control should be the final goal of the lower level controllers [9]. This has proven a difficult issue in aerospace, as the adaptive controllers promise better performance, but the stability analysis is difficult to prove. One way to overcome this problem is to use a Multiple Model Adaptive Control (MMAC) scheme with mixing [83]. This enables the use of LTI tools for stability analysis, and the advantages of adapting to changes in the plant. The method proposed by Kuipers and Ioannou mixes the controllers adaptively. This adaptive law is proven stable and overcomes the stability problems of gain scheduling when switching between controllers.

A subject that has only briefly been touched in this study is navigation. Navigation, and specifically state estimation, will become extremely important if the controller is implemented on a hardware platform. The most likely candidate for this is a Kalman filter. This will change the compensator from an LQR compensator to a Linear Quadratic Gaussian (LQG) compensator.

### 6.2.3 Test environment

The test environment can be expanded to include Hardware-In-The-Loop (HITL) simulations. This will make real sensor data available, and actuator dynamics will be included in the simulations. The controller will then be implemented on the flight computer. This simulation will be realistic in all aspects but the flight dynamics.

## 6.3 Closure

Autonomous flight has been achieved with the use of a model based optimal controller. The methodology and results have been validated in a 3rd party simulation environment. The methodology developed in this study is therefore ideal for new researchers in the field of rotary winged UAVs, who do not have access to hardware. It is inexpensive and easy to set up. It provides a safe way to test flight controllers. The modelling methodology is reliable and the controller development is simple enough to be used for first-flight cases.

# References


[1] Office of the Secretary of Defence, “Unmanned Systems Integrated Roadmap FY2011-2036,” 2011.

[2] W. A. Basson, “Fault Tolerant Adaptive Control of an Unmanned Aerial Vehicle,” Stellenbosch University, 2011.

[3] R. Best, *Intelligence, Surveillance, and Reconnaissance (ISR) Acquisition: Issues for Congress*. DIANE Publishing, 2010, p. 26.

[4] Office of the Secretary of Defence, “Unamnned Aircraft Systems Roadmap 2005-2030,” 2005.

[5] Littoral and Mine Warfare Public Affairs, “Navy Conducts COBRA Flight Test from Fire Scout,” 2010. [Online]. Available: http://www.navy.mil/submit/display.asp?story_id=56748. [Accessed: 10-Nov-2013].

[6] Yamaha, “Yamaha RMAX,” 2013. [Online]. Available: http://rmax.yamaha-motor.com.au/. [Accessed: 11-Nov-2013].

[7] E. Fratkin, “Stanford auonomous helicopter,” 2010. [Online]. Available: http://cs.stanford.edu/groups/helicopter/photos.html. [Accessed: 02-Dec-2013].

[8] H. Huang, K. Pavek, and B. Novak, “A framework for autonomy levels for unmanned systems (ALFUS),” in *AUVSI’s Unmanned Systems*, 2005, no. June, pp. 1–9.

[9] F. Kendoul, “Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems,” *J. F. Robot.*, vol. 29, no. 2, pp. 315–378, Mar. 2012.

[10] “International Aerial Robotics Competition,” 2013. [Online]. Available: http://www.aerialroboticscompetition.org/pastmissions.php. [Accessed: 27-Nov-2013].

[11] P. Gili and M. Battipede, “A comparative design of a MIMO neural adaptive rate damping for a nonlinear helicopter model.,” *Eur. Symp. Artif. Neural Networks*, pp. 159–164, 2000.

[12] A. R. S. Bramwell, G. Done, and D. Balmford, *Bramwell’s helicopter dynamics*, 2nd ed. Avon: Butterworth-Heinemann, 2001.

[13] G. D. Padfield, *Helicopter Flight Dynamics*, 2nd ed. Oxford, UK: Blackwell Publishing Ltd, 2007.

[14] I. A. Raptis and K. P. Valavanis, *Linear and Nonlinear Control of Small-Scale Unmanned Helicopters*, 1st ed., vol. 45. Springer, 2010.

[15] "Tuning - Paparazzi." [Online]. Available: http://paparazzi.enac.fr/wiki/Tuning. [Accessed: 02-May-2012].

[16] H. Shim, "Hierarchical Flight Control System Synthesis for Rotorcraft-based Unmanned Aerial Vehicles," University of California, 2000.

[17] B. Mettler, "Modeling small-scale unmanned rotorcraft for advanced flight control design," Carnegie Mellon University, 2001.

[18] J. Seddon and S. Newman, *Basic Helicopter Aerodynamics*, 3rd ed. Chichester, West Sussex: John Wiley & Sons, 2011.

[19] R. Prouty, *Helicopter performance, stability, and control*. Malabar, Florida: Krieger Publishing Company, 2002.

[20] B. Ren, S. S. Ge, C. Chen, C.-H. Fua, and T. H. Lee, *Modeling, Control and Coordination of Helicopter Systems*. New York, NY: Springer New York, 2012.

[21] C. Munzinger, "Development of a real-time flight simulator for an experimental model helicopter," Georgia Institute of Technology, 1998.

[22] A. Budiyono, T. Sudiyanto, and H. Lesmana, "First principle approach to modeling of small scale helicopter," in *ICIUS*, 2007, p. 11.

[23] R. Cunha and C. Silvestre, "SimModHeli: A Dynamic Simulator for Model-Scale Helicopters," in *11th Mediterranean Conference on Control and Automation*, 2003, pp. 1–6.

[24] U. B. Hald, M. V. Hesselbaek, J. J. T. J. Holmgaard, C. S. Jensen, S. L. Jakobsen, and M. Siegumfeldt, "Autonomous helicopter-modelling and control," Aalborg, 2005.

[25] R. K. Heffley and M. A. Mnich, "Minimum-Complexity Helicopter Math Model Program," Moffet Field, CA, 1986.

[26] K. T. Christensen, K. G. Campbell, C. D. Griffith, C. M. Ivler, M. B. Tischler, and J. W. Harding, “Flight Control Development for the ARH-70 Armed Reconnaissance Helicopter Program,” in *American Helicopter Society 63rd Annual Forum*, 2007.

[27] J. Downs, R. Prentice, and S. Alzell, “Control system development and flight test experience with the MQ-8B fire scout vertical take-off unmanned aerial vehicle (VTUAV),” in *American Helicopter Society 63rd Annual Forum*, 2007.

[28] B. Mettler, M. B. Tischler, and T. Kanade, “System Identification of Small-Size Unmanned Helicopter Dynamics,” in *American Helicopter Society 55th forum*, 1999.

[29] M. La Civita, “Integrated Modeling and Robust Control for Full-Envelope Flight of Robotic Helicopters,” Carnegie Mellon University, 2002.

[30] W. Yuan and J. Katupitiya, “A Time-Domain Grey-Box System Identification Procedure for Scale Model Helicopters,” *Proc. 13th Aust. Conf.*, pp. 1–10, 2011.

[31] G. Franklin, J. Powell, and A. Emami-Naeini, *Feedback control of dynamic systems*. Pearson Education, 2011.

[32] L. Ljung, *System identification: theory for the user*. PTR Prentice Hall, 1987.

[33] P. van Vuuren, “Robustness estimation of self-sensing active magnetic bearings via system identification,” North-West University, 2009.

[34] D. Shim, H. Kim, and S. Sastry, “Hierarchical control system synthesis for rotorcraft-based unmanned aerial vehicles,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2000, no. August.

[35] W. Yuan and J. Katupitiya, “A grey-box system identification procedure for scale model helicopters based on frequency-domain estimation methods,” in *Proceedings of the 28th Congress of the International Council of the Aeronautical Sciences*, 2012, pp. 1–10.

[36] M. B. Tischler and M. G. Cauffman, “Frequency-response method for rotorcraft system identification: Flight applications to BO 105 coupled rotor/fuselage dynamics,” *J. Am. Helicopter Soc.*, vol. 37, no. 3, pp. 3–17, 1992.

[37] L. Ljung, “System Identification Toolbox: User’s Guide,” 2012.

[38] E. N. Johnson and S. K. Kannan, “Adaptive Trajectory Control for Autonomous Helicopters,” *Control*, pp. 1–49.

[39] G. Chowdhary and E. Johnson, “Adaptive neural network flight control using both current and recorded data,” in *AIAA Guidance, Navigation, and Control Conference*, 2007, no. August, pp. 1–20.

[40] A. Coates, P. Abbeel, and A. Ng, “Learning for control from multiple demonstrations,” *Proc. 25th Int. Conf. Mach. Learn.*, 2008.

[41] R. D. Garcia, K. P. Valavanis, and A. Kandel, “Autonomous helicopter navigation during a tail rotor failure utilizing fuzzy logic,” *Mediterr. Conf. Control Autom.*, pp. 1–6, 2007.

[42] I. A. Raptis, K. P. Valavanis, S. Member, and W. A. Moreno, “A Novel Nonlinear Backstepping Controller Design for Helicopters Using the Rotation Matrix,” *IEEE Trans. Control Syst. Technol.*, vol. 19, no. 2, pp. 465–473, 2011.

[43] T. Koo, Y. Ma, and S. Sastry, “Nonlinear control of a helicopter based unmanned aerial vehicle model,” *IEEE Trans. Control Syst. Technol.*, 2001.

[44] V. Gavrilets, I. Martinos, B. Mettler, and E. Feron, “Control logic for automated aerobatic flight of miniature helicopter,” *Guid. Navig. Control Conf.*, no. August, pp. 1–8, 2002.

[45] V. Gavrilets, I. Martinos, B. Mettler, and E. Feron, “Aggressive Maneuvering Flight Tests of a Miniature Robotic Helicopter,” *Exp. Robot. VIII*, 2003.

[46] H. Shim, T. Koo, and F. Hoffmann, “A comprehensive study of control design for an autonomous helicopter,” *Decis. Control*, no. December, pp. 3653–3658, 1998.

[47] “Arducopter,” 2013. [Online]. Available: copter.ardupilot.com. [Accessed: 27-Nov-2013].

[48] M. F. Weilenmann, U. Christen, and H. P. Geering, “Robust Helicopter Position Control at Hover,” *Meas. Control*, pp. 2491–2495.

[49] H. Kim, H. R. Dharmayanda, T. Kang, A. Budiyono, G. Lee, and W. Adiprawita, “Parameter identification and design of a robust attitude controller using H ∞ methodology for the raptor E620 small-scale helicopter,” *Int. J. Control. Autom. Syst.*, vol. 10, no. 1, pp. 88–101, Feb. 2012.

[50] I. A. Raptis, K. P. Valavanis, and G. J. Vachtsevanos, “Linear Tracking Control for Small-Scale Unmanned Helicopters,” *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 4, pp. 995–1010, 2012.

[51] M. V. Kumar, P. Sampath, S. Suresh, S. N. Omkar, and R. Ganguli, “Design of a stability augmentation system for a helicopter using LQR control and ADS-33 handling qualities specifications,” *Aircr. Eng. Aerosp. Technol.*, vol. 80, no. 2, pp. 111–123, 2008.

[52] B. Anderson and J. Moore, *Linear optimal control*, vol. 197, no. 1. Englewood Cliffs, New Jersey: Prentice Hall, Inc, 1971.

[53] C. van Schalkwyk, “Full State Control of a Fury X-Cell Unmanned Helicopter,” University of Stellenbosch, 2008.

[54] E. Mosca, *Optimal, predictive, and adaptive control*, vol. 3. 1995.

[55] A. E. Bryson, “Time-Varying Linear-Quadratic Control,” *J. Optim. Theory Appl.*, vol. 100, no. 3, pp. 515–525, 1999.

[56] K. S. Soo, “Yamaha RMAX heli kills,” *Naver News*, Imsil, 03-Aug-2009.

[57] M. D. Proctor, M. Bauer, and T. Lucario, “Helicopter Flight Training Through Serious Aviation Gaming,” *J. Def. Model. Simul. Appl. Methodol. Technol.*, vol. 4, no. 3, pp. 277–294, Jul. 2007.

[58] L. Ribeiro and N. Oliveira, “UAV autopilot controllers test platform using Matlab/Simulink and X-Plane,” *2010 IEEE Front. Educ. Conf.*, pp. S2H–1–S2H–6, Oct. 2010.

[59] S. Roy, “Selecting the Right Aircraft Simulation Tool,” 2011.

[60] “RTDynamics Rotorlib,” 2010. [Online]. Available: http://www.rtdynamics.com/Release/Whitepapers/4.0/FWL_FDM_Whitepaper.pdf. [Accessed: 27-Nov-2013].

[61] “Presagis,” 2013. [Online]. Available: www.presagis.com. [Accessed: 27-Nov-2013].

[62] E. F. Sorton and S. Hammaker, “Simulated flight testing of an autonomous aerial vehicle using FlightGear,” *Am. Inst. Aeronaut. Astronaut.*, no. September, 2005.

[63] T. A. M. Abdunabi, M. El-Gelani, and N. M. Nasr, “Modeling and autonomous flight simulation of a small unmanned aerial vehicle,” in *13th International Conference on Aerospace Sciences & Aviation Technology*, 2009, pp. 1–10.

[64] J. F. Horn and D. O. Bridges, “Development of a low-cost, multi-disciplinary rotorcraft simulation facility,” *J. Aerosp. Comput. Inf. Commun.*, vol. 2, no. July, pp. 267–284, 2005.

[65] Laminar Research, “X-plane 10: Manual.” 2012.

[66] M. Guillén, “AeroSIMRC.” [Online]. Available: www.aerosimrc.com. [Accessed: 30-Jul-2013].

[67] B. Mettler, M. B. M. Tischler, and T. Kanade, “System identification modeling of a small-scale unmanned rotorcraft for flight control design,” *J. Am. Helicopter Soc.*, vol. 47, no. 1, pp. 50–63, 2002.

[68] K. P. Valavanis, *Advances in Unmanned Aerial Vehicles*, vol. 33. Dordrecht: Springer Netherlands, 2007.

[69] T. Bak, “Lecture notes-modeling of mechanical systems,” *Department of Control Engineering*. 2002.

[70] R. Nelson, *Flight stability and automatic control*, First., vol. 2. McGraw-Hill, 1998.

[71] R. Murray and S. Sastry, “A mathematical introduction to robotic manipulation,” 1994.

[72] R. C. Hibbeler, “Engineering Mechanics: Statics,” 2004.

[73] R. Garcia, “Designing an autonomous helicopter testbed: from conception through implementation,” ProQuest, 2008.

[74] Y. Barlas, “Formal aspects of model validity and validation in system dynamics,” *Syst. Dyn. Rev.*, vol. 12, no. 3, pp. 183–210, 1996.

[75] B. Arain and H. Pota, “Flight Control of a Rotary wing UAV including Flapping Dynamics,” *World Congr.*, vol. 18, no. 1, pp. 10373–10378, 2011.

[76] H. Chao, Y. Cao, and Y. Chen, “Autopilots for small unmanned aerial vehicles: A survey,” *Int. J. Control. Autom. Syst.*, vol. 8, no. 1, pp. 36–44, Feb. 2010.

[77] G. Franklin, J. Powell, and A. Emami-Naeini, *Feedback control of dynamic systems*, 6th ed. 1991.

[78] Anon, “Aeronautical Design Standard Performance Specification Handling Qualities Requirements for Military Rotorcraft - ADS-33E-PRF,” Redstone Arsenal, Alabama, 2000.

[79] M. Cotting, “Evolution of Flying Qualities Analysis: Problems for a New Generation of Aircraft,” Virginia Polytechnic Intstitute and State University, 2010.

[80] K. J. Aström and R. M. Murray, *Feedback systems: an introduction for scientists and engineers*. Princeton university press, 2010.

[81] H. Tischler, J. Lee, and J. Colbourne, “Optimization and comparison of alternative flight control system design methods using a common set of handling-qualities criteria,” *AIAA Pap.*, no. 4266, 2001.

[82] L. van S. Hager, “Adaptive neural control of a single-rotor small-scale helicopter,” North-West University, 2013.

[83] M. Kuipers and P. Ioannou, “Multiple Model Adaptive Control With Mixing,” *IEEE Trans. Automat. Contr.*, vol. 55, no. 8, pp. 1822–1836, 2010.

# Appendix A

## A.1 Stability derivatives

$X_u$, $Y_v$: The speed force damping derivative, or drag damping derivative, has a negative value and increases with speed.

$M_u$: This derivative describes the pitch response to an increase in lateral velocity, or speed stability. The horizontal stabilizer on the tail will affect this derivative. An increase in forward speed causes a positive pitching moment, or nose up movement, which causes a decrease in forward speed. This is a stabilizing effect, but can degrade handling qualities.

$L_v$: The lateral counterpart of $M_u$ is $L_v$, the dihedral effect, or the roll response to an increase longitudinal velocity. $L_v$ is positive for sideslip to the right, causing the helicopter to roll to the left. A positive $L_v$ has a stabilising effect.

$L_u$, $M_v$: The off-axis counterparts of $L_v$ and $M_u$, caused by an increase in the main rotor wake. This increase causes a tilt of the TPP that creates a moment about the CG.

$L_b$, $M_a$: The pitch rotor-spring coefficient and roll rotor-spring coefficient dominate the helicopter dynamics. $M_a$ is the roll response to a positive change in the flapping angle $a$. $L_b$ is the pitch response to a positive change in the flapping angle $b$.

$L_a$, $M_b$: The cross-coupling spring coefficients. In other words, the off-axis response to changes in the longitudinal and lateral flapping. Similar to $L_b$ and $M_a$.

$A_b$, $B_a$: Cross-coupling terms in the rotor flapping equations. The lateral flapping caused by longitudinal flapping and vice versa.

$Z_a$, $Z_b$: The heave response caused by flapping.

$Z_w$: Heave damping derivative, or drag damping derivative. By definition $Z_w$ has a negative value for an increase in heave velocity. $Z_w$ will give an indication of acceleration after a vertical gust is experienced.

$Z_r$: Yaw rate influence on heave response.

$N_w$: Cross coupling from an increase in heave velocity to the yaw response.

$N_r$: The yaw damping derivative is governed by the yaw rate gyro on the helicopter. In the absence of such a gyro, it would be governed by the tail rotor, in much the same way as $Z_w$.

$K_r$: Yaw rate feedback gain. This parameter is governed by the yaw rate gyro on the helicopter.

The control derivatives describe the aircraft response to control inputs.

$A_{lon}$, $B_{lat}$, $Z_{col}$, $N_{ped}$: The gain from the input to the on-axis output.

$A_{lat}$, $B_{lon}$, $N_{col}$: Cross-coupling gain from the input to the off-axis output.

## A.2 Merged experiments

The fit value of the identified model can be plotted on a bar graph for each of the experiments and for each of their output channels. Typically a bar diagram (e.g. Figure A.1) would be used to determine which experiments are unfit for use.

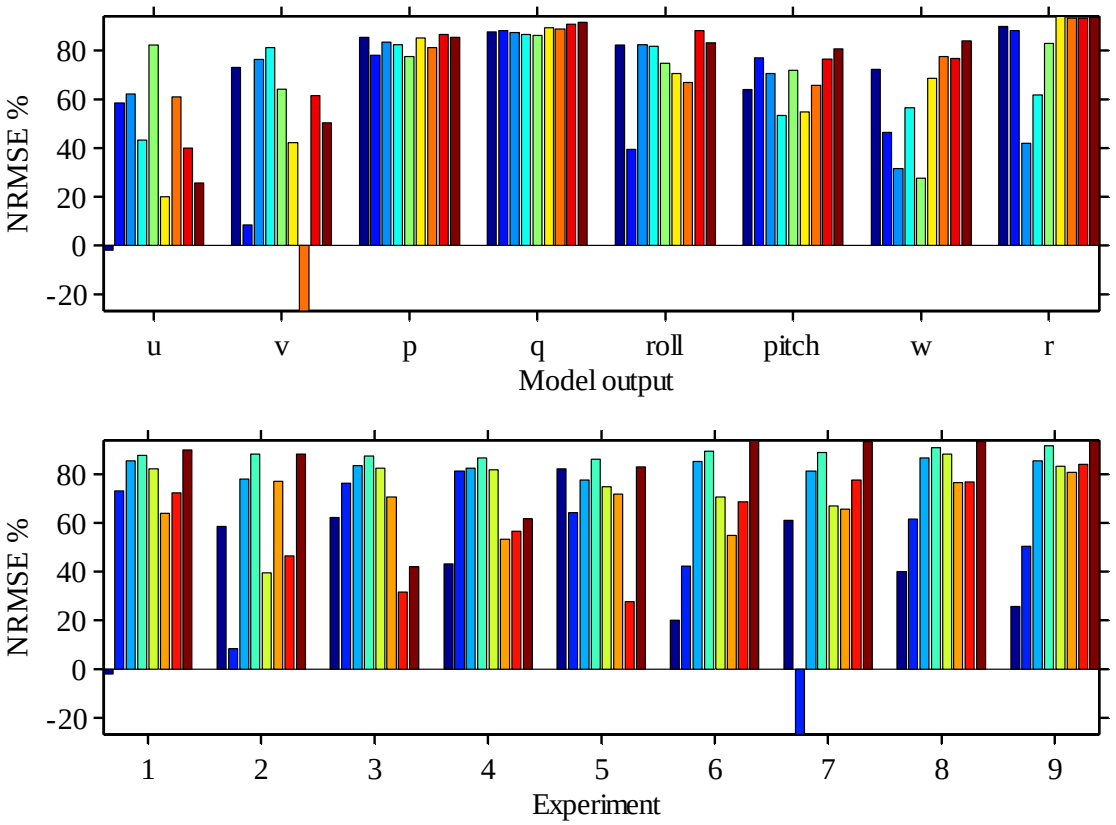



**Figure A.1: Visual method for comparing different experimental results**

The first diagram in Figure A.1 gives a good idea of the experiments that entered a nonlinear region. The bars are organised by experiment and the vertical axis is the fit value of the estimated model to the specific validation data. Each bar is one of the eight measured outputs of the simulator. Experiments 1, 2 and 7 have very low fit values for some of their outputs. Upon further inspection of the second graph in Figure A.1, where the bars are now grouped by output, all of the validation results have relatively good fit values except for two of the experiments. This provides information on which of the experiments should be repeated. The first plot makes it clear that the angular rates for this data set were very well estimated. Further investigation can then be made to why the

translational velocities did not perform so well. The investigation could possibly include returning to the earlier identification steps where the initial values for translational velocity were determined.

These problem experiments are then removed from the dataset and replaced with new ones. There is no exact value for poor performance. Generally, if a model is seen to have NRMSE lower than 30% for any of its outputs, the experiment is removed. These experiments can be removed, because the identified model is linear and time invariant, so it is unable to model any nonlinearity. This process eliminates unwanted experiments from the merged set. The choice of validation data is important, as it will directly influence the choice of final model parameters.

# Appendix B

## B.1 Model validation

Model validation results that were not included in Chapter 3. Figure B.1 shows good fit values except for the heave velocity.

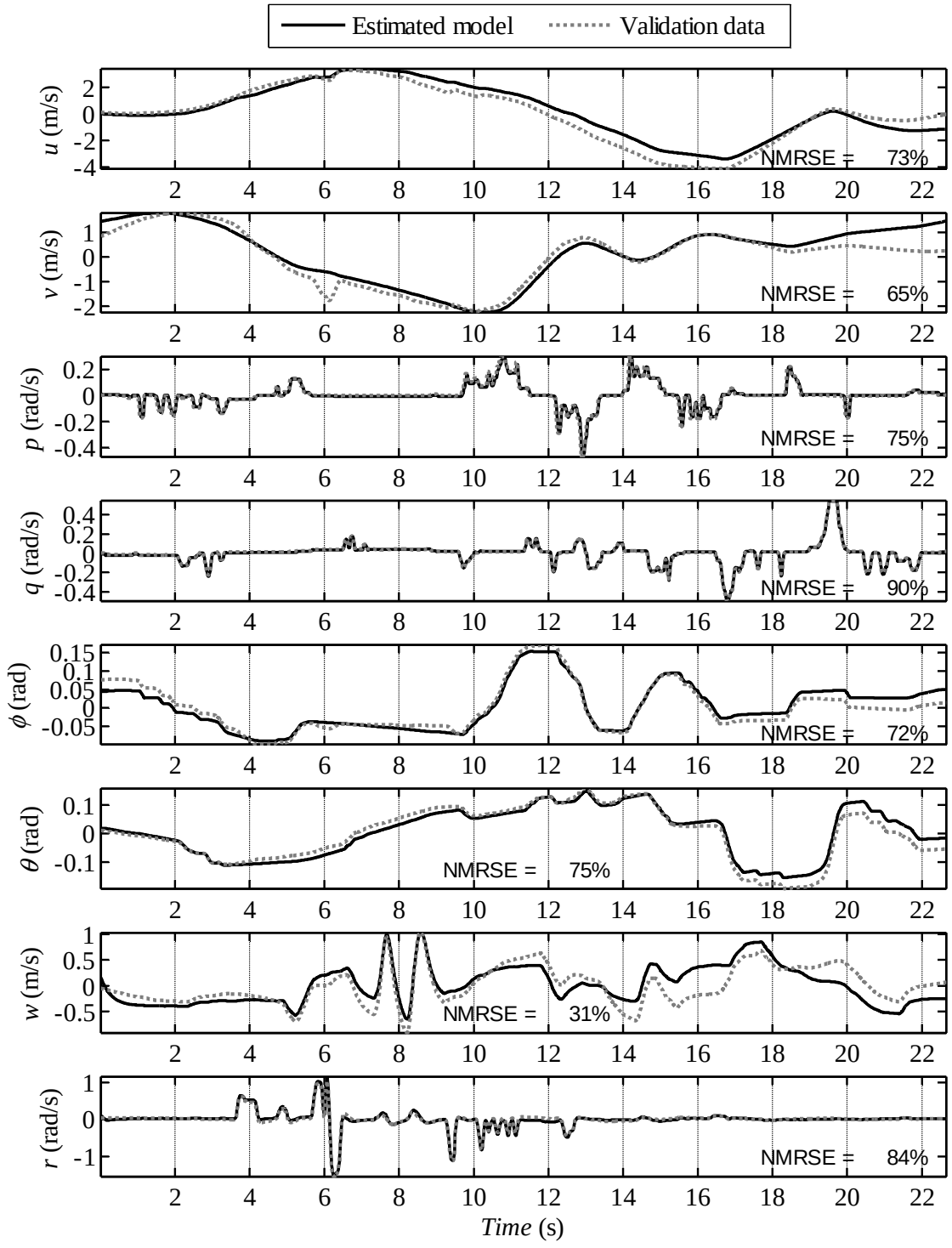



**Figure B.1: Model prediction results for a counter clockwise circular trajectory**

The poor fit to heave velocity is due to the complex inflow dynamics that cannot be linearly modelled. There is some cross coupling that is not modelled. At 6 seconds, an abrupt change in yaw rate can be seen to influence the lateral translational velocity.

Figure B.2 shows that the model is able to predict a complex manoeuvre such as a figure 8. Angular rates estimate very well, proving that the iteratively determined rotor time constant and rotor spring constants are a good representation of the helicopter.

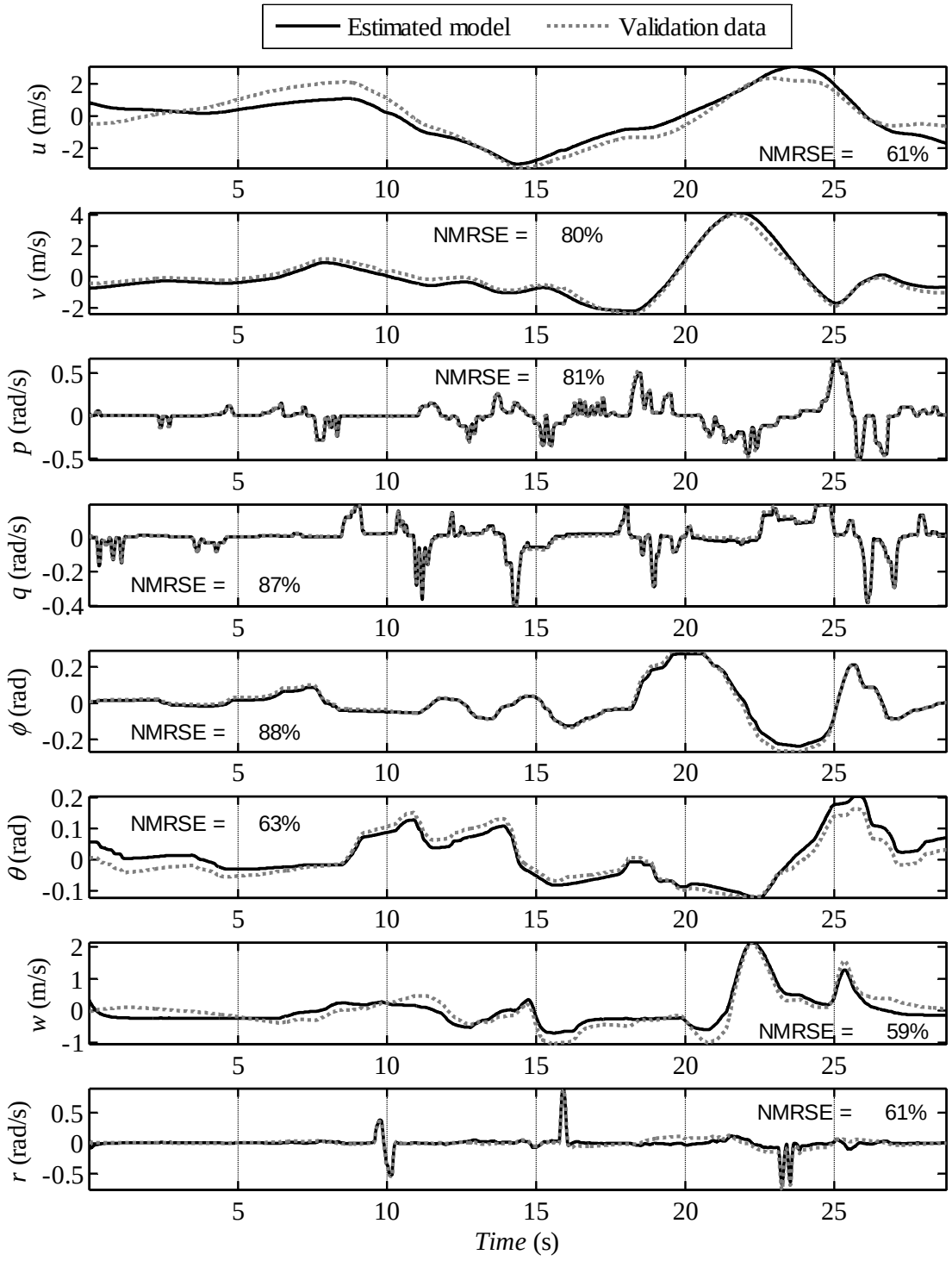



**Figure B.2: Model prediction results for a figure 8 trajectory**

Fit values of 60% are acceptable, especially since it can be seen from Figure B.2 that the model follows the same trend as the flight data. If the model were developed as a simulator, and not for control purposes, higher NRMSE would be desired.

Figure B.3 shows that the model is unable to predict the extreme aerobatic manoeuvres that the RC helicopter is capable of. This flight data set contains highly nonlinear behaviour, with the pilot pushing helicopter to the limits of its flight envelope.

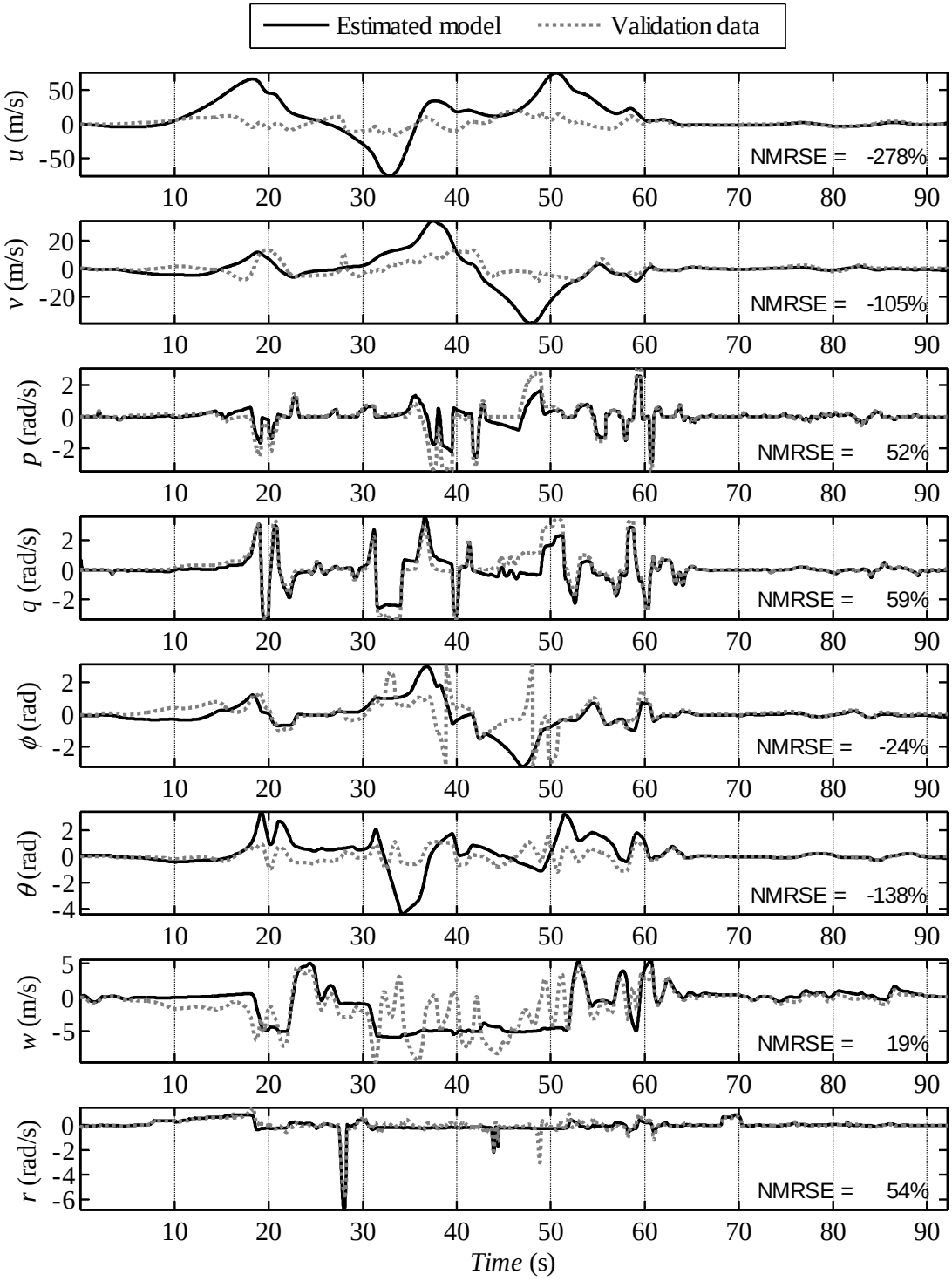



**Figure B.3: Model prediction results for extreme aerobatic flight**

Several linear models would be needed to model such a data set. The angular rates perform better than would be expected. This supports the notion that the simulator's angular dynamics are linear. The model predicts translational velocities that are much higher than the flight-data. This can be expected from the poor angular position estimation, which determines the translational velocities. Negative NRMSE values indicate that the model has worse prediction capabilities than simply taking the mean value of the flight data, which would give a 0% fit value.

# Appendix C

## C.1 ADS-33 bandwidth specifications

Figure C.1 shows the definitions used for the handling quality evaluation in Chapter 5.

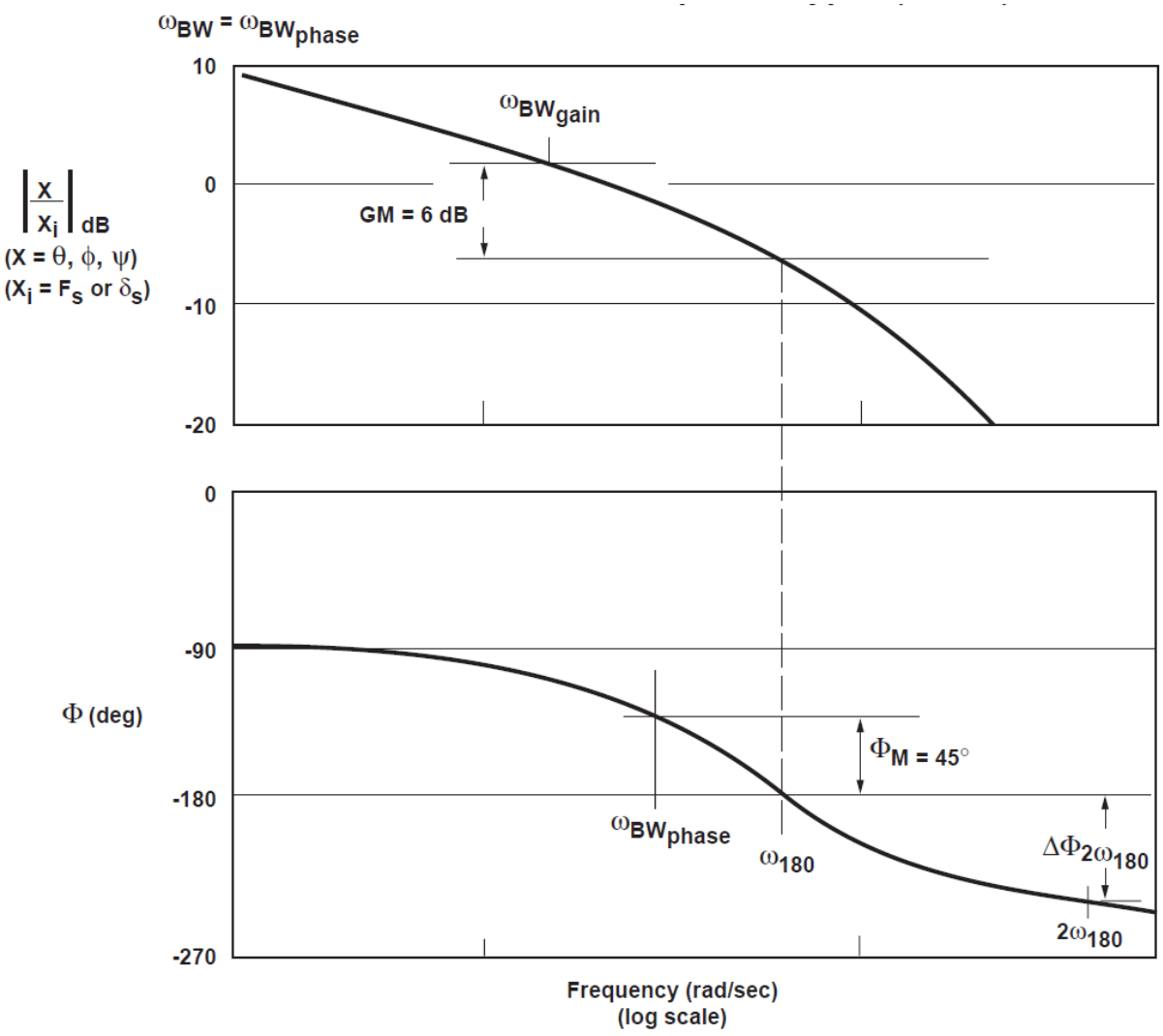



**Figure C.1: ADS-33 definition of bandwidth. From [78]**

$\omega_{BWphase}$ is the frequency at which the phase angle is -135°. $\omega_{BWgain}$ is the frequency at which the magnitude is 6 dB higher than the magnitude at $\omega_{180}$. The difference between the phase angle at $2\omega_{180}$ and at $\omega_{180}$ is given by $\Delta\Phi_{2\omega_{180}}$.

## C.2 Trajectory control (Outer loop)

### C.2.1 Outer loop frequency response validation

The outer loop has four dominant transfer functions indicated by larger entries in the following matrix,

$$\mathbf{T}_{outer}(s)=\begin{bmatrix} \boldsymbol{\frac{x}{x_{ref}}}(s) & \frac{x}{y_{ref}}(s) & \frac{x}{z_{ref}}(s) & \frac{x}{\psi_{ref}}(s) \\ \frac{y}{x_{ref}}(s) & \boldsymbol{\frac{y}{y_{ref}}}(s) & \frac{y}{z_{ref}}(s) & \frac{y}{\psi_{ref}}(s) \\ \frac{z}{x_{ref}}(s) & \frac{z}{y_{ref}}(s) & \boldsymbol{\frac{z}{z_{ref}}}(s) & \frac{z}{\psi_{ref}}(s) \\ \frac{\psi}{x_{ref}}(s) & \frac{\psi}{y_{ref}}(s) & \frac{\psi}{z_{ref}}(s) & \boldsymbol{\frac{\psi}{\psi_{ref}}}(s) \end{bmatrix}. \tag{6.1}$$

The larger font entries indicate the on-axis transfer functions. The frequency domain analysis of the closed loop verifies the time domain response. The frequency response was calculated with the same method as the attitude loop frequency response. Figure C.2 shows the response of the longitudinal transfer function. For the sake of brevity, the other transfer function responses are not included here. There is a very small frequency range that the chirp signal can be applied. The model transfer functions cannot be solved analytically due to the nonlinear transformation of the position coordinates from the body reference frame to the inertial reference frame. The transformation matrix can be linearised, or alternatively the frequency response can be calculated from the input output data as with the flight data.

It can be seen from the bode diagrams in Figure C.2 (next page), that the closed loop gain below 1 rad/s is very high. This makes it hard to tell what exactly these bode plots represent. It can be seen that the model and the flight data in Figure C.2 and Figure C.3 (next page) show good matches. Further conclusion cannot be made.

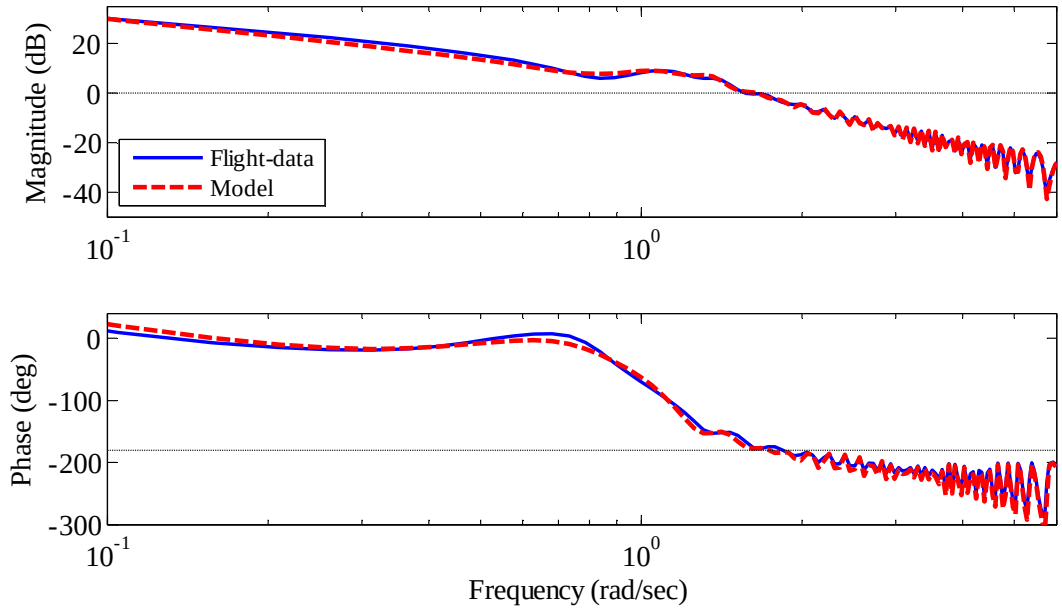



**Figure C.2: Closed loop longitudinal position transfer function of trajectory controller model compared to flight data**

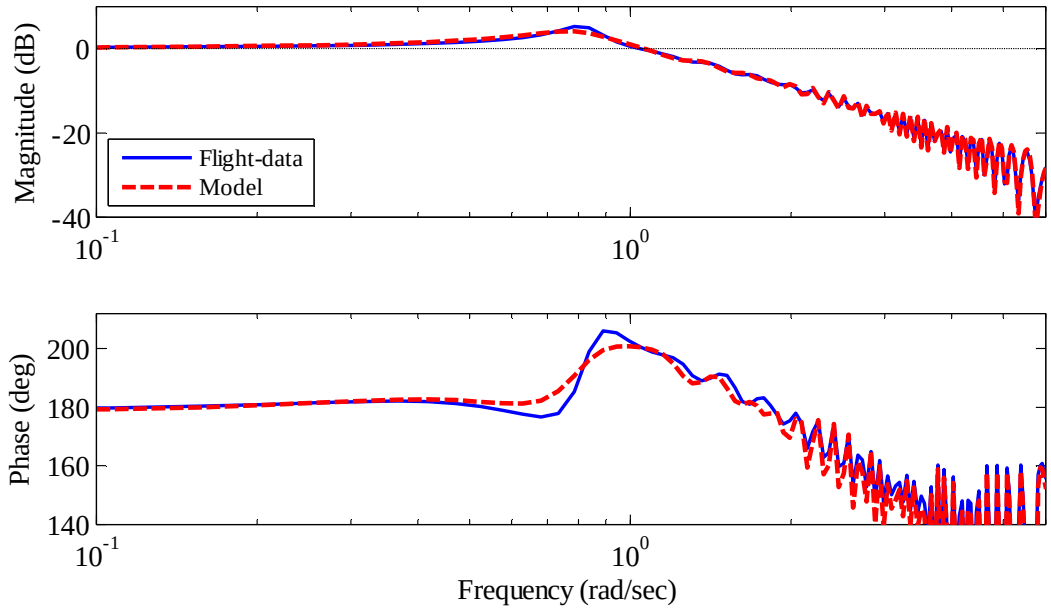



**Figure C.3: Longitudinal loop gain position transfer function of trajectory controller model compared to flight data.**

## C.2.2 Gust rejection

The ability of the controller to reject step disturbances, such as a wind gust is shown in Figure C.4. The gust from a northerly direction is chosen to be greater than helicopter's maximum forward velocity. The helicopter is not able to keep its position, and is forced back by the wind. It is however able to keep stable and return to its original position once the gust has resided. The helicopter reaches wind speeds of 19 m/s.

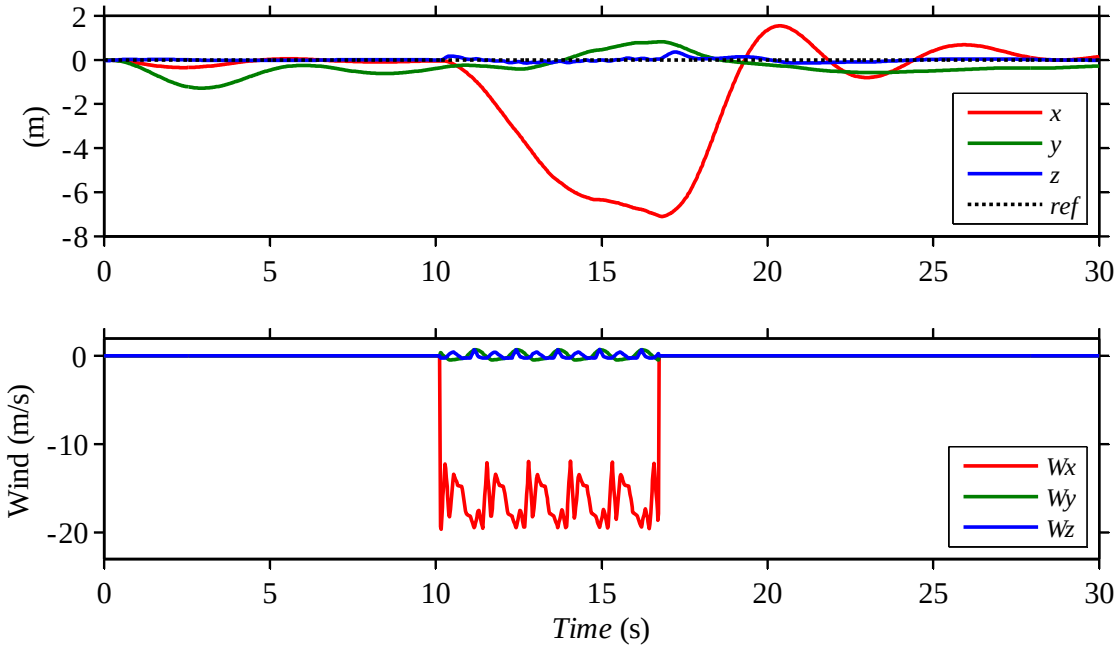



**Figure C.4: Wind gust disturbance rejection in stationary hover.**

With the helicopter in full speed forward flight, there is very little control authority remaining in the collective pitch. The controller's ability to keep the helicopter from being swept away by such a big change in airspeed is a good indication of how robust it is to external disturbances.

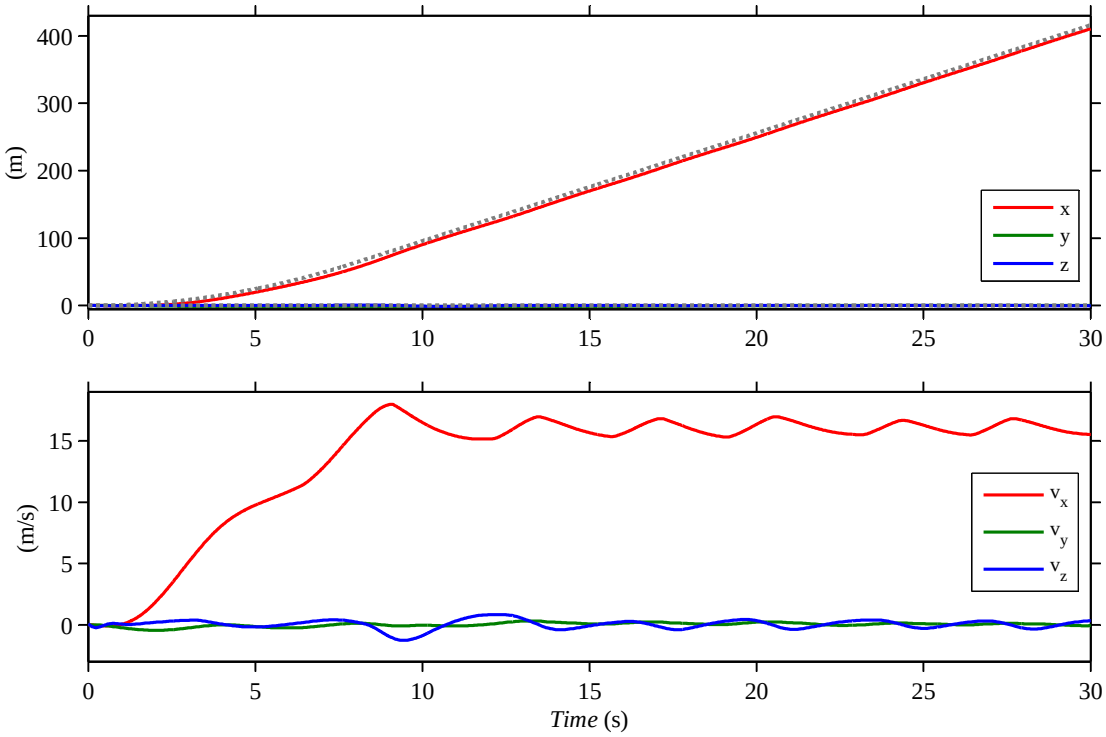



**Figure C.5: Maximum forward flight velocity with position controller. (Dotted grey line: Reference)**

### C.2.3 Maximum air speed

The maximum attainable forward speed in manual flight control is 16 m/s. At this speed, the helicopter acts more like an airplane than a helicopter in hover. Figure C.5 (previous page) shows that the controller can keep the helicopter stable close to its maximum forward speed.

It is important to test the controller's limits. These limits will be needed to define the flight envelope the guidance controller can use to generate trajectories. The velocity has small oscillations. The oscillations can be expected since the velocity is not being controlled, but the position.

### C.2.4 Noise rejection

Noise rejection is evaluated on a figure 8 trajectory. The noise is not filtered, so that the controller can be evaluated and not the filters. Table C.1 shows the noise properties used in the experiment.

**Table C.1:Noise properties used to evaluate trajectory performance**

| State | Mean | Variance |
|---|---|---|
| $u, v, w$ | 0 | 0.7 m/s |
| $p, q, r$ | 0 | 2 deg/s |
| $\phi, \theta$ | 0 | 3 deg |

The noise properties were chosen to be as large as possible with the helicopter still keeping to the trajectory. Dashed lines in Figure C.6 indicate the actual position of the helicopter. Solid lines indicate the estimated position. It is clear that the helicopter is unable to estimate its position correctly. The controller is able to follow the reference trajectory properly, as shown by the solid lines and the black dashed reference line. Estimation, not control, needs to be improved for better results.

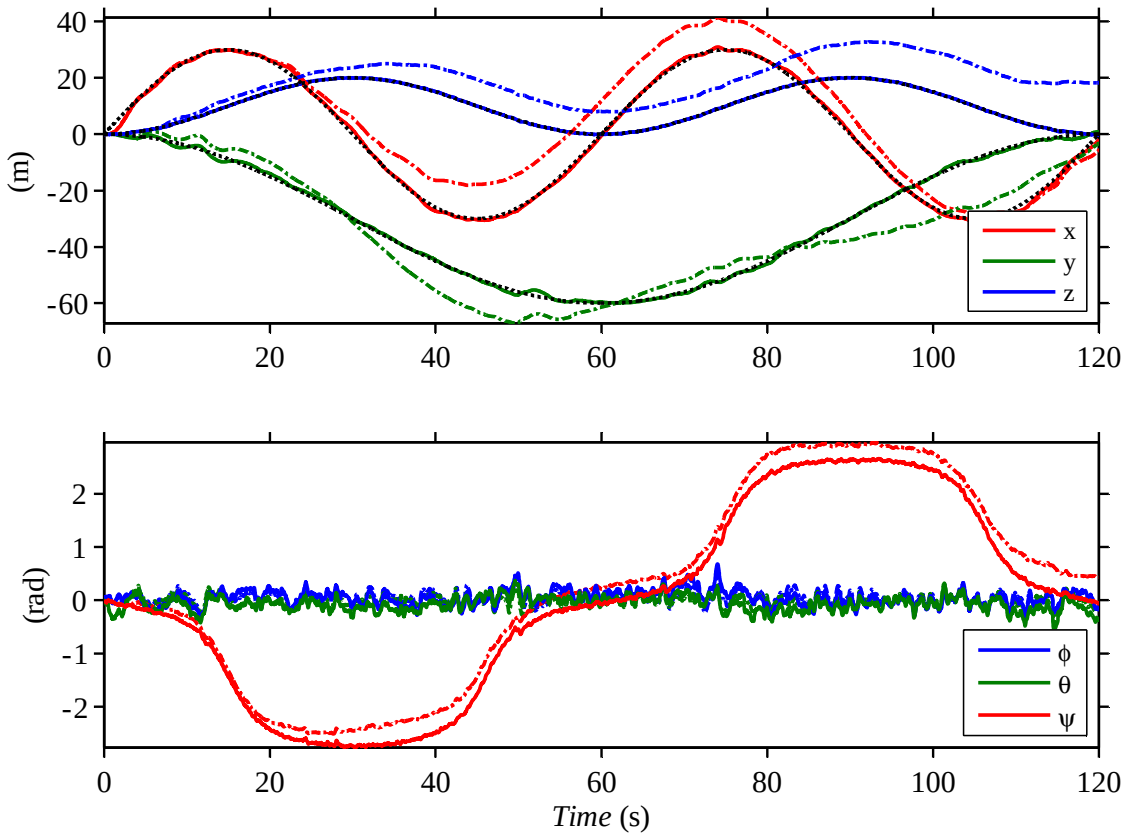



**Figure C.6: Figure 8 trajectory tracking with sensor noise**

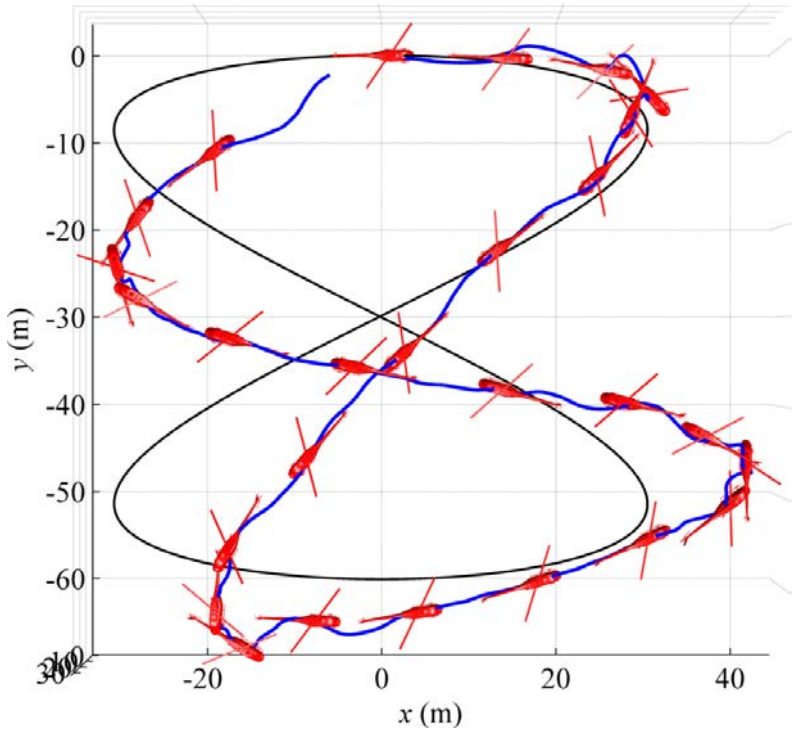



**Figure C.7: Noise rejection 3D view from above (Black: Reference. Blue: Actual position)**

## C.3 AeroSIMRC helicopter

A screenshot of the helicopter flown in AeroSIMRC is shown in Figure C.8.

**Figure C.8: Generic size 30 helicopter in AeroSIMRC**