NTNU – Trondheim
Norwegian University of
Science and Technology

Fault-Tolerant UAV Flight Control System

Kerrin Andre Dybsjord

Master of Science in Engineering Cybernetics
Submission date: June 2013
Supervisor: Thor Inge Fossen, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Faculty of Information Technology,
Mathematics and Electrical Engineering
Department of Engineering Cybernetics

MSC THESIS DESCRIPTION SHEET

Name:	Kerrin Dybsjord
Department:	Engineering Cybernetics
Thesis title (Norwegian):	Feiltolerant styresystem for UAV
Thesis title (English):	Fault-Tolerant UAV Flight Control System

Thesis Description: The purpose of the thesis is to develop and simulate a fault-tolerant flight control system for an unmanned aerial vehicle (UAV) and test the control system in MATLAB Simulink.

The following items should be considered:

1. Literature study on UAV flight control systems with emphasis placed on fault tolerance.
2. Identify and discuss different failures in UAV systems (actuators, sensors, fuselage, etc.)
3. Use an aircraft model to generate flight trajectories such that failure situations can be tested using an arbitrarily flight control system.
4. Choose methods for fault detection and isolation (FDI). Use the FDI to detect faults and implement this in Simulink.
5. Based on the detected failure mode propose reconfiguration mechanisms for feedback control and demonstrate robustness of the system for simulated failure situations. Both single-point failures and multiple failures should be considered.
6. Conclude your results.

Start date:	2013-01-14
Due date:	2013-06-10

Thesis performed at:	Department of Engineering Cybernetics, NTNU
Supervisor:	Professor Thor I. Fossen, Dept. of Eng. Cybernetics, NTNU

Abstract

The main focus of this master's thesis is *fault-tolerant control systems* (FTCSs) for *unmanned aerial vehicles* (UAVs). The goals are to develop an *automatic-flight control system* (AFCS) with *fault detection and isolation* (FDI) and a reconfiguration mechanism for accommodation of faults. The literature study reviews methods for fault-tolerant control and also discusses important faults and failures related to UAVs.

The FTCS is implemented in MATLAB Simulink with a nonlinear model of the Cessna 172SP as a simulation plant. Several case studies test the capabilities of the FTCS using a variety of faults in the absence of noise and disturbances. The AFCS is tested separately in fault-free conditions.

The AFCS comprises of lookahead-based steering for guidance and a *linear-quadratic* (LQ) velocity and rate controller for reference tracking. It was necessary to implement a kinematic controller to reduce stationary errors. This resulted in extended control bandwidth which increased the robustness of the AFCS. The control system gave excellent reference tracking.

Parity space is chosen as the FDI method and variance testing is implemented for residual validation. The results from the FDI showed that there were large separations between simulations with and without faults. Multiple faults during a simulation were also accurately detected.

In a fault situation, fault accommodation is chosen for feedback control. Because of a narrow control bandwidth of the LQ controller, optimal control was not achieved for larger faults. However, the kinematic controller increased the bandwidth substantially; accurate control was achieved within the extended bandwidth. Simulations of larger faults showed that increased errors in the LQ controller lead to rough control signals; this weakens the robustness of the FTCS.

Sammendrag

Hovedfokuset for denne masteroppgaven er *feiltolerante kontrollsystemer* (FTCSs) for *ubemannede fly* (UAV). Målene er å utvikle et automatisk kontrolsystem for fly (AFCS) med *detektering og isolering av feil* (FDI) og en rekonfigureringsmekanisme for kompansering av feil. Litteraturstudiet gjennomgår metoder for feiltolerant kontroll og drøfter også viktige feil relatert til UAVer.

FTCSen er implementert i MATLAB Simulink med en ulineær modell av Cessna 172SP som en simulator. Flere case-studier tester egenskapene til FTCSen med en rekke feil i fravær av støy og forstyrrelser. AFCSen testes separat i feilfrie forhold.

AFCSen består av "lookahead"-basert styring og en *lineær-kvadratisk* (LQ) rate- og hastighetskontroller for referansefølging. Det var nødvendig å implementere en kinematisk kontroller for å redusere stasjonæravvik. Dette resulterte i utvidet kontrollbåndbredde som økte robustheten til AFCSen. Kontrollsystemet ga utmerket referansefølging.

"Parity space" er valgt som FDI-metode og variansetesting er implementert for feildeteksjon. Resultatene fra FDIen viste at det var en stor separasjon mellom simuleringer med og uten feil. På grunn av dette var det også mulig å oppdage flere feil i løpet av en simulering.

"Fault accommodation" er valgt for tilbakekobling i en feilsituasjon. På grunn av smal kontrollbåndbredde på LQ-kontrolleren ble ikke optimal kontroll oppnådd for større feil. Den kinematiske kontrolleren øker båndbredden vesentlig; nøyaktig kontroll ble oppnådd innenfor den utvidede båndbredden. Simuleringer av større feil viste at økt avvik i LQ-kontrolleren fører til grove styresignaler, og dette svekker robustheten til FTCSen.

Preface

This master's thesis is the final part of the Master of Science in Engineering Cybernetics course at the Norwegian University of Science and Technology (NTNU). It was produced during the spring of 2013.

I want to take this opportunity to thank my supervisor, Thor Inge Fossen at the Department of Engineering Cybernetics, for excellent guidance and for being an inspiration in the field of navigation and vessel control systems. I would also like to thank everyone working in room D134, the unmanned vehicle laboratory, for contributing to a very pleasant working environment.

A DVD is attached to this thesis as seen in Appendix C. This includes all MATLAB Simulink diagrams and the necessary scripts to run the simulations. Note that a mex compiler must be installed in order to compile the Simulink program.

Some of the sections in this thesis are expansions of an unpublished project report submitted prior to this thesis. This includes the first part of Section 1.1 and sections from Chapter 4.

MathWorks' MATLAB Simulink R2012b was utilised in this thesis.

Kerrin Dybsjord

Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviations

AFCS	Automatic-Flight-Control System
AFTCS	Active-fault-tolerant control system
ARE	Algebraic Riccati equation
CG	Centre of gravity
CO	Centre of origin
DOF	Degrees of Freedom
ECEF	Earth-centred Earth-fixed
ECI	Earth-centred inertial
FDD	Fault detection and diagnosis
FDI	Fault detection and isolation
FTCS	Fault-tolerant control system
LOS	line of sight
NED	North-east-down
PFTCS	Passiv-fault-tolerant control system
SA	Stability axes
SO	Special Orthogonal group
UAV	Unmanned aerial vehicle

Chapter 1

Introduction

1.1 Motivation

An *Unmanned aerial vehicle* is defined as "An aerial vehicle that does not require a human operator and can fly independently or be operated remotely, (...) and is capable of carrying a lethal or non-lethal payload" (The Oxford Essential Dictionary of the U.S. Military). Though this is a military definition it also applies for the commercial market. Today UAVs are used for terrain mapping, weather monitoring, real estate mapping, law enforcement, crop surveying, etc. The need for aerial monitoring is huge and without UAVs this would be a costly affair. Most of these tasks require a degree of automatic control so the training of the operator is minimal.

The advantage of using UAVs is mostly cost related. There is no need for pilots and the duration of operation depends only on the fuel. A UAV is also able to travel places where pilots cannot or where there is high risk involved, for instance disaster areas after nuclear blasts, areas with unstable weather, volcanoes, etc. Also tedious aerial surveillance can be made fully automatic so that fewer operators are needed.

Oil extraction in Arctic regions has in the recent years become more probable because of the development of unmanned vehicles. Drift ice here will pose a threat for surface vehicles, as it moves freely dependant on wind and currents. Constant aerial surveillance and mapping of the ice is crucial under these circumstances and this can be provided by UAVs.

A UAV with a payload can be an expensive piece of equipment and therefore high demands are set on the *automatic flight control system* (AFCS). Also in harsh environments such as the Arctic, failure situations can arise, for instance icing of a control surface. In these cases the control system should be able to detect and counteract these failures preventing the aircraft from crashing. Taking failure situations in to consideration when designing the AFCS will make it fault tolerant.

1.2 Fault Tolerant Control Systems (FTCS)

There is an increasing demand for safety and performance requirements in modern control systems. Faults occur and conventional control systems may not be able to adapt, resulting in reduced performance or even instabilities. Safety-critical applications such as aircraft, spacecraft, nuclear power plants, chemical plants, etc. must have capabilities beyond conventional control systems where the cost and consequences of a malfunction is too great. New control systems must be capable of maintaining stable control in the event of faults in the system.

Early forms of fault-tolerant control takes shape as monitoring systems, according to [Isermann, 2006]. These would observe the outputs of the system and operate alarms. Typical applications were chemical plants, factories, etc. For aircraft systems fault tolerance has always been in the form of hardware redundancy. This resolves faults by adding additional flight controls, sensors, actuators, etc., which could substitute the broken components during flight. For instance, Boeing 777 has quadruplex actuator redundancy, i.e. four actuators for the same control surface. Additionally, along with the introduction of fly-by-wire systems, analytical redundancy (software redundancy) was introduced in the late 1970s [Zhang and Jiang, 2008]. As flight computers became more advanced the possibility of more complicated control systems made it possible to reduce hardware redundancy.

Low weight is crucial for UAVs, the bigger the UAV the more expensive it is to operate. Therefore, hardware redundancy is very limited and the development of reliable analytical redundancy is important. These computer systems must be computationally light because of limitations on CPUs and onboard power supply.

According to [Zhang and Jiang, 2008] "a closed-loop control system which can tolerate component malfunctions, while maintaining desirable performance and stability properties is said to be a *fault-tolerant control system*" (FTCS). The FTCS can be divided in two main parts:

The first part of the FTCS is detecting that a fault has occurred and determining its origin. Such systems have many names but a list of definitions are given by [Isermann, 2006]:

Fault detection: Determination of faults present in a system and time of detection;

Fault isolation: Determination of kind, location and time of detection of a fault by evaluating symptoms. Follows fault detection;

Fault identification: Determination of the size and time-variant behaviour of a fault. Follows fault isolation;

Fault diagnosis: Determination of kind, size, location and time of detection of a fault by evaluating symptoms. Follows fault detection.

A combination of these terms form the terminology commonly used in this field. *Fault detection and diagnosis* (FDD) is the most comprehensive system and is commonly referred to as fault detection and isolation with identification [Zhang and Jiang, 2008]. *Fault detection and isolation* (FDI) is another design. According to [Isermann, 2006] this does not identify the size of the fault. Due to the field's lack of consistent terminology, these terms are often used interchangeably in the literature.

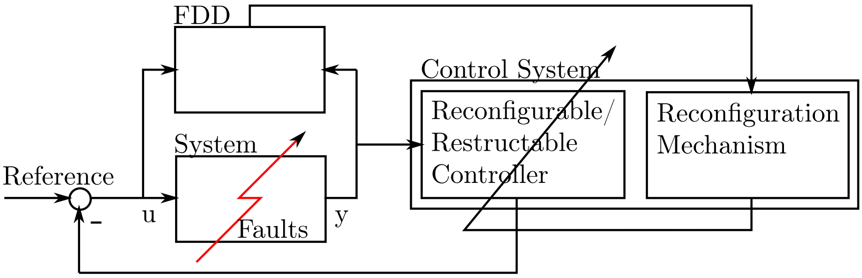


Figure 1.1: An active-fault-tolerant control scheme. u is control inputs and y is system measurements. The figure is adapted from Fig 1 in [Zhang and Jiang, 2008]

The second part of the FTCS is the control system. The reconfiguration mechanism has to reconfigure the parameters of the controller, i.e. retuning the controller. Such systems are called *reconfigurable* control systems according to [Zhang and Jiang, 2008]. The same principle is also covered by [Blanke et al., 2006]; however this is introduced as *fault accommodation* which will be the terminology used in this thesis. In some cases the reconfiguration mechanism has to alter the structure of the controller. This could imply rearranging the order of the controllers or introduce a new controller. This approach, according to [Zhang and Jiang, 2008], is called *restructable* control systems and in [Blanke et al., 2006] referred to as *control reconfiguration*. The problem of *control allocation* also falls into this part of the FTCS. This is the problem of finding the optimal arrangement of actuators to accomplish the desired behaviour.

There are two main classifications of FTCS: *passive* (PFTCS) and *active* (AFTCS). A PFTCS is designed with a selection of faults in mind. Thus a reconfiguration mechanism is not required since predesigned controllers take over when faults are detected. Also FDD techniques may not be necessary. AFTCSs use fault detection and fault accommodation/reconfiguration methods described above to achieve fault tolerance. They are also referred to as reconfigurable, restructable, self-repairing or self-designing control systems and have the capabilities of altering or redesigning a controller.

A simplified figure of a typical *active*-FTCS can be seen in Figure 1.1. From the figure it is seen that the inputs and outputs (I/O) of the system are considered in the FDD, thus a comparison is performed. The result is a decision sent to the reconfiguration-mechanism block which in turn alters the existing control system.

1.3 Contribution of this Thesis

In this thesis several parts related to fault-tolerance will be explored, developed and implemented through simulations:

- Development of an automatic flight control system (AFCS).
- Development of a linear fault detection and isolation (FDI) system using the parity space method.
- Development of fault accommodation for model-based control to counteract occurring faults.
- Implement the fault-tolerant control system on the fixed-wing aircraft Cessna 172SP by using MATLAB Simulink as a simulation platform. The simulation plant used is from [Vistnes, 2012], Section 1.6.

1.4 Outline of the Thesis

This thesis is divided in 9 chapters presented below. Figure 1.2 describes the structure of the system and has references to theory and the implementation of each block.

Chapter 2 Literature Study: In this chapter a literature review will be performed on FTCSs, including types of faults and methods used for detection and isolation. In the last section the methods are discussed and the FTCS in this thesis is defined.

Chapter 3 Aircraft Modelling: Necessary background theory about aircraft modelling is presented in this chapter.

Chapter 4 Control Theory: This chapter presents relevant control theory which will be used later in the thesis.

Chapter 5 Implementation: The theory described in the preceding chapters is implemented, constructing a simulation platform for the FTCS using MATLAB Simulink.

Chapter 6 Results and Discussion on the AFCS: The results from the AFCS are presented and discussed separately assuming a fault-free system.

Chapter 7 Case Study: The FTCS is tested through 7 case studies. Different parts of the FTCS are tested and a variety of faults are used.

Chapter 8 Discussion: A discussion is performed based on the 7 case studies. General topics are discussed at the end.

Chapter 9 Conclusion and Future work: Conclusions are drawn based on the overall results. Also, future work which requires investigation is outlined in order to further develop the FTCS.

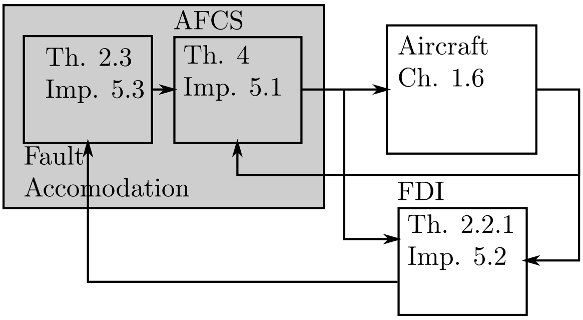


Figure 1.2: This is an overview of the fault-tolerant control system (FTCS) with references to the theory (Th.) and implementation (Imp.) of these systems. The abbreviations are: automatic flight control system (AFCS), fault detection and isolation (FDI).

1.5 Assumptions

Assumptions will be made throughout the thesis; however the most important ones are presented here:

- Euler angles will be used in the development of the aircraft model. It is assumed that $\theta \neq \frac{\pi}{2}$ as this will result in a singular Jacobian matrix as shall be discussed in Section A.3. The aircraft used in this thesis is the Cessna 172SP and is not capable of advanced flight manoeuvres resulting in $\theta = \frac{\pi}{2}$.
- A decoupled aircraft model will be developed in this thesis. It is assumed that the longitudinal channel and the lateral channel do not effect one another.
- Fault identification will not be apart of this thesis. It is therefore assumed that the fault is estimated correctly.

1.6 Aircraft Model

The aircraft model used in this thesis is based on the civil utility aircraft, Cessna 172SP, from [Vistnes, 2012]. The nonlinear model [Vistnes, 2012] is developed with least-squares method for system identification. [Vistnes, 2012] uses both online and offline methods and the raw data is obtained from the X-Plane flight simulator.

Chapter 2

Literature Study

2.1 Faults and Failures

The terms fault and failure are used to describe different degrees of system degradation. According to [Isermann, 2006], in cooperation with IFAC SAFEPROCESS Technical Committee, a fault is defined as "an unpermitted deviation of at least one characteristic property or parameter of the system from the acceptable/usual/-standard condition". A failure is defined as "a permanent interruption of a system's ability to perform a required function under specified operating conditions". Thus, a failure is more severe than a fault. However, these terms are dependent on relativity. For instance, an actuator failure occurs on one of the ailerons leaving this control surface unusable. Relative to the overall control, the failure can be interpreted as a fault. This is because by using the rest of the control surfaces differently the aircraft is still controllable.

In aircraft (or vehicles in general) faults can enter the system in three ways: as actuator faults, system faults or sensor faults. This is illustrated in Figure 2.1. As the plant is an input/output model ($\boldsymbol{y}/\boldsymbol{u}$) the dynamics of each block in Figure 2.1 is viewed as one output. However, it is common to separate the sensors from the rest of the plant and as a result faulty sensors can be solved independently with the use of sensor redundancy. This concept is introduced in [Sørensen, 2011]. [Alwi et al., 2011] lists up some common sensor faults that has to be accounted for in fault-tolerant sensor systems. These are:

- **Bias:** A bias is a constant error on the sensor output.
- **Freezing:** When a sensor malfunctions it can sometimes freeze at an arbitrary value.
- **Drift:** Sensor drift is a steady increase in deviation of the sensor output.
- **Loss of accuracy:** The sensor loses accuracy.
- **Calibration error:** A loss of physical meaning of the sensor readout.

Actuator and system faults, on the other hand, cannot be separated as it is impossible to measure $\boldsymbol{\tau}$ directly. Examples of system faults is structural damage to the vehicle. For aircraft this could be a change in the aerodynamics or a change in the centre of gravity.

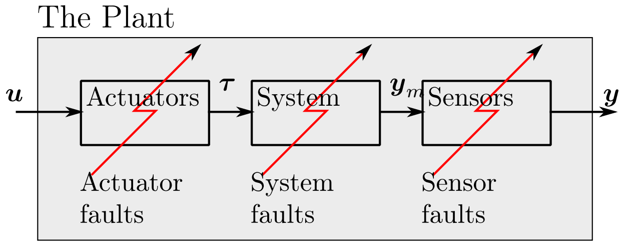


Figure 2.1: Types of faults that influence the vehicle. $\boldsymbol{u}$ is the control command which commands the actuators; $\boldsymbol{\tau}$ is the force generated by the actuators that cause motion in the system; $\boldsymbol{y}_m$ is the measurements received by the sensors and $\boldsymbol{y}$ is the output of the system calculated by the sensor model.

2.1.1 Actuator Faults

According to [Alwi et al., 2011] actuator faults are divided in four categories as seen in Figure 2.2.

(a) Actuator lock is a failure caused by a mechanical jam such that the actuator is locked in place.

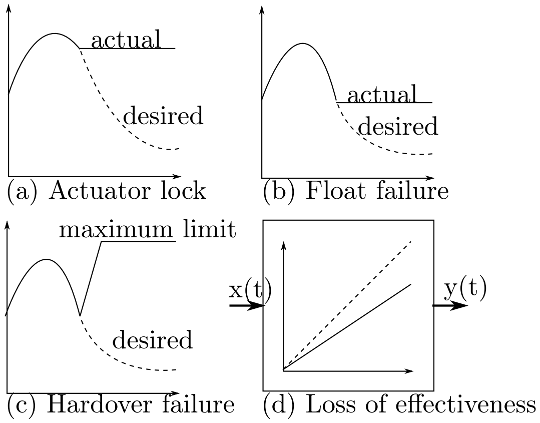
(b) A float failure occurs when the actuator loses total hydraulic or pneumatic pressure. The result is a freely-moving control surface that does not contribute to lift.

(c) A hardover actuator failure is when the actuator moves to maximum deflection due to electrical or software faults.

(d) Loss of effectiveness is an actuator fault which is a result of a partial loss of hydraulic or pneumatic pressure. This fault is modelled as a proportional loss of effect as seen in (2.1.1).

$$u_m^i = (1 - \gamma^i)u_c^i \tag{2.1.1}$$

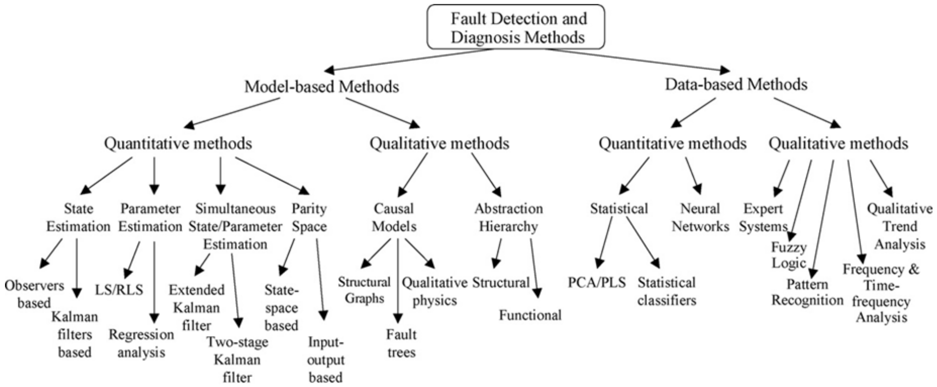
where i denotes the actuator number, u_m is the actual control output when the control surface is faulty, u_c is the calculated control output for a fault-free actuator and γ effect-reduction factor.



(a) Actuator lock (b) Float failure

(c) Hardover failure (d) Loss of effectiveness

Figure 2.2: Types of actuator faults [Alwi et al., 2011]



Note: LS/RLS: Least Squares/Recursive Least Squares; PCA: Principal Component Analysis; PLS: Partial Least Squares.

Figure 2.3: overview of FDD methods [Zhang and Jiang, 2008]

2.2 Fault Detection and Diagnosis

As mentioned preliminary FDD is the most comprehensive system. It includes fault identification which determines the size of the fault, [Isermann, 2006]. Fault identification can be viewed as a different part of the system, separating the fault identification from the FDI.

First and foremost the FDD methods can be divided in two main parts: Model-based and data-based methods. Figure 2.3 gives an overview of methods that can be implemented in their respective categories. The data-based (or model-free) methods bases the decision on presence of faults on data pools. This thesis will focus on quantitative-model-based methods.

The following terminology is defined in [Isermann, 2006]:

Quantitative models: *Use of static and dynamic relation among system variables and parameters in order to describe system's behaviour in quantitative mathematical terms.*

Residual: *a fault indicator based on a deviation between measurements and model-equation based computations.*

Figure 2.4 shows a structure of a typical FDI scheme. The problem of FDI is twofold: *Residual generation* and *residual validation*. A residual is generated by comparing elements from the system with mathematical models. Methods commonly used in FDD/FDI to construct mathematical models are parameter estimation, state estimation, parity space or a combination of them. The first two methods mentioned are often used to construct a post-fault model by estimating the behaviour of the system. In these cases the fault identification can be included in the FDI resulting in a FDD system. Parity space on the other hand compares a fault-free mathematical model with the faulty system.

Residual validation analyses the residual and determines whether a fault has occurred or not. Methods used for this task varies. A generalized list is given in [Zhang and Jiang, 2008]:

- Threshold test on instantaneous or moving average values of the residuals.
- Statistical methods.
- Methods based on fuzzy logic symptom evaluation.
- Methods based on neural network pattern classification.

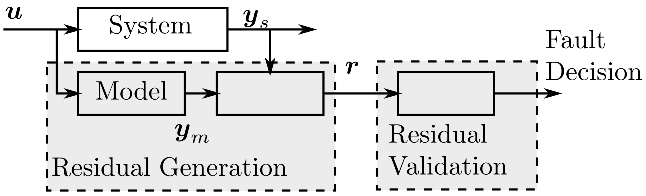


Figure 2.4: A model-based FDI structure divided in two stages residual generation and residual validation. u is the controller input, y_s is the system output, y_m is the model output and r is the residual.

2.2.1 The Parity Space Approach

Parity space is defined in [Patton and Chen, 1994] "as a space in which all elements are residuals (or parity vectors). Residuals and parity vectors are synonyms in this context". The idea of parity space approaches is to compare the outputs of the system with the output of a fault-free model, by generating residuals. In [Patton and Chen, 1994] this is elaborated by using a discrete-linear-state-space model of the monitored system at time k:

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{R}_1\boldsymbol{f}(k) \tag{2.2.1}$$

$$\boldsymbol{y}(k) = \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{u}(k) + \boldsymbol{R}_2\boldsymbol{f}(k) \tag{2.2.2}$$

where $\boldsymbol{x}(k) \in \mathcal{R}^n$ is the state vector, $\boldsymbol{y}(k) \in \mathbb{R}^m$ is the output of the system, $\boldsymbol{u}(k) \in \mathbb{R}^r$ is the control input and $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B} \in \mathbb{R}^{n\times r}$, $\boldsymbol{C} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{D} \in \mathbb{R}^{m\times r}$ are known system matrices. $\boldsymbol{f}(k) \in \mathbb{R}^q$ is a fault vector where each element $\boldsymbol{f}_i(k)$ $(i = 1, 2, \cdots, q)$ corresponds to a specific fault. These are unknown time functions. $\boldsymbol{R}_1 \in \mathbb{R}^{n\times q}$ and $\boldsymbol{R}_2 \in \mathbb{R}^{m\times q}$ are fault entry matrices and represents the effect of the faults on the system.

The input-output description of the system can be developed by inserting the Laplace transformation of (2.2.1) into (2.2.2):

$$(z\boldsymbol{I} - \boldsymbol{A})\boldsymbol{x}(z) = \boldsymbol{B}\boldsymbol{u}(z) + \boldsymbol{R}_1\boldsymbol{f}(z)$$

$$\boldsymbol{x}(z) = (z\boldsymbol{I} - \boldsymbol{A})^{-1}\left[\boldsymbol{B}\boldsymbol{u}(z) + \boldsymbol{R}_1\boldsymbol{f}(z)\right] \tag{2.2.3}$$

and

$$\boldsymbol{y}(z) = \boldsymbol{C}\boldsymbol{x}(z) + \boldsymbol{D}\boldsymbol{u}(z) + \boldsymbol{R}_2\boldsymbol{f}(z) \tag{2.2.4}$$

Inserting $\boldsymbol{x}(z)$ gives:

$$\boldsymbol{y}(z) = \boldsymbol{C}\left[(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{B}\boldsymbol{u}(z)\right] + \boldsymbol{C}\left[(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{R}_1\boldsymbol{f}(z)\right] + \boldsymbol{D}\boldsymbol{u}(z) + \boldsymbol{R}_2\boldsymbol{f}(z) \tag{2.2.5}$$

$$\boldsymbol{y}(z) = \left[\boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{B} + \boldsymbol{D}\right]\boldsymbol{u}(z) + \left[\boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{R}_1 + \boldsymbol{R}_2\right]\boldsymbol{f}(z) \tag{2.2.6}$$

(2.2.6) is written more compactly by the definitions (2.2.8) and (2.2.9):

$$\boldsymbol{y}(z) = \boldsymbol{G}_u \boldsymbol{u}(z) + \boldsymbol{G}_f \boldsymbol{f}(z) \tag{2.2.7}$$

$$\boldsymbol{G}_u \triangleq \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{B} + \boldsymbol{D} \tag{2.2.8}$$

$$\boldsymbol{G}_f \triangleq \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{R}_1 + \boldsymbol{R}_2 \tag{2.2.9}$$

The Parity Space

The parity space or residual $\boldsymbol{r}(z)$ is realized by defining:

$$\boldsymbol{r}(z) \triangleq \boldsymbol{H}_u(z)\boldsymbol{u}(z) + \boldsymbol{H}_y(z)\boldsymbol{y}(z) \tag{2.2.10}$$

Figure 2.5 shows a block diagram of this set-up. For a fault-free system the residual should be zero. To achieve this it is assumed that the model catches the dynamics of the system perfectly. This is called the certainty equivalence principle.

$$\boldsymbol{r}(z) = 0 \qquad \text{and} \qquad \boldsymbol{y}(z) = \boldsymbol{G}_u(x)\boldsymbol{u}(z) \tag{2.2.11}$$

Applying (2.2.11) to (2.2.10) gives:

$$\boldsymbol{H}_u(z) = -\boldsymbol{H}_y(z)\boldsymbol{G}_u(z) \tag{2.2.12}$$

The residual generator is designed by choosing the appropriate transfer matrices $\boldsymbol{H}_u(z)$ and $\boldsymbol{H}_y(z)$. These have to satisfy (2.2.12). These matrices will typically be a selection matrix, containing ones and zeros, which selects the desired states.

Further, by inserting (2.2.7) into (2.2.11) the fault dynamics can be modelled:

$$\boldsymbol{r}(z) = \boldsymbol{H}_u(z)\boldsymbol{u}(z) + \boldsymbol{H}_y(z)\left[\boldsymbol{G}_u\boldsymbol{u}(z) + \boldsymbol{G}_f\boldsymbol{f}(z)\right] \tag{2.2.13}$$

$$= \left[\boldsymbol{H}_u(z) + \boldsymbol{H}_y(z)\boldsymbol{G}_u\right]\boldsymbol{u}(z) + \boldsymbol{H}_y(z)\boldsymbol{G}_f\boldsymbol{f}(z) \tag{2.2.14}$$

By the assumption of certainty equivalence, (2.2.12) is inserted for $\boldsymbol{H}_u(z)$:

$$\boldsymbol{r}(z) = \left[-\boldsymbol{H}_y(z)\boldsymbol{G}_u(z) + \boldsymbol{H}_y(z)\boldsymbol{G}_u\right]\boldsymbol{u}(z) + \boldsymbol{H}_y(z)\boldsymbol{G}_f\boldsymbol{f}(z) \tag{2.2.15}$$

$$\boldsymbol{r}(z) = \boldsymbol{H}_y(z)\boldsymbol{G}_f\boldsymbol{f}(z) \tag{2.2.16}$$

The fault dynamics are then described by inserting $\boldsymbol{G}_f$:

$$\boldsymbol{r}(z) = \boldsymbol{H}_y(z)\left[\boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{R}_1 + \boldsymbol{R}_2\right]\boldsymbol{f}(z) \tag{2.2.17}$$

According to [Frank, 1990] the residual $\boldsymbol{r}$ can be used as the residual directly, however in [Blanke et al., 2006] it is proposed that $\boldsymbol{r}$ is filtered such that:

$$\boldsymbol{\epsilon} = \boldsymbol{H}_{rf}\boldsymbol{r}(z) \tag{2.2.18}$$

where $\boldsymbol{H}_{rf}$ is the residual filter.

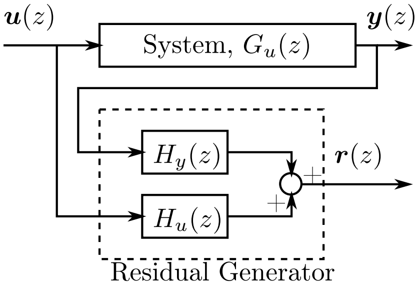


Figure 2.5: The transfer function structure of a residual generator [Patton and Chen, 1994], where $\boldsymbol{H}_y(z)$ and $\boldsymbol{H}_u(z)$ are design matrices, $\boldsymbol{H}_u(z) = -\boldsymbol{H}_y(z)\boldsymbol{G}_u(z)$ and $\boldsymbol{G}_u$ is the transfer function describing the input-output model of the fault-free system, (2.2.8)

2.2.2 Statistical Residual Validation Methods

Windowing

Windowing is a method used in signal processing for looking at a selected part of a signal. Windowing can be used to create a set of data which the analysis can be based upon. Defining the window signal $w(k)$, a rectangular window is given by:

$$w(k) \triangleq \begin{cases} 1 & M_1 \leq k \leq M_2 \\ 0 & \text{otherwise} \end{cases} \tag{2.2.19}$$

where M_1 and M_2 can be time or sample variables which defines the size of the window. Data from a signal $x(k)$ is limited by windowing:

$$y(k) = x(k)w(k) \tag{2.2.20}$$

The result is the limited signal $y(k)$:

$$y(k) = \begin{cases} x(k) & M_1 \leq k \leq M_2 \\ 0 & \text{otherwise} \end{cases} \tag{2.2.21}$$

Variance Testing

The variance of a signal refers to how spread the signal is from the expected value. A signal $x(k)$ is evaluated at time $t = k$. Using a moving window of size n, that is $n-1$ historical values, $x(i)$ is defined where $i = \{k-(n-1), \cdots, k-1, k\}$. The definition of mean and variance is given in [Sørensen, 2011]:

$$\bar{x}_k = \frac{1}{n} \sum_{i=k-(n-1)}^{k} x(i) \tag{2.2.22}$$

$$\sigma_k^2 = \frac{n}{n-1} \left(\sum_{i=k-(n-1)}^{k} x(i)^2 - \bar{x}_k^2 \right) \tag{2.2.23}$$

Derivations of the recursive version of (2.2.22) and (2.2.23) are given below. The idea is that the current mean and variance is to be updated as the moving window moves one step. This means adding a term at $t = k$ and subtracting the term at $t = k - (n - 1)$. At time $t = k$, $\bar{x}_{k-1}$ is known form the previous recursion and can be extracted from (2.2.22):

$$\bar{x}_k = \frac{1}{n} \sum_{i=k-(n-1)}^{k-1} x(i) + \frac{1}{n} x(k) \tag{2.2.24}$$

$$\tag{2.2.25}$$

The first element of x_{k-1} (at $t = k - 1 - (n - 1)$), is missing from the summation and is therefore added and subtracted in the following equation:

$$\bar{x}_k = \frac{1}{n} \sum_{i=k-1-(n-1)}^{k-1} x(i) + \frac{1}{n} x(k) - \frac{1}{n} x(k-1-(n-1)) \tag{2.2.26}$$

$$= \bar{x}_{k-1} + \frac{1}{n} \left[x(k) - x(k-1-(n-1)) \right] \tag{2.2.27}$$

The equation for variation is derived by defining a function $y(k)$:

$$y(k) \triangleq \frac{1}{n} \sum_{i=k+1-(n-1)}^{k} x(i)^2 \tag{2.2.28}$$

The recursive version is derived similar to (2.2.27):

$$y(k) = y(k-1) + \frac{1}{n} \left[x(k)^2 - x(k-1-(n-1))^2 \right] \tag{2.2.29}$$

The variance is then given by:

$$\sigma_k^2 = \frac{n}{n-1} \left[y(k) - \bar{x}_k^2 \right] \tag{2.2.30}$$

2.3 Fault Accommodation for Model-Based Control

This section is based on [Blanke et al., 2006]. Linear quadratic optimal control is derived in Section 4.2 and relies on a linear model. It produces a control law based on the model given by (4.2.1). When the system is fault-free this system can be written as:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \sum_{i \in I} \boldsymbol{B}_i \boldsymbol{u}_i \tag{2.3.1}$$

where I is the set of all actuators and i is the actuator number. When a faults occur the system dynamics change. I_N and I_F are subsets of I where I_N are the normal-functioning actuators and I_F are the faulty ones. The faulty system becomes:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \sum_{i \in I_N} \boldsymbol{B}_i \boldsymbol{u}_i + \sum_{i \in I_F} \boldsymbol{\beta}_i(\boldsymbol{u}_i, \theta_i) \tag{2.3.2}$$

where $\sum\limits_{i \in I_N} \boldsymbol{B}_i \boldsymbol{u}_i$ is the contribution of the healthy actuators and $\sum\limits_{i \in I_F} \boldsymbol{\beta}_i(\boldsymbol{u}_i, \theta_i)$ is the contribution of the faulty actuators.

When a Fault is introduced into the system the fault has to be identified. This falls under the field of fault identification. As mentioned in Section 2.2 this can be done by state or parameter estimation methods. The post-fault model where faults are estimated can be written as:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \sum_{i \in I_N} \boldsymbol{B}_i \boldsymbol{u}_i + \sum_{i \in I_F} \hat{\boldsymbol{\beta}}_i(\boldsymbol{u}_i, \theta_i) \tag{2.3.3}$$

When a linear model is utilized this can be rewritten as:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \sum_{i \in I_N} \boldsymbol{B}_i \boldsymbol{u}_i + \sum_{j \in I_F} \hat{\boldsymbol{B}}_j \boldsymbol{u}_j \tag{2.3.4}$$

Finding the optimal control law of (2.3.4) using LQ methods is presented in Section 4.2.

2.4 Discussion and Definition of Thesis

This thesis will concentrate on actuator faults. Of those introduced in Section 2.1.1 loss of effectiveness will be implemented and be the main focus. More specifically the faults considered will be on the ailerons and elevator. These control the roll and pitch motions as seen in Section 3.5. These are chosen because they are the main stabilizing control surfaces and therefore will give the most interesting results.

As aircraft have fast dynamics the FDI chosen should have a quick reaction time to faults. Also, power supply on an aircraft or drone is limited and the method should also be computationally light to support this. Based on this knowledge a linear parity space is chosen for the FDI [Patton and Chen, 1994]. A perk with using this method is that "parity space does not require knowledge of fault behaviour" [Patton and Chen, 1994]. This implies that the parity space could detect faults that were not accounted for when the FDI was designed. For validation of the residual statistical methods are chosen, more specifically variance testing.

Fault identification will not be implemented in this thesis, which is the determination of the size of the fault. However to be able to apply fault tolerant control, it is assumed that the fault is estimated and a correct value is found. A linear model is derived to be used for the FDI. The same model is used to apply a linear quadratic (LQ) controller for the AFCS. This opens the possibility of using fault accommodation for LQ presented by [Blanke et al., 2006].

Chapter 3

Aircraft Modelling

The dynamics of an aircraft can be divided in two main parts: The kinematics and the kinetics.

The kinematics describe the motion and the geometry of a system without regarding forces and moments. To be able to describe the relations between independently moving objects, reference frames are defined. Appendix A will elaborate on this as well as define vector and matrix notations which will be used throughout this thesis. Also rotation matrices for linear and angular rotation is defined.

The kinetics describe the forces and moments that act on the vehicle. In this chapter the equations of motion (3.1.2) is presented and applied for aircraft. Also a linearisation of the nonlinear aircraft model which was presented in Section 1.6 is derived.

3.1 Equations of Motion

The 6 degrees-of-freedom (DOFs) equation of motion is expressed by [Fossen, 2011a] as:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}_{\Theta}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{3.1.1}$$

$$\boldsymbol{M}_{RB}\dot{\boldsymbol{\nu}} + \boldsymbol{N}_{RB}\boldsymbol{\nu} + \boldsymbol{G}\boldsymbol{\eta} = \boldsymbol{\tau} \tag{3.1.2}$$

where 3.1.1 is the kinematic equation elaborated in the Appendix A. $\boldsymbol{M}_{RB}$ is the rigid-body mass matrix, $\boldsymbol{N}_{RB}$ is the rigid-body Coriolis and centripetal matrix and $\boldsymbol{G}$ is the matrix of gravitational forces. $\boldsymbol{\tau}$ is the aerodynamic forces and moments and also contain contributions of the control surfaces. The derivation of (3.1.2) using the Euler-Newton formulation can be found in [Fossen, 2011a].

The equations of motion for an aircraft is derived in [Fossen, 2011b] and is given on component form as:

$$\begin{aligned} m(\dot{U} + QW - RV + g\sin(\Theta)) &= X \\ m(\dot{V} + UR - WP - g\cos(\Theta)\sin(\Phi)) &= Y \\ m(\dot{W} + VP - QU - g\cos(\Theta)\cos(\Phi)) &= Z \\ I_x\dot{P} - I_{xz}(\dot{R} + PQ) + (I_z - I_y)QR &= L \\ I_y\dot{Q} + I_{xz}(P^2 - R^2) + (I_x - I_z)PR &= M \\ I_z\dot{R} - I_{xz}\dot{P} + (I_y - I_x)PQ + I_{xz}QR &= N \end{aligned} \tag{3.1.3}$$

3.1.1 Inertia Matrix

From [Fossen, 2011a] the inertia matrix can be expressed, around the centre of gravity as:

$$I_{CG} \triangleq \begin{bmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{yx} & I_y & -I_{yz} \\ -I_{zx} & -I_{zy} & I_y \end{bmatrix} \tag{3.1.4}$$

For aircraft it is realistic to assume xz-plane symmetry ($I_{xy} = I_{yz} = 0$), [Fossen, 2011b]. This reduces the inertia matrix to:

$$I_{CG} \triangleq \begin{bmatrix} I_x & 0 & -I_{xz} \\ 0 & I_y & 0 \\ -I_{xz} & 0 & I_y \end{bmatrix} \tag{3.1.5}$$

3.2 Linear State-Space Model Using Perturbation Theory

Perturbation theory is a method used to linearise non-linear equations of motion. Each state is divided in a normal value, which are called the *nominal* value and a *perturbation* (a deviation) from the nominal value. The states are defined as:

$$\begin{bmatrix} \Phi \\ \Theta \\ \Psi \end{bmatrix} \triangleq \begin{bmatrix} \Phi_0 \\ \Theta_0 \\ \Psi_0 \end{bmatrix} + \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}, \quad \boldsymbol{\nu} \triangleq \boldsymbol{\nu}_0 + \delta\boldsymbol{\nu} = \begin{bmatrix} U_0 \\ V_0 \\ W_0 \\ P_0 \\ Q_0 \\ R_0 \end{bmatrix} + \begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix} \tag{3.2.1}$$

$$\boldsymbol{\tau} \triangleq \boldsymbol{\tau}_0 + \delta\boldsymbol{\tau} = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ L_0 \\ M_0 \\ N_0 \end{bmatrix} + \begin{bmatrix} \delta X \\ \delta Y \\ \delta Z \\ \delta L \\ \delta M \\ \delta N \end{bmatrix} \tag{3.2.2}$$

where the left side of the equation is defined in Appendix A.2. The linearised state-space model is derived by inserting (3.2.1) and (3.2.2) into (3.1.3) and separating the equilibrium conditions (3.2.3) from the perturbation equations (3.2.4). In addition it is assumed that ϕ, θ and ψ are small such that $\sin(\cdot) \approx \cdot$ and $\cos(\cdot) \approx 1$. The derivation is given in [Fossen, 2011b] and the result is:

Equilibrium Conditions:

$$\begin{aligned}
m(\dot{U}_0 + Q_0W_0 - R_0V_0 + g\sin(\Theta_0)) &= X_0 \\
m(\dot{V}_0 + U_0R_0 - W_0P_0 - g\cos(\Theta_0)\sin(\Phi_0)) &= Y_0 \\
m(\dot{W}_0 + V_0P_0 - Q_0U_0 - g\cos(\Theta_0)\cos(\Phi_0)) &= Z_0 \\
I_x\dot{P}_0 - I_{xz}(\dot{R}_0 + P_0Q_0) + (I_z - I_y)Q_0R_0 &= L_0 \\
I_y\dot{Q}_0 + I_{xz}(P_0^2 - R_0^2) + (I_x - I_z)P_0R_0 &= M_0 \\
I_z\dot{R}_0 - I_{xz}\dot{P}_0 + (I_y - I_x)P_0Q_0 + I_{xz}Q_0R_0 &= N_0
\end{aligned} \tag{3.2.3}$$

Perturbation Equations:

$$\begin{aligned}
m\left[\dot{u} + Q_0w + W_0q - R_0v - V_0r + g\cos(\Theta_0)\theta\right] &= \delta X \\
m\left[\dot{v} + U_0r + R_0u - W_0p - P_0w - g\cos(\Theta_0)\cos(\Phi_0)\phi + g\sin(\Theta_0)\sin(\Phi_0)\theta\right] &= \delta Y \\
m\left[\dot{w} + V_0p + P_0v - U_0q - Q_0u + g\cos(\Theta_0)\sin(\Phi_0)\phi + g\sin(\Theta_0)\cos(\Phi_0)\theta\right] &= \delta Z \\
I_x\dot{p} - I_{xz}\dot{r} + (I_z - I_y)(Q_0r + R_0q) - I_{xz}(P_0q + Q_0p) &= \delta L \\
I_y\dot{q} + (I_x - I_z)(P_0r + R_0p) - 2I_{xz}(R_0r + P_0p) &= \delta M \\
I_z\dot{r} - I_{xz}\dot{p} + (I_y - I_x)(P_0q + Q_0p) + I_{xz}(Q_0r + R_0q) &= \delta N
\end{aligned} \tag{3.2.4}$$

3.3 Decoupling of Longitudinal and Lateral Channels

It is common to decouple the longitudinal and lateral channels. That is, separating the *degrees of freedom* (DOFs) 1, 3 and 5, from the DOFs 2, 4 and 6. In other words this assumes that the longitudinal motions does not affect the lateral and vice versa. This assumption divides the system in two, which reduces the complexity of the system.

From (3.2.4) the terms with v, p and r is neglected in the longitudinal channel and terms with u, w and q are neglected in the lateral channel.

3.3.1 Longitudinal Channel

From (3.2.4) the longitudinal DOFs are:

$$\begin{aligned} m\left[\dot{u} + Q_0 w + W_0 q + g\cos(\Theta_0)\theta\right] &= \delta X \\ m\left[\dot{w} - U_0 q - Q_0 u + g\sin(\Theta_0)\cos(\Phi_0)\theta\right] &= \delta Z \\ I_y \dot{q} &= \delta M \end{aligned} \tag{3.3.1}$$

On matrix form the longitudinal channel can be expressed as:

$$\boldsymbol{M}_{lo}\delta\dot{\boldsymbol{\nu}}_{lo} + \boldsymbol{N}_{lo}\delta\boldsymbol{\nu}_{lo} + \boldsymbol{G}_{lo}\delta\boldsymbol{\eta}_{lo} = \delta\boldsymbol{\tau}_{lo} \tag{3.3.2}$$

where the matrices are:

$$\begin{aligned} \boldsymbol{M}_{lo} &= \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I_y \end{bmatrix}, \quad \boldsymbol{N}_{lo} = \begin{bmatrix} 0 & mQ_0 & mW_0 \\ -mQ_0 & 0 & -mU_0 \\ 0 & 0 & 0 \end{bmatrix}, \\ \boldsymbol{G}_{lo} &= \begin{bmatrix} mg\cos(\Theta_0) \\ mg\sin(\Theta_0)\cos(\Phi_0) \\ 0 \end{bmatrix}, \end{aligned} \tag{3.3.3}$$

and vectors are

$$\delta\boldsymbol{\nu}_{lo} = \begin{bmatrix} u \\ w \\ q \end{bmatrix}, \quad \delta\boldsymbol{\eta}_{lo} = \theta, \quad \delta\boldsymbol{\tau}_{lo} = \begin{bmatrix} \delta X \\ \delta Z \\ \delta M \end{bmatrix} \tag{3.3.4}$$

3.3.2 Lateral Channel

From (3.2.4) the lateral DOFs are:

$$\begin{aligned} m\left[\dot{v} + U_0 r - W_0 p - g\cos(\Theta_0)\cos(\Phi_0)\phi\right] &= \delta Y \\ I_x\dot{p} - I_{xz}\dot{r} + (I_z - I_y)Q_0 r - I_{xz}Q_0 p &= \delta L \\ I_z\dot{r} - I_{xz}\dot{p} + (I_y - I_x)Q_0 p + I_{xz}Q_0 r &= \delta N \end{aligned} \tag{3.3.5}$$

On matrix form the lateral channel can be expressed as:

$$\boldsymbol{M}_{la}\delta\dot{\boldsymbol{\nu}}_{la} + \boldsymbol{N}_{la}\delta\boldsymbol{\nu}_{la} + \boldsymbol{G}_{la}\delta\boldsymbol{\eta}_{la} = \delta\boldsymbol{\tau}_{la} \tag{3.3.6}$$

where the matrices are

$$\begin{aligned} \boldsymbol{M}_{la} &= \begin{bmatrix} m & 0 & 0 \\ 0 & I_x & -I_{xz} \\ 0 & -I_{xz} & I_z \end{bmatrix}, \quad \boldsymbol{N}_{la} = \begin{bmatrix} 0 & -mW_0 & mU_0 \\ 0 & -I_{xz}Q_0 & (I_z - I_y)Q_0 \\ 0 & (I_y - I_x)Q_0 & I_{xz}Q_0 \end{bmatrix}, \\ \boldsymbol{G}_{la} &= \begin{bmatrix} -mg\cos(\Theta_0)\cos(\Phi_0) \\ 0 \\ 0 \end{bmatrix}, \end{aligned} \tag{3.3.7}$$

and the vectors are

$$\delta\boldsymbol{\nu}_{la} = \begin{bmatrix} v \\ p \\ r \end{bmatrix}, \quad \delta\boldsymbol{\eta}_{la} = \begin{bmatrix} \phi \\ \psi \end{bmatrix}, \quad \delta\boldsymbol{\tau}_{la} = \begin{bmatrix} \delta Y \\ \delta L \\ \delta N \end{bmatrix} \tag{3.3.8}$$

3.4 Aircraft Model using Stability and Wind Axes

The model derived in Section 3.3 contains two rather useless states: V and W that is side and vertical velocity. These are closely related to, and can therefore be replaced by, the sideslip angle and the angle of attack. By including the two angles in the model it is easier to keep the aircraft stable in a control situation, as both of these should be regulated to zero. Also as seen in Section 3.6 these angles are used frequently to describe the model, therefore it is advantageous to include them as states.

Assumption 1 *It is assumed that α and β is small such that* $\cos(\cdot) \approx 1$ *and* $\sin(\cdot) \approx \cdot$*. It follows that $U = V_T$. From equations* (A.4.1) *and* (A.4.2)*, the angle of attack and the sideslip angle can be written as:*

$$\frac{\sin(\alpha)}{\cos(\alpha)} \approx \alpha = \frac{W}{V_T} \qquad \textit{and} \qquad \sin(\beta) \approx \beta = \frac{V}{V_T} \tag{3.4.1}$$

According to [Fossen, 2011b], when linear theory is applied the derivatives can be expressed as:

$$\dot{\alpha} = \frac{1}{V_T}\dot{W} \quad \text{and} \quad \dot{\beta} = \frac{1}{V_T}\dot{V} \tag{3.4.2}$$

The new state vector is:

$$\boldsymbol{\nu} = \begin{bmatrix} U \\ \beta' \\ \alpha' \\ P \\ Q \\ R \end{bmatrix} \triangleq \boldsymbol{\nu}_0 + \delta\boldsymbol{\nu} = \begin{bmatrix} U_0 \\ \beta_0 \\ \alpha_0 \\ P_0 \\ Q_0 \\ R_0 \end{bmatrix} + \begin{bmatrix} u \\ \beta \\ \alpha \\ p \\ q \\ r \end{bmatrix} \tag{3.4.3}$$

The decoupled models can now be derived from (3.3.2) and (3.3.6) by inserting (3.4.1) and (3.4.2):

Longitudinal Model

$$\begin{bmatrix} m & 0 & 0 \\ 0 & V_T m & 0 \\ 0 & 0 & I_y \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{\alpha} \\ \dot{q} \end{bmatrix} + \begin{bmatrix} 0 & V_T m Q_0 & m W_0 \\ -mQ_0 & 0 & -mU_0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ \alpha \\ q \end{bmatrix} + \begin{bmatrix} mg\cos(\Theta_0) \\ mg\sin(\Theta_0)\cos(\Phi_0) \\ 0 \end{bmatrix} \theta = \begin{bmatrix} \delta X \\ \delta Z \\ \delta M \end{bmatrix} \tag{3.4.4}$$

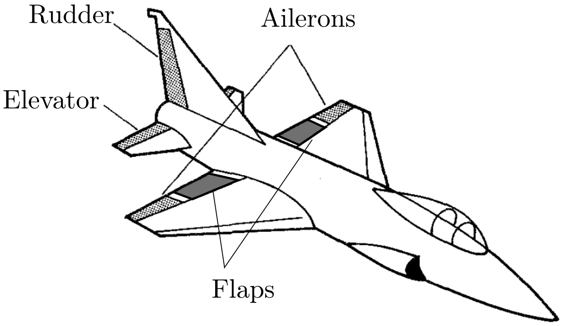


Figure 3.1: Control Surfaces (McLean 1990).

Lateral Model

$$\begin{bmatrix} V_T m & 0 & 0 \\ 0 & I_x & -I_{xz} \\ 0 & -I_{xz} & I_z \end{bmatrix} \begin{bmatrix} \dot{\beta} \\ \dot{p} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} 0 & -mW_0 & mU_0 \\ 0 & -I_{xz}Q_0 & (I_z - I_y)Q_0 \\ 0 & (I_y - I_x)Q_0 & I_{xz}Q_0 \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \end{bmatrix} + \begin{bmatrix} -mg\cos(\Theta_0)\cos(\Phi_0) \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} \phi \\ \psi \end{bmatrix} = \begin{bmatrix} \delta Y \\ \delta L \\ \delta N \end{bmatrix} \tag{3.4.5}$$

Equilibrium Conditions

The equilibrium conditions are found by using Assumption 1 and inserting (3.4.2) in (3.2.3).

3.5 Aerodynamic Forces and Moments

The aerodynamic forces and moments are divided in three parts: aerodynamic added mass, aerodynamic damping and forces and moments due to deflections of the control surfaces. On matrix form (3.5.1) these are $\boldsymbol{M}_F$, $\boldsymbol{N}_F$ and $\boldsymbol{B}$ respectively. The linear system which is the right hand side of equation (3.1.2) can be expressed as:

$$\boldsymbol{\tau} = \boldsymbol{M}_F\dot{\boldsymbol{\nu}} + \boldsymbol{N}_F\boldsymbol{\nu} + \boldsymbol{B}\boldsymbol{u} \tag{3.5.1}$$

For conventional aircraft added mass can be neglected.

To control the attitude, the aircraft is equipped with three main control surfaces: Ailerons (δ_A) for roll, elevator (δ_E) for pitch and rudder (δ_R) for yaw see Figure 3.1. The airspeed is controlled by thrust using the throttle (δ_T) and additional lift can be induced by the flaps (δ_F).

For a nonlinear system the right hand side of (3.1.2) can be expressed as:

$$\tau_{NL} = [X, Y, Z, L, M, N]^T = f(x, u) \tag{3.5.2}$$

These terms are also commonly expressed as aerodynamic coefficients [Klein and Morelli, 2006] $C_{(\cdot)}$.

$$C_i = C_i(\alpha, \beta, \mathbb{M}, h, \delta_s,) \tag{3.5.3}$$

for $i = \{D, Y, L, l, m, n\}$ where α is the angle of attack, β is the sideslip angle, $\mathbb{M}$ is the Mach number, h is the altitude and δ_s is the control surfaces where $s = \{T, E, F, A, R\}$.

$$\boldsymbol{f}^b = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \bar{q}S \begin{bmatrix} C_X \\ C_Y \\ C_Z \end{bmatrix}, \quad \boldsymbol{m}^b = \begin{bmatrix} L \\ M \\ N \end{bmatrix} = \bar{q}S \begin{bmatrix} bC_l \\ \bar{c}C_m \\ bC_n \end{bmatrix} \tag{3.5.4}$$

where $\bar{q}$ is given by (3.6.5) and S, b and $\bar{c}$ are found in Appendix B. Here C_X and C_Z can also be written as a function of the drag coefficient C_D and the lift coefficient C_L such that:

$$C_X = \bar{q}S[-C_D \cos(\alpha) + C_L \sin(\alpha)] \tag{3.5.5}$$

$$C_Z = \bar{q}S[-C_D \sin(\alpha) - C_L \cos(\alpha)] \tag{3.5.6}$$

By linearising (3.5.4) using partial differentiation (4.1.2), the aerodynamic forces and moments can be written on matrix form (3.5.1), where each element of the matrices are aerodynamic derivatives. This results in an equilibrium part $\boldsymbol{\tau}_0$ and a perturbation around the equilibrium $\delta\boldsymbol{\tau}$ such that:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_0 + \delta\boldsymbol{\tau} \tag{3.5.7}$$

Also the control surface vector is defined as

$$\boldsymbol{u} \triangleq \begin{bmatrix} \delta_T \\ \delta_E \\ \delta_F \\ \delta_A \\ \delta_R \end{bmatrix} = \begin{bmatrix} \delta_{T_0} \\ \delta_{E_0} \\ \delta_{F_0} \\ \delta_{A_0} \\ \delta_{R_0} \end{bmatrix} + \begin{bmatrix} \delta_t \\ \delta_e \\ \delta_f \\ \delta_a \\ \delta_r \end{bmatrix} \tag{3.5.8}$$

In the subsequent sections partial differentiation around an equilibrium point, $[x_0, u_0]$, will be expressed as X_U, which is a partial differentiation of X with respect to U:

$$X_U \Leftrightarrow \left.\frac{\partial X}{\partial U}\right|_{[x_0, u_0]} \tag{3.5.9}$$

3.5.1 Longitudinal Channel

$$\boldsymbol{\tau}_{0_{lo}} = f_{lo}(x_0, u_0) \tag{3.5.10}$$

Control surfaces in the longitudinal channel include thrust, elevator and flaps:

$$\delta\boldsymbol{u}_{lo} = \begin{bmatrix} \delta_t \\ \delta_e \\ \delta_f \end{bmatrix} \tag{3.5.11}$$

The perturbation model becomes:

$$\delta \boldsymbol{\tau}_{lo} = \begin{bmatrix} \delta X \\ \delta Z \\ \delta M \end{bmatrix} = \overbrace{\begin{bmatrix} X_U & X_{\alpha'} & X_Q \\ Z_U & Z_{\alpha'} & Z_Q \\ M_U & M_{\alpha'} & M_Q \end{bmatrix}}^{\boldsymbol{N}_{F_{lo}}} \delta \boldsymbol{\nu}_{lo} + \overbrace{\begin{bmatrix} X_{\delta_T} & X_{\delta_E} & X_{\delta_F} \\ Z_{\delta_T} & Z_{\delta_E} & Z_{\delta_F} \\ M_{\delta_T} & M_{\delta_E} & M_{\delta_F} \end{bmatrix}}^{\boldsymbol{B}_{lo}} \delta \boldsymbol{u}_{lo} \tag{3.5.12}$$

3.5.2 Lateral Channel

$$\boldsymbol{\tau}_{0_{la}} = f_{la}(x_0, u_0) \tag{3.5.13}$$

Control surfaces in the lateral channel include ailerons and rudder:

$$\delta \boldsymbol{u}_{la} = \begin{bmatrix} \delta_a \\ \delta_r \end{bmatrix} \tag{3.5.14}$$

$$\delta \boldsymbol{\tau}_{la} = \begin{bmatrix} \delta Y \\ \delta L \\ \delta N \end{bmatrix} = \overbrace{\begin{bmatrix} Y_{\beta'} & Y_P & Y_R \\ L_{\beta'} & L_P & L_R \\ N_{\beta'} & N_P & N_R \end{bmatrix}}^{\boldsymbol{N}_{F_{la}}} \delta \boldsymbol{\nu}_{la} + \overbrace{\begin{bmatrix} Y_{\delta_A} & Y_{\delta_R} \\ L_{\delta_A} & L_{\delta_R} \\ N_{\delta_A} & N_{\delta_R} \end{bmatrix}}^{\boldsymbol{B}_{la}} \delta \boldsymbol{u}_{la} \tag{3.5.15}$$

3.6 Development of Linear Model for the Cessna 172SP

In this section the linear model of the Cessna 172SP is derived. The nonlinear model from [Vistnes, 2012] uses aerodynamic coefficients and is linearised using partial differentiation as described above. The values of the coefficients along with a description are found in Appendix B. In addition to this list $\bar{q}$ is the dynamic pressure given by (3.6.5).

3.6.1 Nonlinear Equations of Aerodynamic Forces and Moments

The following equations are obtained from Appendix E.4 and E.5 in [Vistnes, 2012]. The model for the landing gear is not required as this is designed to only produce forces and moments when the wheels touch the ground.

$$
\begin{aligned}
X =& \bar{q}S\cos(\alpha') \\
& \left[-C_{D_0} - C_{D_\alpha}\alpha' - C_{D_{\alpha^2}}\alpha'^2 - \frac{C_{D_{T_C}}}{\bar{q}S}\delta_T - C_{D_{\delta F}}\delta_F - C_{D_{\delta F^2}}\delta_F^2\right] \\
Y =& \bar{q}S\left[C_{Y_0} + C_{Y_\beta}\beta' + C_{Y_{\delta R}}\delta_R + \frac{C_{Y_{T_C}}}{\bar{q}S} + \frac{b}{2V_T}(C_{Y_p}P) + \frac{b}{2V_T}(C_{Y_r}R)\right] \\
& + \bar{q}S\sin(\alpha')\left[C_{L_0} + C_{L_\alpha}\alpha' + C_{L_{\delta E}}\delta_E + C_{L_{\delta F}}\delta_F + C_{L_{\delta F^2}}\delta_F^2\right] + \delta_T \\
Z =& \bar{q}S\sin(\alpha') \\
& \left[-C_{D_0} - C_{D_\alpha}\alpha' - C_{D_{\alpha^2}}\alpha'^2 - \frac{C_{D_{T_C}}}{\bar{q}S}\delta_T - C_{D_{\delta F}}\delta_F - C_{D_{\delta F^2}}\delta_F^2\right]\bar{q}S \qquad (3.6.1) \\
L =& b\bar{q}S\left[C_{l_\beta}\beta' + \frac{C_{l_{T_C}}}{\bar{q}S}\delta_T + C_{l_{\delta A}}\delta_A + C_{l_{\delta R}}\delta_R + \frac{b}{2V_T}(C_{l_p}P) + \frac{b}{2V_T}(C_{l_r}R)\right] \\
& + \bar{q}S\cos(\alpha')\left[-C_{L_0} - C_{L_\alpha}\alpha' - C_{L_{\delta E}}\delta_E - C_{L_{\delta F}}\delta_F - C_{L_{\delta F^2}}\delta_F^2\right] \\
M =& \bar{q}S\bar{c}\left[C_{m_\alpha}\alpha' + C_{m_{\delta E}}\delta_E + C_{m_q}\frac{\bar{c}}{2V_T}Q\right] + r^b_{T_z}\delta_T \\
N =& b\bar{q}S[C_{n_0} + C_{n_\beta}\beta' + \frac{C_{n_{T_C}}}{\bar{q}S}\delta_T + C_{n_{\delta A}}\delta_A \\
& + C_{n_{\delta R}}\delta_R + \frac{b}{2V_T}(C_{n_p}P) + \frac{b}{2V_T}(C_{n_r}R)]
\end{aligned}
$$

3.6.2 Linearisation

$$
\boldsymbol{x} = \begin{bmatrix} U \\ \beta' \\ \alpha' \\ P \\ Q \\ R \end{bmatrix} = \begin{bmatrix} U_0 \\ \beta_0 \\ \alpha_0 \\ P_0 \\ Q_0 \\ R_0 \end{bmatrix} + \begin{bmatrix} u \\ \beta \\ \alpha \\ p \\ q \\ r \end{bmatrix} \qquad (3.6.2)
$$

The linearized model is derived with the same structure as in Section 3.5. From (4.1.2) the equilibrium and perturbation equations become:

$$
\boldsymbol{\tau}_0 = f_\tau[\boldsymbol{x}_0, \boldsymbol{u}_0] \qquad (3.6.3)
$$

$$
\delta\boldsymbol{\tau} = \left.\frac{\partial f_\tau}{\partial \boldsymbol{x}}\right|_{[\boldsymbol{x}_0, \boldsymbol{u}_0]} \delta\boldsymbol{x} + \left.\frac{\partial f_\tau}{\partial \boldsymbol{u}}\right|_{[\boldsymbol{x}_0, \boldsymbol{u}_0]} \delta\boldsymbol{u} \qquad (3.6.4)
$$

where f_τ is the nonlinear equations of aerodynamics forces and moments, (3.6.1).

Assumption 1 shows that $V_T = U$. The dynamic pressure $\bar{q}$ can be written as:

$$
\bar{q} = \frac{\rho}{2}V_T^2 = \frac{\rho}{2}U^2 \qquad (3.6.5)
$$

The equilibrium equations (3.6.3) can be derived directly from (3.6.1) by inserting $\boldsymbol{x}_0$ and $\boldsymbol{u}_0$ and will not be elaborated. The perturbation models of the longitudinal and the lateral channel is found in the subsequent sections.

Longtudinal Channel

$$
\begin{bmatrix} \delta X \\ \delta Z \\ \delta M \end{bmatrix} = \overbrace{\begin{bmatrix} \frac{\partial X}{\partial U}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} u & \frac{\partial X}{\partial \alpha'}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \alpha & \frac{\partial X}{\partial Q}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} q \\ \frac{\partial Z}{\partial U}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} u & \frac{\partial Z}{\partial \alpha'}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \alpha & \frac{\partial Z}{\partial Q}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} q \\ \frac{\partial M}{\partial U}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} u & \frac{\partial M}{\partial \alpha'}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \alpha & \frac{\partial M}{\partial Q}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} q \end{bmatrix}}^{\boldsymbol{N}_{F_{lo}} \delta \boldsymbol{\nu}_{lo}}
$$
$$
+ \overbrace{\begin{bmatrix} \frac{\partial X}{\partial \delta_T}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_t & \frac{\partial X}{\partial \delta_E}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_e & \frac{\partial X}{\partial \delta_F}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_f \\ \frac{\partial Z}{\partial \delta_T}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_t & \frac{\partial Z}{\partial \delta_E}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_e & \frac{\partial Z}{\partial \delta_F}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_f \\ \frac{\partial M}{\partial \delta_T}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_t & \frac{\partial M}{\partial \delta_E}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_e & \frac{\partial M}{\partial \delta_F}\Big|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_f \end{bmatrix}}^{\boldsymbol{B}_{lo} \delta \boldsymbol{u}_{lo}} \tag{3.6.6}
$$

Each matrix entry is derived below where (3.6.7) to (3.6.9) constitute $\boldsymbol{N}_{F_{lo}}$ and (3.6.10) to (3.6.12) constitute $\boldsymbol{B}_{lo}$:

$$
\begin{aligned}
\frac{\partial X}{\partial U}\bigg|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= S\rho U_0[-C_{D_0} - C_{D_\alpha}\alpha_0 - C_{D_{\alpha^2}}\alpha_0^2 - C_{D_{\delta F}}\delta_{F_0} - C_{D_{\delta F^2}}\delta_{F_0}^2 \\
&\quad + \alpha_0(C_{L_0} + C_{L_\alpha}\alpha_0 + C_{L_{\delta E}}\delta_{E_0} + C_{L_{\delta F}}\delta_{F_0} + C_{L_{\delta F^2}}\delta_{F_0}^2)] \\
\frac{\partial X}{\partial \alpha'}\bigg|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S[-C_{D_\alpha} - 2C_{D_{\alpha^2}}\alpha_0 \\
&\quad + C_{L_0} + 2C_{L_\alpha}\alpha_0 + C_{L_{\delta E}}\delta_{E_0} + C_{L_{\delta F}}\delta_{F_0} + C_{L_{\delta F^2}}\delta_{F_0}^2] \\
\frac{\partial X}{\partial Q}\bigg|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= 0
\end{aligned} \tag{3.6.7}
$$

$$\begin{aligned}
\left.\frac{\partial Z}{\partial U}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= S\rho U_0 \\
&\quad [\alpha_0(-C_{D_0} - C_{D_\alpha}\alpha_0 - C_{D_{\alpha^2}}\alpha_0^2 - C_{D_{\delta F}}\delta_{F_0} - C_{D_{\delta F^2}}\delta_{F_0}^2) \\
&\quad - C_{L_0} - C_{L_\alpha}\alpha_0 - C_{L_{\delta E}}\delta_{E_0} - C_{L_{\delta F}}\delta_{F_0} - C_{L_{\delta F^2}}\delta_{F_0}^2] \\
\left.\frac{\partial Z}{\partial \alpha'}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S \\
&\quad [-C_{D_0} - 2C_{D_\alpha}\alpha - 3C_{D_{\alpha^2}}\alpha^2 - \frac{C_{D_{T_C}}}{\bar{q}S}\delta_{T_0} - C_{D_{\delta F}}\delta_{F_0} - C_{D_{\delta F^2}}\delta_{F_0}^2 \\
&\quad - C_{L_\alpha}] \\
\left.\frac{\partial Z}{\partial Q}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= 0
\end{aligned} \tag{3.6.8}$$

$$\begin{aligned}
\left.\frac{\partial M}{\partial U}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= S\rho U_0\bar{c}[C_{m_\alpha}\alpha_0 + C_{m_{\delta E}}\delta_{E_0}] + S\rho\frac{1}{4}\bar{c}^2 Q_0 \\
\left.\frac{\partial M}{\partial \alpha'}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S\bar{c}[C_{m_\alpha}] \\
\left.\frac{\partial M}{\partial Q}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S\bar{c}[C_{m_q}\frac{\bar{c}}{2V_T}]
\end{aligned} \tag{3.6.9}$$

$$\begin{aligned}
\left.\frac{\partial X}{\partial \delta_T}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= -C_{D_{T_C}} + 1 \\
\left.\frac{\partial X}{\partial \delta_E}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S[C_{L_{\delta E}}\alpha_0] \\
\left.\frac{\partial X}{\partial \delta_F}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S[-C_{D_{\delta F}} - 2C_{D_{\delta F^2}}\delta_{F_0} + \alpha C_{L_{\delta F}} + 2\alpha_0 C_{L_{\delta F^2}}\delta_{F_0}]
\end{aligned} \tag{3.6.10}$$

$$\begin{aligned}
\left.\frac{\partial Z}{\partial \delta_T}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= -C_{D_{T_C}}\alpha_0 \\
\left.\frac{\partial Z}{\partial \delta_E}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S[-C_{L_{\delta E}}] \\
\left.\frac{\partial Z}{\partial \delta_F}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S[-\alpha_0 C_{D_{\delta F}} - 2\alpha_0 C_{D_{\delta F^2}}\delta_{F_0} - C_{L_{\delta F}} - C_{L_{\delta F^2}}\delta_{F_0}]
\end{aligned} \tag{3.6.11}$$

$$\begin{aligned}
\left.\frac{\partial M}{\partial \delta_T}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= r^b_{T_z} \\
\left.\frac{\partial M}{\partial \delta_E}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S\bar{c}[C_{m_{\delta E}}] \\
\left.\frac{\partial M}{\partial \delta_F}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= 0
\end{aligned} \tag{3.6.12}$$

Lateral Channel

$$\begin{aligned}
\begin{bmatrix}\delta Y \\ \delta L \\ \delta N\end{bmatrix} = {} & \overbrace{\begin{bmatrix}
\left.\frac{\partial Y}{\partial \beta'}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \beta & \left.\frac{\partial Y}{\partial P}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} p & \left.\frac{\partial Y}{\partial R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} r \\
\left.\frac{\partial L}{\partial \beta'}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \beta & \left.\frac{\partial L}{\partial P}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} p & \left.\frac{\partial L}{\partial R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} r \\
\left.\frac{\partial N}{\partial \beta'}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \beta & \left.\frac{\partial N}{\partial P}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} p & \left.\frac{\partial N}{\partial R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} r
\end{bmatrix}}^{\boldsymbol{N}_{F_{la}}\delta\boldsymbol{\nu}_{la}} \\
& + \overbrace{\begin{bmatrix}
\left.\frac{\partial Y}{\partial \delta_A}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_a & \left.\frac{\partial Y}{\partial \delta_R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_r \\
\left.\frac{\partial L}{\partial \delta_A}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_a & \left.\frac{\partial L}{\partial \delta_R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_r \\
\left.\frac{\partial N}{\partial \delta_A}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_a & \left.\frac{\partial N}{\partial \delta_R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} \delta_r
\end{bmatrix}}^{\boldsymbol{B}_{la}\delta\boldsymbol{u}_{la}}
\end{aligned} \tag{3.6.13}$$

Each matrix entry is derived below where (3.6.14) to (3.6.16) constitute $\boldsymbol{N}_{F_{la}}$ and (3.6.17) to (3.6.19) constitute $\boldsymbol{B}_{la}$:

$$\begin{aligned}
\left.\frac{\partial Y}{\partial \beta'}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S[C_{Y_\beta}] \\
\left.\frac{\partial Y}{\partial P}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= S\frac{b}{4}\rho V_T[C_{Y_p}] \\
\left.\frac{\partial Y}{\partial R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= S\frac{b}{4}\rho V_T[C_{Y_r}]
\end{aligned} \tag{3.6.14}$$

$$\begin{aligned}
\left.\frac{\partial L}{\partial \beta'}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S[C_{l_\beta}] \\
\left.\frac{\partial L}{\partial P}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= S\frac{b^2}{4}\rho V_T[C_{l_p}] \\
\left.\frac{\partial L}{\partial R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= S\frac{b^2}{4}\rho V_T[C_{l_r}]
\end{aligned} \tag{3.6.15}$$

$$\begin{aligned} \left.\frac{\partial N}{\partial \beta'}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= b\bar{q}S[C_{n_\beta}] \\ \left.\frac{\partial N}{\partial P}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= S\frac{b^2}{4}\rho V_T[C_{n_p}] \\ \left.\frac{\partial N}{\partial R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= S\frac{b^2}{4}\rho V_T[C_{n_r}] \end{aligned} \tag{3.6.16}$$

$$\begin{aligned} \left.\frac{\partial Y}{\partial \delta_A}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= 0 \\ \left.\frac{\partial Y}{\partial \delta_R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= \bar{q}S[C_{Y_{\delta R}}] \end{aligned} \tag{3.6.17}$$

$$\begin{aligned} \left.\frac{\partial L}{\partial \delta_A}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= b\bar{q}S[C_{l_{\delta A}}] \\ \left.\frac{\partial L}{\partial \delta_R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= b\bar{q}S[C_{l_{\delta R}}] \end{aligned} \tag{3.6.18}$$

$$\begin{aligned} \left.\frac{\partial N}{\partial \delta_A}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= b\bar{q}S[C_{n_{\delta A}}] \\ \left.\frac{\partial N}{\partial \delta_R}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]} &= b\bar{q}S[C_{n_{\delta R}}] \end{aligned} \tag{3.6.19}$$

Chapter 4

Control Theory

This chapter introduces necessary background theory on which this thesis will be based. This includes subjects within guidance and control, observers and fault detection. Some of these sections are based on the unpublished project report submitted in advance of this thesis.

4.1 Linearisation

Assumptions made for this linearisation. A non-linear system is represented on state-space from:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) \tag{4.1.1}$$

where $\boldsymbol{x}$ is the states and $\boldsymbol{u}$ are the inputs. The system is linearised in [Balchen et al., 2003] using partial differentiation as:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}_0, \boldsymbol{u}_0) + \left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{[\boldsymbol{x}_0, \boldsymbol{u}_0]} \boldsymbol{\delta x} + \left.\frac{\partial f}{\partial \boldsymbol{u}}\right|_{[\boldsymbol{x}_0, \boldsymbol{u}_0]} \boldsymbol{\delta u} \tag{4.1.2}$$

where x_0 and u_0 are the nominal value of $\boldsymbol{x}$ and $\boldsymbol{u}$, and $\boldsymbol{\delta x}$ and $\boldsymbol{\delta u}$ are perturbations around the equilibrium.

4.2 Linear Quadratic Optimal Control

For system (4.2.1) an optimal control law can be found by using a LQ control strategy which utilizes an *algebraic Riccati equation* (ARE) [Antsaklis and Michel, 1997].

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \\ \boldsymbol{y} &= \boldsymbol{C}\boldsymbol{x} \end{aligned} \tag{4.2.1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^r$ and $\boldsymbol{y} \in \mathbb{R}^m$. A requirement for the implementation is that the system must be controllable, that is $\mathbb{C}$ must be of full rank ($\text{rank}(\mathbb{C}) = n$):

$$\mathbb{C} = \left[\boldsymbol{B} | \boldsymbol{A}\boldsymbol{B} | \cdots | (\boldsymbol{A})^{n-1}\boldsymbol{B}\right] \tag{4.2.2}$$

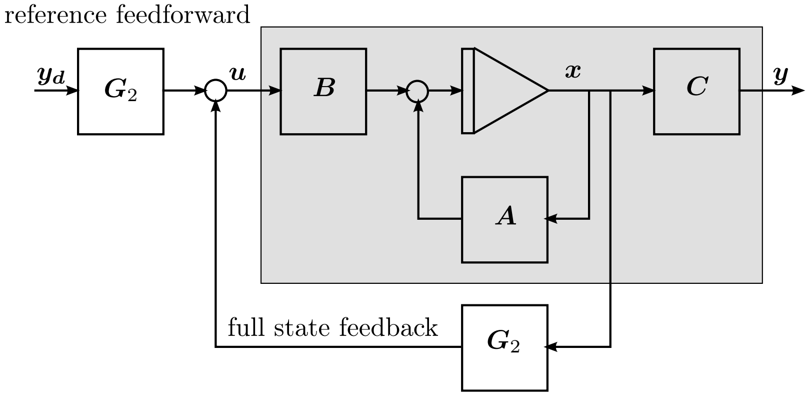


Figure 4.1: Block diagram of LQ Regulator with reference feedforward, [Fossen, 2011a]

4.2.1 Trajectory tracking

A reference signal $\boldsymbol{y}_d$ is generated from a linear reference model:

$$\begin{aligned}\dot{\boldsymbol{x}}_d &= \boldsymbol{A}_d\boldsymbol{x}_d + \boldsymbol{B}_d\boldsymbol{r}\\ \boldsymbol{y}_d &= \boldsymbol{C}\boldsymbol{x}_d\end{aligned} \tag{4.2.3}$$

where $\boldsymbol{r}$ is the setpoint vector. The desired trajectory is followed by driving the state signals to the reference. The error signal is defined as:

$$\boldsymbol{e} \triangleq \boldsymbol{y} - \boldsymbol{y}_d \tag{4.2.4}$$

$$= \boldsymbol{C}(\boldsymbol{x} - \boldsymbol{x}_d) \tag{4.2.5}$$

$$= \boldsymbol{C}\tilde{\boldsymbol{x}} \tag{4.2.6}$$

The optimal solution is found by minimizing a cost function given by (4.2.8). $\boldsymbol{Q} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{R} \in \mathbb{R}^{r\times r}$ are both diagonal design matrices, which respectively *weight* the measured state errors and *punishes* the use of control input. Only the states present in the reference signal $\boldsymbol{y}_d$ is weighted. (4.2.7) ensures that the remaining reference values are set to zero, such that $\tilde{\boldsymbol{Q}} \in \mathbb{R}^{n\times n} \geq 0$. This means that these sates are to be regulated to zero.

$$\tilde{\boldsymbol{Q}} \triangleq \boldsymbol{C}^T\boldsymbol{Q}\boldsymbol{C} \tag{4.2.7}$$

$$J = \min\left\{\frac{1}{2}\int_0^\infty (\tilde{\boldsymbol{x}}^T\tilde{\boldsymbol{Q}}\tilde{\boldsymbol{x}} + \boldsymbol{u}^T\boldsymbol{R}\boldsymbol{u})dt\right\} \tag{4.2.8}$$

where $\tilde{\boldsymbol{Q}} = \tilde{\boldsymbol{Q}}^T$ and $\boldsymbol{R} = \boldsymbol{R}^T > 0$.

From [Fossen, 2011a] the control law (4.2.9) is implemented. This law includes full state feedback and reference feedfoward for trajectory tracking.

$$\boldsymbol{u} = \boldsymbol{G}_1\boldsymbol{x} + \boldsymbol{G}_2\boldsymbol{y}_d \tag{4.2.9}$$

where $\boldsymbol{G}_1 \in \mathbb{R}^{r\times n}$ and $\boldsymbol{G}_2 \in \mathbb{R}^{r\times m}$ are found by:

$$\begin{aligned}\boldsymbol{G}_1 &= -\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P}_\infty \\ \boldsymbol{G}_2 &= -\boldsymbol{R}^{-1}\boldsymbol{B}^T(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{G}_1)^{-T}\boldsymbol{C}^T\boldsymbol{Q}\end{aligned} \tag{4.2.10}$$

$\boldsymbol{P}_\infty$ is found by satisfying the ARE (4.2.12). Because there are no time constraints on the problem (4.2.8) infinite horizon is assumed. As $t \to \infty$ then $\dot{\boldsymbol{P}} \to 0$ and $\boldsymbol{P} \to \boldsymbol{P}_\infty$:

$$\dot{\boldsymbol{P}} = -\boldsymbol{P}\boldsymbol{A} - \boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P} - \tilde{\boldsymbol{Q}} \tag{4.2.11}$$

$$\downarrow$$

$$\boldsymbol{0} = -\boldsymbol{P}_\infty\boldsymbol{A} - \boldsymbol{A}^T\boldsymbol{P}_\infty + \boldsymbol{P}_\infty\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P}_\infty - \tilde{\boldsymbol{Q}} \tag{4.2.12}$$

In MATLAB $\boldsymbol{P}_\infty$ is found by the use of the function *lqr*:

```
        [K,P,eig]lqr = (A,B,C'*Q*C,R)
```

which returns $K = -\boldsymbol{G}_1$, $P = \boldsymbol{P}_\infty$ and *eig* which is the eigenvalues of the new stabilized system (4.2.14). By inserting (4.2.9) in (4.2.1) the system becomes:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}(\boldsymbol{G}_1\boldsymbol{x} + \boldsymbol{G}_2\boldsymbol{y}_d) \tag{4.2.13}$$

$$\dot{\boldsymbol{x}} = (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{G}_1)\boldsymbol{x} + \boldsymbol{B}\boldsymbol{G}_2\boldsymbol{y}_d \tag{4.2.14}$$

The resulting system is illustrated in Figure 4.1.

4.2.2 Tuning

The matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ have to be tuned manually. Looking at each matrix separately a high value of an element will result in a high cost value in (4.2.8). In other words a high value in the $\boldsymbol{Q}$ matrix will result in a stricter regulation for that state, tolerating a smaller error. For the $\boldsymbol{R}$ matrix this means that a high control input for the selected control surface will not be tolerated. In addition the relation between the tuning matrices also matters. The case where $\boldsymbol{Q} >> \boldsymbol{R}$ means that it is very important to keep the state errors as small as possible. This means that the vehicle is free to use more control input to accomplish this; the regulation will be more aggressive. The opposite case $\boldsymbol{Q} << \boldsymbol{R}$ is that the vehicle is to use less control input. This can be interpreted as that accuracy is sacrificed for low energy consumption.

There is a more intuitive way of tuning these matrices instead of picking random numbers. The Bryson's tuning rule (4.2.15) opens a more substantial way of tuning. However this method relies on that the user have a good understanding of the system. Each element in $\boldsymbol{Q}$ and $\boldsymbol{R}$ can be tuned by the rule:

$$\begin{aligned} q_{ii} &= \frac{1}{\text{maximum acceptable value of } \tilde{x}_i^2} \\ r_{ii} &= \frac{1}{\text{maximum acceptable value of } u_i^2}\end{aligned} \tag{4.2.15}$$

4.3 The Proportional-Internal-Derivative Controller

The *Proportional-Integral-Derivative* (PID) controller is the most common feedback controller used in control theory today. It uses the error dynamics of a process to achieve a desired state. The performance of the controller is decided by the tuning constants K_p, K_i and K_d, which have to be tuned manually. The PID controller is given in (4.3.1) where e signifies the error dynamics in position and v the error dynamics of the velocity. The block diagram of the system is seen in Figure 4.2.

$$\delta = K_p e(t) + K_i \int\limits_0^t e(\tau)d\tau + K_d v(t) \tag{4.3.1}$$

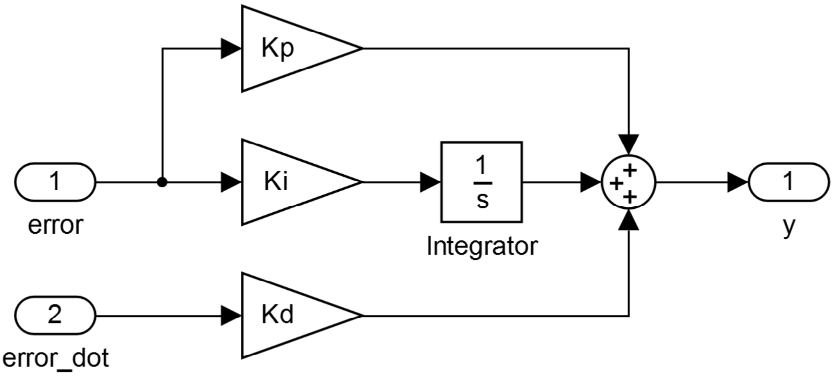


Figure 4.2: Block diagram of the Proportional Internal Derivative Controller

4.3.1 Anti-Windup

Windup occurs when there is a large step in set point, and the controller is not able to bring the error to zero in sufficient time. The integral action will accumulate to an undesirably high value, and the controller will overshoot the reference value. The time response of the system will have an underdamped characteristic. To solve this saturation is the key. Saturating the control output will prevent the signal in becoming too large. However this will not stop the integral winding up. A solution can be seen in Figure 4.3. The integral blocks in MATLAB Simulink have an anti-windup function implemented as saturation.

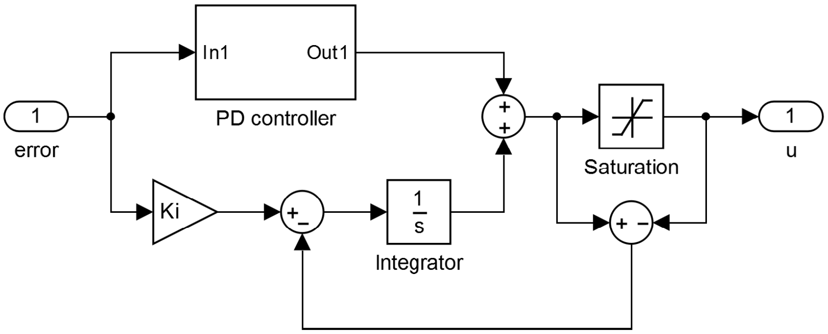


Figure 4.3: Anti-Windup block diagram

4.4 Reference model

Reference models are used to smooth the reference values. When there is a change in setpoint a smooth trajectory is created as seen in Figure 4.4. The purpose is to reduce the error jumps in the controllers, such that they can be tuned to have a fast and accurate response around zero.

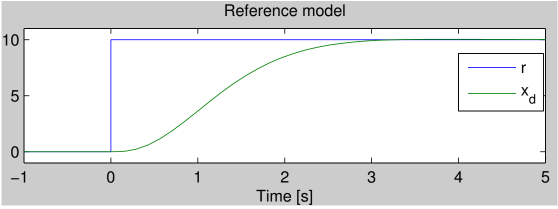


Figure 4.4: Filtering a step with a reference model one achieves a smooth trajectory

When creating a reference model for position and attitude, it is common to use the dynamics of mass-spring-damper systems in cascade with a low-pass (LP)-filter, [Fossen, 2011a]. This results in a third order system, which ensures smooth acceleration and velocity.

The Laplace transformed transfer-function of the cascaded system can be seen in (4.4.1). x_d is the smooth reference value and r is the desired state.

$$\frac{x_d}{r}(s) = \frac{1}{(1+\frac{1}{\omega_n}s)}\frac{\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2} \tag{4.4.1}$$

ω_n is the angular frequency of an undamped system and ζ is the dampening ratio.

Block Diagram Representation

It is necessary to derive the block diagram of (4.4.1) for the implementation in Chapter 5.1. The state space representation for the LP filter and the mass-spring-damper system is derived separately:

$$\frac{x_{LP}}{r} = \frac{1}{1+\frac{1}{\omega_n}s} \tag{4.4.2}$$

$$\frac{x_d}{x_{LP}} = \frac{\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2} \tag{4.4.3}$$

For the LP filter:

$$x_{LP}(1+\frac{1}{\omega_n}s) = r \tag{4.4.4}$$

$$x_{LP}+\frac{1}{\omega_n}\dot{x}_{LP} = r \tag{4.4.5}$$

$$\dot{x}_{LP} = \omega_n(-x_{LP}+r) \tag{4.4.6}$$

For the mass-spring-damper system:

$$x_d(s^2+2\zeta\omega_n s+\omega_n^2) = \omega_n^2 x_{LP} \tag{4.4.7}$$

$$\ddot{x}_d+2\zeta\omega_n\dot{x}_d+\omega_n^2 x_d = \omega_n^2 x_{LP} \tag{4.4.8}$$

Defining $x_1 \triangleq x_d$ and $x_2 \triangleq \dot{x}_d$, the system can be written in state space representation:

$$\dot{x}_1 = x_2 \tag{4.4.9}$$

$$\dot{x}_2 = -2\zeta\omega_n x_2 - \omega_n^2 x_1 + \omega_n^2 x_{LP} \tag{4.4.10}$$

Using the state space representations of the LP filter, (4.4.6), and the mass-spring-damper system, (4.4.9) - (4.4.10), the block diagram of the reference model can be derived as seen in Figure 4.5.

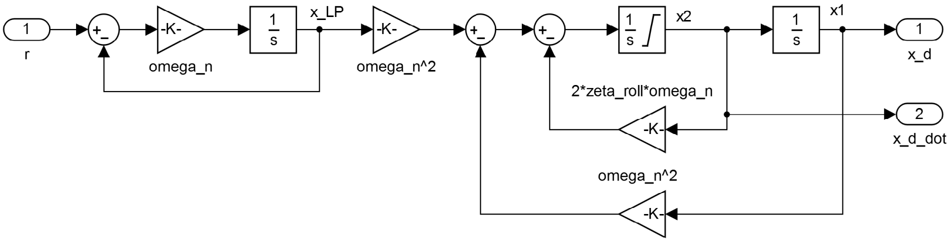


Figure 4.5: Block diagram of a third order reference model, consisting of a low pass filter in cascade with a mass-spring-damper system.

Performance and Tuning

The reference model is designed to have slower dynamics than the aircraft. This way the aircraft is always able to follow the reference signal. The system in Figure 4.5 is tuned with the constants ζ and ω_n, where ζ is the dampening ratio and ω_n is the natural frequencies.

Another benefit of the block diagram representation is the possibility of adding saturations. In Figure 4.5 velocity saturation is added. This will ensure that the velocity of the reference signal does not exceed the capabilities of the aircraft. This can also be implemented in acceleration. By limiting the performance of the reference signal the aircraft will be able to follow it and minimizing the error in the controller.

4.5 Guidance

The guidance calculates the desired attitude based on the aircraft's deviation from the path.

4.5.1 Path following

A path is defined by waypoints in the NED-frame $\boldsymbol{p}_k \triangleq [x_k, y_k, z_k]^T$, where the vector between $\boldsymbol{p}_k$ and $\boldsymbol{p}_{k+1}$ describes the current line segment.

Track errors

To be able to describe the aircraft's position relative to the path, a path-fixed reference frame (*path*) is defined with origin in $\boldsymbol{p}_k$. The x-axis, aligned in the direction of the path, is rotated an angle χ_p to the north-axis as seen in Figure 4.6. The z-axis is rotated an angle ν_D to the down-axis and gives $\nu_p \triangleq \nu_D - 90°$. 90° is subtracted from ν_D such that a path with $\nu_p = 0$ is a horizontal path.

$$\chi_p \triangleq \operatorname{arctan2}\left(y_{k+1} - y_k, x_{k+1} - x_k\right) \tag{4.5.1}$$

$$\nu_p \triangleq \operatorname{arctan2}\left(\sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2}, z_{k+1} - z_k\right) \tag{4.5.2}$$

The function arctan2 is defined in (4.5.3).

$$\operatorname{arctan2}(y, x) \triangleq \begin{cases} \arctan(\frac{y}{x}) & x > 0 \\ \arctan(\frac{y}{x}) + \pi & y \geq 0, x < 0 \\ \arctan(\frac{y}{x}) - \pi & y < 0, x < 0 \\ +\frac{[}{2}\pi] & y > 0, x = 0 \\ -\frac{[}{2}\pi] & y < 0, x = 0 \\ \text{undefined} & y = 0, x = 0 \end{cases} \tag{4.5.3}$$

Two rotation matrices forms the final rotation between the *path*-frame and the NED-frame as shown in (4.5.6).

$$\boldsymbol{R}(\chi_p) = \begin{bmatrix} \cos(\chi_p) & -\sin(\chi_p) & 0 \\ \sin(\chi_p) & \cos(\chi_p) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.5.4}$$

$$\boldsymbol{R}(\nu_p) = \begin{bmatrix} \cos(\nu_p) & 0 & \sin(\nu_p) \\ 0 & 1 & 0 \\ -\sin(\nu_p) & 0 & \cos(\nu_p) \end{bmatrix} \tag{4.5.5}$$

$$\boldsymbol{R}^n_{path} \triangleq \boldsymbol{R}(\chi)\boldsymbol{R}(\nu) \tag{4.5.6}$$

A vector w is defined as the vector from the aircraft to the origin of the *path* reference frame: $w^n \triangleq (\boldsymbol{p}(t) - \boldsymbol{p}_k)$. Both waypoints are given in the NED frame, thus w^n. The aircraft's deviation of the path can then be modelled in the *path*-frame:

$$\boldsymbol{\epsilon}^{path}(t) = \boldsymbol{R}^{path}_n w^n \tag{4.5.7}$$

$$\boldsymbol{\epsilon}^{path}(t) = \begin{bmatrix} s(t) & e(t) & h(t) \end{bmatrix}^T \tag{4.5.8}$$

s, e and h are *along-track*, *cross-track error* and *vertical-track error* respectively, relative to $\boldsymbol{p}_k$.

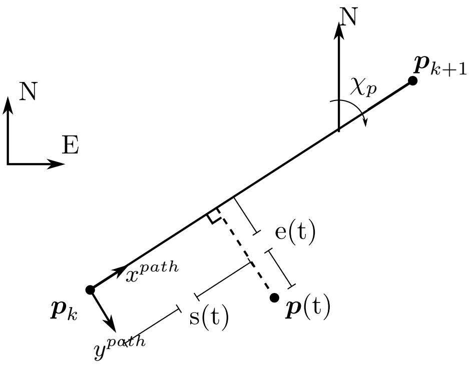


Figure 4.6: A path is given by the waypoints p_k and p_{k+1}, and $p(t)$ is the position of the aircraft. The aircraft's deviation of the path in the *path*-frame is given by $\epsilon = [s(t), e(t), h(t)]^T$, where $s(t)$ and $e(t)$ is shown in the figure.

Lookahead-based steering

Lookahead-based steering, [Breivik and Fossen, 2009], defines a point on the path that the aircraft is directed towards, the *steering point*. The distance between this point and the projection of the aircraft onto the path is called the *lookahead distance*, Δ. The definition of Δ is given in (4.5.9) and is dependent on one tuning variable R. R is the length from the aircraft to the steering point as seen in Figure 4.7, this vector is called the *line of sight* (LOS)-vector. The steering direction is defined by two angles: χ for horizontal and ν for vertical steering. Sideslip β and angle of attack α is included see Section 4.5.2.

$$\Delta(t) \triangleq \sqrt{R^2 - e(t)^2} \tag{4.5.9}$$

$$\chi(e) \triangleq \chi_p + \chi_r(e) + \beta \qquad \nu(h) \triangleq \nu_p + \nu_r(h) + \alpha \tag{4.5.10}$$

where

$$\chi_r(e) \triangleq \arctan\left(-\frac{e(t)}{\Delta}\right) \tag{4.5.11}$$

and

$$\nu_r(h) \triangleq \arctan\left(\frac{h(t)}{\sqrt{e(t)^2 + \Delta^2}}\right) \tag{4.5.12}$$

R is chosen arbitrarily and affects the performance. A small R gives aggressive steering and a large R gives smooth steering. When this variable is chosen the one should consider the steering capabilities of the aircraft.

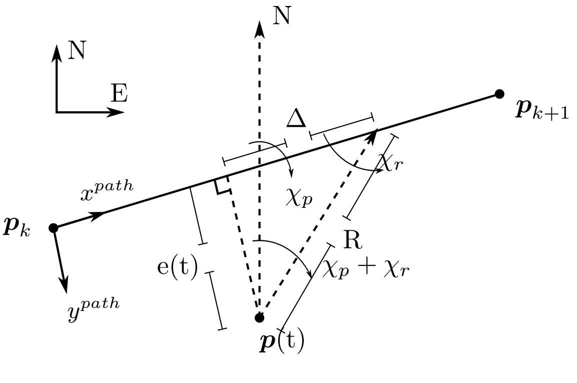


Figure 4.7: The steering laws in the horizontal plane, (4.5.10), is a summation of three angles. χ_p and χ_r is described in the figure. χ_r is given by e(t) and Δ.

4.5.2 The Stability and Wind Axes in Guidance

The calculation of the angle of attack and the sideslip angle can be seen in equations (A.4.2) and (A.4.1).

Because the angle of attack indicates the actual lateral trajectory of the aircraft this angle is vital for accurate control when landing. Also this angle is an important tool to eliminate vertical stationary errors in levelled flight as seen in Figure 4.8. In addition α can help determine the stability of the aircraft; as W becomes increasingly large the aircraft will lose the ability to create lift and enters a stall.

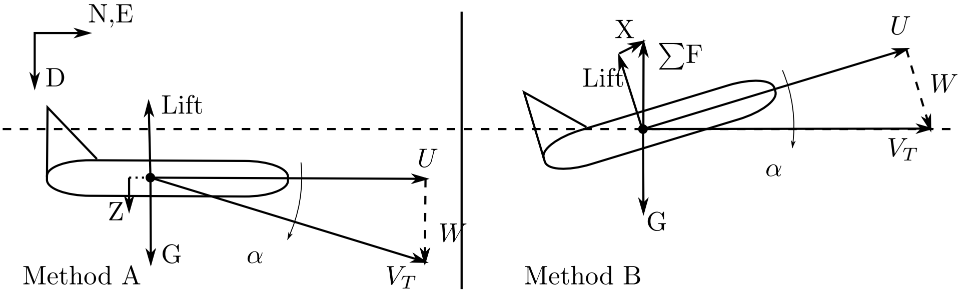


Figure 4.8: In method A the lack of lift induces a downward force ($Z = G - Lift$) throwing the aircraft off the path. In method B the angle of attack α is included in the steering law, by angling the force X slightly upwards. X is the thrust generated by the engine.

The usage of the sideslip angle is similar to the angle of attack: It can be used to compensate for wind in steady flight and reduce stationary errors as seen in Figure 4.9.

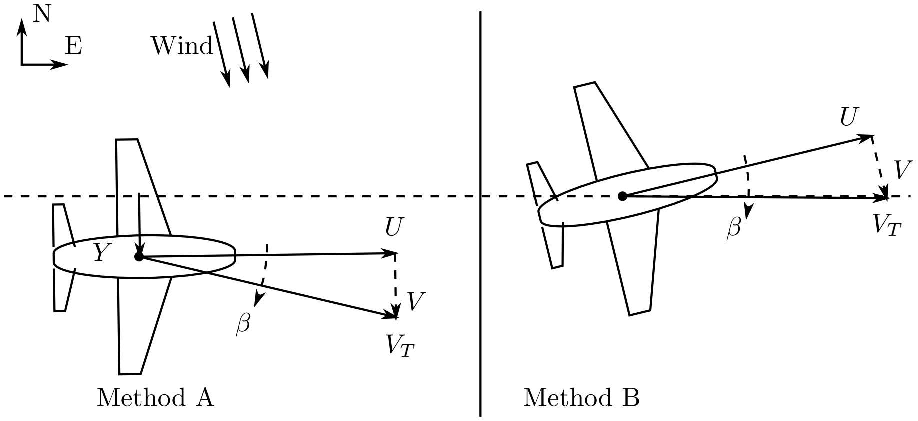


Figure 4.9: In method A the environmental force Y is inducing a stationary error forcing the aircraft off the path. In method B the sideslip angle β is included in the steering law.

Chapter 5

Implementation

This chapter contains the implementation of the automatic-flight control system (AFCS), the fault detection and isolation (FDI) system and the fault-tolerant control system (FTCS) in MATLAB Simulink. The nonlinear aircraft model of the Cessna 172SP from [Vistnes, 2012] is used as a simulation plant. Equations which are used in the MATLAB Simulink model is derived here as well as referenced to the MATLAB code.

5.1 Automatic Flight Control System (AFCS)

In this report the AFCS comprises of four main parts: The setpoint generator, the reference model, the kinematic controller and the velocity and rate controller as seen in Figure 5.1. The AFCS is constructed such that the inner loop stabilizes the velocities and rates. The outer loop consists of a reference signal generated by the guidance law in the setpoint generator. The reference signal is filtered through a reference model which creates the desired velocities and rates for the aircraft to follow. Stationary errors that occur from small errors in the rates propagating to the Euler angles are corrected by the kinematic controller. In this Section each block in Figure 5.1 will be described. Finally the tuning of the AFCS is elaborated in Section 5.1.4.

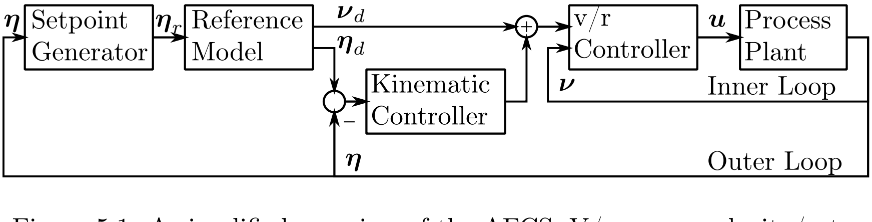


Figure 5.1: A simplified overview of the AFCS. V/r means velocity/rate.

5.1.1 Guidance and Navigation

The guidance and navigation part of the AFCS comprises of two Simulink blocks: The setpoint generator and the reference model as seen in Figure 5.1.

The navigation algorithm is part of the setpoint generator and has two main tasks: Creating the current path which the aircraft is to follow and switching paths when it is required. A path is based on two waypoints in the flight plan. The flight plan contains all waypoints in the flight mission, and these are appointed before the flight. The MATLAB code of the navigation part can be seen in the Appendix D.1. The first part of the code, from line 13, generates the path based on the variable, i. The second part, from line 54, increments i when a waypoint has been reached. This is carried out by Algorithm 1. In the path-frame $|Path|$, s and Δ all points in the same direction (x_{path}). The switch will take place when the steering point is on top of or past waypoint $\boldsymbol{p}_{k+1}$. This ensures smooth transitions when switching paths with respect to χ and ν. However this is not always the case as discussed in Section 6.2.2.

```
if $|Path| \leq s + \Delta$ then
    Switching waypoints by incrementing $i$
end
```

Algorithm 1: An algorithm for switching waypoints. s and Δ are along-track and lookahead distance respectively, defined in Section 4.5.1

The purpose of the guidance part of the setpoint generator is to designate the aircraft's desired trajectory and speed. The desired trajectory consists of two angles χ (the lateral trajectory) and ν (the longitudinal trajectory), and are determined by using the lookahead-based steering (see Section 4.5.1). The steering laws (4.5.10) are

$$\chi(e) \triangleq \chi_p + \chi_r(e) + \beta \qquad \nu(h) \triangleq \nu_p + \nu_r(h) + \alpha \tag{5.1.1}$$

where χ_p and ν_p are the trajectories of the path, χ_r and ν_r are given by equations (4.5.11) and (4.5.12) and α and β are angle of attack and sideslip respectively. The MATLAB code can be seen in Appendix D.1. From line 19 the track errors are found by using (4.5.8). From line 35 the lookahead-based steering equations are implemented.

The reference model block contains the reference models in roll, pitch and yaw. These are meant to smooth the reference values and the roll command (see Section 5.1.3) such that they do not exceed the limitation of the aircraft. Figure 5.2 shows the Simulink diagram of the three reference models. Each contain a similar setup as seen in Section 4.4.

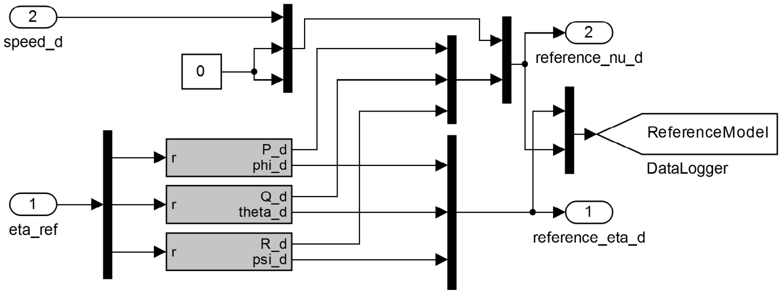


Figure 5.2: Simulink diagram of the reference model

5.1.2 Control System

The control system comprises of the velocity and rate controller and the kinematic controller. As will become clear in the proceeding section the velocity and rate controller does not catch the exact dynamics of the aircraft. There are two reasons for this, where the first is that $\boldsymbol{\eta}$ is neglected and thus gravity is also neglected. The second reason is that the equilibrium conditions from the linearisation is not part of the control scheme. Both of these result in constant forces and moments acting on the aircraft which are not modelled. These forces add up and will cause drift. However, due to state feedback in the controller, the result will be stationary errors in the respective channels. Stationary errors in the Euler rates will propagate to large angle errors over time. This means that there should be an outer control loop that closes these stationary errors; hence a kinematic controller.

Velocity and Rate Control

The purpose of the velocity and rate controller is to primarily stabilize the system's rates but also to control the aircraft's speed. Because a linear aircraft model is available (Section 3), it is natural to choose model-based control. Two LQ controllers are implemented, one for the longitudinal channel and one for the lateral channel.

The longitudinal model is derived by inserting (3.5.15) in (3.4.4)

$$\boldsymbol{M}_{lo}\delta\dot{\boldsymbol{\nu}}_{lo} + \boldsymbol{N}_{lo}\delta\boldsymbol{\nu}_{lo} + \boldsymbol{G}_{lo}\delta\boldsymbol{\eta}_{lo} = \boldsymbol{N}_{F_{lo}}\delta\boldsymbol{\nu}_{lo} + \boldsymbol{B}_{lo}\delta\boldsymbol{u}_{lo} \tag{5.1.2}$$

$$\delta\dot{\boldsymbol{\nu}}_{lo} = \boldsymbol{M}_{lo}^{-1}\left[-\boldsymbol{G}_{lo}\delta\boldsymbol{\eta}_{lo} - (\boldsymbol{N}_{lo} - \boldsymbol{N}_{F_{lo}})\delta\boldsymbol{\nu}_{lo} + \boldsymbol{B}_{lo}\delta\boldsymbol{u}_{lo}\right] \tag{5.1.3}$$

To get (5.1.3) on state space form, (4.2.1), the term $\boldsymbol{G}_{lo}\delta\boldsymbol{\eta}_{lo}$ has to be neglected. (5.1.3) is now rewritten:

$$\delta\dot{\boldsymbol{\nu}}_{lo} = \boldsymbol{M}_{lo}^{-1}\left[-(\boldsymbol{N}_{lo} - \boldsymbol{N}_{F_{lo}})\delta\boldsymbol{\nu}_{lo} + \boldsymbol{B}_{lo}\delta\boldsymbol{u}_{lo}\right] \tag{5.1.4}$$

$$\delta\dot{\boldsymbol{x}}_{lo} = \boldsymbol{A}_{lo}\delta\boldsymbol{x}_{lo} + \boldsymbol{B}_{lo}\delta\boldsymbol{u}_{lo} \tag{5.1.5}$$

where $\delta\boldsymbol{x}_{lo} = \delta\boldsymbol{\nu}_{lo} = [u, \alpha, q]^T$ and

$$\boldsymbol{A}_{lo} = -\boldsymbol{M}_{lo}^{-1}(\boldsymbol{N}_{lo} - \boldsymbol{N}_{F_{lo}}) \quad \text{and} \quad \boldsymbol{B}_{lo} = \boldsymbol{M}_{lo}^{-1}\boldsymbol{B}_{lo}$$

The lateral model: Similar to the above derivation the lateral model is found from equations (3.5.15) and (3.4.5).

$$\delta\dot{\boldsymbol{\nu}}_{la} = \boldsymbol{M}_{la}^{-1}\left[-(\boldsymbol{N}_{la} - \boldsymbol{N}_{F_{la}})\delta\boldsymbol{\nu}_{la} + \boldsymbol{B}_{la}\delta\boldsymbol{u}_{la}\right] \tag{5.1.6}$$

$$\delta\dot{\boldsymbol{x}}_{la} = \boldsymbol{A}_{la}\delta\boldsymbol{x}_{la} + \boldsymbol{B}_{la}\delta\boldsymbol{u}_{la} \tag{5.1.7}$$

where $\delta\boldsymbol{x}_{la} = \delta\boldsymbol{\nu}_{la} = [\beta, p, r]^T$ and

$$\boldsymbol{A}_{la} = -\boldsymbol{M}_{la}^{-1}(\boldsymbol{N}_{la} - \boldsymbol{N}_{F_{la}}) \quad \text{and} \quad \boldsymbol{B}_{la} = \boldsymbol{M}_{la}^{-1}\boldsymbol{B}_{la}$$

The reference model is made for the complete system in Section 5.1.1. From (4.2.3) $\boldsymbol{y}_{d_{lo}}$ and $\boldsymbol{y}_{d_{la}}$ can be calculated using respectively $\boldsymbol{C}_{lo}$ and $\boldsymbol{C}_{la}$:

$$\boldsymbol{y}_{d_{lo}} = \boldsymbol{C}_{lo}\boldsymbol{x}_d = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u_d \\ \beta_d \\ \alpha_d \\ p_d \\ q_d \\ r_d \end{bmatrix} \tag{5.1.8}$$

$$\boldsymbol{y}_{d_{la}} = \boldsymbol{C}_{la}\boldsymbol{x}_d = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_d \\ \beta_d \\ \alpha_d \\ p_d \\ q_d \\ r_d \end{bmatrix} \tag{5.1.9}$$

The control equations: The derivation of the control equations is similar for the longitudinal and lateral channels. The subscripts $_{lo}$ and $_{la}$ will therefore be omitted. The system equations above (5.1.5) and (5.1.7) are actually derived from the perturbation equations and not the total system. With this in mind the control law from Section 4.2, (4.2.9), can be rewritten as:

$$\delta\boldsymbol{u} = \boldsymbol{G}_1\delta\boldsymbol{x} + \boldsymbol{G}_2\delta\boldsymbol{y}_d \tag{5.1.10}$$

where $\delta\boldsymbol{y}_d \triangleq \boldsymbol{y}_d - \boldsymbol{C}\boldsymbol{x}_0$. As $\boldsymbol{u} = \boldsymbol{u}_0 + \delta\boldsymbol{u}$:

$$\boldsymbol{u} = \boldsymbol{u}_0 + \boldsymbol{G}_1\delta\boldsymbol{x} + \boldsymbol{G}_2\delta\boldsymbol{y}_d \tag{5.1.11}$$

As $\boldsymbol{x}$ is measured from the process plant it is more convenient to express this equation as:

$$\boldsymbol{u} = \boldsymbol{u}_0 + \boldsymbol{G}_1(\boldsymbol{x} - \boldsymbol{x}_0) + \boldsymbol{G}_2(\boldsymbol{y}_d - \boldsymbol{C}\boldsymbol{x}_0) \tag{5.1.12}$$

$\boldsymbol{G}_1$ and $\boldsymbol{G}_2$ are found by using (4.2.10) and the MATLAB function *lqr*.

The tuning matrices are defined below using Bryson's tuning rule, (4.2.15), but the values will be derived in Section 5.1.4

$$\boldsymbol{Q}_{lo} = \begin{bmatrix} \frac{1}{u^2_{max}} & 0 \\ 0 & \frac{1}{q^2_{max}} \end{bmatrix}, \boldsymbol{R}_{lo} = \begin{bmatrix} \frac{1}{\delta^2_{t_{max}}} & 0 & 0 \\ 0 & \frac{1}{\delta^2_{e_{max}}} & 0 \\ 0 & 0 & \frac{1}{\delta^2_{f_{max}}} \end{bmatrix}, \tag{5.1.13}$$

$$\boldsymbol{Q}_{la} = \begin{bmatrix} \frac{1}{p^2_{max}} & 0 \\ 0 & \frac{1}{r^2_{max}} \end{bmatrix}, \boldsymbol{R}_{la} = \begin{bmatrix} \frac{1}{\delta^2_{a_{max}}} & 0 \\ 0 & \frac{1}{\delta^2_{r_{max}}} \end{bmatrix}, \tag{5.1.14}$$

$$\tag{5.1.15}$$

The MATLAB code is found in Appendix D.2. Initiation of the controller is done up to line 54. When a fault has occurred and is detected the controller gains has to be calculated again. This is tested at line 65. From line 73 the system matrices and controller gains are calculated for the longitudinal and lateral model. The control signal is calculated at line 148.

Kinematic controller

The kinematic controller consists of three PI-controllers for roll, pitch and yaw as seen in Figure 5.3. These are developed with the same structure as in Section 4.3. The MATLAB code can be seen in Appendix D.3.

$$\tilde{\boldsymbol{\eta}} = \begin{bmatrix} \phi_d - \phi \\ \theta_d - \theta \\ \psi_d - \psi \end{bmatrix} \tag{5.1.16}$$

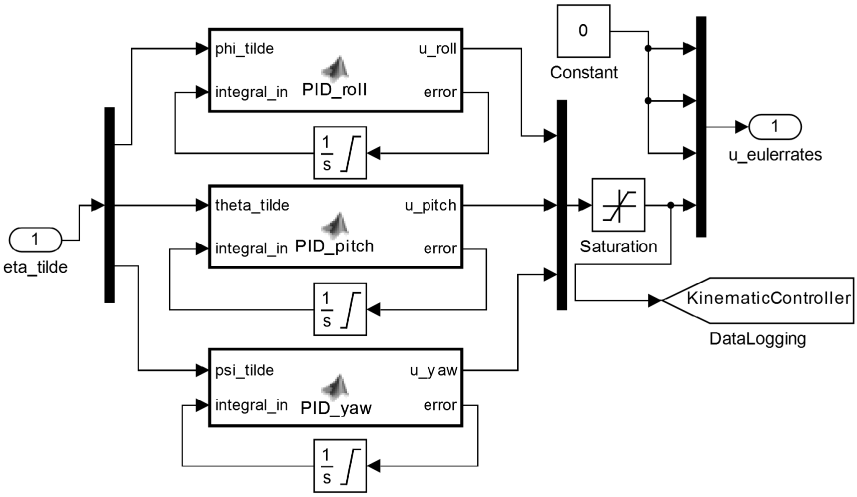


Figure 5.3: Simulink diagram of the Kinematic controller

5.1.3 Banking

A banking turn is a fundamental manoeuvre to change heading. To perform the manoeuvre the aircraft is banked, giving it a roll angle different from zero. The lift generated by the wings are directed away from the vertical axis, giving the aircraft a force sideways, Y as seen in Figure 5.4.

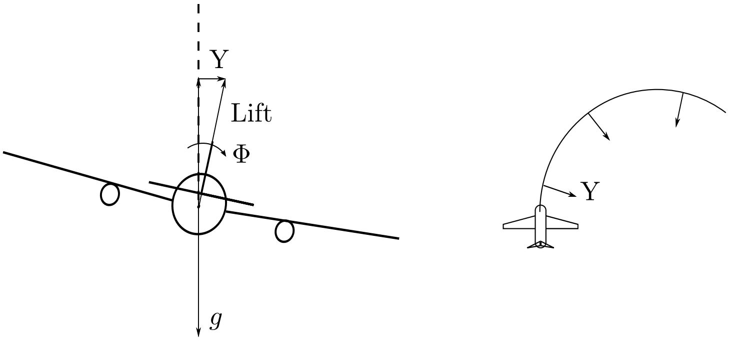


Figure 5.4: A bank turn generates a force Y sideways creating a steady turn.

From [McLean, 1990] yaw rate can be expressed as a function of roll such that:

$$r = \frac{g}{U_0} \sin \phi \tag{5.1.17}$$

where g is the gravitational constant and U_0 is the aircraft's speed.

When *phi* $< 25°$ it is a good assumption that $\sin\phi \approx \phi$. (5.1.17) can therefore be rewritten for small angles as:

$$\phi_r = \frac{U_0}{g} r_d \tag{5.1.18}$$

ϕ_r is the reference value which are fed through the reference model and r_d is the desired value of yaw rate. The implementation of the banked turn can be seen in Figure D.2.

5.1.4 Tuning the AFCS

This section sums up the tuning values of the AFCS. They are obtained based on Chapters 6 and 6.2.

Guidance and Navigation

Steering laws: For the steering law the only tuning variable is R which is set to 1.3 km as seen in line 36 in Appendix D.1. When choosing this value the steering capabilities of the aircraft has to be regarded as becomes clear in Chapter 6.

Speed: From the Cessna website [Cessna Aircraft Company] the Cessna 172SP Skyhawk's maximum cruise speed is set to 124 ktas (knots true airspeed). The reference airspeed is then set below this value at 100 knots ≈ 50 m/s. This is seen in Appendix D.1 line 52.

The reference model has three tuning variables per Euler angle: ω_n, ζ and the velocity saturations. These have to be tuned such that the filtered signals don't exceed the aircraft's limitations. The variables are implemented in the MATLAB script *init_Control_Constants.m* which are found on the attached DVD described in Appendix C. The values can be seen in Table 5.1. Chapter 6 will elaborate these choices. Finding the velocity saturation in yaw was done by considering the maximum bank angle: From (5.1.17) $r(20°) = 3.8°/s$. The maximum yaw rate was set below this number.

Control

The velocity and rate controllers are tuned by the matrices $\boldsymbol{Q}_{lo}$, $\boldsymbol{R}_{lo}$, $\boldsymbol{Q}_{la}$ and $\boldsymbol{R}_{la}$. Because of the systems (5.1.5) and (5.1.7) the tuning is performed based

Table 5.1: Tuning constants for the reference model.

	ω	ζ	velocity Saturation
Roll	1.5	0.8	6 $°/s$
Pitch	2.5	0.8	4 $°/s$
Yaw	0.5	0.8	2.5 $°/s$

Table 5.2: Tuning values for the velocity and rate controller

Channel	State Variables	Control Variables
Longitudinal:	$u_{max} = 2$ $q_{max} = 0.1°$	$\delta_{t_{max}} = 60$ N $\delta_{e_{max}} = 0.7°$ $\delta_{f_{max}} = 1°$
Lateral:	$p_{max} = 1°$ $r_{max} = 40°$	$\delta_{a_{max}} = 2°$ $\delta_{r_{max}} = 1°$

Table 5.3: Eigenvalue of the control plant for the velocity and rate controller.

Longitudinal model		Lateral model	
u	-253	β	-183
α	-18.9	p	$-1.41 + 3.89i$
q	-0.05	r	$-1.41 - 3.89i$

on the perturbation model. This means that the values chosen for the tuning matrices are the maximum perturbation from the nominal values allowed. These values were found by prioritizing which states are most important. For example correct roll rate is much more important than yaw rate. Similarly thrust command can be given much larger values than the elevator and flaps. These values are implemented in Appendix D.2.

The eigenvalues of system (4.2.14) are calculated for the longitudinal and lateral channel the result can be seen in Table 5.3.

Kinematic controller consists of three PI-controllers for each Euler angle. Each of these comprises of three tuning variables: the proportional term, the integral term and the integral saturation term for anti-windup. It is important that the kinematic controller does not drown the reference signal. Therefore the strength of the reference signal has to be considered when tuning.

The tuning variables are found in Table 5.4 and are implemented in the MATLAB script *init_Control_Constants.m* which are found on the attached DVD described in Appendix C.

Table 5.4: Tuning variables for the kinematic controller's three PIs. Respectively the proportional term, the integral term and the integral saturation term.

	Roll controller	Pitch controller	Yaw controller
P	10	0.01	0.2
I	0	0.5	0.05
I_{sat}		0.01	0.1

5.2 Fault Detection and Isolation (FDI)

The fault detection and isolation consists of three main parts: The observer, the residual generator and the residual validator as seen in Figure 5.5. The purpose of the FDI is to detect that an actuator is faulty. The resulting vector $\boldsymbol{f}_{ac} \in \mathbb{R}^r$ where r is the number of actuators, contains information on which actuator is faulty and is zero when the system is fault-free.

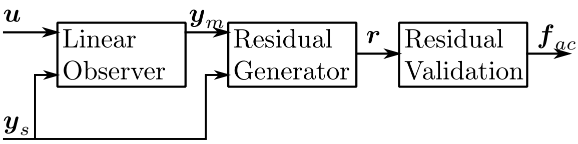


Figure 5.5: A scheme of the fault detection and isolation. $\boldsymbol{u}$ is the output of the controller, $\boldsymbol{y}_s$ is the system output, $\boldsymbol{y}_m$ is the observer output and $\boldsymbol{r}$ is the residual or parity vector. $\boldsymbol{f}_{ac}$ is a vector with each row representing an actuator which becomes one when a fault occur.

5.2.1 The Linear Observer

The Parity Space approach introduced in Section 2.2.1 relies on an accurate linear model of the system. The aircraft model is derived and linearised in Chapter 3. In this Section the linear model is implemented in code-form using forward and backward Euler integration.

Equation (3.1.2) can be rewritten as:

$$\dot{\boldsymbol{\nu}} = \boldsymbol{M}^{-1}\left[-\boldsymbol{G}\boldsymbol{\eta} - \boldsymbol{N}\boldsymbol{\nu}\right] + \boldsymbol{M}^{-1}\left[\boldsymbol{\tau}(\boldsymbol{x},\boldsymbol{u})\right] \tag{5.2.1}$$

The right side of (5.2.1) is divided in two and using the superposition principle these can be derived separately.

The linearisation of the first part is covered in Section 3.4, with the equilibrium conditions (5.2.7). The linearisation of $\boldsymbol{\tau}$ (the aerodynamic forces and moments) is performed in Section 3.6 using (4.1.2). The combined system can be expressed as:

$$\begin{aligned}\dot{\boldsymbol{\nu}} =& \boldsymbol{M}^{-1}\left[f(\boldsymbol{x}_0,\boldsymbol{u}_0) - \overbrace{\left.\frac{\partial f}{\partial \boldsymbol{\eta}}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]}}^{\boldsymbol{G}}\boldsymbol{\delta\eta} - \overbrace{\left.\frac{\partial f}{\partial \boldsymbol{\nu}}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]}}^{\boldsymbol{N}}\boldsymbol{\delta\nu}\right] + && (5.2.2)\\ &\boldsymbol{M}^{-1}\left[\boldsymbol{\tau}(\boldsymbol{x}_0,\boldsymbol{u}_0) + \overbrace{\left.\frac{\partial \boldsymbol{\tau}}{\partial \boldsymbol{\nu}}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]}}^{\boldsymbol{N}_F}\boldsymbol{\delta\nu} + \overbrace{\left.\frac{\partial \boldsymbol{\tau}}{\partial \boldsymbol{u}}\right|_{[\boldsymbol{x}_0,\boldsymbol{u}_0]}}^{\boldsymbol{B}}\boldsymbol{\delta u}\right] && (5.2.3)\end{aligned}$$

where

$$\boldsymbol{x} = \begin{bmatrix}\boldsymbol{\eta}\\ \boldsymbol{\nu}\end{bmatrix} = \underbrace{\begin{bmatrix}\boldsymbol{\eta}_0\\ \boldsymbol{\nu}_0\end{bmatrix}}_{\boldsymbol{x}_0} + \underbrace{\begin{bmatrix}\boldsymbol{\delta\eta}\\ \boldsymbol{\delta\nu}\end{bmatrix}}_{\boldsymbol{\delta x}} \quad \text{and} \quad \boldsymbol{u} = \boldsymbol{u}_0 + \boldsymbol{\delta u} \tag{5.2.4}$$

On the form of (3.1.2) and (3.5.1) the resulting system can be written as:

$$\begin{aligned}\dot{\boldsymbol{\eta}} =& \boldsymbol{J}_{\boldsymbol{\Theta}}(\boldsymbol{\eta}_0)\boldsymbol{\nu}_0 + \boldsymbol{J}_{\boldsymbol{\Theta}}(\boldsymbol{\eta}_0)\boldsymbol{\delta\nu}\\ \dot{\boldsymbol{\nu}} =& \boldsymbol{M}^{-1}\left[\boldsymbol{F}_0(\boldsymbol{x}_0,\boldsymbol{u}_0) - \boldsymbol{G}\boldsymbol{\delta\eta} - (\boldsymbol{N}-\boldsymbol{N}_F)\boldsymbol{\delta\nu} + \boldsymbol{B}\boldsymbol{\delta u}\right]\end{aligned} \tag{5.2.5}$$

where

$$\boldsymbol{F}_0(\boldsymbol{x}_0,\boldsymbol{u}_0) = f(\boldsymbol{x}_0,\boldsymbol{u}_0) + \boldsymbol{\tau}(\boldsymbol{x}_0,\boldsymbol{u}_0) \tag{5.2.6}$$

Equilibrium Conditions

From Section 3.4 the equilibrium conditions $\boldsymbol{F}_0(\boldsymbol{x}_0, \boldsymbol{u}_0)$ are derived: Note that $I_{xz} = 0$ (Appendix B).

$$\begin{aligned}
\dot{U}_0 &= -Q_0 W_0 + R_0 V_0 - g\sin(\Theta_0) + \frac{1}{m} X_0 \\
\dot{\beta}_0 &= \frac{1}{V_T}\left[-U_0 R_0 + W_0 P_0 + g\cos(\Theta_0)\sin(\Phi_0) + \frac{1}{m} Y_0\right] \\
\dot{\alpha}_0 &= \frac{1}{V_T}\left[-V_0 P_0 + Q_0 U_0 + g\cos(\Theta_0)\cos(\Phi_0) + \frac{1}{m} Z_0\right] \\
\dot{P}_0 &= \frac{1}{I_x}\left[-(I_z - I_y) Q_0 R_0 + L_0\right] \\
\dot{Q}_0 &= \frac{1}{I_y}\left[-(I_x - I_z) P_0 R_0 + M_0\right] \\
\dot{R}_0 &= \frac{1}{I_z}\left[-(I_y - I_x) P_0 Q_0 + N_0\right]
\end{aligned} \tag{5.2.7}$$

X_0, Y_0, Z_0, L_0, M_0 and N_0 are found from (3.6.3).

Forward and Backward Euler Integration

From [Fossen, 2011a] the discrete-time equations for nonlinear systems are:

$$\boldsymbol{\nu}(k+1) = \boldsymbol{\nu}(k) + h\dot{\boldsymbol{\nu}}(k) \tag{5.2.8}$$

$$\boldsymbol{\eta}(k+1) = \boldsymbol{\eta}(k) + h\left[\boldsymbol{J}_{\boldsymbol{\Theta}}(\boldsymbol{\eta}(k))\boldsymbol{\nu}(k+1)\right] \tag{5.2.9}$$

where h is the sampling interval given by (5.2.14). However to be able to apply Equations (5.2.8) and (5.2.9) to the linearized system (5.2.5), the state and control vector (5.2.4) has to be inserted. The result is the Euler integration equations:

$$\begin{aligned}
\boldsymbol{\nu}(k+1) = &\boldsymbol{\nu}_0(k) + \boldsymbol{\delta\nu}(k) \\
&+ h\boldsymbol{M}^{-1}\left[\boldsymbol{F}_0(\boldsymbol{x}_0, \boldsymbol{u}_0) - \boldsymbol{G}\boldsymbol{\delta\eta} - (\boldsymbol{N} - \boldsymbol{N}_F)\boldsymbol{\delta\nu} + \boldsymbol{B}\boldsymbol{\delta u}\right]
\end{aligned} \tag{5.2.10}$$

$$\boldsymbol{\eta}(k+1) = \boldsymbol{\eta}_0(k) + \boldsymbol{\delta\eta}(k) + h\left[\boldsymbol{J}_{\boldsymbol{\Theta}}(\boldsymbol{\eta}(k))\boldsymbol{\nu}(k+1)\right] \tag{5.2.11}$$

Model and Trajectory Update

Because the linear observer does not catch the nonlinear dynamics exactly the model will drift with respect to the system. The solution to this problem is to update the state values (5.2.11) and (5.2.10) periodically. The nominal part of the state is constant and therefore it is the perturbation part which is updated. From (5.2.4) where $\boldsymbol{x}$ now is the measurements from the system $\boldsymbol{x}_s$:

$$\boldsymbol{\delta x} = \boldsymbol{x}_s - \boldsymbol{x}_0 \tag{5.2.12}$$

$$\tag{5.2.13}$$

Table 5.5: Initial trajectory values for the model.

State	value
U_0	51.4 m/s
α_0	-0.0121 [rad]
β_0	0.0204 [rad]
δ_{T_0}	750 N

$\boldsymbol{\delta x}$, $\boldsymbol{x}_0$ are the model states.

The model's trajectory is given by the nominal values: $\boldsymbol{x}_0$ and $\boldsymbol{u}_0$. These also has to be updated to complement the desired trajectory of the aircraft. For instance the aircraft require some thrust to keep the aircraft at 100 kn. The initial values can be found in Table 5.5.

Implementation

The MATLAB code of the linear observer can be seen in the Appendix D.4. Initiation of the code is performed above line 56. The current step is updated at line 56. The logic after line 67 decides when the linearisation is performed. Linearisation is performed after initiation and after that given by a period of the constant *update_interval*. Trajectory update and model update are specified by the simulation scenario and is therefore separated. Further the model matrices are found at line 122. The integration equations (5.2.11) and (5.2.10) are found at line 156 and line 170.

The sampling interval used is given by (5.2.14). This is because the solver used in MATLAB is ode 4 (Runge-Kutta 4). This means that each script is run through 4 times per step.

$$h = \text{step-size} \times \frac{1}{4} \tag{5.2.14}$$

5.2.2 Residual Generator

As presented in Section 2.4, the faults will be applied to the ailerons and elevator. These control surfaces control the aircraft's roll and pitch and therefore these channels will be monitored for faults. Both the angle and the rates will be considered.

(2.2.10) defines the residual as:

$$\boldsymbol{r}(z) \triangleq \boldsymbol{H}_u(z)\boldsymbol{u}(z) + \boldsymbol{H}_y(z)\boldsymbol{y}_s(z) \tag{5.2.15}$$

Inserting (2.2.12) the residual becomes:

$$\boldsymbol{r}(z) = -\boldsymbol{H}_y(z)\boldsymbol{G}_u(z)\boldsymbol{u}(z) + \boldsymbol{H}_y(z)\boldsymbol{y}_s(z) \tag{5.2.16}$$

Further defining the model output as $\boldsymbol{y}_m = \boldsymbol{G}_u(z)\boldsymbol{u}$ the residual can be written on state space form as:

$$\boldsymbol{r}(t) = -\boldsymbol{H}_y\boldsymbol{y}_m(t) + \boldsymbol{H}_y\boldsymbol{y}_s(t) \tag{5.2.17}$$

Table 5.6: Threshold levels for variance testing...

state	threshold
roll (ϕ)	$1e-5$
pitch (θ)	$1e-5$
roll rate (P)	$5e-7$
pitch rate (Q)	$5e-7$

$\boldsymbol{y}_m$ and $\boldsymbol{y}_s$ has the same states which is $[\boldsymbol{p}, \boldsymbol{\Theta}, \boldsymbol{v}, \boldsymbol{\omega}]^T$ according to Section A.2.

The MATLAB code can be seen in Appendix D.5 and is an implementation of (5.2.17). $\boldsymbol{H}_y$ is a selection matrix which selects the states ϕ, θ, P and Q such that:

$$\boldsymbol{H}_y = \begin{bmatrix} & 1 & 0 & 0 & & 0 & 0 & 0 \\ & 0 & 1 & 0 & & 0 & 0 & 0 \\ \mathbf{0}_{4\times3} & 0 & 0 & 0 & \mathbf{0}_{4\times3} & 1 & 0 & 0 \\ & 0 & 0 & 0 & & 0 & 1 & 0 \end{bmatrix} \tag{5.2.18}$$

5.2.3 Residual Validator

The validation of the residual is performed by variance testing. When a fault has occurred the dynamics is given by (2.2.17) which is different from zero. When the variance exceeds a predefined threshold a fault has presumably occurred.

From Section 2.2.2 the statistical methods are derived. The recursive equation for variance is given by (2.2.30) on a selected window. The MATLAB implementation is given in Appendix D.6. The variance is calculated in line 25 and the threshold test is performed in line 44 resulting in the faulty actuator vector, $\boldsymbol{f}_{ac} \in \mathbb{R}^r$. $\boldsymbol{f}_{ac_i}$ is 1 if actuator i is faulty.

Finding the threshold values are done in Chapter 7. This is done by regarding the fault-free system and comparing it to the faulty one. The values are found in Table 5.6.

5.3 Fault-Tolerant Linear Quadratic Control

In this section the implementation of the fault-tolerant linear quadratic control is performed. In this thesis the fault identification is not performed and therefore it is assumed that the fault is known.

The post-fault model is derived in Section 2.3 resulting in:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \sum_{i \in I_N} \boldsymbol{B}_i \boldsymbol{u}_i + \sum_{j \in I_F} \hat{\boldsymbol{B}}_j \boldsymbol{u}_j \tag{5.3.1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^r$. The fault modelled in this thesis is loss of effectiveness and is given by (2.1.1). The post-fault model can then by expressed as:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \sum_{i \in I_N} \boldsymbol{B}_i \boldsymbol{u}_i + \sum_{j \in I_F} \hat{\boldsymbol{B}}_j \theta_j \boldsymbol{u}_j \tag{5.3.2}$$

where θ_j is the effect-reduction factor and the effect-reduction vector is defined:

$$\boldsymbol{\theta} \triangleq [\theta_k, \theta_{k+1} \cdots \theta_r]^T \tag{5.3.3}$$

The faulty actuator vector $\boldsymbol{f}_{ac}$ is introduced in Section 5.2.3 and the effect-reduction matrix is defined:

$$\boldsymbol{\Gamma} \triangleq \boldsymbol{I} - \text{diag}(\boldsymbol{f}_{ac})\text{diag}(\boldsymbol{\theta}) \tag{5.3.4}$$

By defining the new post-fault control matrix

$$\boldsymbol{B}_f \triangleq \boldsymbol{B}\boldsymbol{\Gamma} \tag{5.3.5}$$

the post-fault model can be written as:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}_f\boldsymbol{u} \tag{5.3.6}$$

The implementation of the fault-tolerant control in MATLAB code can be seen in Appendix D.2 in line 102 and line 133 for the longitudinal and the lateral model respectively. The values of the vectors $Theta\_lo$ and $Theta\_la$ should preferably come from a parameter or state estimator. However, fault identification is not a part of this thesis and thus these values are fictional.

Chapter 6

Results and Discussion on the Automatic Flight Control System

In this chapter results and a discussion will be presented on the performance of the automatic flight control system (AFCS).

6.1 Results

A new notation is introduced in the following figures: Subscript r is the reference value, d is the desired value from the reference model, s is the value from the system.

6.1.1 Guidance Steering Law

The tuning variable, R, is given three different values according to Table 6.1 and Figure 6.1 shows the results of the simulations. Scenario 1 results in large deviations from the path and scenario 3 results in an overshoot. Scenario 2 results in neither.

Using the tuning variable found in scenario 2, the reference and desired values are plotted in red in Figure 6.2. Ψ_d is not able to keep up with the reference signal as it becomes linear in the periods 55-90 s and 125-160 s. In blue in Figure 6.2 $R = 1000$ and a clear overshoot of Ψ_d is seen in the periods 100-120 s and 170-200 s.

Table 6.1: The steering law is tuned with three different values.

Tuning scenario:	1	2	3
R [km]:	1.5	1.3	1

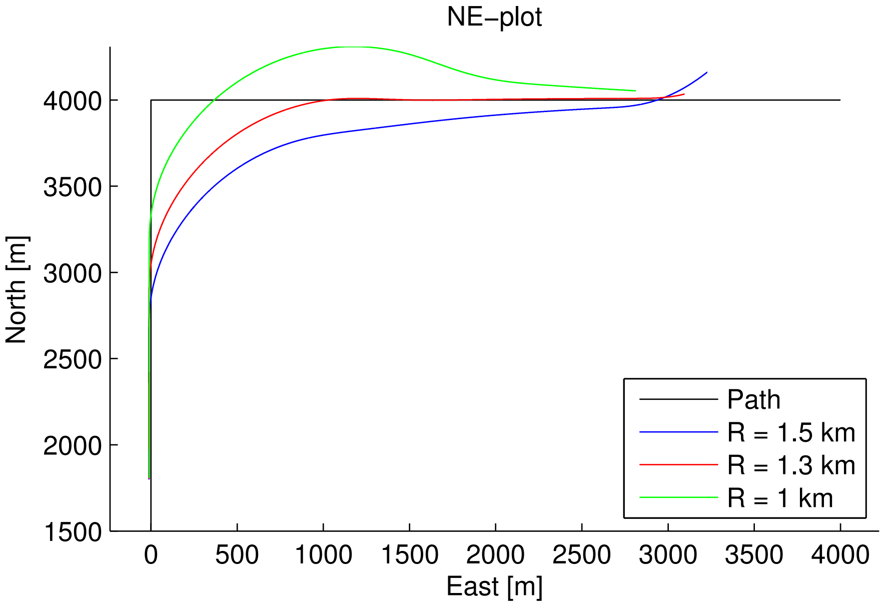


Figure 6.1: The steering law tuning variable R is given three different values.

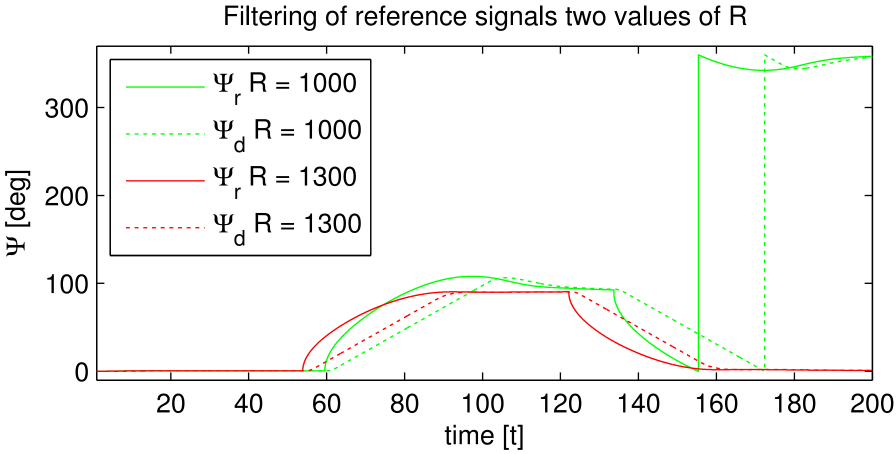


Figure 6.2: The reference values from the guidance law are filtered through the reference model with two values of R.

6.1.2 Navigation

Figure 6.3 shows the yaw response of Ψ_r and Ψ_d with two kinds of turns. The first turn has an angle of $>> 90$ and the second turn $<< 90$. The yaw response of turn one is very smooth with a slight stationary error. There is also a minimal deviation from the path. This is not the case for the second turn. First of all there is a jump in Ψ_r at 53 seconds. This results in a large gap between Ψ_r and Ψ_d. In the plot of the position of the aircraft one can see that the aircraft has experienced a large overshoot.

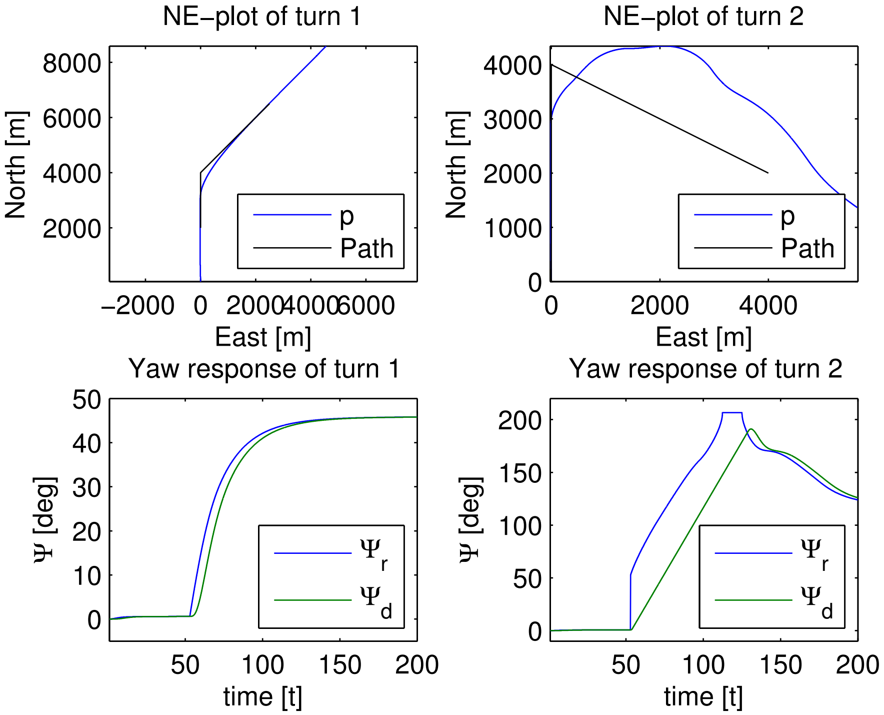


Figure 6.3: Yaw response of two different kinds of turns. p is the aircraft's position.

6.1.3 Reference Model

Steps are feed into the reference model of the three Euler angles. The desired values are plotted as well as the response of the aircraft. The reference model is tuned with the variables from Table 5.1.

In Figure 6.4 the reference model in roll is fed a step of 20°. The reference model in roll uses four seconds to adjust, and the aircraft follows the desired signal perfectly. The same can be said for the roll rate.

In Figure 6.5 the reference model in pitch is fed a step of 10°. The aircraft's pitch is adjusted in three seconds with perfect following of the desired values.

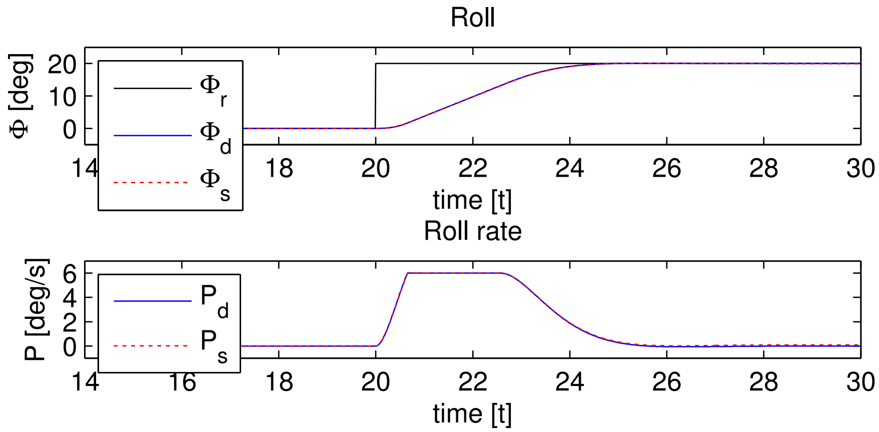


Figure 6.4: Reference model step in roll channel.

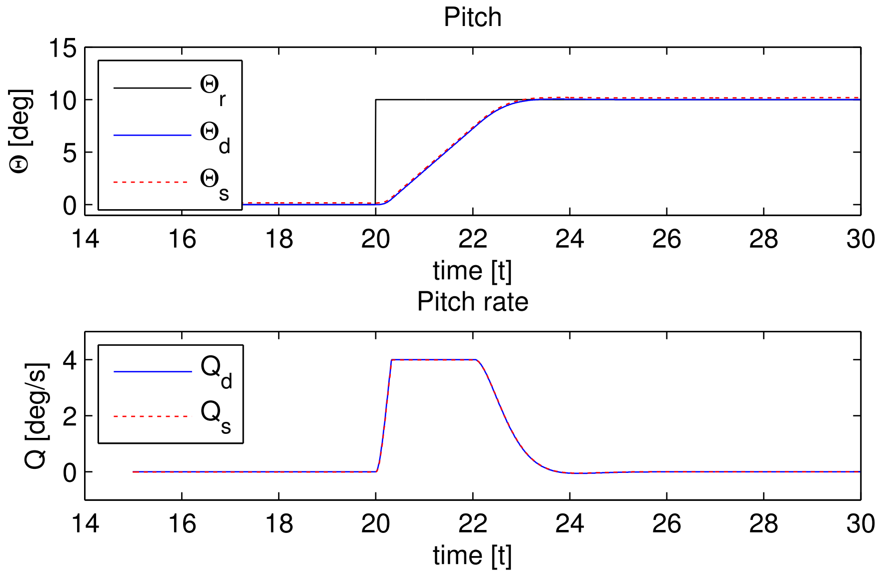


Figure 6.5: Reference model in pitch channel

Table 6.2: Tuning scenarios for velocity saturation in yaw for the reference model.

Tuning scenario:	1	2	3
Velocity saturation $[^\circ/s]$:	3	2.5	2

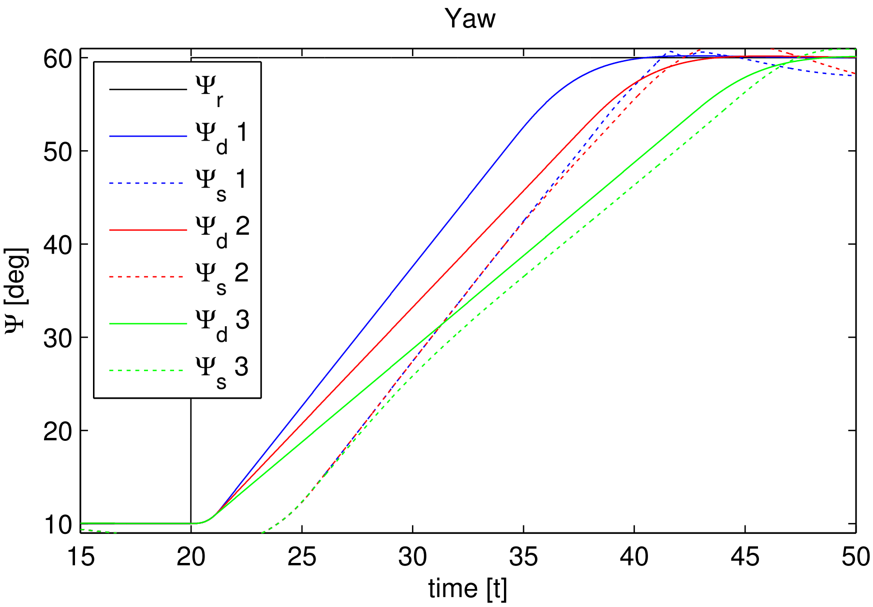


Figure 6.6: The response of the yaw angle for the reference model and the aircraft with three different velocity saturations.

Three tuning scenarios for velocity saturation is used for yaw as seen in Table 6.2. Scenario 1 and 3 represent the extremes and scenario 2 is from Table 5.1. Figure 6.6 shows the plot of the yaw angle as a response to the step. For scenario 1 the aircraft is not able to catch up with the reference signal. In scenario 2 and 3 the aircraft is able to do this however a stationary error is observed in both of these scenarios. The aircraft response in scenario 1 and 2 is similar, but are much steeper than in scenario 3. The response of the yaw rate for the three scenarios can be seen in Figure 6.7. In scenario 2 and 3 the aircraft is able to exceed the velocity saturation of the response model.

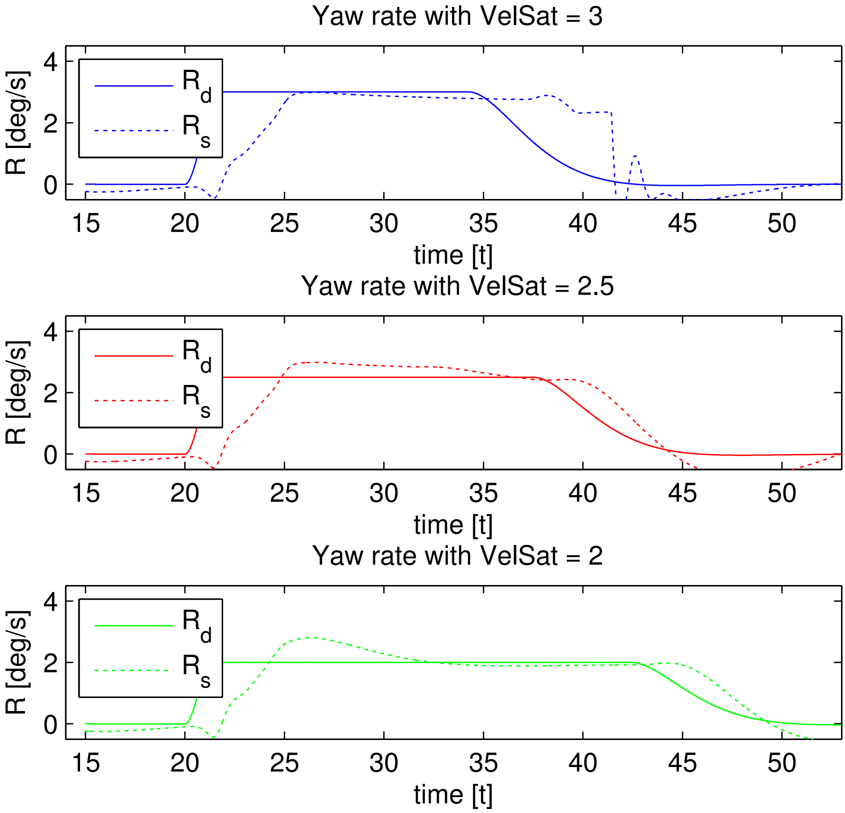


Figure 6.7: The response of the yaw rate for the reference model and the aircraft with three different velocity saturations.

6.1.4 Kinematic controller

Without the Kinematic Controller

Simulations of the control system without the kinematic controller is seen in figures 6.8 to 6.10.

One can clearly see that the Euler angles in Figure 6.8 are drifting from the desired values. Also oscillations are seen in the pitch plot. Figure 6.9 only show the 80 first seconds of the simulation. Stationary errors are seen in all three rates. Figure 6.10 shows the NE-plot of the simulation, the aircraft has large deviations from the path during the whole simulation.

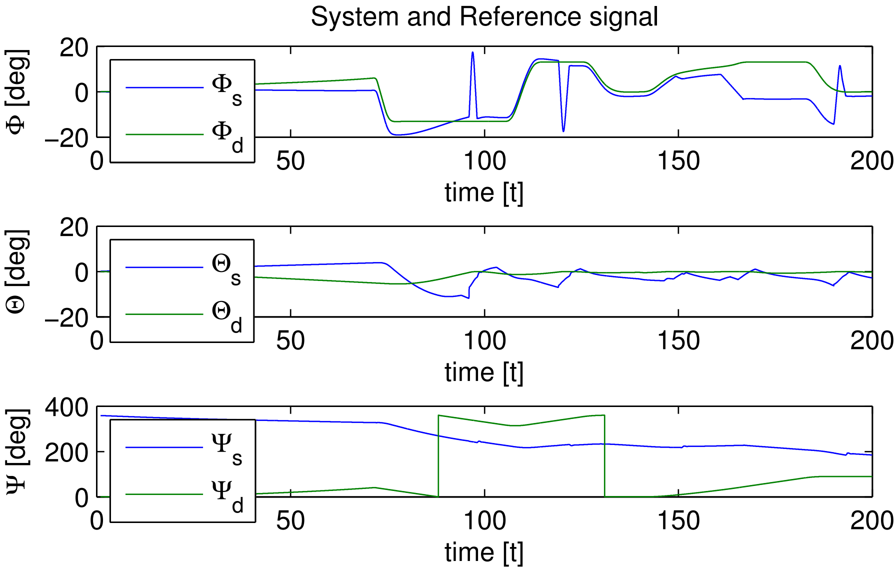


Figure 6.8: Response of the Euler angles without the kinematic controller.

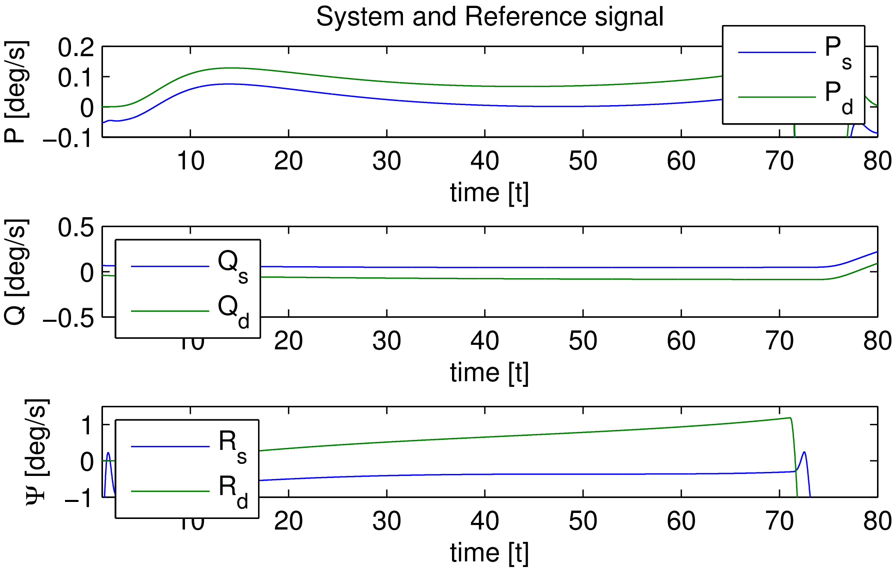


Figure 6.9: Response of the Euler rates without the kinematic controller.

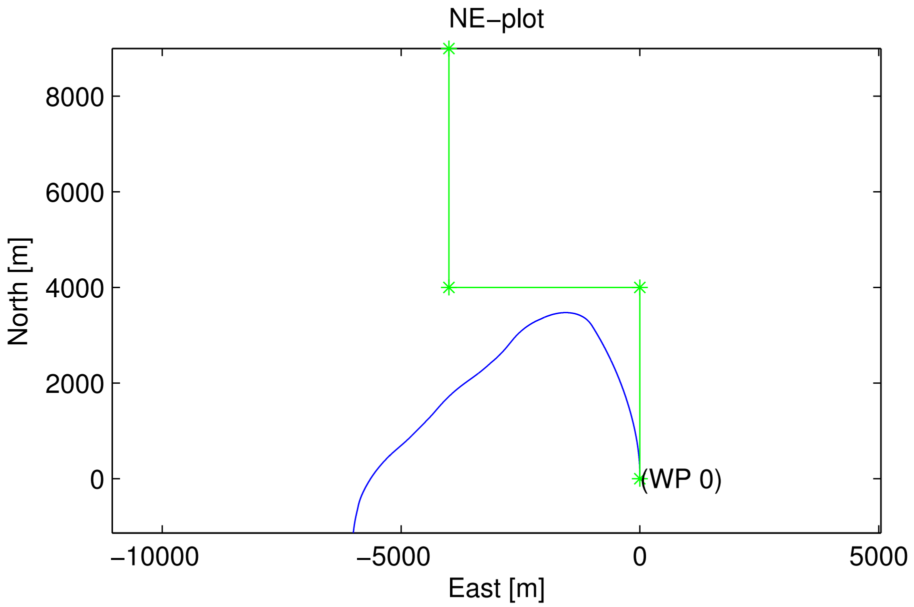


Figure 6.10: Response of the aircraft without the kinematic controller.

With the Kinematic Controller

Two different tuning configurations are tested for the kinematic controller in pitch with constants given by Table 6.3. Results from the first tuning scenario is given by the figures 6.11-6.14, and results from the second is given by the figures 6.16-6.20. A comparison of the pitch angles and the pitch rates of both scenarios is given in Figure 6.21.

Scenario 1: Figure 6.11 shows the control signal from the kinematic controller and the desired rates. The signal from the kinematic controller becomes dominating in Q at 60 seconds. From Figure 6.12 it can be verified that a turn is initiated at 54 seconds. The aircraft experiences major unwanted edgy oscillations in roll and pitch (both angles and rates) at 70-100 seconds. This is seen in figures 6.12 and 6.13.

Table 6.3: Two different tuning configurations for the kinematic controller in pitch.

Channel:	Pitch	
Scenario:	1	2
K_P	0.1	0.5
K_I	0.01	0.01
$saturation$	0.1	0.5

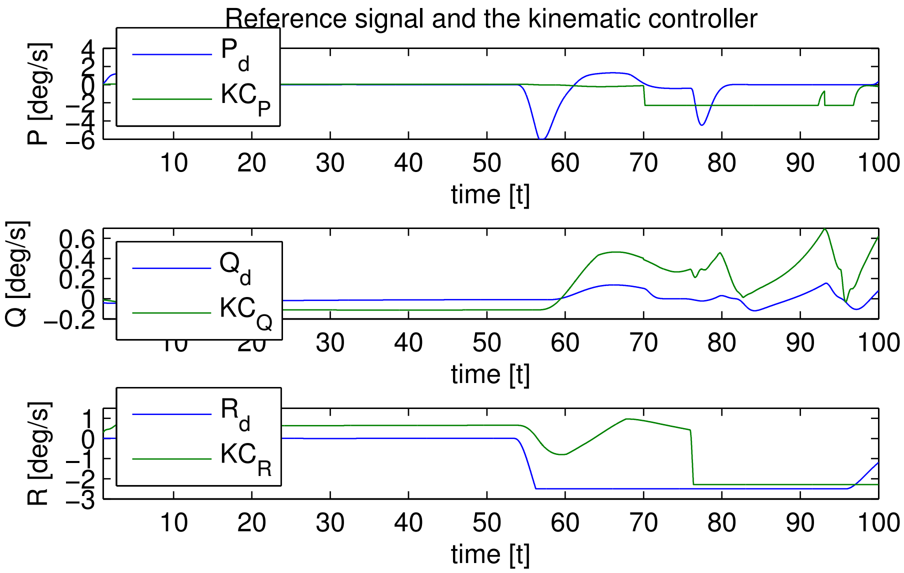


Figure 6.11: Tuning Scenario 1: The control signal from the kinematic controller (KC) and the desired rates.

Scenario 2: In Figure 6.16 the control signal from the kinematic controller is plotted alongside the desired rates. In roll rate the control signal is very week compared to P_d. The zoomed in plot, Figure 6.17, shows a slight stationary command of $0.05°/s$. The kinematic control signal in pitch rate and yaw rate is much stronger. In Figure 6.18 a large error in pitch is experienced at 60 s and 120 s. The same error is also seen in pitch rate in Figure 6.19.

Comparison: Figure 6.21 shows the comparison of the pitch channel in the two scenarios. It is clear that the tuning in scenario 2 results in better following of the pitch angle.

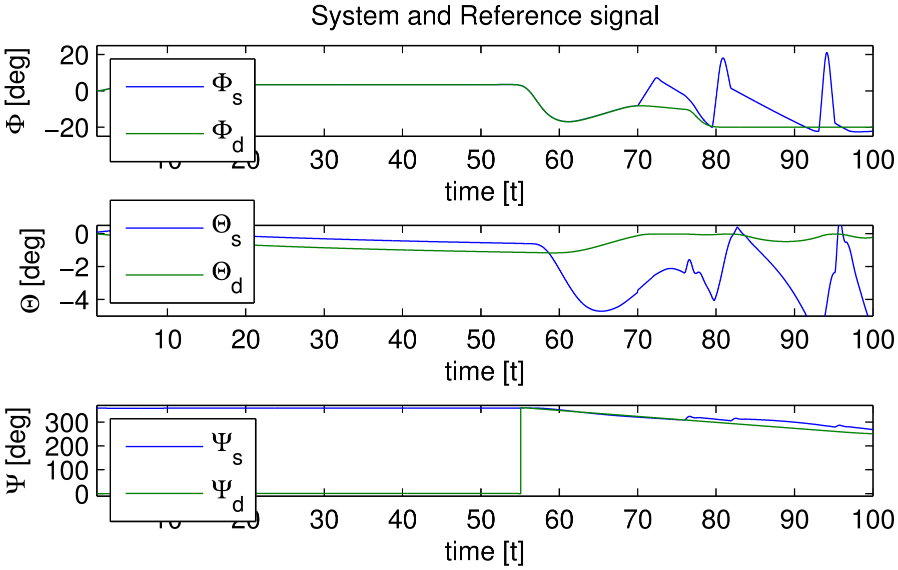


Figure 6.12: Tuning Scenario 1: Response of the aircraft with the kinematic controller.

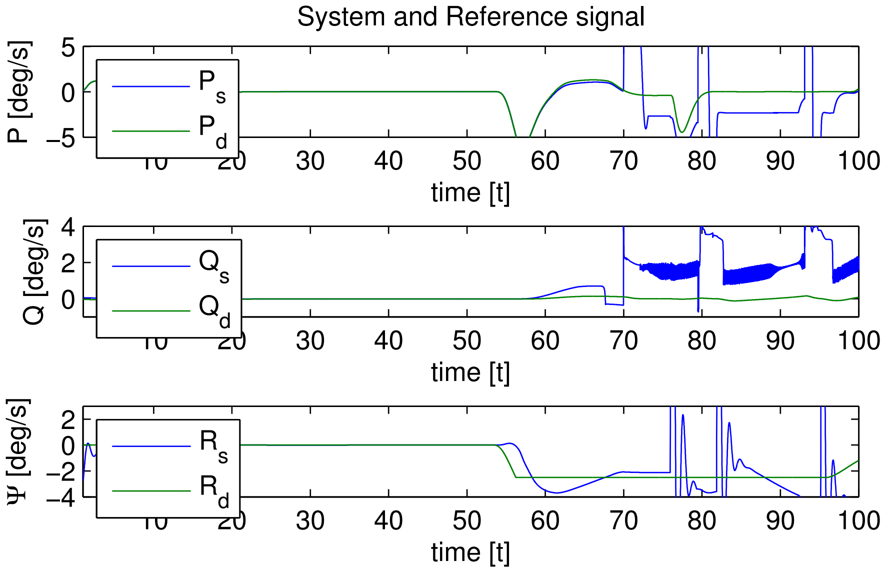


Figure 6.13: Tuning Scenario 1: Response of the aircraft with the kinematic controller.

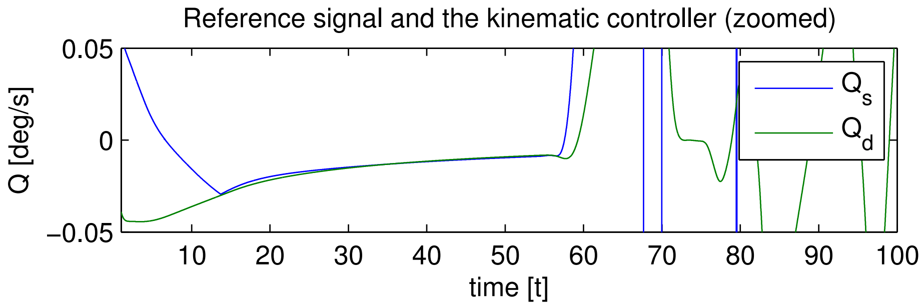


Figure 6.14: Tuning Scenario 1: Response of the aircraft with the kinematic controller.

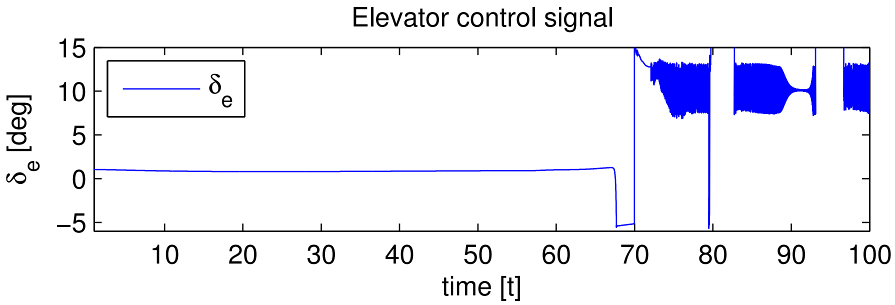


Figure 6.15: Tuning Scenario 1: Elevator control signal.

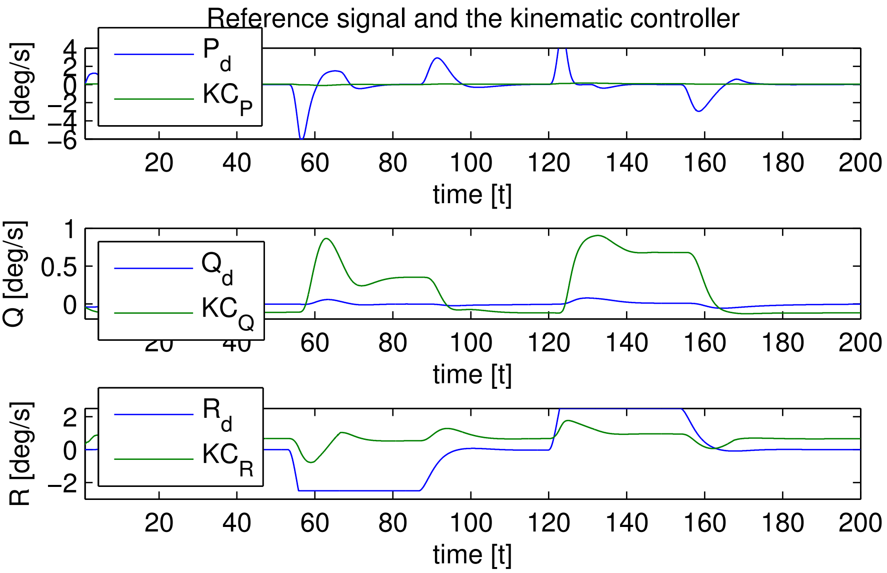


Figure 6.16: Tuning Scenario 2: The control signal from the kinematic controller (KC) and the desired rates.

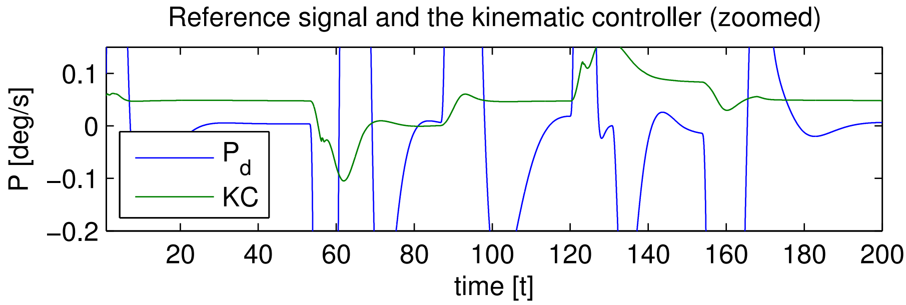


Figure 6.17: Tuning Scenario 2: The control signal from the kinematic controller (KC) and the desired rate in roll.

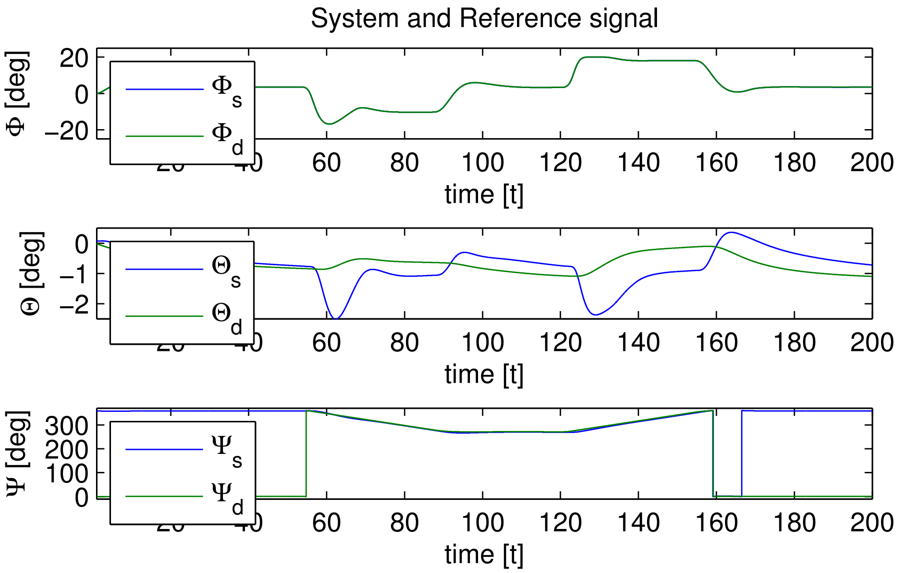


Figure 6.18: Tuning Scenario 2: Response of the aircraft with the kinematic controller.

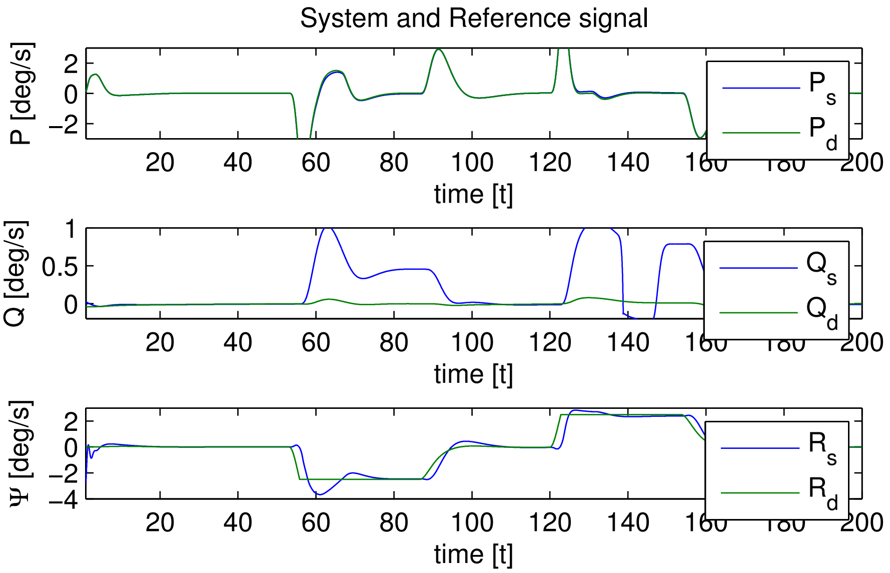


Figure 6.19: Tuning Scenario 2: Response of the aircraft with the kinematic controller.

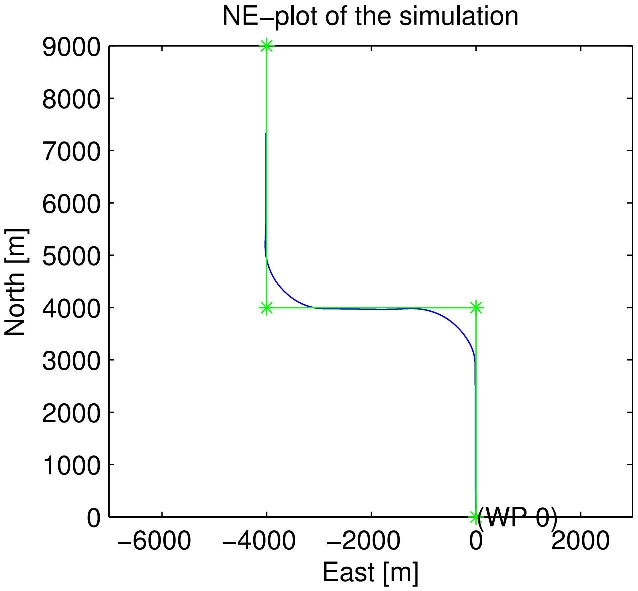


Figure 6.20: Tuning Scenario 2: Response of the aircraft with the kinematic controller.

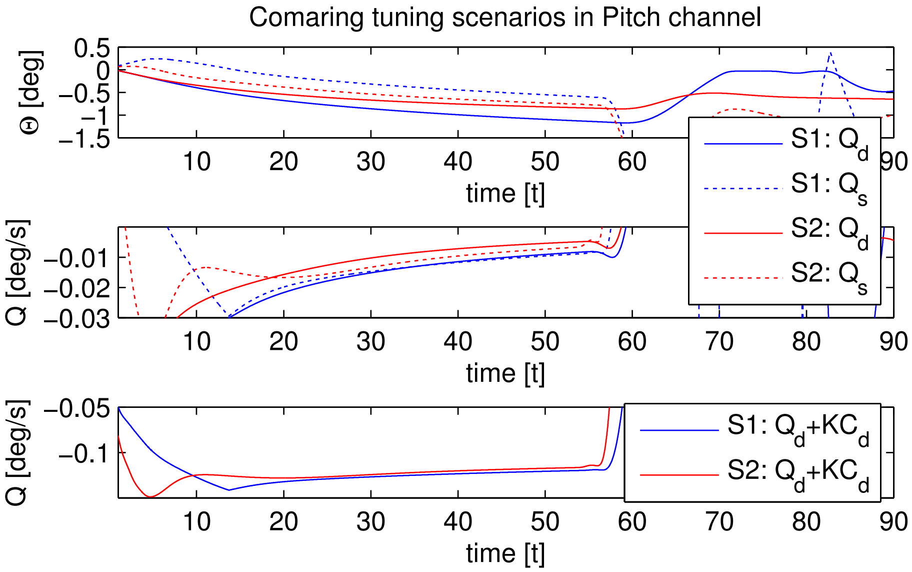


Figure 6.21: Comparing the tuning scenarios in pitch for both angles and rates.

6.1.5 Velocity and Rate controller

The velocity and rate controller is tested through a simulation with two turns as seen in Figure 6.22. The results are analysed in the longitudinal and lateral motions. It is important to note that the states and control outputs plotted in figures 6.23 to 6.26 are the perturbation. The nominal values are constant during this simulation and is seen in Table 6.4.

Longitudinal Channel: In Figure 6.23 it can be seen that the aircraft's forward velocity is about 2 m/s above the desired value. The pitch rate has a stationary error during the simulation. The control output in Figure 6.24 is mostly smooth, however some jumps can be seen.

Lateral Channel: p follows the reference perfectly in Figure 6.25. There is a small phase shift in r as well as a stationary error. The control output in Figure 6.26 show smooth graphs.

Table 6.4: The nominal state and control output values for the simulation. The remaining are set to zero.

state/control output	Nominal value
U_0	51.4 m/s
α_0	-0.7 °
δ_{T_0}	750 N

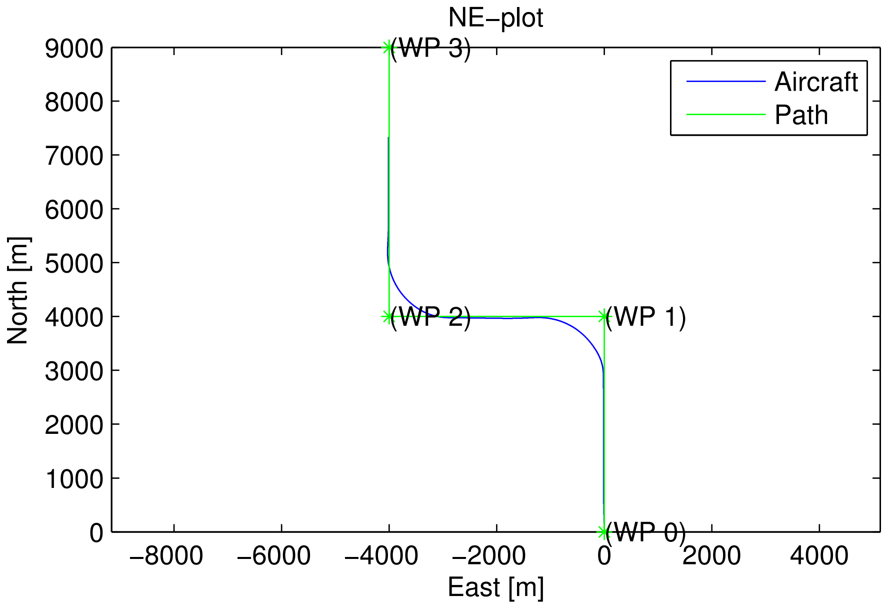


Figure 6.22: A north-east plot of the simulation.

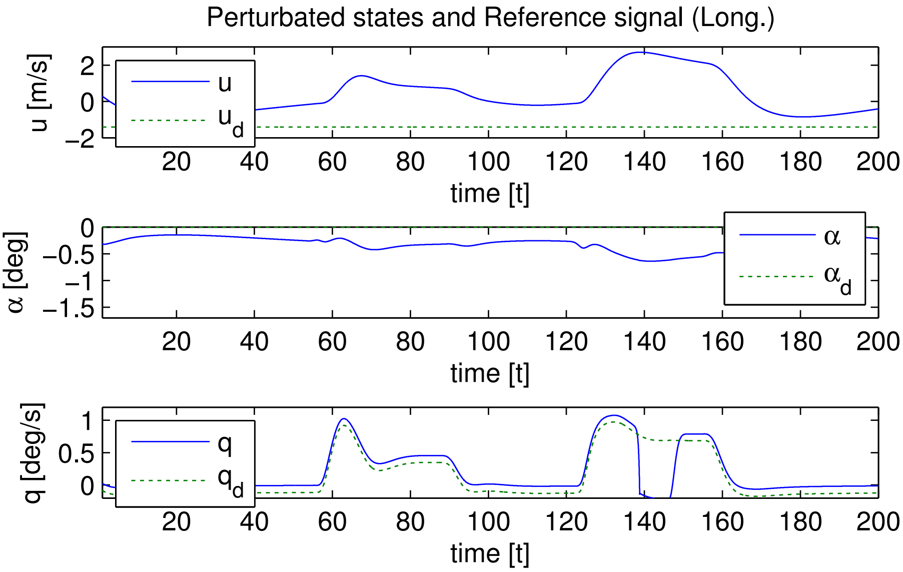


Figure 6.23: The states of the velocity and rate controller in the longitudinal channel.

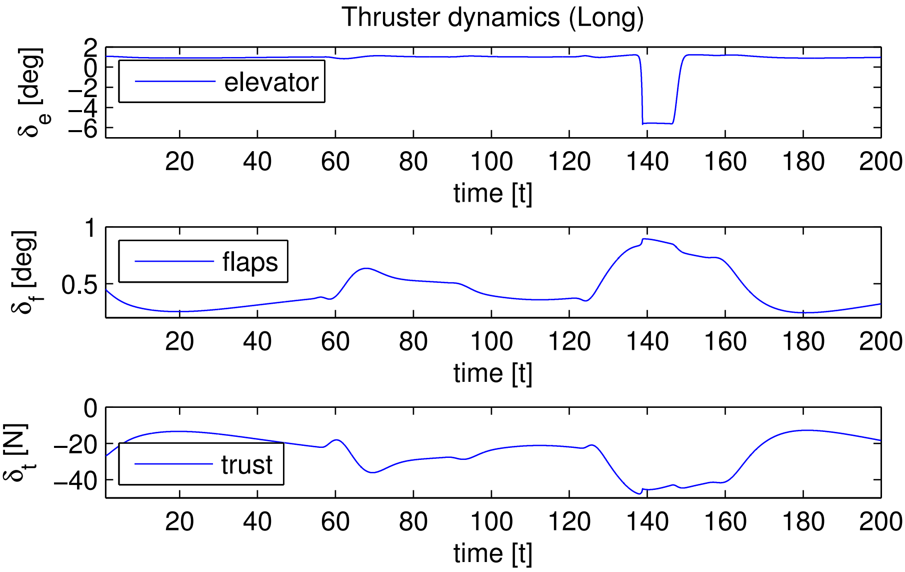


Figure 6.24: The thrust dynamics from the velocity and rate controller in the longitudinal channel.

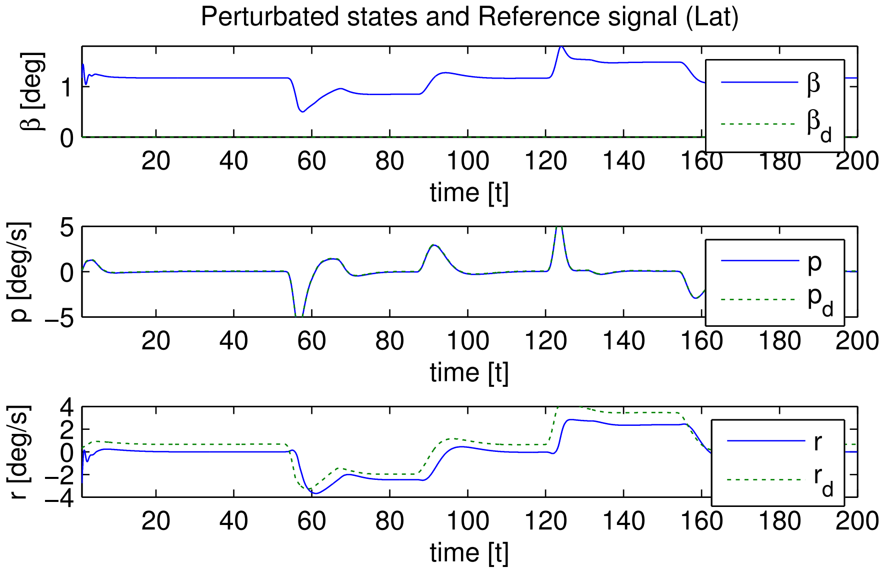


Figure 6.25: The states of the velocity and rate controller in the lateral channel.

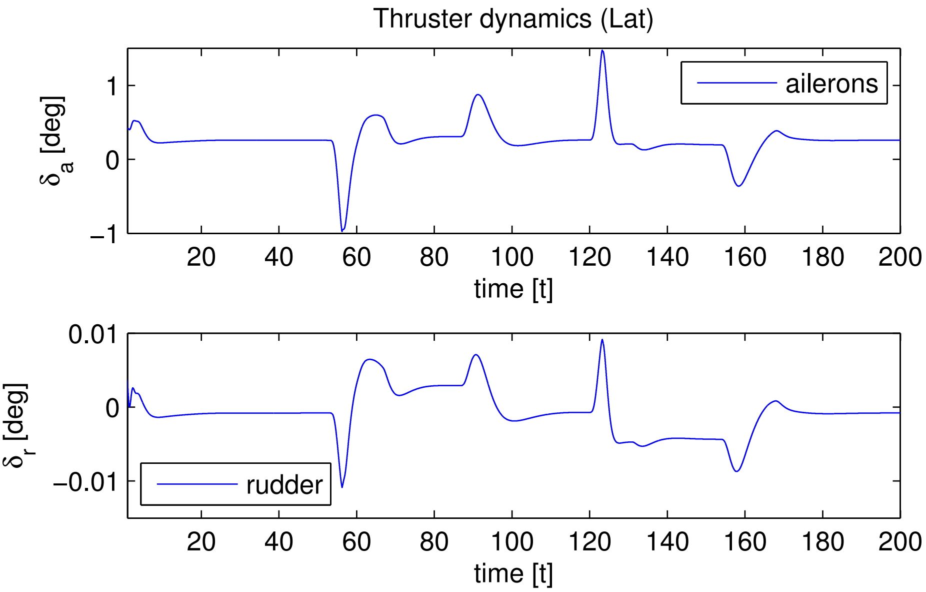


Figure 6.26: The thrust dynamics from the velocity and rate controller in the lateral channel.

6.2 Discussion

6.2.1 Guidance Law

The tuning variable R has to be tuned when considering the aircraft's turning rate. If this is too small and the aircraft will not have enough room for making a clean turn as seen when $R = 1000$ in Figure 6.1. The result is an overshoot of the second path as also becomes evident in Figure 6.2 from 90 to 110 s. When R is too big (scenario 1) the aircraft will suffer from large path deviations. $R = 1300$ was found to be perfect as seen in Figure 6.1. The velocity saturation for Ψ_d is demonstrated; as Ψ_d becomes linear in the period 60-100 s and 140-180 s of Figure 6.2.

6.2.2 Navigation

Considering Figure 6.3 containing turn two, the step causes Ψ_d to experience a large error with respect to Ψ_r. In terms of position the large jump causes an overshoot because the average rate of Ψ_r is much larger than the velocity saturation of Ψ_d. The reason for this jump lies in the waypoint switching algorithm 1. As long as the angle between LOS vector and the next path is larger than 90°, Ψ_r will not experience a jump. This is illustrated in Figure 6.27. Considering Figure 6.3 along with figures 6.1 (the red line) and 6.2 (the red line) it is clear that the flight plan should be constructed such that there are no turns which are less than 90°. This

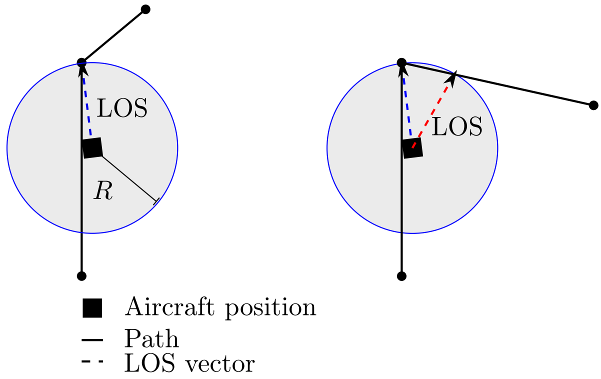


Figure 6.27: An illustration of the problem with the path switching algorithm. The blue dashed line is the line of sight (LOS) vector before path switch and the red dashed line is after. In the left part of the figure the red and blue dashed line are on top of each other.

is the only demand needed assuming that the cross-track error (e) is zero before the transition.

6.2.3 Reference Model

In Figure 6.6 the response of scenarios 1 and 2 are similar. As scenario 1 is much steeper than scenario 2 this indicates that the aircraft has met its limit. This is also supported by Figure 6.7 as the aircraft is able to go faster than the reference in scenarios 2 and 3 and not in scenario 1. Considering scenario 2, the aircraft is able to follow the reference as well as catching up to it, signifying this as best tuning scenario to choose.

6.2.4 Kinematic controller

Because the main controller of AFCS is a rate based controller, the Euler angles are not considered in the control law. The small stationary errors in rates in Figure 6.9, propagate to large errors in the Euler angles in Figure 6.8 and the position in Figure 6.10.

The first tuning scenario in pitch is much lower than the second. This becomes evident when comparing Q_d with the kinematic control signal in pitch (KC_P) in figures 6.11 and 6.16. The consequence of lower gains can be seen when comparing Figure 6.12 and Figure 6.18 where the stationary error is double for lower gains. Figure 6.21 shows that scenario 2 has a much closer following in the pitch angle than scenario 1.

As seen in Section 5.1.2 the kinematic controller alters the reference signals which are fed to the velocity controller. However, if the control signal from the kinematic controller becomes too large it will drown the dynamics of the reference signal. To be able to have low gains in the kinematic controller the velocity and rate controller must be accurate.

The edgy oscillations discovered for the first scenario is a result of large deviations in pitch rate around 70 seconds. Figure 6.15 shows a large jump at 68 s of the elevator control surface. These violent manoeuvres propagate to roll and cause unwanted oscillations in all channels.

However, the large deviation in pitch and pitch rate which occurs at 56 s is a result of the banking manoeuvre. This is also evident in figures 6.18 and 6.19. However in this scenario the kinematic controller issues a large control signal to counter act this motion as seen in Figure 6.16. It is therefore concluded that higher gains in pitch in the kinematic controller is necessary. Model couplings is discussed in Section 6.2.6.

6.2.5 Rate and Velocity Controller

The states in the longitudinal channel follow the references with a small stationary error in all channels as seen in Figure 6.23. In pitch rate it is the kinematic controller which "pulls" the rates down by tuning the reference down. The average speed of the aircraft is a couple of m/s too high. To adjust this the nominal value for thrust is set to 740 N for levelled flight. In the lateral model, Figure 6.25, p has perfect following. Yaw rate on the other hand suffers from phase lag. This is due to the fact that roll is calculated by yaw rate and therefore has to pass through the reference model two times before the aircraft reacts.

The thruster output in Figure 6.24 has some rapid movements during the simulation. This is due to the LQ optimal control that does not include constraints. However in reality thruster dynamics should be considered as movements of the control surfaces are restricted. The rapid movements could have been dampened by including thruster dynamics in the control plant and would help ensuring stability.

6.2.6 Model-based Control

It has been observed frequently that the banking motion of the aircraft affects the aircraft's pitch. This is best illustrated in Figure 6.18 at both 60 and 130 s. In this example the aircraft first initiates a left turn and then a right turn. In both cases the pitch drops 1 ° which is interpreted as the nose rising 1 °. In Figure 6.12 (with lower kinematic controller gains in pitch) this drop is -4 ° and a large deviation occurs in pitch rate during the drop. The controller corrects the rate error by a large elevator command which induces the consequent oscillations.

Several conclusions can be drawn by this incident. First of all the main reason for the oscillatory behaviour is due to the large errors in the controller. However, these could be managed if the controller did not respond with violent control outputs. This brings to light that thruster dynamics should have been included in the control plant model.

Additionally it was assumed that the system could be decoupled in longitudinal and lateral motions. It was presumed that feedback control would suffice to correct the resulting deviations. However this is not the case for the bank-to-turn manoeuvre mentioned above; thus for large roll angles the model in pitch is inaccurate. Another way of correcting this could be to include a feedforward control in the longitudinal channel which is activated as a turn is initiated.

Chapter 7
Case Studies

The fault-tolerant control system (FTCS) is tested with seven case studies which are presented below. A discussion and conclusion of the results are presented in the chapters 8 and 9.

Faults simulated are on the control surfaces and are reduction of effectiveness. These could represent a loss of hydraulic pressure in the actuators.

Case 1: Two Turns A simulation with two turns is performed. The performance of the FDI is analysed in a fault-free situation. Also, filtered and unfiltered residuals are compared.

Case 2: Two Turns with a Fault Running the same simulation as for case 1, a fault is simulated in the ailerons. The results from the FDI are analysed.

Case 3: Elevation regulator A simulation is performed with a change in elevation. Guidance using the angle of attack is presented, and the performance of the FDI is analysed.

Case 4: Elevation regulator with a Fault The same simulation as in case 3 is performed. A fault is simulated on the elevator and the results from the FDI are analysed.

Case 5: Multiple Faults The flight plan from case 3 and 4 are simulated with two faults. The fault occur with 20 seconds apart, first in the elevator and then in the ailerons. The results from the FDI are analysed.

Case 6: Different Magnitude of Faults Different magnitudes of fault is simulated on a small part of the flight plan from case study 1 and 2. An analysis is performed on the results.

Case 7: Fault Accommodation A fault is simulated and the performance of the control system is then compared with and without fault accommodation enabled.

7.1 Case Study 1: Two Turns

The flight plan of this case study can be seen in Figure 7.1 along with the aircraft's trajectory. The nominal values in the controller are constant during this simulation and is seen in Table 6.4. Figures 7.2-7.5 show the state of the system and compares them to the states from the observer. The residual and variance are presented in figures 7.6-7.7. A comparison is be made between the raw and the filtered values.

Figure 7.2 shows the Euler angles from the system and the observer during the simulation. There is a slight drift in the roll estimate during the period 70-80 s and 130-140 s. The pitch estimate is oscillating during the periods 60-80 s and 130-160. Figure 7.3 shows the Euler rates. Oscillations are present in the pitch rate estimate during 60-100 s and 120-160 s. A small drift in yaw rate is also seen in the periods 60-80 s and 120-160. According to Figure 7.4 a small drift is present in roll rate during the periods 60-80 s and 130-155 s. Figure 7.5 shows the forward velocity, sideslip angle and angle of attack during the simulation. Oscillations in $\hat{U}$ are present during the periods 60-100 s and 130-140 s. $\hat{\beta}$ experiences drift during the periods 60-80 s and 120-160 s. And finally a stationary error is present in $\hat{\alpha}$ during the periods 60-70 s and 120-160 s.

Figure 7.6 shows the residuals in roll, pitch, roll rate and pitch rate. The periods with largest deviations from zero are from 60-100 s and 120-160 s.

Further, looking at the unfiltered variance signals in Figure 7.7 (left) spikes are observed at 40, 100, 140 and 160 seconds. These spikes disappear in the right part of this figure and the variance is smooth. Also in this plot the largest variance is observed at the periods 60-100 s and 120-160 s.

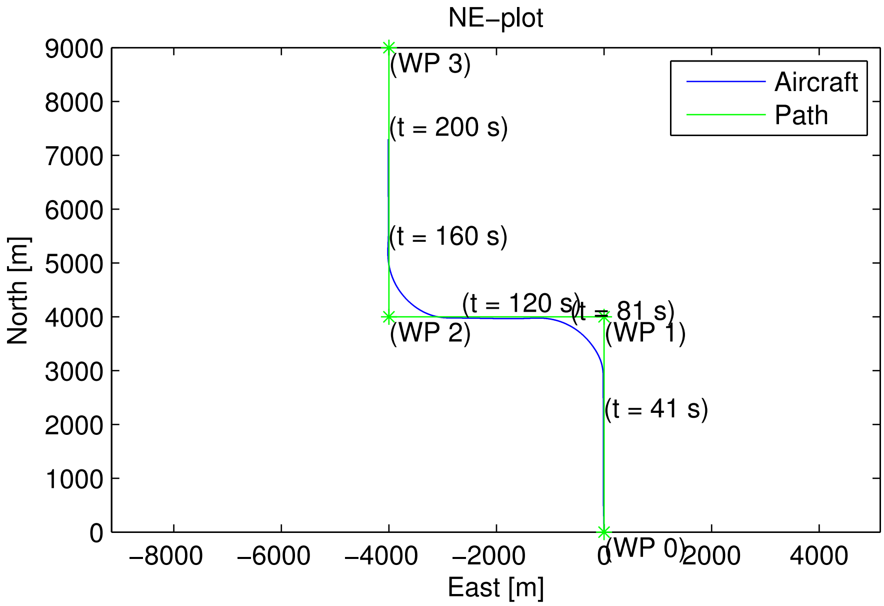


Figure 7.1: Case study 1: A plot of the simulation.

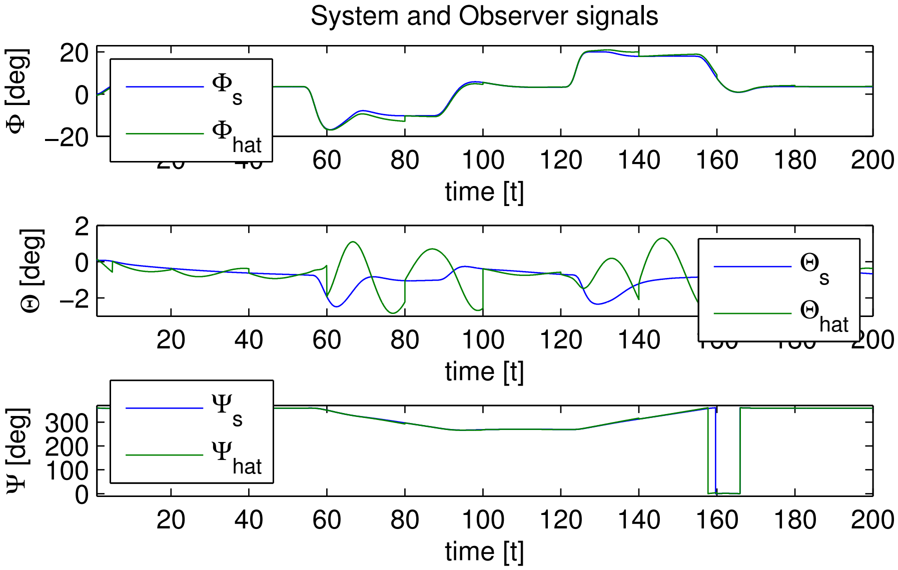


Figure 7.2: Case study 1: The Euler angles from the system and observer.

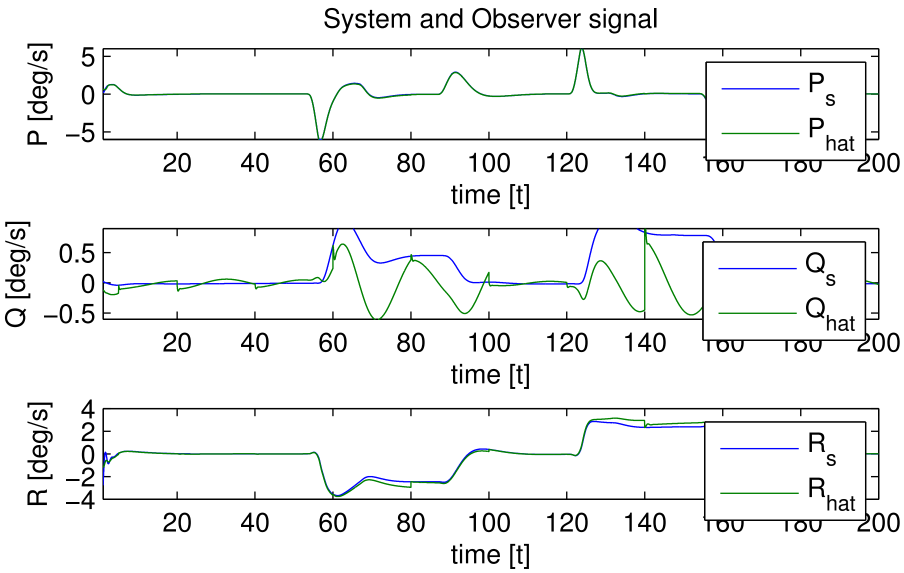


Figure 7.3: Case study 1: The Euler rates from the system and observer.

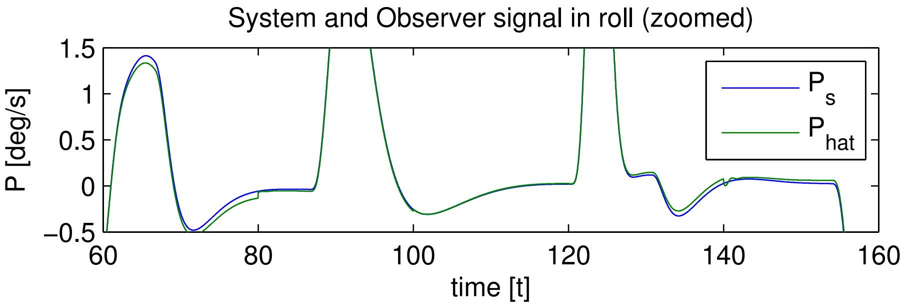


Figure 7.4: Case study 1: A zoomed in graph of the Roll rate from the system and observer.

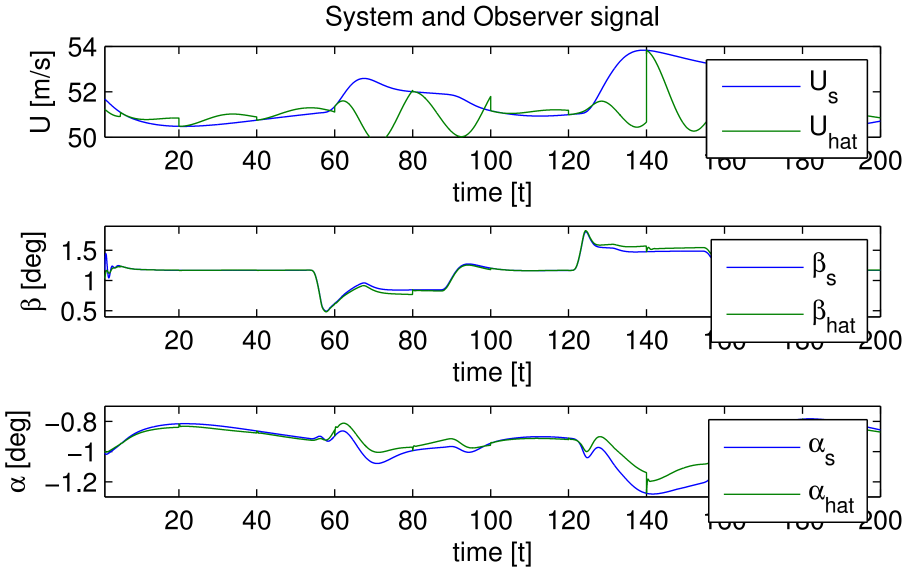


Figure 7.5: Case study 1: Forward velocity, sideslip angle and angle of attack from the system and observer.

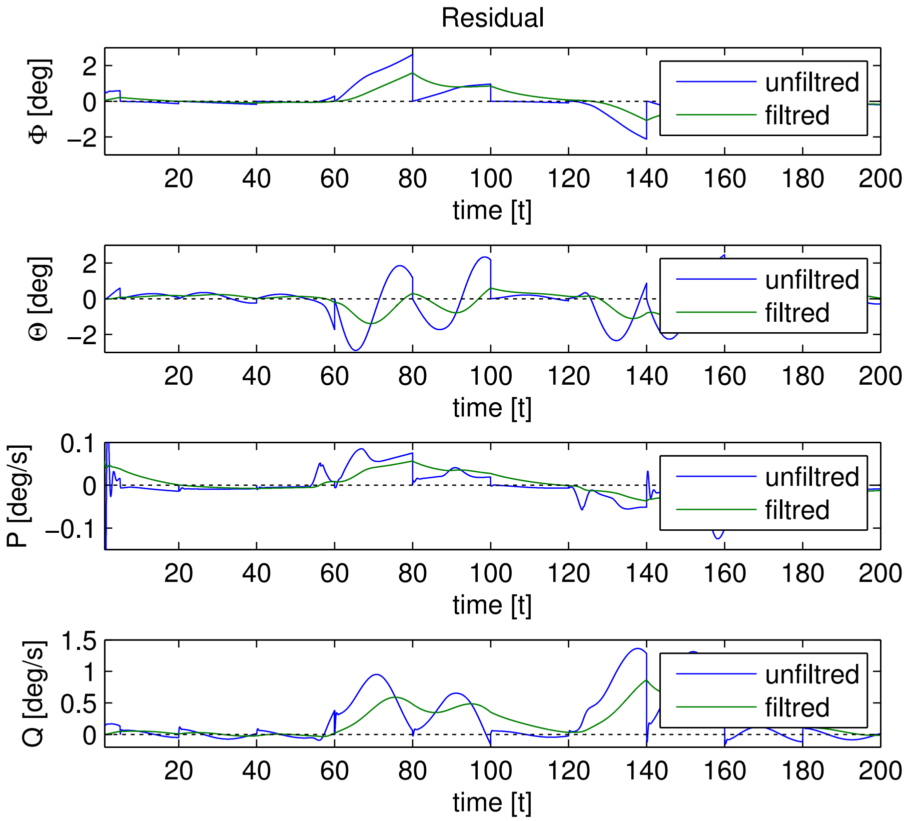


Figure 7.6: Case study 1: The residual from the FDI. Unfiltered values are compared with filtered values.

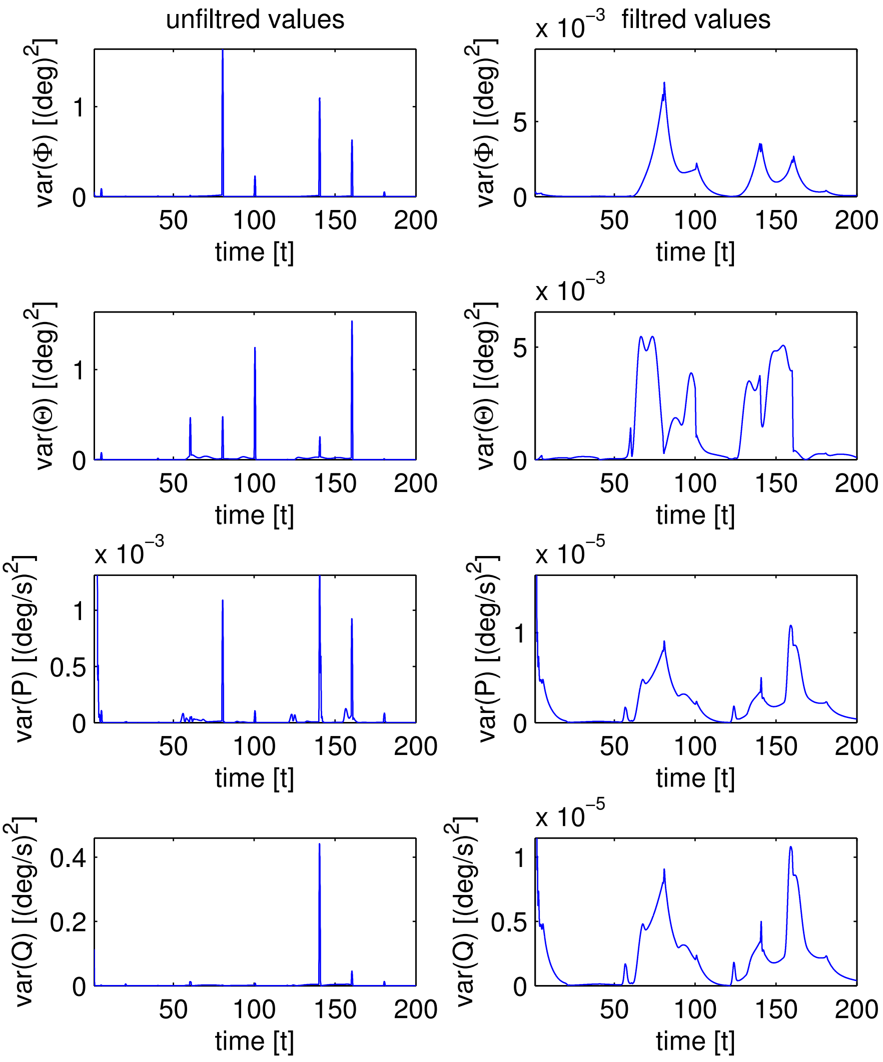


Figure 7.7: Case study 1: The variance of the residual from the FDI. Unfiltered (left) and filtered (right) values are presented.

7.2 Case Study 2: Two Turns with Fault Present

A fault is introduced in the ailerons at 60 s. During 5 seconds the control surfaces are reduced to 10 % of normal effect. Variance thresholds for fault detection is set according to Table 7.1. The flight plan and a plot of the simulation is seen in Figure 7.8.

Figures 7.9 - 7.10 shows the results of the FDI.

Figure 7.9 shows the residual and the variance of the residual of the simulation. It is clear that the residual in roll and roll rate increases considerably at 65 seconds. First the roll rate and followed by roll. Looking at the variance plot roll and roll rate crosses the threshold at 66.5 and 64.5 seconds which is also reflected in the plot of the FDI's decision, Figure 7.10. In the longitudinal model the residuals increase during the periods 60-100 s and 140-180 s. During the second of these periods a fault is detected in pitch and pitch rate according to Figure 7.10. Figure 7.11 shows the roll and roll rate from the system, desired values and estimated values. At 150 s a large aileron command is produced.

Now the fault accommodation algorithm is included in the simulation.

Table 7.1: Threshold values for fault detection in the FDI.

Thresholds	values:	
Φ_{th}	$1 \times 10^{-5}\text{rad}^2$	$= 3.28 \times 10^{-2}\text{deg}^2$
Θ_{th}	$1 \times 10^{-5}\text{rad}^2$	$= 3.28 \times 10^{-2}\text{deg}^2$
P_{th}	$5 \times 10^{-7}(\text{rad}/s)^2$	$= 1.6 \times 10^{-3}(\text{deg}/s)^2$
Q_{th}	$5 \times 10^{-7}(\text{rad}/s)^2$	$= 1.6 \times 10^{-3}(\text{deg}/s)^2$

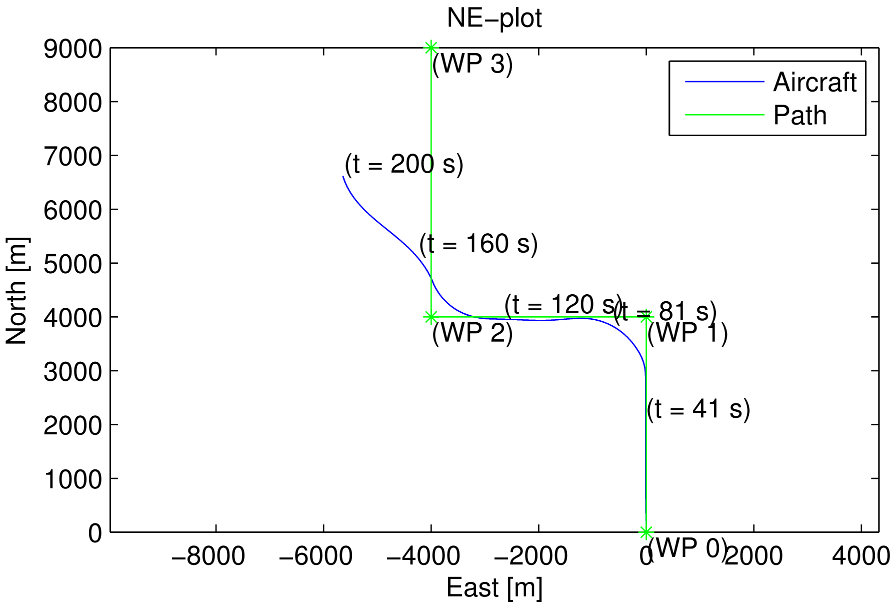


Figure 7.8: Case study 2: A plot of the simulation.

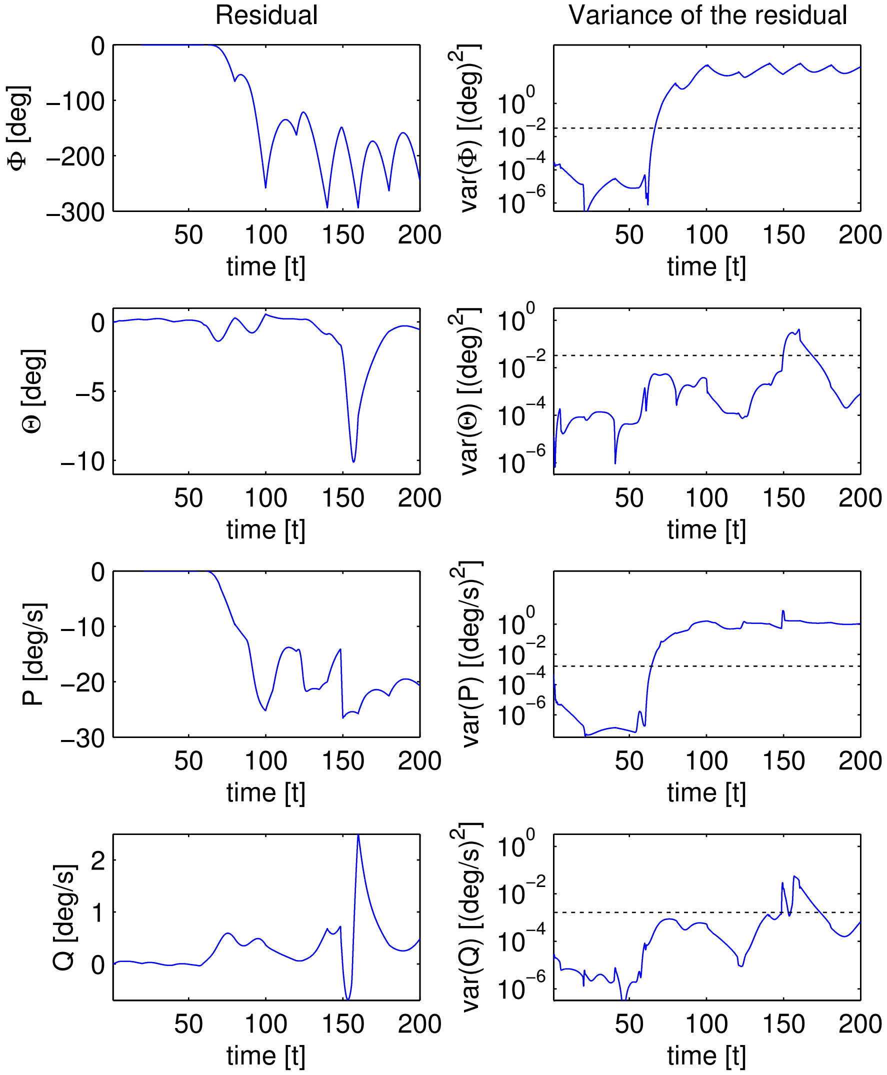


Figure 7.9: Case study 2: The residuals and the variance of the residuals. The variance is plotted with a logarithmic y-axis

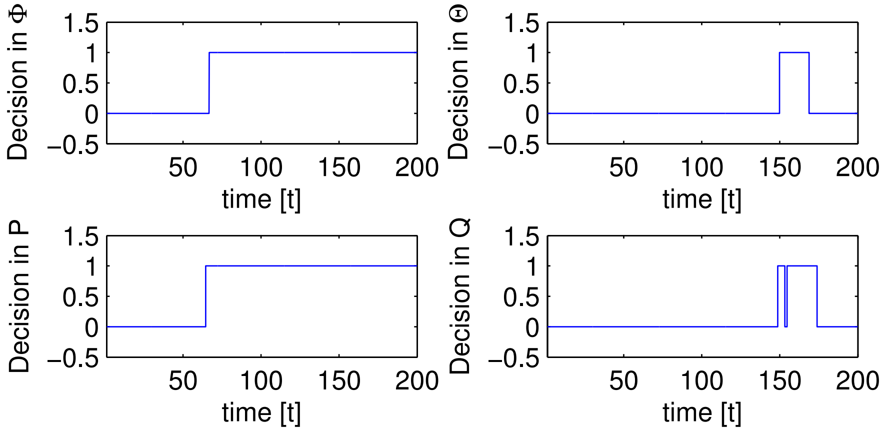


Figure 7.10: Case study 2: The decision that a fault has occurred in the respective channels.

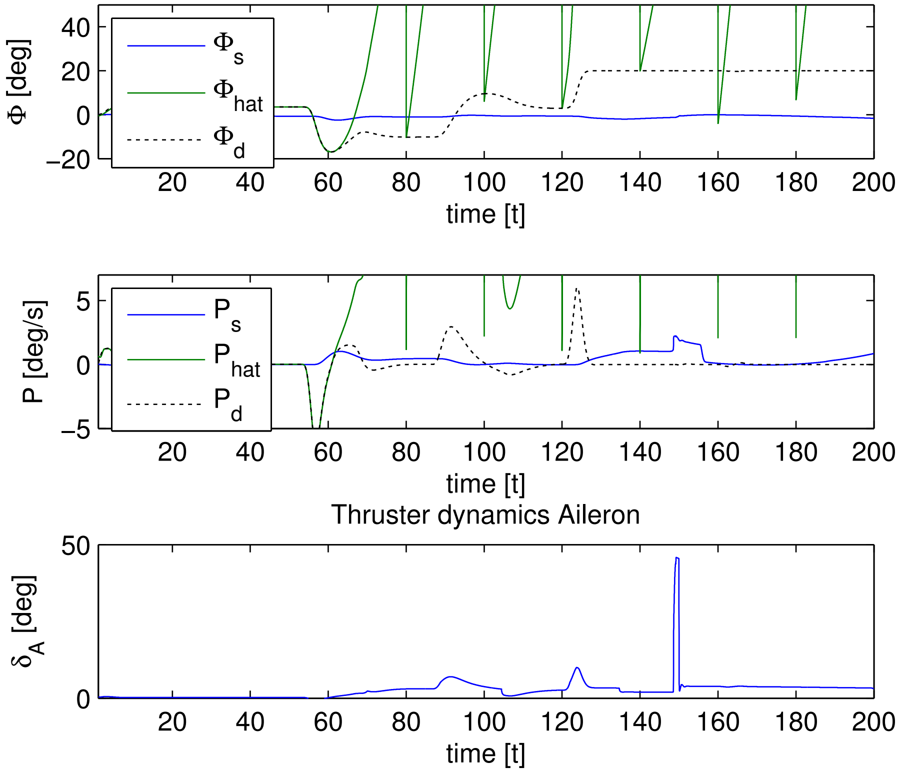


Figure 7.11: Case study 2: Roll and roll rate states and aileron dynamics.

7.3 Case Study 3:

The flight plan of this case study can be seen in Figure 7.12 and is straight in the NE-plane, the aircraft's trajectory can also be viewed. The nominal values in the controller are constant during this simulation and are seen in Table 6.4. In Figure 7.12 a large deviation from the path is seen for the blue graph. The red is following the path perfectly. Every 20 seconds the $\hat{\Phi}$ bounces fast away from Φ_s as seen in Figure 7.13. The bursting like behaviour is only seen in the lateral model, Figure 7.14. Figure 7.15 shows the residual and variance of the residual from the simulation. the bursting is most prominent in P.

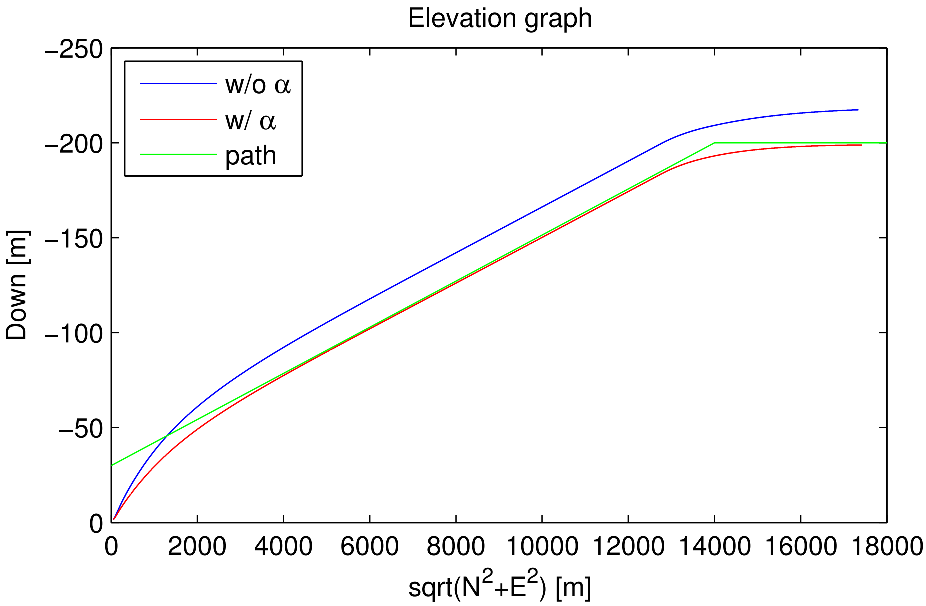


Figure 7.12: Case study 3: A plot of the simulation. Where the blue line is without feedback of α, and the red is with.

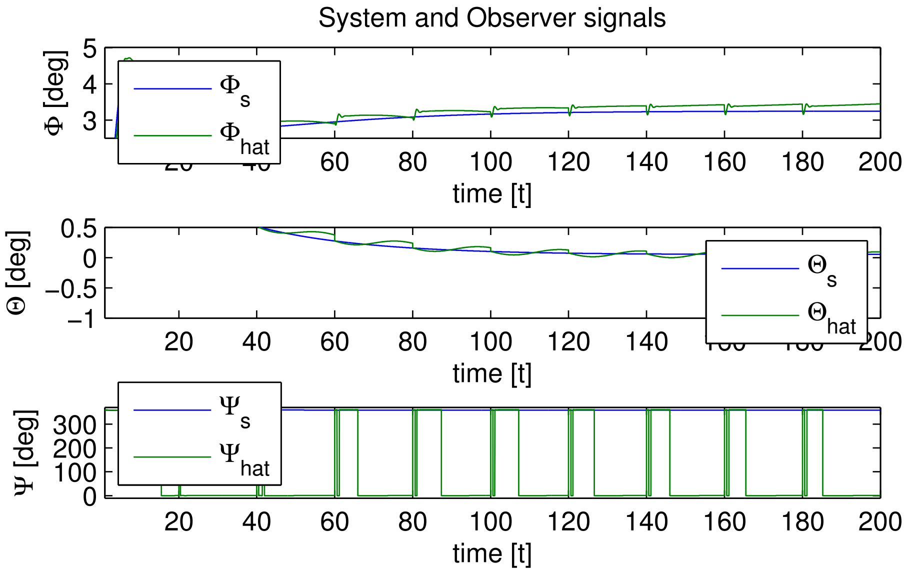


Figure 7.13: Case study 3: The Euler angles from the system and observer.

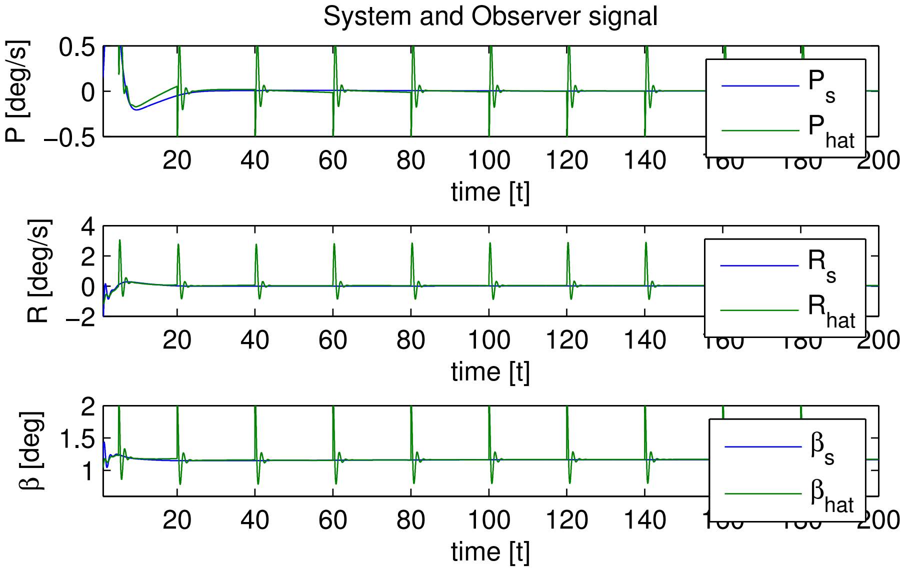


Figure 7.14: Case study 3: A burst like effect is observed in the lateral model.

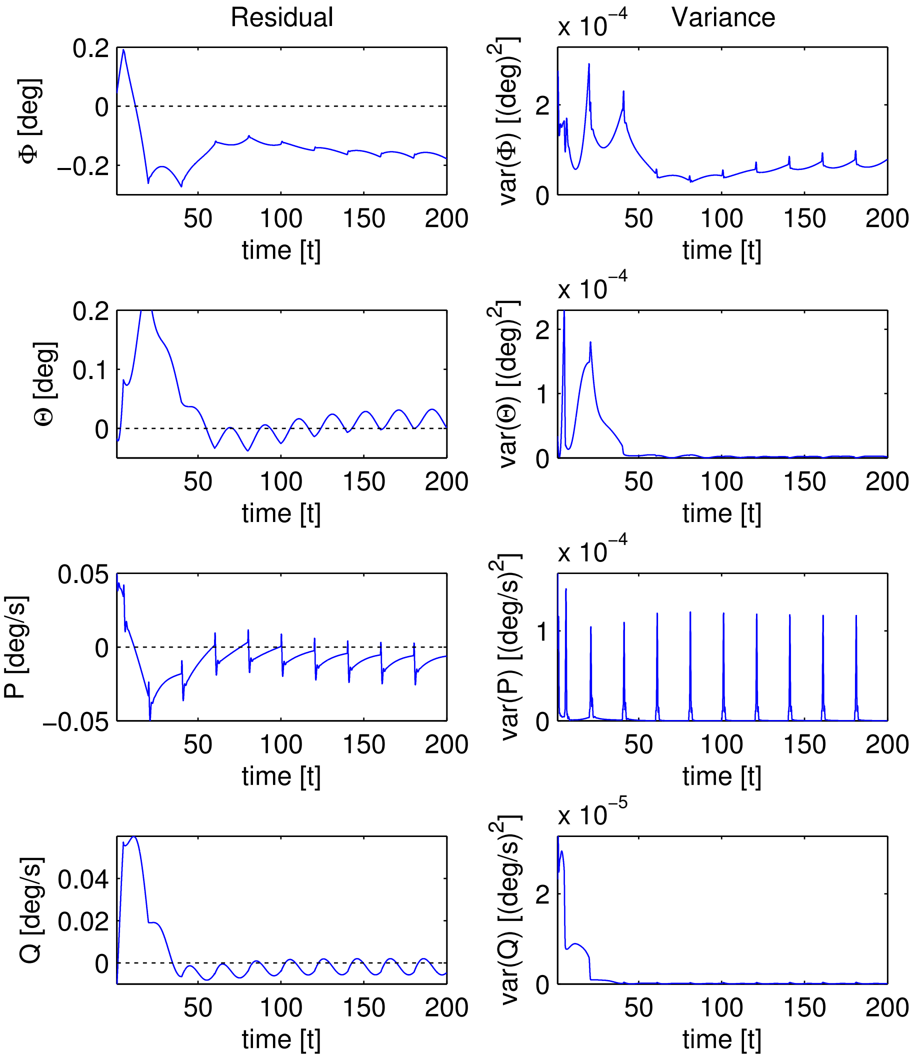


Figure 7.15: Case study 3: The residuals and the variance of the residuals.

7.4 Case Study 4

The purpose of this study case is to see the effect of a 90 % loss of effectiveness in the elevator control surface. The fault occurs at 60 s and the trajectory of the aircraft can be seen in Figure 7.16. The aircraft experiences a stationary error during the latter part of the graph.

The residual and variance of the residual are presented in Figure 7.17. After 60 seconds oscillations are seen in the residuals of $\hat{\theta}$ and Q. The variance in pitch is close to the threshold after 60 seconds.

Errors are detected at 68 seconds in roll and 66.5 seconds in Q in accordance with Figure 7.18. Also in the roll channel there are jumps in the decision graph at 120, 140, 160 and 180 seconds.

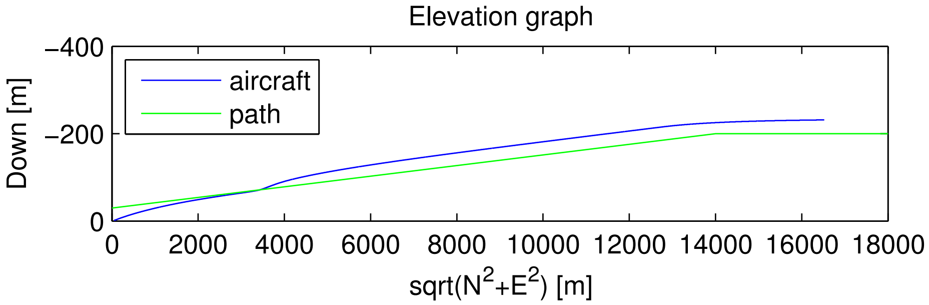


Figure 7.16: Case study 4: A plot of the simulation.

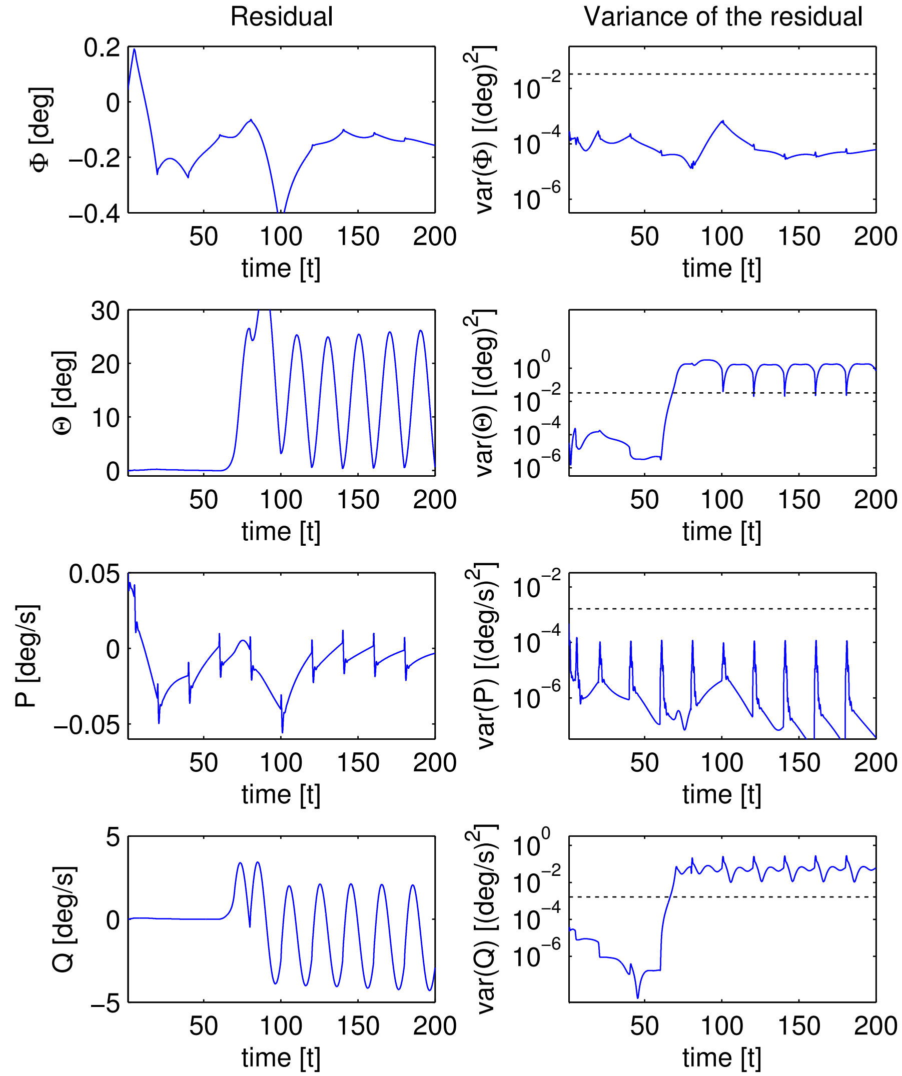


Figure 7.17: Case study 4: The residuals and the variance of the residuals.

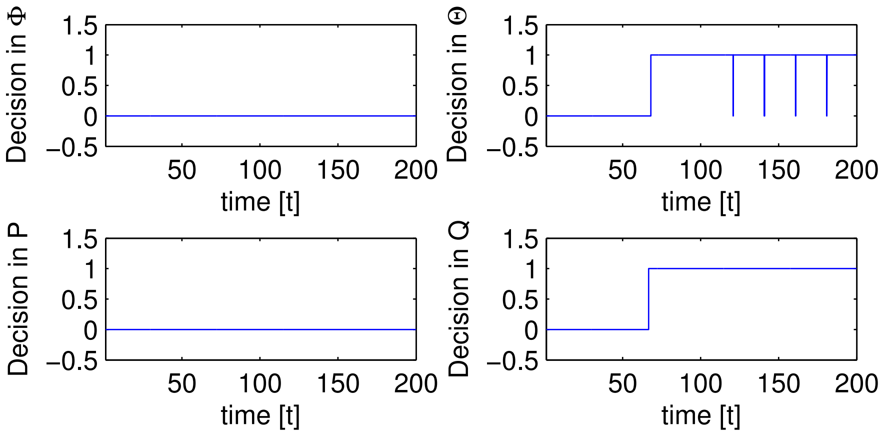


Figure 7.18: Case study 4: The decision that a fault has occurred in the respective channels.

7.5 Case Study 5

The same simulation as in case study 3 and 4 is performed. In this case study two faults occurs, first on the elevator at 60 s and then on the ailerons at 80.

The results are seen in Figure 7.19. In roll and pitch the faults are detected at respectively 88 and 68 seconds. Further, the variance crosses the threshold some times after the fault is detected.

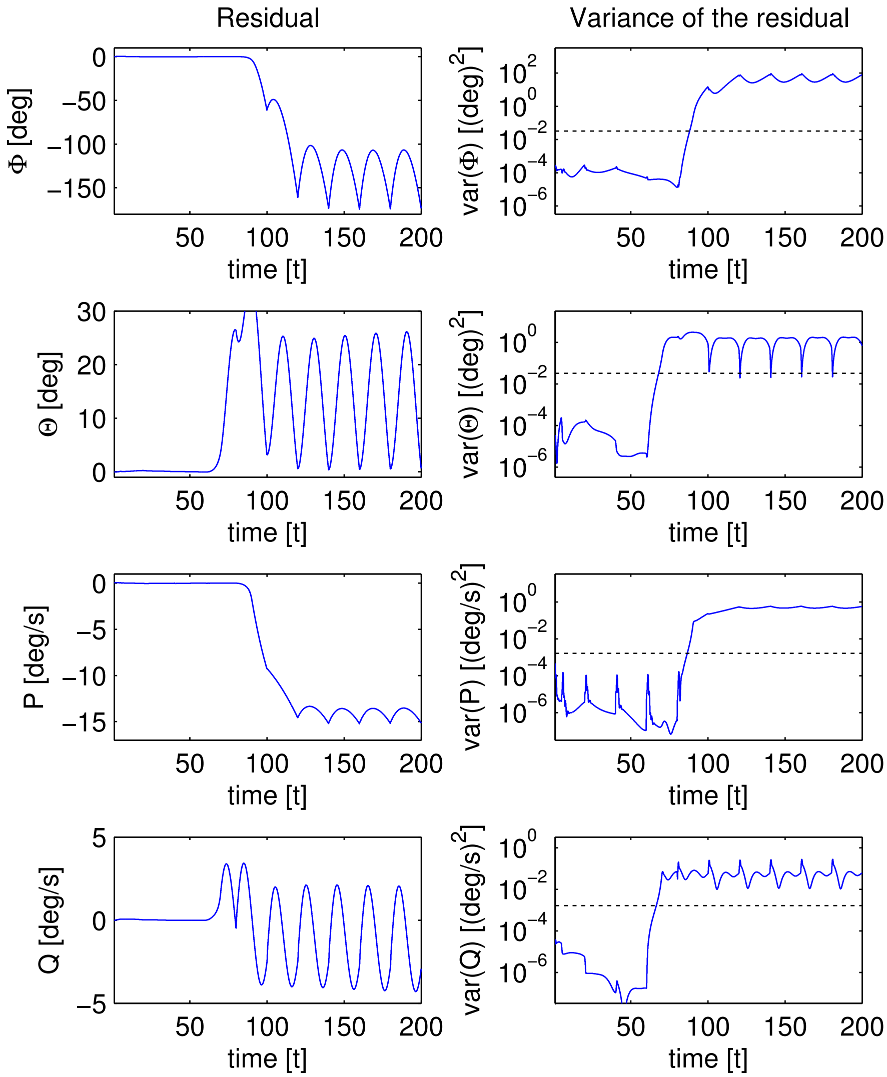


Figure 7.19: Case study 5: The residuals and the variance of the residuals.

7.6 Case Study 6

For the same scenario from case study 1 and 2, four different magnitudes of faults were tested according to Table 7.2. These occurred at 60 seconds in the ailerons during the aircraft's first turn. Figure 7.21 shows the residual and variance as well as the effect the fault has in roll. There is a good spread in the residuals and variances. The effect the fault has on roll is none for cases 1, 2 and 3. For case 4 with the biggest effect reduction Φ_s is not able to follow Φ_d during the last part of the plot.

Looking at the output from the kinematic controller Figure 7.20 it is clear that the contribution increases as the fault magnitude increases.

Table 7.2: Four different fault magnitudes.

1:	No fault (0 % reduction) (blue)
2:	30 % reduction (green)
3:	60 % reduction (red)
4:	90 % reduction (turquoise)

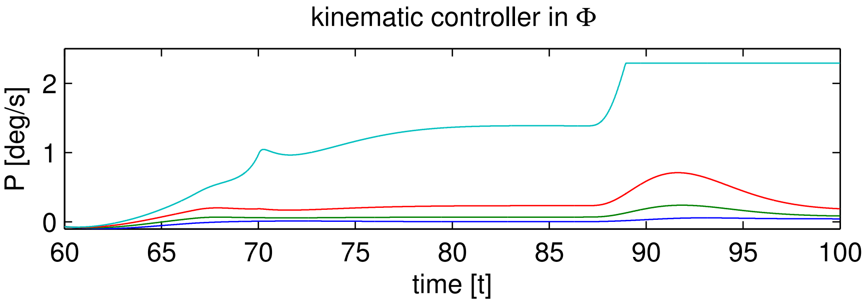


Figure 7.20: Case study 6: The output of the kinematic controller in each case. Four different magnitudes of faults in aileron were tested: No faults (blue), 30 % reduction (green), 60 % reduction (red) and 90 % reduction (turquoise).

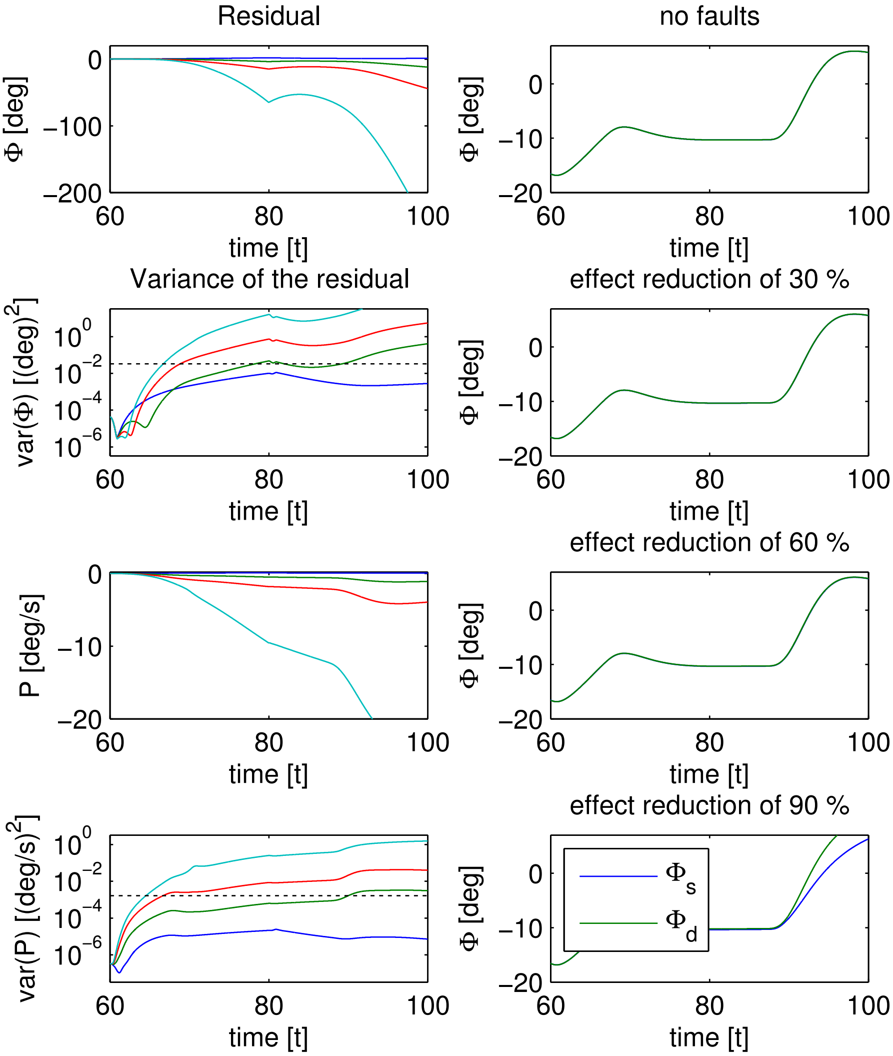


Figure 7.21: Case study 6: Four different magnitudes of faults in aileron were tested: No faults (blue), 30 % reduction (green), 60 % reduction (red) and 90 % reduction (turquoise). The residuals and the variance of the residuals is plotted on the left and the right shows the loss of accuracy in roll.

7.7 Case Study 7 Fault Accommodation

In this case study a fault occurs at 60 seconds in ailerons which reduces the control surface to 10 % effectiveness. The flight plan from case study 1, 2 and 6 is simulated. Two simulations are compared; first without fault accommodation and then with.

In both simulations in Figure 7.22 the aircraft is not able to follow the reference value. With fault accommodation the performance is worse.

Figure 7.23 shows that the kinematic controller has a bigger contribution with fault accommodation enabled.

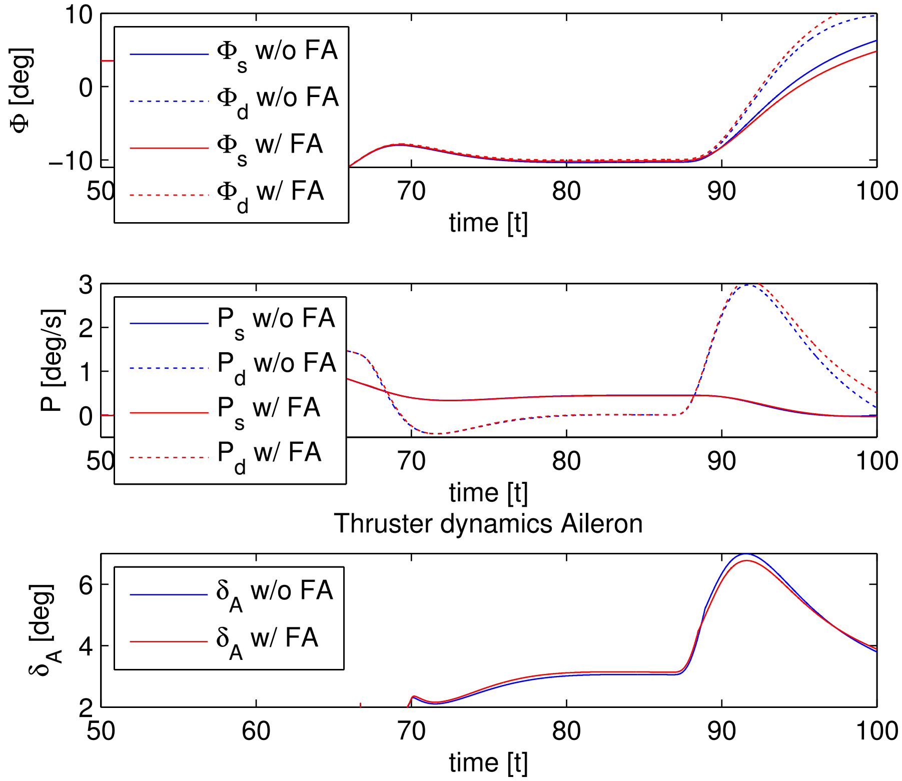


Figure 7.22: Case study 7: Simulations are done with and without fault accommodation (FA).

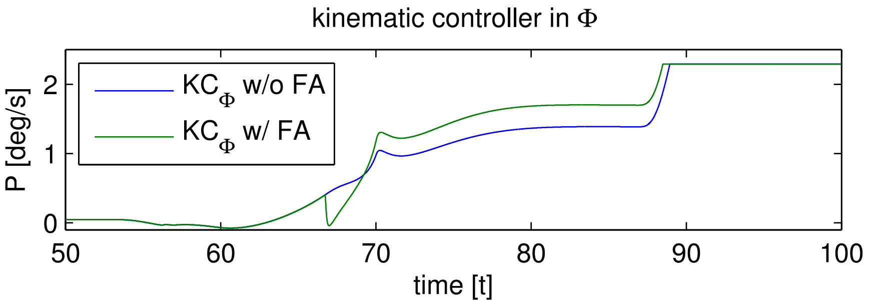


Figure 7.23: Case study 7: A comparison is done of the kinematic controller (KC) with and without fault accommodation (FA).

Chapter 8

Discussion

8.1 Case Study 1

First of all it is clear that the largest residuals occur during the two turns. The observer does an excellent job in estimating the states in the lateral model. There is a drift in $\hat{P}$ during the banking which propagates to $\hat{\Phi}$, $\hat{P}$ and $\hat{\beta}$. The reason for this drift is that $|\Phi| >> 0$ and the assumption that $\sin(\cdot) \approx \cdot$ and $\cos(\cdot) \approx 1$ becomes weaker. However the effect is so small that it is easily corrected when the model is updated as can be seen every 20 s.

The longitudinal model is not as accurate as the lateral. When banking is initialised Θ_s drops 1° as discussed in Section 6.2.6 and the controller corrects this error. Because this dynamic is not modelled in the observer, this incident induces the oscillations seen in figures 7.2 and 7.3. Furthermore, because the nose is elevated the aircraft's forward velocity and angle of attack are also affected, as seen in Figure 7.5.

The spikes observed in the unfiltered variance signal Figure 7.7 occur when the model is updated and are most prominent when the residual is large. To suppress these spikes a filter is implemented. Smoother residuals result in a more even variance.

8.2 Case Study 2

The fault in aileron occurs at 60 seconds and gradually increases for 5 seconds until the elevator is reduced to 10 % effect. According to Figure 7.9 a fault is detected in roll channel at 66.5 seconds and in roll rate at 64.5 s. As it takes time for the fault to propagate to roll it is natural that it is first detected in the rates. A fault is also detected in pitch and pitch rate. The reason for this is a large aileron command at 150 s. As the roll rate is corrected, this motion propagates to the pitch channel which has to be corrected by the controller. Also, because the roll induced motion is not modelled in the longitudinal model, the result is an increase in the residuals in the lateral channel. The origin of the aileron step is an increased error between the desired value and the system value in P as seen in Figure 7.11 during the period 148-156 s.

8.3 Case Study 3

Figure 7.12 shows that without feedback of the angle of attack, α the stationary error will be substantially large. By adding α to the guidance law, as seen in Section 4.5.1, perfect following can be achieved.

A bursting like behaviour is seen in the lateral model in figures 7.13 and 7.14. Since this happens every 20 seconds, it is clear that it is related to the updating of the model. Every 20 seconds $\hat{\phi}$ is reset to the system value $\Phi - \phi_0$. As this is not the equilibrium of the linear model, the signal bounces back to its equilibrium resulting in bursting effects in the roll rate. R and β are also affected.

The bursting behaviour is also present in the residuals. Compared to the variance in Figure 7.7, the variances of P and Q are larger.

8.4 Case Study 4

Faults are detected at 68 seconds in roll and 66.5 seconds in roll rate. The variance in roll rate has a good separation between healthy and faulty residuals. However because the variance of roll is closer to the threshold, some false decisions occur. This could be solved by implementing a lower threshold.

8.5 Case Study 5

The results from this simulation resembles the results from the previous case studies. The roll channel resembles Figure 7.9 and pitch channel resembles Figure 7.17.

The results from the FDI show that there is a clear separation between a faulty and a fault-free system. This enables the FDI to pick up multiple faults and isolate them to the correct channel.

8.6 Case Study 6

From this case study it becomes clear that different magnitudes of faults affects the magnitude of the residual which is natural. The variance of P has the clearest separation of all the variances. Keeping in mind that the y-axis is logarithmic it is easy to part the fault with 30 % reduction from the fault-free case. However as the threshold is set at $1.6 \times 10^{-3}(\text{deg}/s)^2$ the fault is not picked up right away.

The effect of the fault from case 2 and 3 is nevertheless not severe. Looking at the effect on the roll channel it becomes clear that perfect following is still achieved. This is explained when regarding the control output of the kinematic controller. As the performance of the rate controller is impaired the non-model-based-kinematic controller drives the stationary errors to zero. The rate controller using LQ optimal control is limited to a relative small bandwidth. This becomes evident as the contribution of the kinematic controller increases drastically when faults occur. Thus, the kinematic controller adds some control bandwidth. However when this controller reaches saturation as seen for the simulation with 90 % effect reduction, the system is not able to follow the reference. The kinematic controller adds some degree of robustness to the control system. However, it is not possible to come to a conclusion on this because the LQ does not include physical constraints in the optimization problem.

Even though perfect following is achieved, the fault can still be reflected in the residuals. This is because the increased use of control surfaces due to the kinematic controller, separates the response of the system from the response of the model. As the response of the system is slower, the system and the observer will behave differently. This means that a fault can be picked up even if the system seems to behave as if it were fault-free.

8.7 Case Study 7

Looking at Figure 7.22 it becomes clear that with fault accommodation enabled, the system has a slower response than without. The reason for this lies in the gain matrices.

8.7.1 Excluding Rudder

For illustrative purposes the rudder will be excluded in this section. Without fault accommodation enabled the controller gains are calculated based on the following control matrix in the lateral channel:

$$\boldsymbol{B} = \begin{bmatrix} 0 & -91 & -2.9 \end{bmatrix}^T \tag{8.7.1}$$

$$\boldsymbol{G_1} = \begin{bmatrix} -0.1385 & 1.8175 & 0.0416 \end{bmatrix} \qquad \boldsymbol{G_2} = \begin{bmatrix} -1.9899 & 0.0001 \end{bmatrix} \tag{8.7.2}$$

With fault accommodation the new control matrix becomes:

$$\boldsymbol{B} = \begin{bmatrix} 0 & -9.1274 & -0.2925 \end{bmatrix}^T \tag{8.7.3}$$

which result in the following controller gains:

$$\boldsymbol{G_1} = \begin{bmatrix} -0.3879 & 0.8495 & 0.1341 \end{bmatrix} \qquad \boldsymbol{G_2} = \begin{bmatrix} -1.4063 & 0.0000 \end{bmatrix} \tag{8.7.4}$$

To analyse the control gains the control equation (4.2.9) has to be regarded:

$$\boldsymbol{u} = \boldsymbol{G}_1\boldsymbol{x} + \boldsymbol{G}_2\boldsymbol{y}_d \tag{8.7.5}$$

Looking at the feedback of state p (second entry in the $\boldsymbol{G}_1$-matrix) and the reference feedforward of p (first entry in the $\boldsymbol{G}_2$-matrix), it is clear that these controller gains have been reduced after the introduction of fault accommodation. By reducing the control matrix to describe the faulty system the control signal is also reduced. This means that the relationship between the state signal and the control signal in the cost function (4.2.8) is altered. The tuning matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are tuned with the fault-free system in mind and therefore the controller has a limited control bandwidth. These matrices do not hold when the fault is this severe so the $\boldsymbol{R}$ matrix should be tuned up.

8.7.2 Including Rudder

The rudder is included and the controller gains without fault accommodation, is calculated to be:

$$\boldsymbol{G}_1 = \begin{bmatrix} -0.14 & 1.8 & 0.042 \\ 0.0003 & -0.013 & 0.0016 \end{bmatrix} \qquad \boldsymbol{G}_2 = \begin{bmatrix} -1.2 & 0.0001 \\ 0.011 & -0.0001 \end{bmatrix} \tag{8.7.6}$$

With fault accommodation the gains become:

$$\boldsymbol{G}_1 = \begin{bmatrix} -0.39 & 0.85 & 0.14 \\ -0.014 & -0.041 & 0.045 \end{bmatrix} \qquad \boldsymbol{G}_2 = \begin{bmatrix} -1.4 & 0.0000 \\ -0.067 & -0.0001 \end{bmatrix} \tag{8.7.7}$$

In this case it can be seen that the contribution of the rudder increases when the fault accommodation is enabled. While the aileron control is reduced the contribution of the rudder will increase because of the relations in the tuning matrix. Since the rudder is not as effective the aileron the overall performance of the aircraft is still impaired. If the aircraft had been over actuated, for example if flaperons had been used (flaps that also can be used as ailerons), in addition to ailerons, some of the control load could be transferred to these. However, this will merely increase the control bandwidth and make the control system more robust.

[Blanke et al., 2006] mentions that the solution of the fault tolerant control problem is admissible and not optimal. This means that the performance of the fault tolerant control system is reduced with respect to the fault-free case. In order to have an optimal solution of the faulty system the tuning matrices have to be tuned in advance for different kinds of faults.

Is it better to have lower gains and a correct control plant than have higher gains and a wrong control plant? From Figure 7.22 it can be seen that the difference between with and without fault accommodation is not a big. This is mainly due to the kinematic controller, which has a larger contribution with fault accommodation to account for the larger stationary errors. It is easier to ensure stability when the correct model is implemented.

8.8 General Discussion

8.8.1 Parity Space as FDI

By studying the outputs from the observer from the case studies, small oscillations are always observed. For instance in Figure 7.3 during the period 0 - 60 seconds small oscillations can be seen in Q. This is also reflected in the residuals in Figure 7.6 in all channels during the same time period. These oscillations are present because of model uncertainty in the observer. The equilibrium in the linear model is not the same as that of the system. When the model states are updated to that of the system they will begin to oscillate between the two equilibriums. Further, when a fault is present the gap between these equilibriums gets bigger and subsequently the oscillations become bigger. These oscillations are picked up by the residual validation.

8.8.2 Actuator and System Faults

The fault detection and isolation (FDI) developed in this thesis bases the analysis on roll and pitch measurements from the system. As mentioned in Section 2.1 it is hard to differentiate between system faults and actuator faults. In this thesis the faults regarded are effect reductions of the control surfaces. A fault on the fuselage of the aircraft can result in similar effects. The FDI merely picks up the difference between the system and the observer and is therefore able to find faults that were not accounted for when the FDI was designed. However, after the fault is isolated measures have to be taken to counteract the fault induced effects. This includes post analysis of the results which may not be as trivial.

8.8.3 Injection Term vs Model Updating

In this thesis the model update is done every 20 seconds and this is performed by updating the observer's states to the measurements from the system. Another common way of making the observer follow the system is by using an injection term. The injection term is given by $\boldsymbol{K}$ times the estimate error (8.8.1) and drives the states in the observer towards that of the system.

$$\dot{\hat{\boldsymbol{x}}} = f(\hat{\boldsymbol{x}}) + \boldsymbol{K}(\boldsymbol{x} - \hat{\boldsymbol{x}}) \tag{8.8.1}$$

If an injection term had been used, the observer given by (2.2.1) would instead become:

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{R}_1\boldsymbol{f}(k) + \boldsymbol{K}(\boldsymbol{x}(k) - \hat{\boldsymbol{x}}\boldsymbol{(k)}) \tag{8.8.2}$$

This system is not a model of the fault-free system and if a fault occurs the injection term will try to hide it.

By periodically updating the states in the model, the fault-free property of the observer will stay intact. As discussed earlier in this section, the model updating also induce oscillations between the equilibriums of the system and the model which can be exploited. Also by resetting the states periodically, jumps are induced in the residual which can be filtered out.

8.8.4 Wind and Turbulence

In case study 2 it is seen that a correction in roll rate propagates to the pitch channel and causes a false fault detection. If this case is only regarded in the longitudinal model one can conclude that an external force has affected the aircraft. Similar effects could be induced by first order wind or turbulence and these are not regarded in this thesis. However, in a real setting these have to be accounted in the FDI to minimize false fault detections.

Chapter 9

Conclusion And Future work

9.1 Conclusion

The main goal of this thesis is to implement a fault-tolerant control system (FTCS) on an unmanned aerial vehicle (UAV). The FTCS includes an automatic flight control system (AFCS) with fault accommodation and a fault detection and isolation (FDI) system. In this chapter conclusions are drawn based on the discussion of the results in chapter 8.

- The model-based velocity and rate controller implemented results in acceptable tracking of the reference signal with the bonus of feedforward control. The kinematic controller corrects for stationary errors in the rates and adds control bandwidth. It is observed that when large errors are present in the velocity and rate controller, rough control signals can occur. This compromises the robustness outside the bandwidth of the kinematic controller.

- Couplings in the longitudinal and lateral model are observed and these have affected the accuracy of the controller as well as the FDI. This could have been prevented if a coupled control plant had been enabled.

- The results from the FDI indicate that there is a large separation between a faulty and a fault-free system. Static thresholds are implemented and these were easily tuned with simulations of a faulty- and a fault-free system. The FDI is also able to pick up faults even though the aircraft follows the reference signal perfectly. This is because the control system is operating outside the bandwidth of the rate controller but inside the bandwidth of the kinematic controller. In other words as the control signal from the kinematic controller becomes large a fault has occurred. Due to the clear separation of a faulty system from a fault-free, the FDI is also able to pick up multiple faults and isolate them to their respective channels. However, first order wind and turbulence are not regarded as these will reduce the accuracy of the FDI.

- It has been noticed that by updating the observer, oscillations are induced in the residual. The FDI is able to exploit this by regarding the variance of this signal.

- As a consequence of the limited bandwidth of the LQ controller, optimal control is not achieved when fault accommodation is enabled during a fault situation. Because the kinematic controller adds control bandwidth, the aircraft is still able to follow trajectories when the fault is not severe. Also, it has been observed that the LQ controller can generate rough output signals as a result of large errors. This weakens the robustness of the control system and stability is not guaranteed. However, instabilities have not been observed in the study cases, only rapid corrections of large errors in the LQ controller.

9.2 Future Work

- In this thesis static thresholds have been implemented. These are in a very basic form of fault detection and can be replaced by more advanced methods. One example is adaptive thresholds as discussed in [Patton and Chen, 1994].
- The simulations performed have been done without noise. Noise can have many forms, the most important ones are measurement noise, wind and turbulence. To be able to implement this system in a real setting these have to be accounted for as they can contribute to false alarms.
- In this thesis the fault is assumed to be known. This is done in order to be able to implement fault accommodation. In reality the fault has to be estimated, i.e. fault identification has to be implemented. This would upgrade the system from a FDI to a fault detection and diagnosis (FDD) system.
- The limited bandwidth of the LQ controller resulted in an admissible control law which was not an optimal solution. Additional work should be done within robust control systems to improve control in fault situations.

Bibliography

Kåre Vistnes. *Modeling and Identification of Aircraft.* PhD thesis, Norwegian University of Science and Technology, 2012.

Youmin Zhang and Jin Jiang. Bibliographical review on reconfigurable fault-tolerant control systems. *Annual Reviews in Control*, 32(2):229–252, December 2008.

Halim Alwi, Christopher Edwards, and Chee Pin Tan. Fault Detection and Fault-Tolerant Control Using Sliding Modes. pages 7–28, 2011.

R. J. Patton and J. Chen. Review of Parity Space Approaches. *Jorunal of Guidance, Control and Dynamics*, 17(2), 1994.

Thor I Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control.* John Wiley and Sons Ltd, 2011a.

Rolf Isermann. *Fault-Diagnosis Systems.* Springer-Verlag, Berlin/Heidelberg, 2006.

Morgens Blanke, Michel Kinnaert, Jan Lunze, and Marcel Staroswiecki. *Diagnosis and Fault-Tolerant Control.* Springer-Verlag, 2nd edition, 2006.

Asgeir J. Sørensen. *Marine Control System.* Department of Marine Technology, NTNU, 2nd edition, 2011.

Paul M. Frank. Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy - a survey and some new results. *Automatica*, 26(3), 1990.

TI Fossen. Mathematical Models for Control of Aircraft and Satellites. *Department of Engineering Cybernetics*, (January), 2011b.

Vladislav Klein and Eugene A. Morelli. *Aircraft System Identification: Theory and Practice.* AIAA Education Series, 2006.

Jens G. Balchen, Trond Andresen, and Bjarne A. Foss. *Reguleringsteknikk.* Institutt for teknisk kybernetikk, 5 edition, 2003.

Panos J. Antsaklis and Anthony N. Michel. *Linear Systems.* McGraw-Hill, 1997.

Morten Breivik and Thor I Fossen. Guidance laws for autonomous underwater vehicles. In Alexander V. Inzartsev, editor, *Underwater Vehicles.* InTech, 2009.

D. McLean. *Automatic Flight Control Systems.* Prentice Hall International, 1990.

Cessna Aircraft Company. Cessna Skyhawk. URL `www.cessna.com/single-engine/skyhawk`. Accessed data: Maximum Cruise Speed 124 ktas.

Mark W. Spong, Seth Hutchinson, and M. Vidyasagar. *Robot Modeling And Control.* John Wiley and Sons Ltd, 2006.

Appendix A
Kinematics

Kinematics describe the motion and the geometry of a system without regarding forces and moments. To be able to describe the relations between independently moving objects, reference frames are defined. These are divided in Earth-centred, geographical and vehicle reference frames. In this chapter vector and matrix notations are defined which will be used throughout this thesis. Also definitions of important terms will be introduced.

A.1 Reference Frames

This report follows definitions used in [Fossen, 2011a] and [Fossen, 2011b].

Earth-Centred Reference Frames

ECI: The *Earth-centred inertial* (ECI) frame $\{i\} = (x_i, y_i, z_i)$ is a non-rotating and therefore non-accelerating reference frame with the origin in the Earth's centre of mass. Since Newton's laws of motion does apply this reference frame is used for terrestrial navigation.

ECEF: The *Earth-centred Earth-fixed* (ECEF) frame $\{e\} = (x_e, y_e, z_e)$ is rotating with respect to the surface of the Earth with the origin in the Earth's centre of mass. GPS positions are given in ECEF, and the x-axis is pointing to the crossing of 0° longitude and latitude.

Geographic Reference Frames

NED: The *North-East-Down* (NED) frame $\{n\} = (x_n, y_n, z_n)$ is a local tangent plane used for local navigation. The x-axis is pointing to true north, the y-axis to the east and the z-axis is pointing to the earth centre perpendicular to the surface.

Path: The *path* frame $\{p\} = (x_{path}, y_{path}, z_{path})$ is a path-fixed reference frame, with origin in a waypoint $\boldsymbol{p}_k$ and is used for calculating the vehicles deviation from the path. As a path consist of two waypoints, $\boldsymbol{p}_k$ and $\boldsymbol{p}_{k+1}$, the x_{path}-axis points along this vector. the y_{path}-axis points to the right parallel to the horizon and the z_{path}-axis points downward finishing the right hand rule.

Vehicle Reference Frames

BODY: The body-fixed frame $\{b\} = (x_b, y_b, z_b)$ with origin in the aircraft's *centre of origin* (CO). The x-axis is pointing in the vehicle's forward direction, the z-axis is pointing straight down relative to the vehicle and the y-axis to the right, completing the right-hand rule (see Figure A.1 for a graphical description).

WIND: The wind frame $\{w\} = (x_w, y_w, z_w)$ is a reference frame used for navigation in the horizontal-plane. Wind affects the aircraft and alters the direction the aircraft travels with an angle β, as seen in Figure A.2. The origin is located in CO.

SA: The stability frame $\{s\} = (x_s, y_s, z_s)$ is given by an angle α according to Figure A.2. This angle becomes significant when the airflow over the wings is reduced. To to compensate for the lack of lift the aircraft has to angle it's nose upwards. The origin is located in CO.

Body-fixed Reference Points

CO: *Centre of origin* (CO) is a reference point defined arbitrarily.

CG: *Centre of gravity* (CG) is the centre of mass of the aircraft. It reduces the workload when calculating of forces and moments.

A.2 Vector Notation

This section defines important vectors used in aircraft modelling. Also the vectorial notation, including super- and subscripts are also explained.

$$\boldsymbol{\eta} \triangleq \begin{bmatrix} x \\ y \\ z \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}^n_{b/n} \\ \boldsymbol{\Theta}_{nb} \end{bmatrix} \tag{A.2.1}$$

The position vector $\boldsymbol{p}^n_{b/n}$ given in the {n}-frame, shows the position of $\{b\}$ relative to the $\{n\}$-frame. The attitude is given by Euler angles $\boldsymbol{\Theta}_{nb}$ and shows the relations between the $\{b\}$-frame and the $\{n\}$-frame.

$$\boldsymbol{\nu} \triangleq \begin{bmatrix} U \\ V \\ W \\ P \\ Q \\ R \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}^b_{b/n} \\ \boldsymbol{\omega}^b_{b/n} \end{bmatrix} \tag{A.2.2}$$

The velocities are relative to the inertial-frame, which in this report is the NED-frame. $\boldsymbol{v}_{b/n}^{b}$ shows the linear velocity of $\{b\}$ relative to the $\{n\}$-frame, given in the $\{b\}$-frame. $\boldsymbol{\omega}_{b/n}^{b}$ shows the angular velocities of $\{b\}$ relative to the $\{n\}$-frame, given in the $\{b\}$-frame.

$$\begin{bmatrix} X \\ Y \\ Z \\ L \\ M \\ N \end{bmatrix} = \begin{bmatrix} \text{forces in } x^b\text{-direction} \\ \text{forces in } y^b\text{-direction} \\ \text{forces in } z^b\text{-direction} \\ \text{roll moment} \\ \text{pitch moment} \\ \text{yaw moment} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_b^b \\ \boldsymbol{m}_b^b \end{bmatrix} \tag{A.2.3}$$

$\boldsymbol{f}_b^b$ is the forces acting on the CO, expressed in the $\{b\}$-frame. $\boldsymbol{m}_b^b$ is the moment vector about CO, given in the $\{b\}$-frame.

A.3 Rotation Matrices

Rotation matrices are used to rotate a frame of reference to another. These matrices has the properties of the *Special Orthogonal* group for three dimensions (SO(3)) and are defined in [Spong et al., 2006] as:

- $\boldsymbol{R}^T = \boldsymbol{R}^{-1} \in SO(3)$
- The columns (and therefore the rows) of $\boldsymbol{R}$ are mutually orthogonal
- Each column (and therefore the rows) of $\boldsymbol{R}$ is a unit vector
- $\det \boldsymbol{R} = 1$

A rotation of a vector from one frame of reference to another is expressed as

$$\boldsymbol{v}^{to} = \boldsymbol{R}_{from}^{to} \boldsymbol{v}^{from} \tag{A.3.1}$$

A rotation of an arbitrary angle α around the x-axis is given as:

$$\boldsymbol{R}_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \tag{A.3.2}$$

similarly a rotation of α around the y- and z-axis is given by:

$$\boldsymbol{R}_y(\alpha) = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \boldsymbol{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.3.3}$$

A.3.1 Rotation Matrices Between BODY and NED

The three Euler Angles describes the relations between the NED- and the BODY frame. This is called the vehicles attitude and the angles are roll (ϕ), pitch (θ) and yaw (ψ) as seen in Figure A.1. The 6 DOF kinematic equations from [Fossen, 2011a] are seen in (A.3.4) and shows the relation between BODY and NED. The matrix $\boldsymbol{J}_\Theta(\boldsymbol{\eta})$ is referred to as the Jacobian.

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}_\Theta(\boldsymbol{\eta})\boldsymbol{\nu} \tag{A.3.4}$$

$$\Updownarrow$$

$$\begin{bmatrix} \dot{\boldsymbol{p}}^n_{b/n} \\ \dot{\boldsymbol{\Theta}}_{nb} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^n_b(\boldsymbol{\Theta}_{nb}) & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{T}_\Theta(\boldsymbol{\Theta}_{nb}) \end{bmatrix} \begin{bmatrix} \boldsymbol{v}^b_{b/n} \\ \boldsymbol{\omega}^b_{b/n} \end{bmatrix} \tag{A.3.5}$$

Rotation of Linear Velocity

The relation between the NED velocity vector $\dot{\boldsymbol{p}}^n$ and the BODY velocity vector $\boldsymbol{v}^b_{b/n}$ is given as:

$$\dot{\boldsymbol{p}}^n = \boldsymbol{R}^n_b(\boldsymbol{\Theta}_{nb})\boldsymbol{v}^b_{b/n} \tag{A.3.6}$$

where $\boldsymbol{R}^n_b(\boldsymbol{\Theta}_{nb})$ is the rotation matrix and is calculated by three separate rotations:

$$\begin{aligned} \boldsymbol{R}^n_b(\boldsymbol{\Theta}_{nb}) &\triangleq \boldsymbol{R}_{z,\psi}\boldsymbol{R}_{y,\theta}\boldsymbol{R}_{x,\phi} \\ &= \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\phi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\phi s\theta s\psi & -c\psi s\phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \end{aligned} \tag{A.3.7}$$

where $s\cdot = \sin(\cdot)$ and $c\cdot = \cos(\cdot)$.

Rotation of Angular Velocity

The relationship between the BODY fixed rates $\boldsymbol{\omega}^b_{b/n}$ and the Euler rates $\dot{\boldsymbol{\Theta}}_{nb}$ is given in [Fossen, 2011a] as:

$$\dot{\boldsymbol{\Theta}}_{nb} = \boldsymbol{T}_\Theta(\boldsymbol{\Theta}_{nb})\boldsymbol{\omega}^b_{b/n} \tag{A.3.8}$$

where

$$\boldsymbol{T}_\Theta(\boldsymbol{\Theta}_{nb}) = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{bmatrix} \quad \text{where} \quad \theta \neq 90^\circ \tag{A.3.9}$$

where $t\cdot = \tan(\cdot)$. It is important to recognize a disadvantage of using Euler Angles: The matrix $\boldsymbol{T}_\Theta(\boldsymbol{\Theta}_{nb})$ becomes singular when $\theta = \frac{\pi}{2}$. In practice this means when the aircraft is pointing straight down towards the ground or straight up. Quaternions is another method used which does not have this flaw. However this method introduces an additional state to describe attitude which complicates the kinematics.

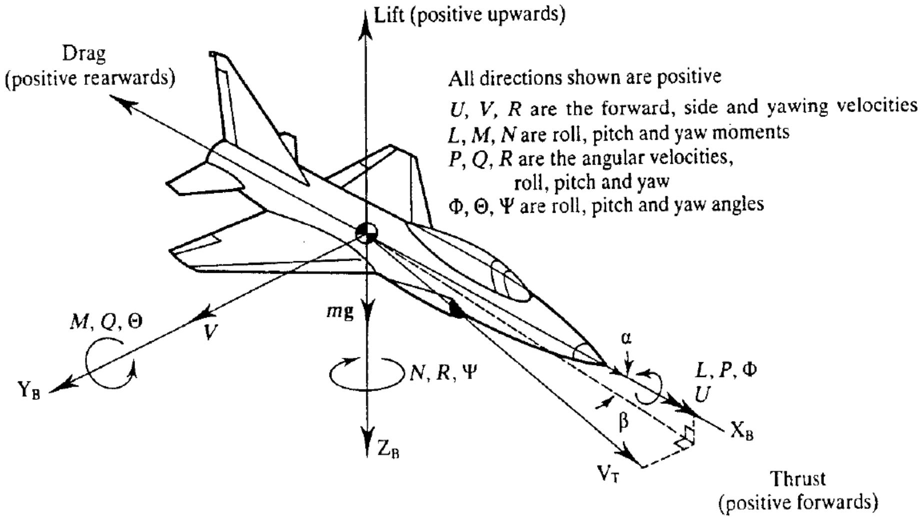

Lift (positive upwards)
Drag
(positive rearwards)
All directions shown are positive
U, V, R are the forward, side and yawing velocities
L, M, N are roll, pitch and yaw moments
P, Q, R are the angular velocities,
roll, pitch and yaw
Φ, Θ, Ψ are roll, pitch and yaw angles
M, Q, Θ
V
mg
Y_B
N, R, Ψ
α
L, P, Φ
U
β
X_B
Z_B
V_T
Thrust
(positive forwards)

Figure A.1: Definition of the Euler angles, velocities, forces and moments in the *body*-axis (McLean 1990)

A.4 Angle of Attack and Side Slip

The stability axis is given by a rotation of α around the y^b-axis as seen in Figure A.2. α is the angle of attack and is defined as:

$$\alpha \triangleq \arctan \frac{W}{U} \tag{A.4.1}$$

The wind axis is given by a rotation of β around the z^b-axis and is seen in Figure A.2. The angle β is defined in (A.4.2) where $V_T = \sqrt{U^2 + V^2 + W^2}$ is the absolute speed of the aircraft relative to the inertial frame.

$$\beta \triangleq \arcsin \frac{V}{V_T} \tag{A.4.2}$$

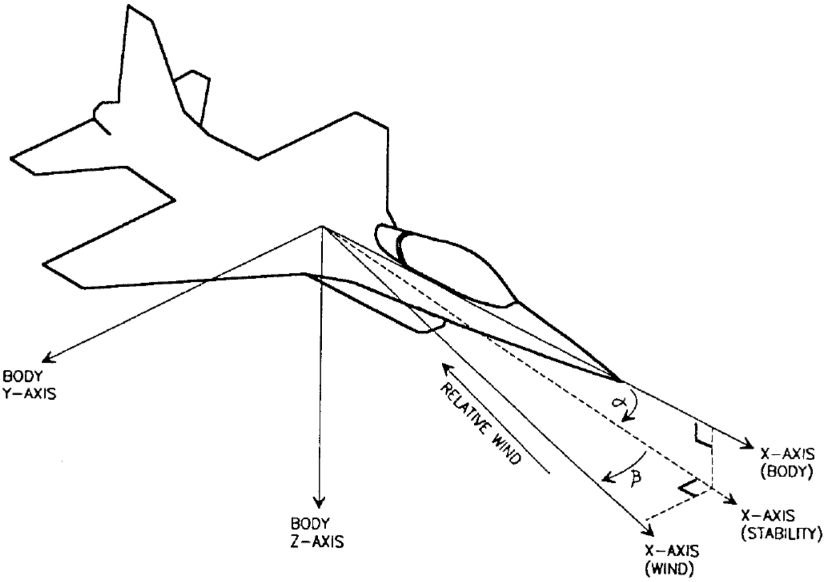


Figure A.2: Definitions of the stability axis and the wind axis given by the angles α and β respectively. (Stevens and Lewis 1992)

Appendix B
Parameter Values for the Cessna 172SP Model

These tables contain parameter values for the Cessna 172SP model developed in [Vistnes, 2012].

Table B.1: Geometry and Mass Properties of the Cessna 172SP, [Vistnes, 2012]

Parameter	**Description**	**Value**
Aircraft Mass and Moments of Inertia		
m	Aircraft mass	745 kg
I_x	Moment of inertia about x-axis	1502.2 kgm^2
I_y	Moment of inertia about y-axis	2862.0 kgm^2
I_z	Moment of inertia about z-axis	4044.5 kgm^2
Location of Propulsion System		
$\boldsymbol{r}^b_{t/b}$	Location of thrust relative to CO	$\begin{bmatrix} 2.225 \\ 0.000 \\ 0.122 \end{bmatrix}$
Wing-Shape Parameters		
b	Wing span	5.9 m
$c = \bar{c}$	Wing chord	1.6 m
S	Wing area	7.5 m^2

Table B.2: 1/2 - Aerodynamics Parameters for the Cessna 172SP: Force and Moment Coefficients, [Vistnes, 2012]

Parameter	**Description**	**Value**
Drag Coefficient		
C_{D_0}	Bias term	0.06
C_{D_α}	Angle of attack	0.58
$C_{D_{\alpha^2}}$	angle of attack squared	1.99
$C_{D_{T_C}}$	Trust coefficient	0.09
$C_{D_{\delta_f}}$	Flaps	0.35
$C_{D_{\delta_f^2}}$	Flaps squared	-0.13
Lift Coefficient		
C_{L_0}	Bias term	0.78
C_{L_α}	Angle of attack	12.43
$C_{L_{\delta_e}}$	Elevator	1.59
$C_{L_{\delta_f}}$	Flaps	3.44
$C_{L_{\delta_f^2}}$	Flaps squared	-2.34
Sideforce Coefficient		
C_{Y_0}	Bias term	0.001
C_{Y_β}	Sideslip Angle	-2.13
$C_{Y_{T_C}}$	Trust coefficient	0.07
$C_{Y_{\delta_r}}$	Rudder	0.40
C_{Y_p}	Damping Derivative	-0.51
C_{Y_r}	Damping Derivative	1.50
Rolling Moment Coefficient		
C_{l_β}	Sideslip Angle	-0.31
$C_{l_{T_C}}$	Trust coefficient	-0.04
$C_{l_{\delta_a}}$	Aileron	-1.97
$C_{l_{\delta_r}}$	Rudder	0.06
C_{l_p}	Damping Derivative	-6.57
C_{l_r}	Damping Derivative	1.59
Pitching Moment Coefficient		
C_{m_α}	Angle of Angle	-5.51
$C_{m_{\delta_e}}$	Elevator	-5.50
C_{m_q}	Damping Derivative	-43.85

Table B.3: 2/2 - Aerodynamics Parameters for the Cessna 172SP: Force and Moment Coefficients, [Vistnes, 2012]

Parameter	**Description**	**Value**
Yawing Coefficient		
C_{n_0}	Bias term	-0.02
C_{n_β}	Sideslip angle	0.91
$C_{n_{T_C}}$	Trust coefficient	0.01
$C_{n_{\delta_a}}$	Aileron	-0.17
$C_{n_{\delta_r}}$	Rudder	-0.37
C_{n_p}	Damping Derivative	-1.57
C_{n_r}	Damping Derivative	-2.05

Appendix C
DVD-attachment

A DVD is attached to this thesis. This includes all MATLAB Simulink diagrams and the necessary scripts to run the simulations. The files are listed below. Note that a mex compiler must be installed in order to compile the Simulink program.

- **cessna172.mdl:** This MATLAB Simulink model contains the complete simulator. In order to compile this file a mex-compiler has to be installed in MATLAB. The MATLAB script *run.m* initiates necessary constants.
- **Cessna172SP.png:** This picture is used by the Cessna 172SP simulation plant in *cessna172.mdl.*
- **flightplan.m:** This MATLAB script initiates the aircraft's flight plan given by the matrix $\boldsymbol{WP}$. Each column consists of a waypoint in the NED-frame.
- **init_Control_Constants.m:**. This MATLAB script initiates the PI constants of the kinematic controller. Constants for the reference model is also set here.
- **init_Model_Cessna172SP.m:** This MATLAB script initiates necessary constants for the simulation plant [Vistnes, 2012]. Also constants for the linear observer and the control plant is also initialized. These are position, velocity, Euler angles and Euler rates.
- **LinearModelMatricesLongitudinal.m** and **LinearModelMatricesLateral.m** are functions which returns linear state matrices for the decoupled model. These functions are used by the velocity and rate controller and the linear observer.
- **plottingsimulation.m:** This script can be used to plot the trajectories of the aircraft after a simulation.
- **run.m:** This MATLAB script initializes all necessary constants.

Appendix D

MATLAB-code

In this chapter the MATLAB-code from the Simulink diagram *cessna172.mdl* are presented. The Simulink diagrams can be found in the DVD-attachment in Appendix C. The figures D.1 to D.3 are simplified versions of the Simulink diagrams and show references to the MATLAB-code below.

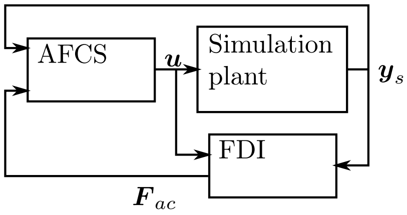


Figure D.1: A top level representation of the MATLAB Simulink diagram. The abbreviations are automatic flight control system (AFCS) and fault detection and isolation (FDI).

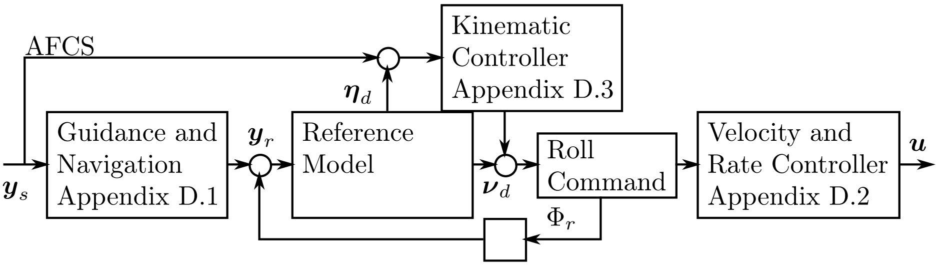


Figure D.2: The automatic flight control system (AFCS), with references to the MATLAB-codes

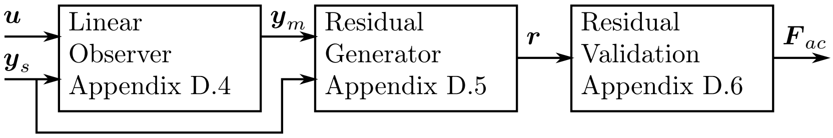

$\boldsymbol{u}$
$\boldsymbol{y}_s$
Linear Observer Appendix D.4
$\boldsymbol{y}_m$
Residual Generator Appendix D.5
$\boldsymbol{r}$
Residual Validation Appendix D.6
$\boldsymbol{F}_{ac}$

Figure D.3: The fault detection and isolation (FDI), with references to the MATLAB-codes

D.1 Guidance and Navigation

```
function [eta_ref, speed_d] = SetPoint_Generator...
(eta, alpha, beta, acceptanceCircles, WP)
persistent chi_p; %Path direction relative to north (NE-plane)
persistent nu_p; %Path direction relative to down
persistent R_rot;
persistent epsilon;
persistent i; %Current waypoint
if(isempty(i))
    i = 1;
    epsilon = [0 0 0]';
end

%% Navigation
Path = zeros(size(WP));
Path(:,1) = WP(:,i);
Path(:,2) = WP(:,i+1);


%% Guidance
% Determining track errors
△_N = Path(1,2)-Path(1,1);
△_E = Path(2,2)-Path(2,1);
△_D = Path(3,2)-Path(3,1);
chi_p = atan2(△_E,△_N);
nu_p = atan2( sqrt(△_N^2 + △_E^2),△_D ) - pi/2;
R_chi= [cos(chi_p)    -sin(chi_p)    0;
        sin(chi_p)     cos(chi_p)    0;
        0              0             1];
R_nu = [cos(nu_p)     0              sin(nu_p);
        0             1              0;
       -sin(nu_p)     0              cos(nu_p)];
R_rot = R_chi*R_nu;
epsilon = R_rot'*(eta(1:3)-Path(:,1));

% lookahead-based steering
R = 1300; %Tuning variable for LOS Steering
temp = R^2 - epsilon(2)^2;
if(temp<0)
    temp = 0;
end
```

```
Delta = sqrt(temp); %lookahead distance
nu_r = atan(epsilon(3)/sqrt(epsilon(2)^2 + (Delta)^2));
chi_r = atan(-epsilon(2)/Delta);

chi = chi_p + chi_r; % beta;
nu = nu_p + nu_r - alpha;
nu = min(15*pi/180, max(-15*pi/180, nu)); %easy stall control

chi = mod(chi,2*pi);  %0-2pi
eta_ref = [0 nu chi]';

speed_d = 50; %approx 100 knts

%Navigation
% Switching waypoint if steering point reaches next waypoint
if ( sqrt((Path(1,1)-Path(1,2))^2 + (Path(2,1)-Path(2,2))^2)...
 < epsilon(1)+Delta)
    if (i)≠ length(WP(1,:))-1
        i=i+1;
    end
end
end
```

D.2 Velocity and Rate Controller

```
function △ = Velocitycontroller(eta_system,nu_system,
    alpha_system,beta_system, nu_d,
    F_ac, kinetics,other,aero,thrust,init)
eta_system = [0, 0, 0, eta_system(1), eta_system(2),eta_system(3)]';
coder.extrinsic('lqr'); %between mex and matlab
stepsize = 0.01/4;% Stepsize Because of runge kutta 4,
    % the matlab func is running 4 times per stepsize.
persistent step state lin_fault
if(isempty(step))
    step = 0;
    state = 1; %Initation of matrices
    lin_fault = 1; %linearize with fault;
end
step = step + 1;
% Initiating the System Matrices
    persistent M_lo N_lo B_bar_lo C_lo C_la N_F_lo
        M_la N_la B_bar_la N_F_la A_lo B_lo A_la
        B_la G1_lo G2_lo G1_la G2_la;
    if(isempty(M_lo))
        M_lo = zeros(3);
        N_lo = zeros(3);
        B_bar_lo = zeros(3);
        N_F_lo = zeros(3);
        M_la = zeros(3);
        N_la = zeros(3);
        B_bar_la = zeros(3,2);
        N_F_la = zeros(3);
        A_lo = zeros(3);
        A_la = zeros(3);
        B_lo = zeros(3,3);
        B_la = zeros(3,2);
        C_lo = [1 0 0; 0 0 1];
        C_la = [zeros(2,1), eye(2)];
        G1_lo = zeros(3,3);
        G2_lo = zeros(3,2);
        G1_la = zeros(2,3);
        G2_la = zeros(2,2);
    end
%Initiating the Nominal and Perturbation state values
    persistent U_0 V_0 W_0 P_0 Q_0 R_0 alpha_0 beta_0
    phi_0 theta_0 psi_0;
    if(isempty(U_0))
        alpha_0 = init.alpha;
        beta_0 = 0;%init.beta;
        U_0 = init.v(1);
        V_0 = 0; W_0 = 0; P_0 = 0; Q_0 = 0; R_0 = 0; phi_0 = 0;
        theta_0 = 0; psi_0 = 0;
    end
% Initiating the Nominal Control Values
    persistent △_T_0 △_E_0 △_F_0 △_A_0 △_R_0;
    if(isempty(△_T_0))
        △_T_0 = 750; △_E_0 = 0; △_F_0 = 0; △_A_0 = 0;
        △_R_0 = 0;
        end

        %% State Machine
        linearization = 0;
        switch state
            case 1 %initiation of matrices
                linearization = 1;
                state = 0;
            otherwise %No linearization
        end
    %% If Fault has occured linarization is necessery
    if(and((F_ac(1)|F_ac(2)|F_ac(3)|F_ac(4)),
       step*stepsize>init.initPeriod))
      if(lin_fault)
         linearization = 1;
         lin_fault = 0;
      end
    end

    if(linearization == 1)
        % Constants
        m = kinetics.m;
        g = other.g;
        c_bar = aero.wingShape.c_bar;
        S = aero.wingShape.S;
        b = aero.wingShape.b;
        rho = other.rho_air;
        [J,J11,J22] = eulerang(phi_0,theta_0,psi_0);

        %% Longitudinal Channel
        [M_lo, N_lo, G_lo, B_bar_lo, N_F_lo, nu_0_dot_lo] =
        LinearModelMatricesLongitudinal(alpha_0, U_0, V_0, W_0, P_0, Q_0,
        R_0, phi_0, theta_0,△_T_0, △_E_0, △_F_0,
        m,g,c_bar,S,rho,aero,thrust,kinetics);

        M_inv_lo = M_lo^(-1);
        A_lo = -M_inv_lo*(N_lo - N_F_lo);
        B_lo = M_inv_lo*B_bar_lo;

        % Tuning according to bryson rule:
            u_max = 2;
            q_max = 0.1*pi/180;
            thrust_max = 60;
            elevator_max = 0.7 *pi/180;
            flaps_max = 1 *pi/180;
        Q_lo = diag([1/u_max^2, 1/q_max^2]);
        R_lo = diag([1/thrust_max^2, 1/elevator_max^2, 1/flaps_max^2]);

        %Fault accomodation
        %on form △_a △_e △_f △_r △_t
        C_acc_lo = [0 0 0 0 1;
                    0 1 0 0 0;
                    0 0 1 0 0];
        Theta_lo = [0 0.9 0];% loss of effectiveness
        Gamma_lo = eye(3)-diag(C_acc_lo*F_ac)*diag(Theta_lo);
        B_lo_new = B_lo*Gamma_lo;
        B_lo = B_lo_new;
    % Calculation of Gains
    [K_lo,P_lo,eig_lo] = lqr(A_lo,B_lo,C_lo'*Q_lo*C_lo,R_lo);
    G1_lo = -R_lo^-1* B_lo'* P_lo;
    G2_lo = -R_lo^-1*B_lo'*((A_lo + B_lo*G1_lo)')^-1*C_lo'*Q_lo;

    %% Lateral Channel
    [M_la,N_la,G_la,B_bar_la,N_F_la,nu_0_dot_la ] =
    LinearModelMatricesLateral(beta_0, U_0, W_0, P_0, Q_0, R_0,
    Δ_T_0, Δ_A_0, Δ_R_0, phi_0, theta_0,
    m,g,b,S,rho,aero,kinetics);
    M_inv_la = M_la^(-1);
    A_la = -M_inv_la*(N_la - N_F_la);
    B_la = M_inv_la*B_bar_la;

    % Tuning according to bryson rule:
        p_max = 1*pi/180;
        r_max = 40*pi/180;
        ailerons_max = 2 *pi/180;
        rudder_max = 1 *pi/180;
    Q_la = diag([1/p_max^2, 1/r_max^2]);
    R_la = diag([1/ailerons_max^2, 1/rudder_max^2]);

    %Fault accomodation
    %on form Δ_a Δ_e Δ_f Δ_r Δ_t
    C_acc_la = [1 0 0 0 0;
                0 0 0 1 0];
    Theta_la = [0.9 0]';% loss of effectiveness

    Gamma_la = eye(2)-diag(C_acc_la*F_ac)*diag(Theta_la);
    B_la_new = B_la*Gamma_la;
    B_la = B_la_new;
    % Calculation of Gains
    [K_la,P_la,eig_la] = lqr(A_la,B_la,C_la'*Q_la*C_la,R_la);
    G1_la = -R_la^-1* B_la'* P_la;
    G2_la = -R_la^-1* B_la'* ((A_la+B_la*G1_la)')^-1 * C_la' * Q_la;
end

%% Control calculations

% Longitudinal model
y_d_lo = [nu_d(1), nu_d(5)]';
x_lo_0 = [U_0, alpha_0, Q_0]';
u_lo_0 = [Δ_T_0,Δ_E_0,Δ_F_0]';
x_system_lo = [nu_system(1), alpha_system, nu_system(5)]';
u_lo = u_lo_0+G1_lo*(x_system_lo-x_lo_0)+G2_lo*(y_d_lo-C_lo*x_lo_0);

% Lateral model
y_d_la = [nu_d(4), nu_d(6)]';
x_la_0 = [beta_0, P_0, R_0]';
u_la_0 = [Δ_A_0, Δ_R_0]';
x_system_la = [beta_system, nu_system(4), nu_system(6)]';
u_la = u_la_0+G1_la*(x_system_la-x_la_0)+G2_la*(y_d_la-C_la*x_la_0);

% Control Signal
Δ = [-u_la(1), u_lo(2), u_lo(3), -u_la(2), u_lo(1)]';
```

D.3 Kinematic Controller

Roll:

```
function [u_roll, error] = PID_roll(PID, phi_tilde, integral_in)
%PID constants:
Kp = PID.roll.Kp;
Ki = PID.roll.Ki;
error = phi_tilde;
u_roll = Kp*error + Ki*integral_in;
end
```

Pitch:

```
function [u_pitch, error] = ...
    PID_pitch(PID, theta_tilde,integral_in)
%PID constants:
Kp = PID.pitch.Kp;
Ki = PID.pitch.Ki;
error = theta_tilde;
u_pitch = Kp*error + Ki*integral_in;
end
```

Yaw:

```
function [u_yaw, error] = PID_yaw(PID, psi_tilde, integral_in)
%PID constants for bank turn
Kp = PID.yaw.Kpbank;
Ki = PID.yaw.Kibank;
error = psi_tilde;
% 0 - 360 problem fix
if(abs(error)>pi)
   error = error - 2*pi*error/abs(error);
end
u_yaw = Kp*error + Ki*integral_in;
end
```

D.4 The Linear Observer

```
function [eta_out, nu_out,y_m] = LinearObserver(control_surfaces,
  eta_system, nu_system, alpha_system, beta_system, kinetics,
  aero, other, thrust, init,Scenario)
stepsize = 0.01/4;% Stepsize Because of runge kutta 4,
  %the matlab func is running 4 times per stepsize.
persistent step state
if(isempty(step))
  step = 0;
  state = 1; %Initation of matrices
end
step = step + 1;

% Initiating the Nominal Control Values
  persistent ∆_T_0 ∆_E_0 ∆_F_0 ∆_A_0 ∆_R_0;
  if(isempty(∆_T_0))
    ∆_T_0 = 750; ∆_E_0 = 0; ∆_F_0 = 0; ∆_A_0 = 0; ∆_R_0 = 0;
  end
% Initiating the System Matrices
  persistent M_lo N_lo G_lo B_lo N_F_lo nu_0_dot_lo
    M_la N_la G_la B_la N_F_la nu_0_dot_la;
  if(isempty(M_lo))
     M_lo = zeros(3);
     N_lo = zeros(3);
     G_lo = zeros(3,1);
     B_lo = zeros(3);
     N_F_lo = zeros(3);
     nu_0_dot_lo = zeros(3,1);
     M_la = zeros(3);
     N_la = zeros(3);
     G_la = zeros(3,1);
     B_la = zeros(3,2);
     N_F_la = zeros(3);
     nu_0_dot_la = zeros(3,1);
  end
%Initiating the Nominal and Perturbation state values
  persistent eta_lo_next nu_lo_next eta_la_next nu_la_next
    U_0 V_0 W_0 P_0 Q_0 R_0
    alpha_0 beta_0 phi_0 theta_0 psi_0;
  if(isempty(eta_lo_next))
    alpha_0 = init.alpha;
    beta_0 = init.beta;
    U_0 = init.v(1);
    V_0 = 0;
    W_0 = 0;
    P_0 = 0;
    Q_0 = 0;
    R_0 = 0;
    phi_0 = 0;
    theta_0 = 0;
    psi_0 = 0;
    %initial eta and nu vectors => ∆_eta/∆_nu = 0.
    eta_lo_next = theta_0;
    nu_lo_next = [U_0 , alpha_0, Q_0]';
```

```
                eta_la_next = [phi_0 psi_0]';
                nu_la_next = [beta_0 P_0 R_0]';
            end
        %This step
        eta_lo_0 = theta_0;
        nu_lo_0 = [U_0, alpha_0, Q_0]';
        eta_la_0 = [phi_0, psi_0]';
        nu_la_0 = [beta_0, P_0, R_0]';
        ∆_eta_lo = eta_lo_next - eta_lo_0;
        ∆_nu_lo = nu_lo_next - nu_lo_0;
        ∆_eta_la = eta_la_next - eta_la_0;
        ∆_nu_la = nu_la_next - nu_la_0;

        %% State Machine
        update_interval = 20; %[s] updating nominal values
        if(state ≠ 1)
            if(step*stepsize == init.initPeriod)
                state = 2; %linearizing after initiation period
            else if(mod(step*stepsize,update_interval) == 0)
                    %Linearize with a period of "update_interval" s
                    state = 2;
                else
                    state = 0; %Not linearizing
                end
            end
        end

        linearization = 0;
        switch state
            case 1 %initiation of matrices
                linearization = 1;
                state = 0; %normal opperation

            case 2 %Trajectory update and linearization U
                 % updating model with system parameters

            switch Scenario
                case 1
                    linearization = 1;
                    ∆_eta_lo = eta_system(5) - theta_0;
                    ∆_nu_lo = [nu_system(1) - U_0,
                        alpha_system - alpha_0, nu_system(5)]';
                    ∆_eta_la = [eta_system(4) - phi_0,
                        eta_system(6) - psi_0]';
                    ∆_nu_la = [beta_system - beta_0,
                        nu_system(4), nu_system(6)]';
                    ∆_T_0 = control_surfaces(5);
                    ∆_F_0 = control_surfaces(3);
                case 2
                    linearization = 1;
                    theta_0 = eta_system(5);
                    U_0 = nu_system(1);
                    alpha_0 = atan(nu_system(3)/nu_system(1));
                    ∆_eta_lo = 0;
                    ∆_nu_lo = [0 0 0]';
                    ∆_eta_la = [eta_system(4) eta_system(6)]';
                    ∆_nu_la = [
                    asin(nu_system(2)/(sqrt(nu_system(1)^2
                        + nu_system(2)^2 + nu_system(3)^2)))
                    nu_system(4) nu_system(6)]';
                Δ_T_0 = control_surfaces(5);
                Δ_F_0 = control_surfaces(3);
            case 3
            case 4
        end
        otherwise %No linearization
    end

    if(linearization == 1)
        % Constants
        m = kinetics.m;
        g = other.g;
        c_bar = aero.wingShape.c_bar;
        S = aero.wingShape.S;
        b = aero.wingShape.b;
        rho = other.rho_air;
        % Linearization
        [M_lo, N_lo, G_lo, B_lo, N_F_lo, nu_0_dot_lo] =
            LinearModelMatricesLongitudinal(alpha_0,
            U_0, V_0, W_0, P_0, Q_0, R_0,
            phi_0, theta_0,Δ_T_0, Δ_E_0, Δ_F_0,
            m,g,c_bar,S,rho,aero,thrust,kinetics);
        [M_la,N_la,G_la,B_la,N_F_la,nu_0_dot_la ] =
            LinearModelMatricesLateral(beta_0,
            U_0, W_0, P_0, Q_0, R_0,
            Δ_T_0, Δ_A_0, Δ_R_0, phi_0, theta_0,
            m,g,b,S,rho,aero,kinetics);
    end
    %Control inputs
    Δ_a = control_surfaces(1) - Δ_A_0;
    Δ_e = control_surfaces(2) - Δ_E_0;
    Δ_f = control_surfaces(3) - Δ_F_0;
    Δ_r = control_surfaces(4) - Δ_R_0;
    Δ_t = control_surfaces(5) - Δ_T_0;

    %Calculation of the next steps
    V_T = U_0;
    %Longitudinal Channel
    u_lo = [Δ_t   Δ_e     Δ_f]';
    eta_lo_0 = theta_0;
    nu_lo_0 = [U_0, alpha_0, Q_0]';

    nu_lo_next = Δ_nu_lo + nu_lo_0 +
        stepsize*(nu_0_dot_lo +
        M_lo^(-1)*(-G_lo*Δ_eta_lo -
        (N_lo - N_F_lo)*Δ_nu_lo + B_lo*u_lo));
    eta_lo_next = Δ_eta_lo + eta_lo_0 + stepsize*(nu_lo_next(3));

    eta_out_lo = eta_lo_next;
    nu_out_lo = nu_lo_next;

    %Lateral Channel
    u_la = [Δ_a Δ_r]';
    eta_la_0 = [phi_0, psi_0]';
```

```
nu_la_0 = [beta_0, P_0, R_0]';

nu_la_next = ∆_nu_la + nu_la_0 +
    stepsize*(nu_0_dot_la +
    M_la^(-1)*(-G_la*∆_eta_la(1) -
    (N_la - N_F_la)*∆_nu_la + B_la*u_la));
eta_la_next = ∆_eta_la + eta_la_0 +
    stepsize*([1 tan(phi_0);
    0 1/cos(theta_0)]*(nu_la_next(2:3,1)));

%% Output
y_m = [eta_out;nu_out];
end
```

D.5 Residual Generator

```
function residual      = ResidualGeneration(y_m, y_s)

H_y = [0 0 0 1 0 0 0 0 0 0 0 0;
       0 0 0 0 1 0 0 0 0 0 0 0;
       0 0 0 0 0 0 1 0 0 0 0 0;
       0 0 0 0 0 0 0 0 0 1 0 0;
       0 0 0 0 0 0 0 0 0 0 1 0]; %phi theta U P Q
H_u = -H_y;
residual = H_u*y_m + H_y*y_s;
end
```

D.6 Residual Validation

```
function [var,Threshold_vec, decision, FaultyActuators]
     = ResidualValidation(residual,init)
stepsize = 0.01/4;% Stepsize Because of runge kutta 4,
    % the matlab func is running 4 times per stepsize.
persistent step
if(isempty(step))
    step = 0;
end
step = step + 1;

%% Variance testing
numb_res = length(residual);
persistent y n x x_mean;
if(isempty(y))
   y = zeros(numb_res,1);
   x_mean = zeros(numb_res,1);
   %% Tuning variable WINDOW size:
   n = 400;
   x = zeros(numb_res,n-1);
end
x_mean_last = x_mean;
x_mean = x_mean_last + 1/n*(residual-x(:,1));
y_last = y;
y = y_last + 1/n*(residual.*residual - x(:,1).*x(:,1));
var = n/(n-1)*(y - x_mean.*x_mean);

%historical window update
for i=1:n-2;
    x(:,i) = x(:,i+1);
end
x(:,n-1) = residual;

%% decision
% Variance Tresholds
% defining tresholds:
roll_threshold = 2*10^-5;
pitch_threshold = 2*10^-5;
U_threshold = 5*10^-2;
roll_r_threshold = 5*10^-5;
pitch_r_treshold = 2*10^-5;

Threshold_vec = [roll_threshold, pitch_threshold, ...
    U_threshold, roll_r_threshold, pitch_r_treshold]';
decision = var>Threshold_vec;
if(step*stepsize > init.initPeriod + 5) %initation period
    %% △_a △_e △_f △_r △_t
    FaultyActuators = [decision(1), decision(2), 0 0 0]';
else
    FaultyActuators = [0 0 0 0 0]';
end
end
```