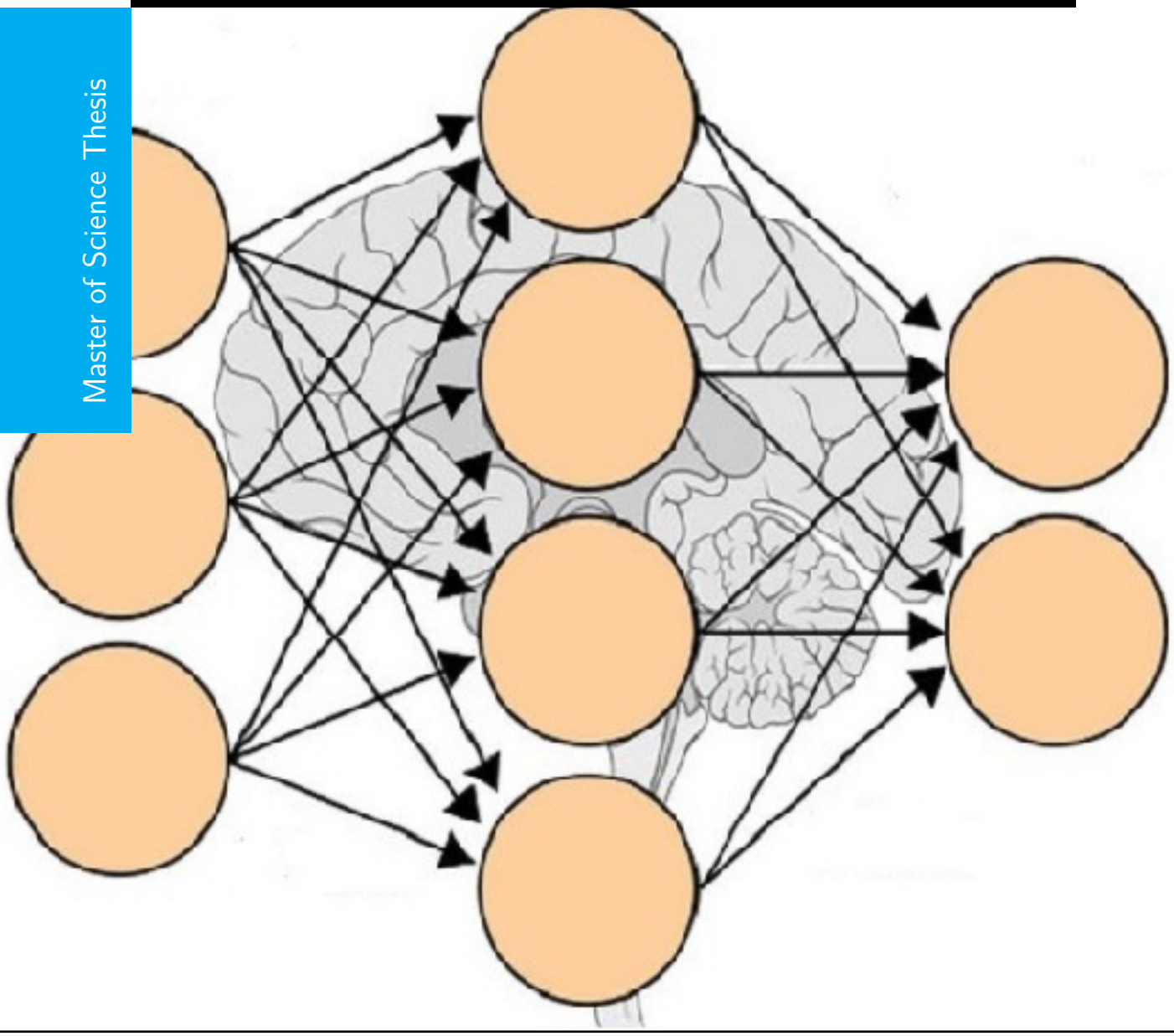
Knowledge-based Control Systems for Re-entry Vehicles

D. Brinkman

Master of Science Thesis



TU Delft Delft University of Technology

Astrodynamics and Space Missions

Knowledge-based Control Systems for Re-entry Vehicles

Master of Science Thesis

For the degree of Master of Science in Aerospace Engineering at Delft University of Technology

D. Brinkman

January 2, 2017

Faculty of Aerospace Engineering · Delft University of Technology

Copyright © Astrodynamics and Space Missions
All rights reserved.

The cover image is a model of the basic structure of a neural network, on top of an image of a cross section of a human brain, which the neural network is a model off. The picture is adapted from pictures found in Babuska (2010).

Preface

This thesis describes the work undertaken to obtain the Master of Science degree at the faculty of Aerospace Engineering, at the Delft University of Technology. The hypersonic atmospheric re-entry phase is the most challenging phase of a return flight from space. Attitude control systems are required to ensure a safe return of the entry vehicle. In the non-aerospace industry, a type of *knowledge-based* or *expert* control systems are commonly used to improve the performance of conventional control system. Can it do the same for the attitude control of an entry vehicle? I very much enjoyed working on such a interesting and challenging topic related to applying intelligent control systems to entry vehicles.

This report is written first and foremost for researchers in the field of intelligent control systems for entry vehicles. In addition, this report can be used by others in unrelated projects who wish to gain knowledge on the intricacies of applying knowledge-based control systems in numerical simulations.

I am grateful to my thesis supervisor Dr. Ir. E. Mooij, for his valuable support and guidance during the whole thesis. Additionally, a special thanks goes to the parents and (step-)brother(s) for putting up with not seeing me as much as any of us would have liked, and for giving me the support and guidance I needed during my studies.

Dennis Brinkman
Delft, January 2, 2017

Summary

Returning to Earth from space is no easy feat, because all of the orbital energy has to be dissipated into the atmosphere to be able to safely land. To accomplish this, the entry-vehicle requires the use of a Guidance, Navigation and Control (GNC) system to ensure mission success. This thesis focused on the control system and the many different techniques that exist to determine the required control action to eliminate differences between the actual and command state. Almost always, conventional control techniques such as Proportional, Integral, and Derivative (PID) control and the Linear Quadratic Regulator (LQR) are used to eliminate the differences between the two states. However, in the industry, a type of *knowledge-based* or *expert* control systems are used that, unlike the conventional controllers, do not require the use of mathematical model to describe the behaviour of the system.

In the past, aerospace vehicles have been designed to employ such Knowledge-based Control Systems (KBCSs), such as Fuzzy Logic Control (FLC) and Neural Network Control (NNC). Furthermore, another type of knowledge-based control methodology that can be used is known as Model Predictive Control (MPC). Essential in MPC is the explicit use of a model that can simulate dynamic behaviour of the process at a certain operation range. After a literature study, a knowledge-gap was identified relating to the comprehensive comparison of conventional and knowledge-based control systems when applied to entry-GNCs. Based on the knowledge-gap, the research question of this thesis was defined as:

Can knowledge-based control systems outperform conventional control systems when applied to entry-GNC systems?

The main research question can be split into three sub-questions which are formulated as follows:
When compared to conventional control systems,

1. **can knowledge-based control systems improve the performance of entry-GNC systems?**
2. **can knowledge-based control systems improve the robustness of entry-GNC systems?**
3. **can the increase in implementation effort for knowledge-based control systems be justified with a corresponding increase in control performance?**

To obtain answers to the research questions, KBCSs must be compared to conventional control system for an entry vehicle. To this end, the HORUS-2B was chosen as a reference entry vehicle. The last bank-reversal of the nominal control commands was selected to be the point along the reference trajectory about which the performance of the control systems were evaluated. For this bank-reversal, all control surfaces and yaw thrusters were active. Additionally, for the best three controllers, a one degree disturbance in the angle of attack and sideslip angle were evaluated at the same initial conditions as for the bank-reversal. Furthermore, the control performance was evaluated for a point along the reference trajectory where the vehicle used all reaction control thrusters and control surfaces simultaneously.

A selection was made of the KBCSs that were to be compared to conventional control system. The KBCSs selected was an NNC optimized using the NEAT algorithm, an FLC optimized using a pattern-search algorithm, and two MPCs. The first MPC used a so-called Input-Output (IO)-model and the second an Increment-Input-Output (IIO)-model. The later model uses information of the states of both the current and previous time-steps, and computed the control actions as the difference (increment) between the current and previous time-steps. These KBCSs were compared to LQR benchmark controllers mirroring the original controller of the reference vehicle.

Because only a subset of the control actuators of the reference vehicle is active at any given time, dependent on the dynamic pressure, the concept of moment-fractions was introduced to allow the controller architecture to be independent of which actuators are active. This is particularly useful for the NNCs and FLCs, whose controller architecture depends on the number of control outputs. Therefore, the required moment-fractions were used as the control outputs, and in a later stage were allocated to the active actuators. Based on these moment-fractions, an additional benchmark controller, the Moment-fraction Controller (MFC), was defined which used the moment-fractions as the control variables. Furthermore, integral terms were added to the original benchmark controller to obtain the Linear Quadratic Integral Controller (LQIC). A fourth benchmark controller, the Moment-fraction Integral Controller (MFIC), was designed incorporating integral terms and used moment-fractions as the control variables. Additionally, NNCs and FLCs with and without integral were designed. Furthermore, MPCs with an IO- and IIO-model were created analogous to each of the benchmark controllers. Hence, a total of 16 controller were designed.

The moment-fractions output of the control systems needed to be transformed to a set of elevon, rudder, and thruster commands. To this end, an actuator assignment algorithm was developed that inverted the numerical aerodynamic database of the reference vehicle and computed the actuator commands from a set of commanded moment coefficients, which are computed directly from the moment-fractions.

To compute the behaviour of the vehicle during a bank-reversal, numerical simulations are required. Therefore, a detailed overview was given of the theoretical basis used to simulate the environment which the reference vehicle interacts with. Furthermore, a mathematical description and details of the numerical implementation of the different control systems was given. Based on the obtained theoretical knowledge, a numerical simulator was constructed to allow simulations to evaluate the performance of the controllers. The inner details of the simulator were described and its components were verified for correct behaviour and implementation.

A comparative analysis was made of the designed controllers to derive insights into the performance and the sensitivity to system variations of the controllers. The obtained insights directly relate to the answers of the sub-questions of the research question. It was shown that introducing moment-fractions and/or integral terms to the benchmark controller could significantly increase its nominal state-response performance, at the cost of a higher control effort. Additionally, it was found that the NNC and the FLC have a faster nominal bank angle response than the MFC, which comes at the cost of a larger induced sideslip angle and angle of attack, and an increased total control effort. When integral terms where added to the NNC and the FLC, it was found that the former optimized in such a way that the effect of the integral terms were decreased. The resulting nominal performance of the Neural Network Integral Controller (NNIC) was worse than that of the MFIC. The Fuzzy Logic Integral Controller (FLIC), however, performed similar to the

MFIC, albeit at the cost of increased control performance and control oscillations. Comparison of the MPC_{IO}^{mfc} and the MPC_{IIO}^{mfc} with the MFC revealed that the MPCs have a higher overall performance. Additionally, the MPC_{IO}^{mfic} and the MPC_{IIO}^{mfic} have a nominal performance similar to that of the MFIC, and all three have a higher overall performance than the MFC and its MPC analogues. Hence, it was concluded that KBCSs *can* improve the nominal performance of entry-GNC systems when compared to conventional control systems, but more complex systems of the latter type shows equal performance improvement. This answered the first sub-question of the main research question.

Furthermore, a comparative analysis was made of the state and control response sensitivity to system variations, including an analysis on the variations of control oscillations. It was shown that the NNCs and the FLCs are equally sensitive to system variations as the corresponding benchmark controllers. In general, however, the MPCs are more robust than all benchmark controllers. Hence, it was concluded that the KBCSs *can* improve the robustness of entry-GNC systems when compared to conventional control system, thereby answering the second sub-question of the main research question. Furthermore, the MPCs, in particular the MPC_{IIO}^{mfc} clearly demonstrated increased robustness when compared to the (conventional) benchmark controllers.

Lastly, a discussion was made on the implementation effort of the KBCSs compared to that of conventional control systems. It was found that, for the NNC and FLC, the increased design effort did not correspond with an increase in nominal performance. Furthermore, for the MPCs, the implementation effort increased only marginally, but that was the price paid for a significant reduction in the sensitivity to system variations. Hence, it was concluded that the drastic increase in implementation effort of the NNCs and FLCs was not justified by a corresponding increase in performance and robustness. However, it was concluded that the marginally increased implementation effort of the MPCs, the MPC_{IO}^{mfic} in particular, is justified by a corresponding increase in robustness of the controller compared to the benchmark controllers.

It was found that all three sub-question were answered positively. Hence, the main conclusion is that KBCSs *can* outperform conventional control system when applied to entry-GNC systems. In particular, the MPC_{IO}^{mfic} shows the greatest potential of the KBCSs analysed in this thesis to improve the performance and robustness of conventional control systems.

Table of Contents

Nomenclature

Acronyms

ACTIVE	Advanced Control Technology for Integrated VEhicles
AI	Artificial Intelligence
ANN	Artificial Neural Network
CoG	Centre of Gravity
CoM	Centre of Mass
DoF	Degrees of Freedom
EoM	Equations of Motion
FLC	Fuzzy Logic Control
FLIC	Fuzzy Logic Integral Controller
GA	Genetic Algorithm
GGNCSim	Generic Guidance, Navigation, and Control Simulator
GMST	Greenwich Mean Sidereal Time
GNC	Guidance, Navigation and Control
IIO	Increment-Input-Output
IO	Input-Output
KBCS	Knowledge-based Control System
LQIC	Linear Quadratic Integral Controller
LQPC	Linear Quadratic Predictive Control
LQR	Linear Quadratic Regulator
LTI	Linear-Time-Invariant
MFC	Moment-fraction Controller
MFIC	Moment-fraction Integral Controller
MIMO	Multiple Input, Multiple Output
MPC	Model Predictive Controller
MPC	Model Predictive Control
NA	Not Active
NEAT	Neuro-Evolution of Augmenting Topologies
NFC	Neuro-Fuzzy Control
NNC	Neural Network Control
NNIC	Neural Network Integral Controller
PD	Proportional and Derivative
PID	Proportional, Integral, and Derivative

RCS	Reaction Control System
RNG	Random Number Generator
SAD	Software Architecture Diagram
SISO	Single Input, Single Output
SPCP	Standard Predictive Control Problem
TAEM	Terminal Area Energy Management
TS	Takagi-Sugeno
TUDAT	TU Delft Astrodynamics Toolbox
UAV	Unmanned Aerial Vehicle
ZMWN	Zero-Mean White Noise

Latin Symbols

$\mathbf{A}$	state matrix	various
$\mathbf{B}$	input matrix	various
b_{ref}	reference length for roll/yaw	m
$\mathbf{C_{BA}}$	transformation matrix from frame A to frame B	-
$\mathbf{C}$	output matrix	various
C_l	roll-moment coefficient	-
C_{m_0}	trimmed pitch-moment coefficient	-
C_m	pitch-moment coefficient	-
C_n	yaw-moment coefficient	-
C_D	drag coefficient	-
C_L	lift force coefficient	-
C_S	side force coefficient	-
c_{ref}	reference length for pitch	m
$\mathbf{D}$	transmission matrix	various
D	number of disjoint genes	-
D	drag force	N
d	dead time of the process	s
E	number of excess genes	-
e	ellipticity of ellipsoid	-
f_i'	adjusted fitness of a genome	various
f_i	fitness of a genome	various
h	altitude	m
$\mathbf{I}$	mass moment of inertia tensor	kg m^2
$\mathbf{I}$	identity matrix	-
$\mathbf{I}$	identity matrix	-
I	mass moment of inertia	kgm^2
J	cost function	various
$\mathbf{K}$	gain matrix	various
$\mathcal{L}$	Lagrangian	various
$\mathcal{L}$	roll-moment	Nm
L	lift force	N
$\tilde{\mathbf{M}}_{cm}$	sum of external, Coriolis, and relative moments about the CoM, expressed in components along the body	Nm
$\mathbf{M_A}$	moment vector of aerodynamic origin	Nm
M_{nav}	Mach number as given by the navigation system	-
M_p	maximum overshoot	-
$M_{T,x}$	roll-thruster moment	Nm
$M_{T,y}$	pitch-thruster moment	Nm
$M_{T,z}$	yaw-thruster moment	Nm

M	pitch-moment	Nm
N_c	control horizon	-
N_m	minimum cost-horizon	-
N	prediction horizon	-
N	yaw-moment	Nm
n	dimension of the system	-
net	neuron output	various
o	neuron activation level	various
p	roll-rate	rad/s
$\mathbf{Q}$	state weighting-matrix	various
$\bar{q}$	dynamic pressure	Nm^{-2}
Q	attitude quaternion	-
q	pitch-rate	rad/s
$\mathbf{R}$	input weighting-matrix	various
R_e	planetary radius at the equator	m
R_p	planetary radius at the poles	m
$\mathbf{r}_{cm}$	location of the CoM	m
r	yaw-rate	rad/s
$\mathbf{S}$	active constraint indicator	-
S_{ref}	reference area	m^2
T_s	sample time	s
t_d	delay time	s
t_p	peak time	s
t_r	rise time	s
t_s	settling time	s
$\mathbf{u}$	input vector	various
V	speed	m/s
$\check{\mathbf{v}}$	concatenated vector of the predictions of the input vector	various
$\mathbf{v}$	input signal vector	various
$\bar{W}$	average weight difference of matching genes	-
w	neural network connection weight	various
$\mathbf{x}$	state signal vector	various
$\mathbf{x}$	position vector	m
x	neural net input	various
$\mathbf{y}$	process output vector	various
$\hat{\mathbf{z}}$	prediction of the performance signal vector	various
$\tilde{\mathbf{z}}$	concatenated vector of the predictions of the performance signal vectors	various
$\mathbf{z}$	performance signal vector	various

Greek Symbols

α_c	commanded angle-of-attack	rad
α	angle of attack (no wind assumption)	rad
β	degree of fulfilment	-
β	sideslip-angle (no wind assumption)	rad
$\bar{\mathbf{\Gamma}}$	block diagonal matrix of the diagonal selection matrix	-
$\mathbf{\Gamma}$	diagonal selection matrix	-
γ	flight-path angle	rad
$\delta^\star$	geodetic latitude	rad
δ_a	aileron deflection	rad
$\delta_{e,l}$	left elevon deflection	rad

$\delta_{e,r}$	right elevon deflection	rad
δ_e	elevator deflection	rad
$\delta_{r,l}$	left rudder deflection	rad
$\delta_{r,r}$	right rudder deflection	rad
δ_r	rudder deflection	rad
$\delta_{b_{trim}}$	body-flap deflection required for trim	rad
$\delta_{e_{trim}}$	elevon deflection required for trim	rad
δ_t	compatibility threshold	-
δ	compatibility distance	-
η	moment-fraction	-
μ_A	membership function of the set A	-
$\boldsymbol{\Xi}$	matrix relating the rotation vector of the body frame to the time derivative of the attitude quaternion	-
σ	bank-angle (no wind assumption)	rad
$\hat{\boldsymbol{\phi}}$	prediction of the equality constraints	various
$\tilde{\boldsymbol{\phi}}$	concatenated vector of the predictions of the equality constraints	various
φ	neuron activation function	-
$\tilde{\boldsymbol{\Psi}}$	concatenated vector of the upper boundaries of the inequality constraints	various
$\hat{\boldsymbol{\Psi}}$	predictions of the upper boundaries of the inequality constraints	various
$\hat{\boldsymbol{\psi}}$	prediction of the inequality constraints	various
$\tilde{\boldsymbol{\psi}}$	concatenated vector of the predictions of the inequality constraints	various
$\boldsymbol{\omega}$	rotation vector	rad/s
ω_b	process bandwidth	Hz
ω_s	sampling frequency	Hz

Chapter 1

Introduction

The atmosphere surrounding the Earth makes space flight harder than it would be if it did not exist. It is sometimes said, half-jokingly, that getting into orbit is like getting "halfway to anywhere", because of the large amounts of energy necessary to reach it. Generally overlooked, however, are the difficulties that come with coming back to Earth from orbit safely. Such a re-entry flight consists of three phases (Mooij, 2014a):

- **Hypersonic atmospheric re-entry phase**
  The hypersonic atmospheric re-entry phase starts at an altitude of approximately 120 km and ends at an altitude of about 25 km. During this phase, the vehicle flies with a large angle of attack and resembles a typical blunt re-entry vehicle like, for example, the Orion. During this phase, the vehicle decelerates from an initial velocity of up to Mach 25 down to Mach 2.5.
- **Terminal area energy management phase**
  During the Terminal Area Energy Management (TAEM) phase, the vehicle is brought to the correct interface position with the landing phase with the correct amount of energy (velocity and altitude) and the correct heading.
- **Approach and landing phase**
  The last phase, the approach and landing phase, is responsible for the final part of the runway approach and the actual landing.

The hypersonic atmospheric re-entry phase is the focus of this thesis. Because this phase of the re-entry flight is usually un-powered, all of the energy used to get to orbit is dissipated into the atmosphere in the form of extreme heating, some of which will be transferred to the re-entry vehicle. For the thermo-mechanical loads to be as favourable to the vehicle as possible, an optimal trajectory, also known the *nominal* trajectory, is computed, and ensures that the trajectory constraints, such as maximum deceleration, thermal load, and landing place, are satisfied. Once the optimal trajectory has been computed, it must be verified that the vehicle can execute the required manoeuvres without violating any constraints. To ensure mission success, it must be guaranteed that the vehicle will still be able to fulfil its mission when encountering (unforeseen) disturbances, which force the vehicle away from its nominal trajectory. Therefore, the entry vehicle is equipped with a Guidance, Navigation and Control (GNC) system. “The task of the guidance system is to generate steering commands, taking a reference state, trajectory constraints, and/or a final state into account. For this task, the system requires information from the outside world, for

instance the current actual state. These data have to be provided by the navigation system, using sensor information and predefined theoretical models. The control system then has to guarantee that the steering commands are carried out, such that the actual attitude equals the command attitude in a reasonably short time and that this attitude is dynamically stable (trim stability)" (Mooij, 2014a).

This thesis will focus on the many different techniques that exist to determine the required control action to eliminate differences in the actual state w.r.t. the commanded state, known as the *state error*, at a point along the nominal reference trajectory. Most commonly, conventional control techniques such as Proportional, Integral, and Derivative (PID) control and the Linear Quadratic Regulator (LQR) are used to determine the mapping from the state error to the control action required to correct for the state error. PID control techniques have been used for the original attitude controller of the HORUS-2B (MBB, 1998), the X-33 (Hall et al., 1998), and recently also for the IXV (Zaccagnino et al., 2011). These vehicles are shown in Figures 1-1, 1-2, and 1-3, respectively. After linearisation of the control laws of the HORUS-2B, Mooij (1998) applied the LQR method to determine the optimal feedback gains.

Figure 1-1: The HORUS-2B as the second stage of the Sänger concept (de Ridder, 2009).

Figure 1-2: The X-33. Image courtesy of NASA.

Figure 1-3: The IXV. Image courtesy of ESA.

In conventional control theory, a mathematical model of the process to be controlled and specifications of the closed-loop behaviour are used to design a controller. This approach may not succeed if a model of the process is difficult to obtain, (partly) unknown, or highly non-linear. As a consequence, many processes in the industry that are controlled by humans cannot be automated

using conventional control techniques, since the performance of such controllers is often inferior to that of the human operators. Furthermore, humans combine various kinds of information and control strategies that cannot be integrated into a single analytical control law (Babuska, 2010). Due to these reasons, in the non-aerospace industry a type of *knowledge-based* or *expert* control systems are commonly used to automate the process while still having the higher performance a human operator provides, compared to conventional control techniques.

While the list of knowledge-based methodologies themselves is not extensive, the list of options within each methodology *is*. Not all options can be studied, however, and therefore a selection must be made. For Knowledge-based Control Systems (KBCSs), a variety of methodologies can be chosen, namely: Neural Network Control (NNC), Fuzzy Logic Control (FLC), Neuro-Fuzzy Control (NFC), and Model Predictive Control (MPC). Each can be described as follows (Klancar and Skrjanc, 2007; Rutkowski, 2004):

NNC Neural network systems are simple models imitating the function of biological neural systems. Information is implicitly coded in the network and its parameters. Their main strength is the ability to learn complex functional relations by generalizing from a limited amount of training data. If such training data is not available, one can train the neural network by using optimization techniques such as Genetic Algorithms (GAs) to achieve a specified performance.

FLC Fuzzy logic systems describe relations by means of if-then rules, such as 'if heating valve is open then temperature is high'. The ambiguity in the definition of the linguistic terms (e.g., *high temperature*) is represented by using *fuzzy sets*, which are sets with overlapping boundaries. In the fuzzy set framework, an element can simultaneously belong to multiple fuzzy sets with different degrees of membership. They form a suitable framework for representing qualitative knowledge.

NFC Fuzzy logic systems and neural networks are often combined in *Neuro-Fuzzy systems*, which effectively integrate qualitative rule-based techniques with data-driven learning. Neuro-Fuzzy systems exploit the fact that, at a computational level, a fuzzy system can be seen as a network, similar to neural networks. Therefore, it allows for the use of training algorithms commonly used for neural networks to train the NFC.

MPC Model predictive control is based on three main concepts: *i)* A model is used to predict the process output at future discrete time instant over an *prediction horizon*, *ii)* A sequence of future control action is composes over a *control horizon* by minimizing a given objective function, and *iii)* Only the first control action in the sequence is applied, the horizons are moved towards the future and the procedure is repeated. The main reason for its popularity is its constraint-handling capabilities.

Several studies have been performed analysing the performance of KBCSs when applied to entry vehicles. For example, Wu et al. (2000) implemented a fuzzy logic controller with an unique rulebase for each body-axis, and connected it to the simulator of the X-38, which was a re-entry demonstrator vehicle and a prototype for a space station crew return vehicle, and is shown in Figure 1-4. This controller takes the attitude error and its derivative for an axis as input, and outputs a single value encoding the percentage needed of the maximum actuator capacity for that axis. It showed that the body-flaps and rudders are able to conduct their part in the appropriate flight regions along the nominal trajectory. However, the bank-reversals in the early part of re-entry are too demanding for the thrusters, which was shown to be a hardware problem, as the controller assigned maximum actuator capacity to perform the bank-reversals. Furthermore, Wallner and Well (2001) have implemented a non-linear adaptive controller based on the architecture of a dynamic inversion controller, and applied it to the non-linear simulation of the X-38. The adaptive element is composed of a neural network and adaptive bounding terms. They showed that the neural networks functional approximation capabilities are sufficient for the tracking of a reference

trajectory. Additionally, it was shown that the inclusion of the adaptive element improves the tracking performance of the non-adaptive dynamic inversion controller. Wu et al. (2000) and Wallner and Well (2001) thus have shown that KBCSs can be successfully applied to entry vehicles.

Figure 1-4: The X-38. Image courtesy of NASA.

Figure 1-5: The LoFLYTE. Image courtesy of NASA.

Furthermore, several studies have been performed regarding the application of NNCs as controllers for a variety of systems. For example, McFarland and Calise (1995) presented an adaptive neural network that was applied to correct for linearisation errors in a standard PD-controller through on-line learning for an air-to-air missile. Similarly, McDowell and Irwin (1995) showed an adaptive neural network that was used in parallel with a linear controller to compensate for the difference in accelerations by the linear controller and a non-linear reference model, and applied it to the autopilot of a guided missile. Both studies involved the applications of on-line updating the neural network throughout the flight and have similar architectures. Both studies showed that the use of neural networks can greatly diminish the need for gain-scheduled designs and accurate aerodynamic modelling in conventional applications, and that they can improve the performance of the controller when compared to non-adaptive controllers.

Additionally, Cox et al. (1998) demonstrated LoFLYTE, which was a prototype vehicle that tested the subsonic airworthiness of the waverider shaped aircraft, and is shown in Figure 1-5. It used a neural network to adapt the gains of a Proportional and Derivative (PD) controller based on the output of a reference model, the output of the vehicle, and the difference between the two. It is interesting to note that the neural network learned to control the vehicle while it was flying, without prior information about the dynamics of the vehicle. Furthermore, Saini and Balakrishnan (1996) applied adaptive critic-based neural networks, which used successive adaptations of two neural networks until closed loop optimal control was achieved. The controller was applied to the auto-landing problem and deals with the longitudinal dynamics of an aircraft, which is to be landed in a specified touchdown region. It was shown through simulations that the neural network controller had much smoother behaviour than a PID-controller for the same problem and parameters. Furthermore, Zhang et al. (2008) showed an artificial neural network controller designed to control a fish-robot's locomotion. It used an artificial neural oscillator, which consists of two neurons, and it has been shown that it had excellent control properties compared to a PID-controller. Moreover, Yildirim (2008) used a neural network as an inverse controller for a four-legged planar walking robot. It showed that, when compared to a standard PID-controller, the neural networks inverse control was better at rejecting large load disturbances. These studies, and many others, have shown that NNCs can be successfully applied to a variety of control problems, ranging from the control of a walking robot to the automatic landing system of a large aircraft, and often have a higher control performance than a standard PID-controller. Additionally, these studies showed that they can be used as the main controller, similar to the human brain, or used to augment more conventional controllers, such as PID-controllers and dynamic inversion controllers.

Training of a neural network can be done by least-squares, iterative back-propagation methods or through genetic algorithms. In most training situations, the structure (number of neurons in the hidden layer and its connections) is fixed, however, Stanley and Miikulainen (2002) developed a method where both the structure and weights were optimized through genetic algorithms. They applied this method to optimize an NNC to balance a double-pole cart system. It showed that Neuro-Evolution of Augmenting Topologies (NEAT) can be successfully applied to the design of NNCs, and can do so faster than other neuro-evolution methods (Stanley and Miikulainen, 2002). Additionally, this technique has been successfully applied by Sethbling[1] to create an Artificial Intelligence (AI) for Mario to successfully complete a level of the video game Super Mario World.

FLCs have also been the subject of many control related studies. For example, Duerksen (1997) showed the successful implementation of the same fuzzy controller for bank angle control on both a piston powered single engine aileron equipped aircraft simulation, and on a business jet simulation which used spoilers for primary roll control. These studies showed that the same fuzzy controller architecture can be used for the control of different systems when tuned appropriately for each system. Furthermore, Nho and Agarwal (2000) developed a fuzzy logic control system for the automatic landing of an aircraft. It was shown that a simple tuned fuzzy controller designed for a longitudinal aircraft model is robust enough to give satisfactory performance in terms of stability, for both linear and non-linear aircraft dynamics. These studies showed that the fuzzy controller architecture is not dependent on the chosen actuators and system parameters, and that a relatively simple architecture can provide sufficient behaviour of the system. Additionally, Nour et al. (2007) implemented a fuzzy logic controller to control an inverted pendulum system, a commonly used benchmark system. Simulation results showed that the fuzzy logic controllers are superior compared to PID controllers in terms of overshoot, settling time, and response to parameter changes. Gaurav (2012) obtained similar results for a liquid flow controller. Additionally, Farid and Barakti (2013) designed a neuro-fuzzy controller to control the attitude of an Unmanned Aerial Vehicle (UAV). It was shown that the neuro-fuzzy controller has better pulse response than a conventional PID controller. Nour et al. (2007) and Gaurav (2012) also reported that the tuning of an FLC can be a daunting task when no expert-knowledge is available and/or there are many inputs and outputs. In this case, it is often chosen to use training methods commonly used for neural networks to train/tune the fuzzy logic controllers. Such a controller is known as an NFC. Training an NFC in this manner requires the use of training data which the controller learns to copy, or it has to be taught how to perform using *unsupervised learning* methods, such as GAs or grid searches (Babuska, 2010).

NFCs, like NNCs and FLCs, have been studied for the application to control systems. For example, Hui et al. (2006) developed several neuro-fuzzy approaches to determine a time-optimal, collision-free path of a car-like mobile robot navigating through a dynamic environment. It showed that tuned neuro-fuzzy controllers outperform poorly tuned fuzzy logic controllers and more standard controllers. Furthermore, it showed the possibility that such neuro-fuzzy controllers are fast enough that they could be implemented as on-line controllers. Additionally, Rutkowski (2004) showed a generalized neuro-fuzzy system that contained weighted and flexible parameters, allowing for a more finely tuned controller than traditional neuro-fuzzy systems allow.

It was mentioned by van den Boom (2013) that in the industry, for supervisory optimizing control of multi-variable processes, MPC is often preferred over other controller design methods, such as PID and LQR. A PID-controller is easily tuned but can only be applied in a straightforward manner to Single Input, Single Output (SISO) systems. LQRs can be applied to Multiple Input, Multiple Output (MIMO) systems, but cannot handle signal constraints in an adequate way. These techniques also exhibit difficulties in realizing robust performance for varying operating conditions. MPC is an easy-to-tune method, in principle there are only three basic parameters to be tuned, can handle constraints in an systematic way during the design and implementation of the controller, and can handle structural changes, such as changes in sensors and actuator, changes in

[1] Sethbling,"MarI/O - Machine Learning for Video Games," Youtube Video, https://www.youtube.com/watch?v=qv6UVOQ0F44 [Retrieved 18-10-2016], June 2015

system parameters and system structure by adapting the control strategy on a sample-by-sample basis. Essential in MPC is the explicit use of a model that can simulate dynamic behaviour of the process at a certain operating point. Because of the explicit use of a model, the input must be (among others) the state of the system. It is noted that the requirement of knowing the state of the system can be a serious drawback in the case of systems where not all of the states are known, such as for flexible vehicles. In such cases, predictions of future states are made based on the states that are known (van den Boom, 2013).

The major drawback of MPC is that the methodology is open, and many variations have led to a large number of MPC-methods. However, a single method can be selected. In the case when there is no reference signal, one can select a performance index that will be analogous to the cost-functional of the LQR-method. This *performance index* is known as Linear Quadratic Predictive Control (LQPC) (Garcia et al., 1989). Although the methodology of MPCs is very open, several studies have been performed regarding the application of MPCs to control problems. For example, Ghahramani and Towhidkhah (2009) showed that, if one takes both current and previous steps into account when predicting futures states, the control action provides faster response for a close loop system and improves robust stability. While Ghahramani and Towhidkhah (2009) did not apply their methodology to an LQPC, it can be derived for it. Furthermore, Klancar and Skrjanc (2007) designed a model-predictive trajectory-tracking control system and applied it to a mobile robot. It was shown that the MPC-controller performed similar to a time-varying state-tracking controller, although the MPC-controller gave better control results, which was as expected, as control effort was taken into account during the optimization process.

In the above paragraphs, it was shown that knowledge-based control systems can be successfully applied to an entry vehicle and other aerospace vehicles, and that in the non-aerospace industry the use of knowledge-based control systems can improve the performance of the controlled process, when compared to conventional control techniques. Therefore, it naturally raises the question of whether KBCSs could also improve the control performance for entry vehicles. However, no comprehensive study has been performed that compares the performance of a variety of knowledge-based techniques to conventional control techniques for entry vehicles. Besides studying whether KBCSs can outperform conventional control techniques for entry problems, one can also compare the different controllers for their ability to control a set of systems whose parameters deviate from the nominal system. If the controller remains stable for all deviations, the controller is said to have *robust stability* (Mooij, 1998). Because a real system will always deviate from the numerical model of the same system, it is an important property of a controller to be robust to such system deviations. Additionally, if only a relatively small improvement can be made to the control performance of conventional controllers at the cost of a much larger *implementation effort*, it needs to be evaluated whether this improvement is worth the increased design effort. Based on these observations and arguments, the main research question is formulated as follows:

Can knowledge-based control systems outperform conventional control systems when applied to entry-GNC systems?

The main research question can be split into three sub-questions which are formulated as follows:
When compared to conventional control systems,

1. **can knowledge-based control systems improve the performance of entry-GNC systems?**
2. **can knowledge-based control systems improve the robustness of entry-GNC systems?**
3. **can the increase in implementation effort for knowledge-based control systems be justified with a corresponding increase in control performance?**

From the above paragraphs, one can make a selection of KBCSs that can be compared to a benchmark controller of a reference entry vehicle, such that an answer to the research questions can be derived. Stanley and Miikulainen (2002) showed that their developed method is a powerful method for artificially evolving neural networks, called NEAT, and is capable of evolving solutions of minimal complexity, and that it can outperform other neuro-evolution methods. Because of these arguments, it is decided to choose NEAT as a method for constructing a NNC for further study. It is, however, noted that this neuro-evolution algorithms require a significant amount of simulation time. Furthermore, Duerksen (1997), Nho and Agarwal (2000), Nour et al. (2007), and Gaurav (2012) have shown that FLCs can be applied to aircraft and can outperform PID-controllers. For these reasons, it is decided to study the implementation of an FLC as a controller for a reference entry vehicle. To ensure that this FLC will have optimal performance, it will be tuned by a pattern-search algorithm to determine the optimal set of membership functions.

Additionally, it was shown by Strejc (1981) that MPC is a generalization of an LQR, capable of handling constraints on the states and inputs of the system in a systematic way. Furthermore, (Ghahramani and Towhidkhah, 2009) showed that the robustness of the controller can be improved by using information of the previous states. Taking into account that robustness is an important property of a controller, it is decided to investigate the implementation of a Model Predictive Controller (MPC) which only takes the current values of the state as inputs, and an MPC which takes both the current and previous values of the state as input.

The *research objective* of this thesis can be said to be answering the research questions by making a comparative numerical analysis between conventional and knowledge-based attitude controllers when applied to an entry vehicle. From the research objective and the chosen control methodologies, a list of tasks can be constructed. This list of tasks will then automatically coincide with the different chapters of the thesis. The tasks to be completed in this report are:

- The first task is to select a reference entry vehicle which will be used as the numerical system that needs to be controlled. Chapter 2 will present the chosen reference vehicle and discuss the implications of that choice.
- After a reference vehicle has been chosen and discussed, it is necessary to present the theoretical basis used to simulate the environment which the reference vehicle interacts with. Hence, the second task will be to give an overview of the theoretical basis used to construct the numerical environment to simulate the behaviour of the reference vehicle. This overview will be given in Chapter 3.
- The research question asks to compare knowledge-based control systems to conventional control systems and, hence, the third task will be to design a conventional control system for the reference vehicle. To this end, Chapter 4 will present the theoretical basis of a feedback control system and discuss the designed *benchmark* controllers.
- The fourth task will be to present the inner workings of the chosen knowledge-based control methodologies and to discuss and present how to implement the methodologies as an attitude controller of the reference vehicle. To this end, Chapter 5 will present an actuator assignment algorithm that transforms so-called *moment-fractions* to a set of actuator commands for the reference vehicle. Additionally, Chapters 6, 7, and 8 will present the inner workings and numerical implementations of the NNCs, FLCs, and MPCs, respectively.
- To answer the research questions, simulations must be carried out. Hence, the theoretical basis of the environment, the reference vehicle, and the controllers need to be implemented in a numerical simulator, which will be the fifth task of this thesis. Once the numerical simulator has been constructed, it needs to be verified for correct implementation. Chapter 9 will present the Software Architecture Diagrams (SADs) and the verification procedure of the simulator and its components.

- The sixth and final task will be to analyse and discuss the numerical results and to derive an answer to the research questions, which will be the topic of Chapter 10. Chapter 11 will conclude this thesis and give an overview of the shortcomings of the work presented in this report, and gives recommendations for future research and improvements based on these shortcomings.

Chapter 2

Reference Vehicle

For flight simulation studies, it is necessary to have a consistent database with the properties of the vehicle that is being simulated. To that end, a reference vehicle must be selected that has consistent database available in the literature that is easy to obtain and implement. Aerodynamic data is made available by Weiland (2014) for the Space Shuttle Orbiter , the X-33, the X-38, and many others. The data is, however, represented in the form of graphs. A consistent numerical aerodynamic base of the HORUS-2B is made available by Mooij (1995). Additionally, several previous studies have used this vehicle as well, . Hence, it is chosen that the HORUS-2B will serve as the reference vehicle for this thesis. Section 2-1 will present the physical characteristics of the vehicle including the control modes. The corresponding mission and system requirements will be presented in Section 2-2. This chapter will then be concluded in Section 2-3 with an overview of the numerical aerodynamic database of the HORUS-2B.

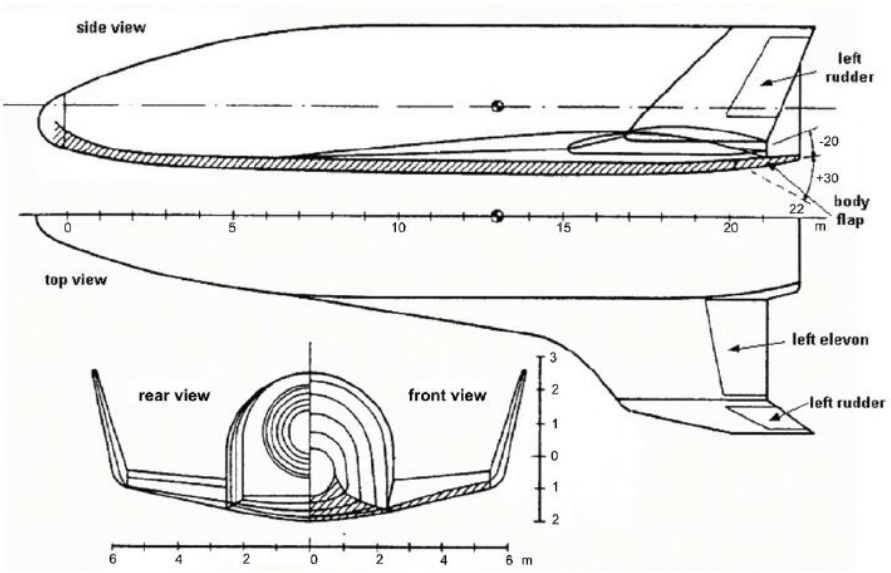


Figure 2-1: The HORUS-2B vehicle (MBB, 1998).

2-1 HORUS-2B

The configuration of the HORUS-2B vehicle is shown in Figure 2-1. It can be seen that it has a number of control surfaces, namely two rudders, two elevons and one body-flap. The rudders are outward moveable only, and such an outboard movement is considered to be a positive deflection. The elevons combine both the elevator and the aileron function. The deflections commanded by the attitude controller should be combined to give the corresponding left and right elevon deflections (Mooij, 1998). A downwards deflection is considered to be positive for the elevons and the body-flap (MBB, 1998).

The entry control modes are shown in Figure 2-2. The Reaction Control System (RCS) thrusters are operated during the early phase of the re-entry, when dynamic pressure is too low to allow for effective use of the aerodynamic actuators. The aileron, the elevator, and the body-flap start operating once the dynamic pressure reaches a value of 100 N/m^2. The rudder, which effectiveness is low, starts operating when the dynamic pressure if above 150 N/m^2. To support yaw control, the yaw thrusters continue to operate down to a Mach number of 1 (Mooij, 2014a).

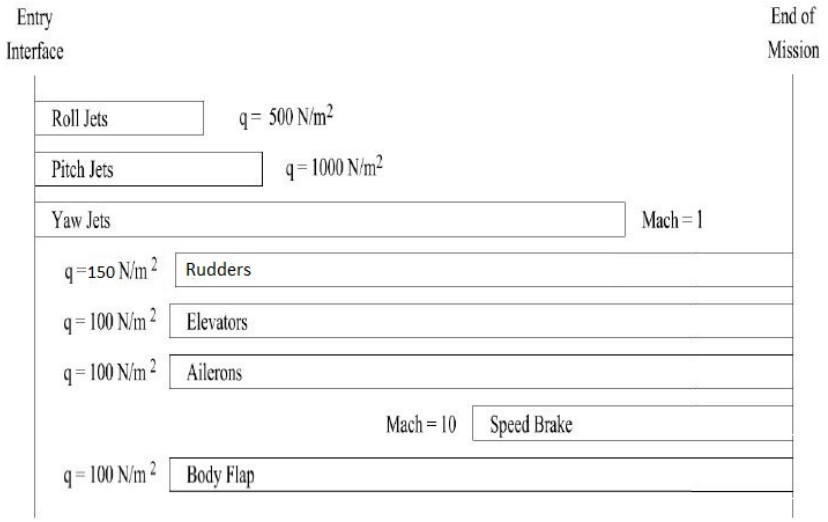


Figure 2-2: Control modes for HORUS-2B (Mooij, 2014a).

Table 2-1: Main Characteristics of the HORUS-2B (MBB, 1998).

Total vehicle Length :	25.0	[m]
Maximum fuselage width :	5.4	[m]
Maximum fuselage height :	4.5	[m]
Wing span :	12.0	[m]
Wing chord :	23.0	[m]
Wing area :	110.0	[m^2]
Maximum payload mass :	7,000	[kg]
Re-entry mass :	26,029	[kg]
I_{xx} :	199,605	[kg m^2]
I_{xy} :	0	[kg m^2]
I_{xz} :	-20,372	[kg m^2]
I_{yy} :	769,000	[kg m^2]
I_{yz} :	0	[kg m^2]
I_{zz} :	805,395	[kg m^2]

The elevator deflection can be defined as the symmetric combination of the left and right elevon and the aileron deflection can be defined as the asymmetric combination of the left and right elevon, *i.e.* (Mooij, 2014a):

$$\delta_e = \frac{\delta_{e,l} + \delta_{e,r}}{2},\ \delta_a = \frac{\delta_{e,l} - \delta_{e,r}}{2} \tag{2-1}$$

where $\delta_{e,l}$ and $\delta_{e,r}$ are the left and right elevon deflections, respectively, and δ_e and δ_a are the effective elevator and aileron deflections, respectively. The rudder deflection δ_r is defined to be equal to $\delta_{r,l}$ when $\delta_{r,l}$ is positive, and equal to -$\delta_{r,r}$ when $\delta_{r,r}$ is positive.

The complete database for HORUS is given in numerical form in a memorandum on the reference vehicle by Mooij (1995), including tables for the aerodynamic coefficients. In Table 2-1, the main characteristics of the vehicle are summarized.

2-2 Mission and System Requirements

When a reference vehicle is selected, one must take into account the corresponding mission and system requirements, which the vehicle is subject to. Mission requirements relate to entry and final conditions, reference trajectories, and nominal commanded steering variables. Therefore, Section 2-2-1 will present and discuss the mission requirements. System requirements relate to maximum heat-flux, maximum g-load, maximum aerodynamic surface deflections and rates, and maximum attitude and angular rates. To this end, Section 2-2-2 will introduce the system requirements of the chosen reference vehicle.

2-2-1 Mission Requirements

Because the selected reference vehicle automatically brings about mission requirements, these must be considered. The most obvious mission requirements are the state of the vehicle at the entry-interface. These entry conditions of the reference trajectory are shown in Table 2-2. Furthermore, a targeting point is selected as the final condition of the re-entry phase, and therefore a landing site is defined by the runway parameters and the distance to the targeting point, which are shown in Table 2-3.

Table 2-2: Entry conditions of reference trajectory (MBB, 1998).

Variable	Value	
Altitude	122	[km]
Latitude	-22.3	[deg]
Longitude	-106.58	[deg]
Velocity	7435.5	[m/s]
Heading	70.75	[deg]
Flight Path Angle	-1.43	[deg]

Table 2-3: Landing site definition and runway parameters (MBB, 1998).

Variable	Value	
Longitude	-53	[deg]
Latitude	5	[deg]
Altitude	20	[m]
Azimuth of landing direction w.r.t. North	180	[deg]
Targeting Point Distance to Runway	12	[km]

As mentioned, a reference trajectory is defined which, nominally, the vehicle tries to follow. The nominal altitude-velocity profile is plotted in the top figure of Figure 2-3, and the commanded angle of attack and bank angle are shown plotted against time in the bottom figure.

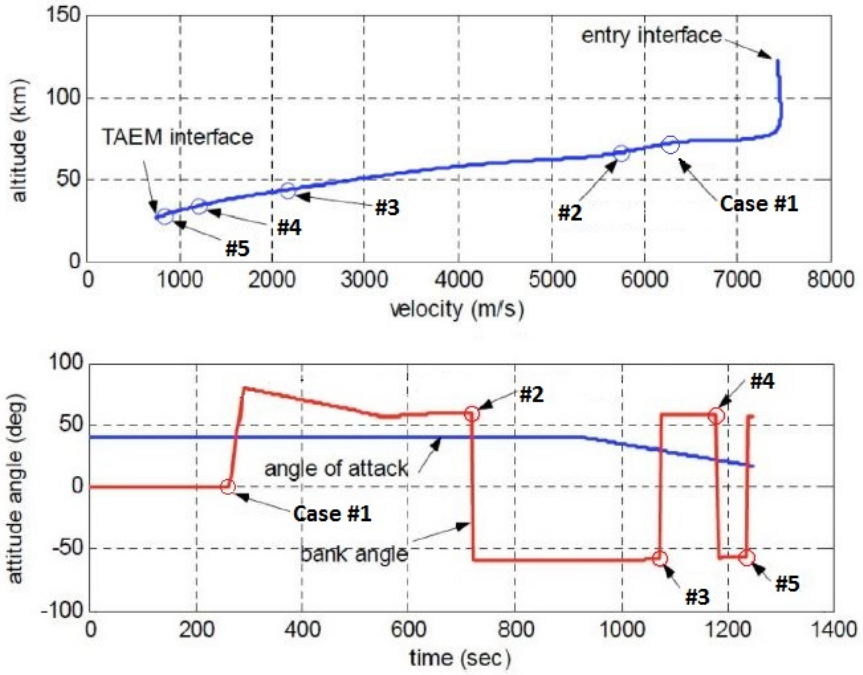


Figure 2-3: Overview of the points along the altitude-velocity profile (top) and nominal control history(bottom) which are tested for control performance. Modified from Mooij (2014a).

From the top figure of Figure 2-3 it can be seen that the re-entry of HORUS starts at an altitude of 122 km at a velocity of 7435.5 m/s, as given in Table 2-2. It is assumed to be heading towards a runway in Kourou (French Guyana). Some 80 km from the landing site the guidance system switches to the TAEM logic. The point where this occurs is known as the *TAEM interface* (Mooij, 2014a). However, the main focus of this thesis is on the hypersonic entry and descent of HORUS.

The nominal control history is shown in bottom figure of Figure 2-3. It can be seen that HORUS enters the atmosphere with a high angle of attack of 40 degrees, such that the maximum heat-flux can be kept sufficiently small. Further down the trajectory, the angle of attack is decreased to meet the cross- and downrange requirements. The nominal bank angle history consists of absolute values only, guaranteeing a certain descent rate. The corresponding sign is determined by flying towards the targeting point near Kourou, while keeping the heading error within dead-band limits. Whenever the heading error exceeds the dead-band limit, the sign of the bank angle is inverted. This manoeuvre is known as a *bank-reversal* (Mooij, 2014a). It is at the points along the reference trajectory where these bank-reversals occur that an attitude controller can best be tested, because at these points, the demand on the control system will be the highest. The cases of interest have been indicated in Figure 2-3. The values of the dynamic pressure at the points of interest along the reference trajectory are shown in Table 2-4.

2-2-2 System Requirements

Every vehicle is limited by its system requirements. These system requirements are derived from the vehicle structural limitations and from the trimming and manoeuvrability restrictions imposed by flight dynamics. Constraints of the chosen reference vehicle are found to be the maximally achievable aerodynamic attitude angles and rates, maximum heat-flux, and maximum g-load. In Table 2-5 the maximum system capabilities for the heat-flux and g-load are shown. The maximum heat-flux and g-load constraints determine the lower boundaries of the entry corridor

for HORUS. If the vehicle violates the maximum heat-flux constraint, its heat shield no longer provides sufficient thermal protection, and this will results in the destruction of the vehicle. If the vehicle violates the maximum g-load constraint, either, if the constraint is a structural constraint, its structure will become too weak to deal with the mechanical loads, or, if the constraint is a payload constraint (human or material payload), the payload will be compromised. Therefore, it is of utmost importance that the trajectory of HORUS does not violate these constraints.

Table 2-4: Dynamic Pressures at the points of interest along the reference trajectory.

Case	$\bar{q}$ [Pa]
1	637
2	2,332
3	5,488
4	7,269
5	8,387

Table 2-5: Maximum system capabilities (MBB, 1998; Mooij, 1995).

Variable	Value
Maximum heat-flux [kW/m^2]	530
Maximum G-load [-]	2.5
Maximum bank angle [deg]	87
Minimum rudder deflection [deg]	0
Maximum rudder deflection [deg]	40
Minimum elevon deflection [deg]	-40
Maximum elevon deflection [deg]	40
Minimum body-flap deflection [deg]	-20
Maximum body-flap deflection [deg]	30
Maximum roll thruster moment [Nm]	1,600
Maximum pitch thruster moment [Nm]	10,400
Maximum yaw thruster moment [Nm]	7,600

Furthermore, Table 2-5 shows the maximum system capabilities of the bank angle and the minimum and maximum actuator deflections. The maximum bank angle constrains the vehicle from decreasing its lift vector component normal to the surface of the Earth, making it more difficult to not skip out of the atmosphere. The minimum and maximum actuator deflections are system limitations that limit the change of the moments coefficients due to actuator deflections.

Table 2-6 shows the Mach-number-dependent confinement of the angle of attack. It shows that not only the minimum and maximum angle of attack is Mach number dependent, is also shows that its maximum variation is confined and Mach number dependent. Table 2-7 shows that the maximum roll rate is dependent on the dynamic pressure.

Table 2-6: Mach-number-dependent confinement of the angle of attack (MBB, 1998).

Mach Number	0	13	15	30
Upper limit angle of attack [deg]	43.9	43.9	43.9	43.9
Lower limit angle of attack [deg]	12	12	30	30
Maximum change in angle of attack [deg]	20	10	1	1

Table 2-7: Maximum roll rate as function of dynamic pressure (MBB, 1998).

Dynamic Pressure [N/m^2]	0	2,500	10,000
Roll rate [deg/s]	5	10	15

Further limitations on the reference vehicle are the maximum achievable aerodynamic coefficients, which are dependent on Mach number, angle of attack, aerodynamic surface deflections, and altitude. A full overview of these variations are given in the reference vehicle memorandum by Mooij (1995). It is noted that no extrapolation is required for the aerodynamic coefficients, as they are assumed constant after the given boundaries (Mooij, 1995).

2-3 Numerical Aerodynamic Database

To perform simulations within the atmosphere, one needs to make use of a numerical aerodynamic database. For the chosen reference vehicle, the HORUS-2B, such a database is given in a memorandum by Mooij (1995). This aerodynamic database is based on some simplifying assumptions. The coefficients are steady coefficients for the untrimmed super/hypersonic flight regime, neglecting the following aerodynamic effects (Mooij, 1995):

- no aero-elastic effects (the vehicle is a rigid body);
- the influence of the Reynolds numbers on the skin-friction drag is not presented explicitly but included in the drag coefficients as averaged skin-friction drag along a standard trajectory down to an altitude of 20 km. For trajectory calculations at lower altitudes, a drag altitude decrement is included in the data set. This decrement should be added to the total drag coefficients valid for 20 km;
- no special landing aerodynamics (undercarriage drag, ground effect on pitch, lift, and drag);
- no interference effects of the flaps;
- the influence of sideslip is simplified by linearisation over two degrees of sideslip angle β. It is noted that the β-derivatives or β-dependent coefficients can be extremely non-linear for $\beta > 2^\circ$.

The aerodynamic force-coefficients are defined as follows:

$$C_D = C_{D_0} + \Delta C_{D_{r,l}} + \Delta C_{D_{e,l}} + \Delta C_{D_b} + \Delta C_{D_{e,r}} + \Delta C_{D_{r,r}} - \Delta C_{D_h} \tag{2-2}$$

$$C_S = \Delta C_{S_{r,l}} + \Delta C_{S_{e,l}} + \Delta C_{S_{e,r}} + \Delta C_{S_{r,r}} + \left[\left(\frac{\partial C_S}{\partial \beta} \right)_0 + \Delta \left(\frac{\partial C_S}{\partial \beta} \right)_{e,l} + \Delta \left(\frac{\partial C_S}{\partial \beta} \right)_{e,r} \right] \beta \tag{2-3}$$

$$C_L = C_{L_0} + \Delta C_{L_{e,l}} + \Delta C_{L_{e,b}} + \Delta C_{L_{e,r}} \tag{2-4}$$

where the subscript $()_0$ refers to the coefficient when no flaps are used, and the subscripts $()_{e,l}, ()_{e,r}, ()_{r,l}, ()_{r,r}$, and $()_b$ refer to the left and right elevon, the left and right rudder, and the body-flap, respectively. The functional arguments of all components are listed in Table 2-8.

Table 2-8: Aerodynamic force-coefficients functional arguments list (Mooij, 1995).

Drag Coefficients	Sid-force Coefficients	Lift Coefficients
$C_{D_0} = f(\alpha, M)$	$\Delta C_{S_{r,l}} = f(\alpha, \delta_{r,l}, M)$	$C_{L_0} = f(\alpha, M)$
$\Delta C_{D_{r,l}} = f(\alpha, \delta_{r,l}, M)$	$\Delta C_{S_{e,l}} = f(\alpha, \delta_{e,l}, M)$	$\Delta C_{L_{e,l}} = f(\alpha, \delta_{e,l}, M)$
$\Delta C_{D_{e,l}} = f(\alpha, \delta_{e,l}, M)$	$\Delta C_{S_{e,r}} = f(\alpha, \delta_{e,r}, M)$	$\Delta C_{L_b} = f(\alpha, \delta_b, M)$
$\Delta C_{D_b} = f(\alpha, \delta_b, M)$	$\Delta C_{S_{r,r}} = f(\alpha, \delta_{r,r}, M)$	$\Delta C_{L_{e,r}} = f(\alpha, \delta_{e,r}, M)$
$\Delta C_{D_{e,r}} = f(\alpha, \delta_{e,r}, M)$	$\left(\frac{\partial C_S}{\partial \beta}\right)_0 = f(\alpha, M)$	
$\Delta C_{D_{r,r}} = f(\alpha, \delta_{r,r}, M)$	$\Delta \left(\frac{\partial C_S}{\partial \beta}\right)_{e,l} = f(\alpha, \delta_{e,l}, M)$	
$\Delta C_{D_h} = f(\alpha, h)$	$\Delta \left(\frac{\partial C_S}{\partial \beta}\right)_{e,r} = f(\alpha, \delta_{e,r}, M)$	

The aerodynamic moment coefficients are defined as follows:

$$C_l = \Delta C_{l_{e,l}} + \Delta C_{l_{e,r}} + \left(\frac{\partial C_l}{\partial \beta}\right)_0 \beta \tag{2-5}$$

$$C_m = C_{m_0} + \Delta C_{l_{e,l}} + \Delta C_{m_b} + \Delta C_{m_{e,r}} \tag{2-6}$$

$$C_n = \Delta C_{n_{r,l}} + \Delta C_{n_{e,l}} + \Delta C_{n_{e,r}} + \Delta C_{n_{r,r}} + \left[\left(\frac{\partial C_n}{\partial \beta}\right)_0 + \Delta\left(\frac{\partial C_n}{\partial \beta}\right)_{r,l} + \Delta\left(\frac{\partial C_n}{\partial \beta}\right)_{r,r}\right]\beta \tag{2-7}$$

where the functional arguments of all components are listed in Table 2-9.

Table 2-9: Aerodynamic moment coefficients functional arguments list (Mooij, 1995).

Roll Coefficients	Pitch Coefficients	Yaw Coefficients
$\Delta C_{l_{e,l}} = f(\alpha, \delta_{e,l}, M)$	$C_{m_0} = f(\alpha, M)$	$\Delta C_{n_{r,l}} = f(\alpha, \delta_{r,l}, M)$
$\Delta C_{l_{e,r}} = f(\alpha, \delta_{e,r}, M)$	$\Delta C_{m_{e,l}} = f(\alpha, \delta_{e,l}, M)$	$\Delta C_{n_{e,l}} = f(\alpha, \delta_{e,l}, M)$
$\left(\frac{\partial C_l}{\partial \beta}\right)_0 = f(\alpha, M)$	$\Delta C_{m_{e,r}} = f(\alpha, \delta_{e,r}, M)$	$\Delta C_{n_{e,r}} = f(\alpha, \delta_{e,r}, M)$
	$\Delta C_{m_b} = f(\alpha, \delta_b, M)$	$\Delta C_{n_{r,r}} = f(\alpha, \delta_{r,r}, M)$
		$\left(\frac{\partial C_n}{\partial \beta}\right)_0 = f(\alpha, M)$
		$\Delta\left(\frac{\partial C_n}{\partial \beta}\right)_{r,l} = f(\alpha, \delta_{r,l}, M)$
		$\Delta\left(\frac{\partial C_n}{\partial \beta}\right)_{r,r} = f(\alpha, \delta_{r,r}, M)$

To reduce the amount of aerodynamic data for the control surfaces, use is made of the fact that the two rudders and the two elevons are identical, and their location is symmetric w.r.t the vehicle's XZ-plane (Mooij, 1995). The resulting symmetry conditions are tabulated in Table 2-10. It is noted that the symmetry conditions only hold for equal values of the flap deflections.

Table 2-10: Symmetry conditions for right rudder and elevon (Mooij, 1995).

Zero derivatives right elevon	first derivatives right elevon	zero derivatives right rudder	first derivatives right rudder
$C_{D_{e,r}} = C_{D_{e,l}}$	$\left(\frac{\partial C_S}{\partial \beta}\right)_{e,r} = \left(\frac{\partial C_S}{\partial \beta}\right)_{e,l}$	$C_{D_{r,r}} = C_{D_{r,l}}$	$\left(\frac{\partial C_n}{\partial \beta}\right)_{r,r} = \left(\frac{\partial C_n}{\partial \beta}\right)_{r,l}$
$C_{S_{e,r}} = -C_{S_{e,l}}$	$\left(\frac{\partial C_n}{\partial \beta}\right)_{e,r} = \left(\frac{\partial C_n}{\partial \beta}\right)_{e,l}$	$C_{S_{r,r}} = -C_{S_{r,l}}$	
$C_{L_{e,r}} = C_{L_{e,l}}$		$C_{n_{r,r}} = -C_{n_{r,l}}$	
$C_{l_{e,r}} = -C_{l_{e,l}}$			
$C_{m_{e,r}} = C_{m_{e,l}}$			
$C_{n_{e,r}} = -C_{n_{e,l}}$			

It is noted that, due to the summed nature of the moment coefficients an additional property occurs. For example, at a given angle of attack, Mach number, and body-flap deflection, the gradient of the pitch moment coefficient in the direction of the left elevon deflection is independent of the value of the right elevon deflection. This is because, if one takes the derivative of the pitch moment coefficients with respect to the left elevon deflection, the contribution due to the right elevon deflection is a constant and hence, disappears. This means that the pitch moment

contribution of the combined left and right elevon deflection C_{m_δ}, for a given angle of attack and Mach number, is a piece-wise linear surface in three-dimensional space, where each linear plane-segment of the surface is flat[1]. Similar arguments exists for the elevon and rudder contributions to the roll and yaw moment coefficients, respectively. This flatness property will be exploited in Chapter 5.

Finally, the aerodynamic forces and moments are then computed with:

$$D = C_D \bar{q} S_{ref} \tag{2-8}$$
$$S = C_S \bar{q} S_{ref} \tag{2-9}$$
$$L = C_L \bar{q} S_{ref} \tag{2-10}$$
$$\mathcal{L} = C_l \bar{q} S_{ref} b_{ref} \tag{2-11}$$
$$M = C_m \bar{q} S_{ref} c_{ref} \tag{2-12}$$
$$N = C_n \bar{q} S_{ref} b_{ref} \tag{2-13}$$
$$\tag{2-14}$$

where the numerical values for the reference geometry are (MBB, 1998):

$$S_{ref} = 110 \text{ m}^2 \tag{2-15}$$
$$b_{ref} = 12 \text{ m} \tag{2-16}$$
$$c_{ref} = 23 \text{ m} \tag{2-17}$$

[1]Flat planes in this context means that all four boundaries are parallel to another boundary and perpendicular to the remaining two boundaries.

Chapter 3

Flight Dynamics

To test the performance of a control system, one must have a model of the behaviour of the vehicle and of the environment the vehicle is interacting with. To this end, this chapter will present the different aspects of flight dynamics. Section 3-1 the most commonly used reference frames will be presented. In Section 3-2 the different representations of the state variables will be discussed. Section 3-4 will then present the different environmental factors that play a roll in the accurate description of flight dynamics. In Section 3-5, the external forces and moments a re-entry vehicle will encounter are presented, as they are needed to describe the motion of a vehicle through the previously discussed environment. Section 3-6 will then present the general set of the equations of motion used to describe translational and rotational dynamics. Section 3-7 will then conclude with the equations of motion linearised and presented in a state-space representation, which will be used in later chapters.

3-1 Reference Frames

In the context of flight dynamics, a reference frame is a coordinate system or a set of axes which are used to measure the state of the vehicle. In flight dynamics, several reference frames are used to define the different forces acting on a body, and this section will give an overview of the most commonly used references frames in re-entry flight dynamics studies. The references frames can be divided into two categories: planet-fixed reference frames with the origin at the centre of mass of the central body, and vehicle-fixed reference frames with the origin located at the centre of mass of the vehicle. The two planet-fixed reference frames will be presented and discussed first, after which the vehicle-fixed reference frames will be defined.

3-1-1 Inertial Planetocentric Reference Frame, F_I

The first planet-fixed reference frame considered is the inertial planetocentric reference frame, denoted as the I-frame and with the subscript I. It is a right-handed orthogonal axis-system with its origin located at the centre of mass of the central body. The $OX_I\,Y_I$-plane coincides with the equatorial plane of the central body, the Z_I-axis is aligned with the rotation axis of the central body and the X_I-axis passes through the equator at the location whether the ecliptic and the equator cross; the vernal equinox. The Y_I-axis then completes the right-handed coordinate system.

It is noted that the spin-axis of the Earth does not remain in the same direction through time. Due to gravitational influences of the Moon and other celestial bodies, the spin-axis 'wobbles' about a fictive mean spin-axis. This motion is known as polar motion. The definition of the inertial reference frame still holds although the choice of spin-axis orientation can differ per country. For purpose of clarity, an international celestial reference system has been defined. The coordinate system for the inertial planetocentric reference frame is defined by *the mean equator and equinox of J2000.0* (The standard epoch J2000.0 is 12hr on January 1, 2000). The mean equator and equinox are the fictitious equator and equinox, derived after removing the effects of nutation.

3-1-2 Rotating Planetocentric Reference Frame, F_R

The second planet-fixed reference frame is the rotating planetocentric reference frame, denoted as the R-frame and with the subscript R. Like the I-frame, it is a right-handed orthogonal axis-system with its origin located at the centre of mass of the central body. Similarly, the $OX_R Y_R$-plane is in the equatorial plane of the central body, the Z_R-axis is aligned with the rotation axis of the central body. The X_R-axis crosses the equator at the zero longitude line, and Y_R completes the right-handed coordinate system.

3-1-3 Vertical Reference Frame, F_V

The next vehicle-fixed reference frame is the vertical reference frame, denoted as the V-frame and with the subscript V. The Z_V-axis points towards the centre of mass of the central body, along the direction of the gravitational attraction and the X_V-axis lies in a meridian plane, perpendicular to Z_V, and points towards north. The Y_V then completes the right-handed coordinate system. The $X_V Y_V$-plane is commonly referred to as the local horizontal plane, which follows the variation of the planet's curvature, and, therefore, does not follow the vehicle's rotational motion. Figure 3-1 shows the relations between the I-, R- and V-frames

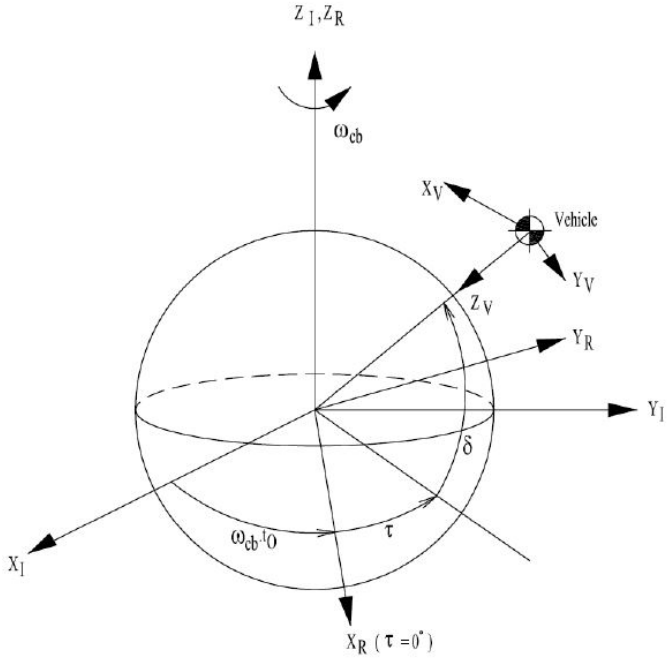


Figure 3-1: Graphical representation of the relation between the I-, $R-$ and V-frames (Mooij, 2014a).

3-1-4 Body Fixed Reference Frame, F_B

The first vehicle-fixed reference frame is the body fixed reference frame, denoted as the B-frame and with the subscript B. It is assumed that there is a (geometrical) symmetry in the longitudinal direction and this plane of symmetry is then chosen to be the X_BZ_B-plane. The X_B-axis then points forward whereas the Z_B-axis points downward. The Y_B-axis then completes the right-handed coordinate system. This reference frame is commonly used to describe rotation around the axis of the vehicle, namely roll, pitch and yaw.

3-1-5 Trajectory Reference Frame, F_T

The trajectory reference frame is the third vehicle-fixed reference frame to be considered and is denoted as the T-frame and with the subscript T. The X_T-axis points in the direction of the velocity vector relative to the R-frame, the Z_T-axis is in the vertical plane pointing downwards, perpendicular to X_T, and the Y_T completes the right-handed coordinate system. Figure 3-2 shows the relation between the V-frame and the T-frame and the TA-frame..

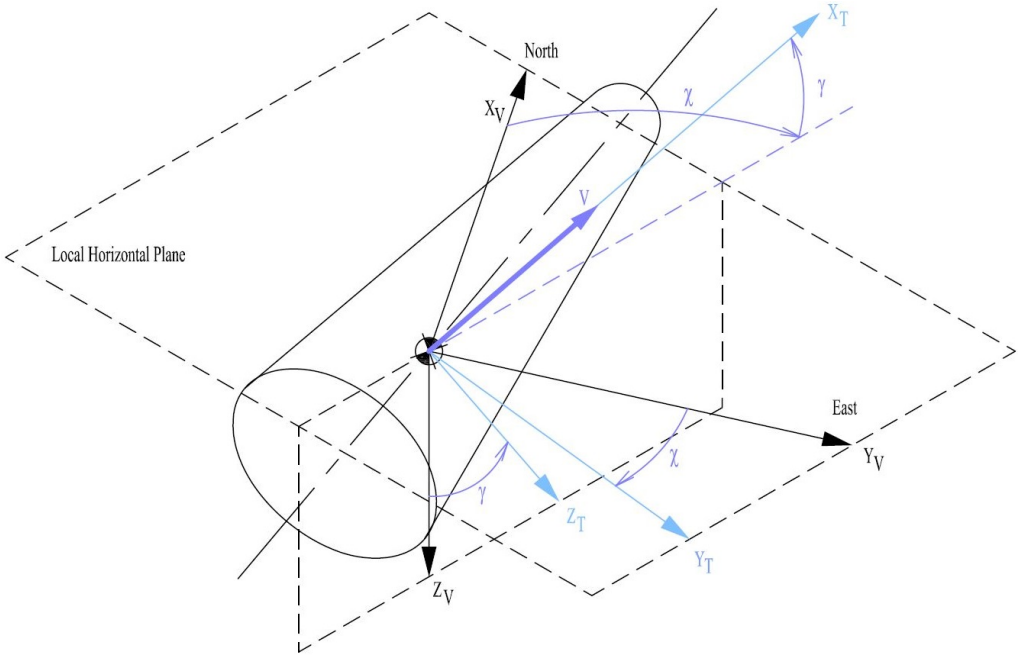


Figure 3-2: Graphical representation of the relation between the V-frame and the T-frame[1].

3-1-6 Aerodynamic Reference Frame, F_A

The next vehicle-fixed reference frame considered is the aerodynamic reference frame, denoted as the A-frame and with the subscript A. The X_A-axis points in the direction of the velocity vector relative to the R-frame. This means that the X_A-axis and the X_T-axis coincide. The Z_A-axis points in the negative direction of the aerodynamic lift force and the Y_A then completes the right-handed coordinate system. It is noted that the A-frame and the T-frame are different only when there is a rotation around the X-axis, i.e. when the vehicle is banking, otherwise the A-frame and the T-frame are identical. Figure 3-3 shows the relation between the A-frame, the B-frame, and the T-frame.

[1] E. Mooij, Private Communication, Oct 2016

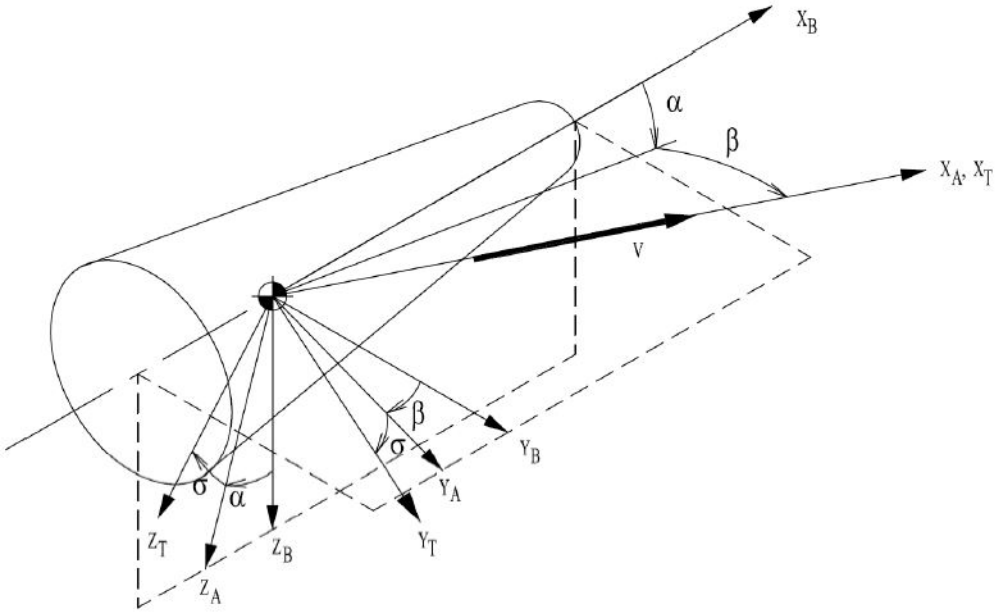


Figure 3-3: Graphical representation of the relation between the A-frame, the B-frame, and the T-frame (Mooij, 2014a).

3-2 State Variables

To correctly describe the complete six Degrees of Freedom (DoF), one needs a description of the state of the vehicle, that is, a complete description of position, velocity, attitude and angular rate. In this section an overview will be given of a selection of state variable descriptions. However, in this thesis the performance of a control system will be evaluated at a single point along the reference trajectory, with fixed position and velocity. Hence, in the remainder of this section, only an overview of the state descriptions for the attitude and angular rates will be given.

Attitude

The attitude of the vehicle can be expressed in different ways. For example, one can use Euler angles, quaternions or modified Rodriquez parameters. The main reason for selecting quaternions or modified Rodriquez parameters is the fact that, when one uses Euler angles to describe the equation of motion, a singularity can occur at $-90°$ and $90°$. This phenomenon is known as "Gimbal Locking" or “Euler-angle singularity”. For quaternions, no singularity occurs thanks to its four elements, whereas the modified Rodriquez parameters have a singularity at $360°$ (Crassidis and Markley, 1996). Furthermore, Euler angles suffer from ambiguity, as its not clear which angle corresponds to a rotation about which axis. However, for quaternions and modified Rodriquez parameters, contrary to the Euler angles, it is hard to visualize the rotation associated with their numerical values. Additionally, Euler angles are good for decomposing rotations into individual degrees of freedom.

Mooij (2014b) advised to use quaternions within a numerical simulation in the I-frame, hence it is chosen to use quaternions within the numerical framework. Furthermore, the set of Euler angles known as the aerodynamic angles will be used to allow for the linearisation of the Equations of Motion (EoM) with these angles. The aerodynamic angles are the angle of attack α ($-180° \leq \alpha <$

180°, positive when "nose-up"), the sideslip angle β ($-90° \leq \beta \leq 90°$, positive when "nose-left") and the bank angle σ ($-180° \leq \sigma < 180°$, positive when banking to the right). The commonly used rotation sequence for the aerodynamic angles is the sequence 2-3-1: a rotation about the Y-axis, followed by a rotation about the (local) Z-axis, and concluded by a rotating around the (local) X-axis. It is noted that when using the aerodynamic angles, a singularity occurs at a sideslip angle of ±90° (Mooij, 2014a).

The attitude description used within the numerical framework are quaternions. A quaternion is a 4-dimensional hyper-complex number consisting of a real part and an imaginary vector part (Kuipers, 2002). The imaginary numbers are different square roots of -1 and obey the following constraint:

$$i^2 = j^2 = k^2 = ijk = -1 \tag{3-1}$$

It can be found that four parameters of the quaternion consist of the rotational angle Φ and the normalized rotational axis $\mathbf{a}$. Or in equation form:

$$Q = \begin{pmatrix} q_1, q_2, q_3, q_4 \end{pmatrix}^T = \begin{pmatrix} \mathbf{q} \\ q_4 \end{pmatrix} = \begin{pmatrix} \mathbf{a} \sin \frac{\Phi}{2} \\ \cos \frac{\Phi}{2} \end{pmatrix} \tag{3-2}$$

The four quaternion elements are not mutually independent as they satisfy the following constraint:

$$Q^T Q = q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1 \tag{3-3}$$

The conjugate of $Q^\star$ of quaternion Q is defined as (Kuipers, 2002);

$$Q^\star = (-\mathbf{q}, q_4) \tag{3-4}$$

Furthermore, the product between two quaternions Q and P is;

$$Q \circ P = (q_4\mathbf{p} + p_4\mathbf{q} + \mathbf{q} \times \mathbf{p}, p_4 q_4 - \mathbf{p} \cdot \mathbf{q}) \tag{3-5}$$

where, in this context, $\circ$ is used to describe the quaternion product. It is noted that the quaternions, thanks to their four parameters, have no singularity for any rotation.

Angular Rate

The angular rate of the vehicle is defined as the rotational velocity of the body frame w.r.t the inertial frame, expressed along the components of the body axis. The rotation vector, $\boldsymbol{\omega}$, is defined as:

$$\boldsymbol{\omega} = \begin{pmatrix} p, q, r \end{pmatrix}^T \tag{3-6}$$

where p is the roll rate, q is the pitch rate, and r is the yaw rate. A graphical representation of the definition of the angular rate can be found in Figure 3-4.

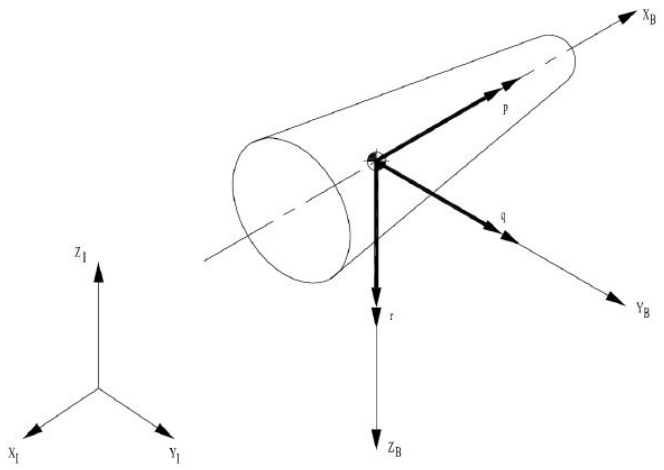


Figure 3-4: Definition of the angular rate of the vehicle, $\boldsymbol{\omega} = (p, q, r)^T$. The I-frame is the inertial planetocentric frame, which has its origin in the centre of mass of the central body, whereas the origin of the B-frame is located in the centre of mass of the vehicle (Mooij, 2014a).

The time derivative of the quaternion can be expressed as a differential equation for the quaternion and is given as (Kuipers, 2002);

$$\dot{Q} = \frac{1}{2} Q \circ \boldsymbol{\omega} \tag{3-7}$$

This can be rewritten in matrix notation as (Crassidis and Markley, 1996);

$$\dot{Q} = \frac{1}{2} \boldsymbol{\Xi}(Q) \boldsymbol{\omega} \tag{3-8}$$

with

$$\boldsymbol{\Xi}(\mathbf{Q}) = \begin{bmatrix} q_4 \mathbf{I}_{3x3} + [\mathbf{q}\times] \\ -\mathbf{q}^T \end{bmatrix}, \quad [\mathbf{q}\times] = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix} \tag{3-9}$$

where $\mathbf{I}_{3x3}$ is a $3x3$ identity matrix and $[\mathbf{q}\times]$ is referred to as the cross product matrix since $\mathbf{a} \times \mathbf{b} = [\mathbf{a}\times]\,\mathbf{b}$.

3-3 Frame Transformations

The change in orientation between reference frames can be described using Euler angles. These angles can then be used to create a transformation matrix $\mathbf{C_{BA}}$ With this transformation matrix, any vector in reference frame A can be transformed to another reference frame B. In equation form, a vector $\mathbf{x_A}$ expressed in reference frame A, can be transformed to $\mathbf{x_B}$ expressed in reference frame B, using:

$$\mathbf{x_B} = \mathbf{C_{BA}} \mathbf{x_A} \tag{3-10}$$

where $\mathbf{C_{BA}}$ is the transformation matrix containing information regarding the orientation of reference frame B with respect to reference frame A. When using Euler angles, one can chain transformation matrices together to obtain a matrix of the complete rotation sequence. This can easily be shown as follows:

$$\mathbf{x_C} = \mathbf{C_{CB}}\mathbf{x_B} = \mathbf{C_{CB}}\mathbf{C_{BA}}\mathbf{x_A} = \mathbf{C_{CA}}\mathbf{x_A} \tag{3-11}$$

where $\mathbf{C_{CA}}$ is the transformation matrix describing the transformation from reference frame A to reference frame C, and is constructed from $\mathbf{C_{CB}}$ and $\mathbf{C_{BA}}$. This shows that complete transformation matrices can be constructed from simpler transformation sub-matrices. Furthermore, an useful property of transformation matrices is that the inverse transformation matrix is equal to the transpose of the transformation matrix. This is due to the fact that the transformation matrices are orthonormal matrices. The product of two orthonormal matrices is also an orthonormal matrix (Mooij, 2014a).

One can define so-called unit-axis rotations to be a rotation about a single axis with an arbitrary angle α (positive rotations according to the right-hand rule). The resulting unit rotation matrices are found to be (Mooij, 2014a):

$$\mathbf{C_x}(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix}$$

$$\mathbf{C_y}(\alpha) = \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{bmatrix}, \quad \mathbf{C_z}(\alpha) = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-12}$$

Any rotation from frame A to frame B can always be constructed from a number of sequential unit-axis rotations. This implies that any transformation matrix can be written as a combination of the matrices $\mathbf{C_x}$, $\mathbf{C_y}$, and $\mathbf{C_z}$. The resulting transformations between the reference frames described in Section 3-1 are summarized in Table 3-1. More a more detailed description of these transformations the interested reader is referred to (Mooij, 2014a).

Table 3-1: Basic frame transformations (Mooij, 1998).

Frame A to B	Notation	Sequence of unit-axis rotations
Rotating to inertial planetocentric frame	$\mathbf{C_{I,R}}$	$\mathbf{C_z}(-\omega_{cb}t_0)$
Vertical to rotating planetocentric frame	$\mathbf{C_{R,V}}$	$\mathbf{C_z}(-\tau)\,\mathbf{C_y}(\pi/2+\delta)$
Trajectory to vertical frame	$\mathbf{C_{V,T}}$	$\mathbf{C_z}(-\chi)\,\mathbf{C_y}(-\gamma)$
Aerodynamic to trajectory frame	$\mathbf{C_{T,A}}$	$\mathbf{C_x}(\sigma)$
Aerodynamic to vertical frame	$\mathbf{C_{V,A}} = \mathbf{C_{V,T}}\mathbf{C_{T,A}}$	$\mathbf{C_z}(-\chi)\,\mathbf{C_y}(-\gamma)\,\mathbf{C_x}(\sigma)$
Body to aerodynamic frame	$\mathbf{C_{A,B}}$	$\mathbf{C_z}(\beta)\,\mathbf{C_y}(-\alpha)$
Vertical to inertial planetocentric frame	$\mathbf{C_{I,V}} = \mathbf{C_{I,R}}\mathbf{C_{R,V}}$	$\mathbf{C_z}(-\omega_{cb}t_0)\,\mathbf{C_z}(-\tau)\,\mathbf{C_y}(\pi/2+\delta)$
Aerodynamic to rotating planetocentric frame	$\mathbf{C_{R,A}} = \mathbf{C_{R,V}}\mathbf{C_{V,A}}$	$\mathbf{C_z}(-\tau)\,\mathbf{C_y}(\pi/2+\delta)\,\mathbf{C_z}(-\chi)$ $\mathbf{C_y}(-\gamma)\,\mathbf{C_x}(\sigma)$
Body to rotating planetocentric frame	$\mathbf{C_{R,B}} = \mathbf{C_{R,V}}\mathbf{C_{V,A}}\mathbf{C_{A,B}}$	$\mathbf{C_z}(-\tau)\,\mathbf{C_y}(\pi/2+\delta)\,\mathbf{C_z}(-\chi)$ $\mathbf{C_y}(-\gamma)\,\mathbf{C_x}(\sigma)\,\mathbf{C_z}(\beta)\,\mathbf{C_y}(-\alpha)$

3-4 Flight Environment

The motion of an object is determined by the forces originating from the environment the object is interacting with. Therefore, if one wants to accurately simulate the motion of an object through its flight environment, one needs to have a detailed description of the environment. To this end, this section will present the relevant flight environment for re-entry vehicles. Section 3-4-1 will present the methods used to model the planetary shapes of celestial bodies. Here only regular bodies will be discussed and Section 3-4-2 will give an overview of the different methods used to model the atmospheres of the celestial bodies with a significant atmosphere. It is noted that, since the rotational motion about a single point of a reference trajectory is studied in this thesis, gravity is not a relevant force, since it only acts on the Centre of Mass (CoM) and hence, produces no moments.

3-4-1 Planetary shape

For all celestial bodies, the planetary shapes differ from a perfect sphere. This is primarily due to the rotational rate that makes the shape deviate from that of a perfect sphere. Even though many factors contribute to the shape of the planet, for the purpose of flight simulation an adequate approximation is to consider the body as an ellipsoid of revolution. The ellipsoid is defined by the ellipticity e, which can be computed as (Mooij, 1998):

$$e = 1 - \frac{R_p}{R_e} \tag{3-13}$$

where R_p is the planetary radius at the poles and R_e is the planetary radius at the equator. The ellipticity can be used to derive an expression for the radius at an arbitrary point along the surface. The following expression can be derived:

$$R_s = R_e \left[1 - \frac{e}{2}\left(1 - \cos 2\delta^\star\right) + \frac{5}{16}e^2\left(1 - \cos 4\delta^\star\right) - \dots \right] \tag{3-14}$$

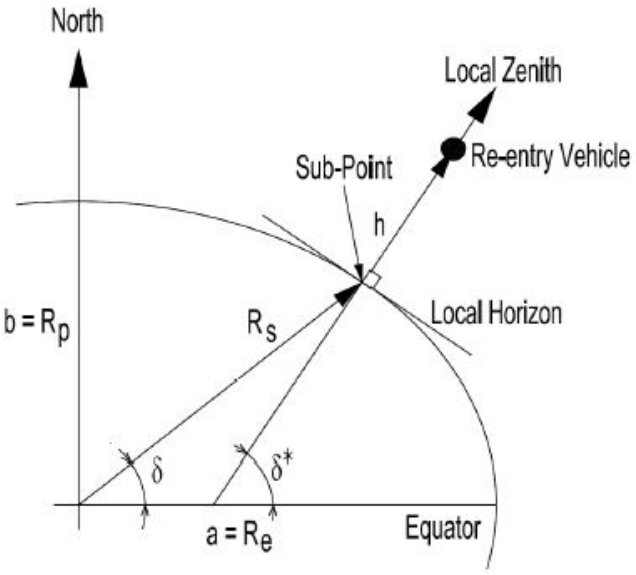


Figure 3-5: Graphical representation of the difference between the geodetic and geocentric latitude (Mooij, 2014a).

Here $\delta^\star$ is the geodetic latitude, which is not identical to the geocentric latitude δ, as shown in Figure 3-5. Since the difference between the geodetic and geocentric latitude is very small, it is justified to approximate the former with the latter. Furthermore, for first-order analysis it is sufficient to approximate Equation 3-14 with:

$$R_s = R_e \left(1 - e \sin^2 \delta\right) \tag{3-15}$$

This expression can be used to compute the altitude, h, of an object above the surface of the central body with:

$$h = R - R_e \left(1 - e \sin^2 \delta\right) \tag{3-16}$$

3-4-2 Atmosphere

The most interesting part of the flight envelope of a re-entry vehicle is the part where the vehicle enters the atmosphere and through its interacting with it, slows down while following a specified trajectory. As will be discussed in Section 3-5, the aerodynamic forces and moments are depended on atmospheric properties at the location of the vehicle, in particular the atmospheric density. For this reason, an atmospheric model needs to be included in the flight simulation.

For analytical purposes, it is common to assume a so-called, *exponential atmospheric model*. This is a simplified model that assumes that the atmosphere is an ideal gas at constant temperature in a hydrostatic equilibrium. The constant temperature assumptions will lead to errors for computing the Mach number, which is why more accurate atmosphere models may be needed.

There exist several more complete atmospheric models for different celestial bodies. A distinction must be made here between a *standard* and a *reference* atmosphere. A standard atmosphere is a theoretical vertical distribution of atmospheric properties, including temperature, pressure and density, which is a rough representative of year average conditions. As they are average values, there is no temporal variations in the atmospheric properties and the only spatial dependence is altitude. This implies that numerical simulations will always give the same results, independent of when or where (except altitude) the simulation starts. The most commonly used standard atmosphere for the Earth is the United States standard atmosphere of 1976 (Mooij, 2014a; NASA et al., 1976). A reference atmosphere does take into account latitudinal, seasonal and solar effects, amongst others, which do have a spatial and temporal dependence. The most common examples are the Jacchia model (Jacchia, 1977) for the upper layers of the atmosphere (90 to 2500 km) and the Earth-GRAM model (Leslie and Justus, 2011), both for the Earth.

The US76 model was used by Mooij (1998) to study the hypersonic re-entry of HORUS. Hence, for this study, the same US76 model will be used to be consistent. This model is an idealized, steady-state representation of the Earth's atmosphere. Winds in the atmosphere will not be taken into account, which means the airspeed is equal to the groundspeed. This has been done to study the nominal behaviour of the entry vehicle.

3-5 External Forces and Moments

To correctly predict the motion of the vehicle, one needs to model the external forces and moments that act on the vehicle. However, as mentioned before, the focus in thesis is on the rotational motion about a single point along the reference trajectory and hence, only the external moments that act on the vehicle are relevant. Therefore only the aerodynamic moments, $\mathbf{M_A}$, have to be considered.

The aerodynamic moment vector due to the aerodynamic moments, expressed in the B-frame, is defined as (Mooij, 1998):

$$\mathbf{M}_{\mathbf{M,A|B}} = \begin{pmatrix} \mathcal{L} \\ M \\ N \end{pmatrix} = \begin{pmatrix} C_l \bar{q} S_{ref} b_{ref} \\ C_m \bar{q} S_{ref} c_{ref} \\ C_n \bar{q} S_{ref} b_{ref} \end{pmatrix} \tag{3-17}$$

with

$\mathcal{L}$: roll moment [Nm]
M : pitch moment [Nm]
N : yaw moment [Nm]
C_l : roll moment coefficient [−]
C_m : pitch moment coefficient [−]
C_n : yaw moment coefficient [−]
b_{ref} : reference length for roll/yaw [m]
c_{ref} : reference length for pitch [m]
$\bar{q}$: dynamic pressure [N/m^2]
S_{ref} : reference area [m^2]

For winged vehicles, b_{ref} and c_{ref} are usually taken to be the wingspan and the mean aerodynamic cord, respectively. For axisymmetric re-entry capsules the reference lengths are most commonly taken to be maximum diameter of the capsule (Mooij, 1998).

Usually, the aerodynamic forces act on a point called the *aerodynamic reference point*. In general, this point does not coincide with the centre of mass of the vehicle, and thus will create a moment around it, in addition to the aerodynamic moments defined above. The aerodynamic moment vector due to the aerodynamic forces, expressed in the B-frame, can be computed as follows:

$$\mathbf{M}_{\mathbf{F,A|B}} = \mathbf{r}_{\mathbf{cm}} \times \mathbf{F}_{\mathbf{A|B}} \tag{3-18}$$

where $\mathbf{r}_{cm}$ is the location the centre of mass w.r.t. the aerodynamic reference point and $\mathbf{F}_{\mathbf{A|B}}$ are the aerodynamic forces expressed in the body frame. The find the total aerodynamic moment on the vehicle, one can sum the aerodynamic moments and the moments induced by the aerodynamic forces. The aerodynamic force, written in the A-frame, is given as:

$$\mathbf{F}_{\mathbf{A|A}} = \begin{pmatrix} -D \\ -S \\ -L \end{pmatrix} = \begin{pmatrix} -C_D \bar{q} S_{ref} \\ -C_S \bar{q} S_{ref} \\ -C_L \bar{q} S_{ref} \end{pmatrix} \tag{3-19}$$

with

D : drag force [N]
S_{ref} : side force [N]
L : lift force [N]
C_D : drag force coefficient [−]
C_S : side force coefficient [−]
C_L : lift force coefficient [−]

3-6 General Equations of Motion

In the previous sections, the tools required to write down the equations of motion have been stated. This section will present the general form of the equations of motion, which can be used to

develop a simulator of the reference vehicle. In general, the equations of motion can be separated into *translational* and *rotational* parts. However, only the rotational equations of motion are of relevance in this thesis and, hence, only those will be discussed in this section.

For a rigid body with constant mass, the equations of rotational motion expressed in the B-frame can be written as the so-called *Euler equations*, which gives information about the angular accelerations. The Euler equations are given as:

$$\frac{\mathrm{d}\boldsymbol{\omega}}{\mathrm{d}t} = \mathbf{I}^{-1}\left(\tilde{\mathbf{M}}_{cm} - \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}\right) \tag{3-20}$$

with the kinematic equation as:

$$\dot{Q} = \frac{1}{2}\Xi(Q)\boldsymbol{\omega} \tag{3-21}$$

and with

$\tilde{\mathbf{M}}_{cm} = (M_x, M_y, M_z)^T$ = sum of external, Coriolis and relative moments about the centre of mass, expressed in components along the body axis.

$\mathbf{I} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix}$ = inertia tensor of the vehicle, referenced to the body frame.

$\boldsymbol{\omega} = (p, q, r)^T$ = the rotation vector of the body frame with respect to the inertial frame, expressed in components along the body axis.

and where Q is the quaternion used to describe the attitude of the vehicle and $\boldsymbol{\Xi}$ is defined to be the matrix relating the rotation vector of the body frame with respect to the inertial frame, to the time derivative of the attitude quaternion (see Equation 3-9 for the definition).

3-7 State-Space Formulation

For first-order predictions of future states of the vehicle, a linear form of the equations of motion can be used. Such predictions can then be used to compute an optimal control strategy. For such applications, it is useful to have a linear state-space representation of the dynamics of the vehicle. However, to create a linear state-space representation, the non-linear equations of motion need to linearised. To facilitate the linearisation process, some additional assumptions are made to simplify the equations (Mooij, 1998, 2014a).

- A non-rotating Earth will be considered only ($\omega_{cb} = 0$). This is justified as the rotational rate of the vehicle is much larger than the rotational rate of Earth. This has as an effect that the centrifugal and Coriolis forces are zero.
- The vehicle is assumed to be rotationally symmetric w.r.t. mass, causing the products of inertia, I_{xy}, I_{xz}, and I_{yz}, to be zero[2].
- Trim is always guaranteed. This has as a consequence that the nominal pitch coefficient $C_m = 0$ and that an external trim law must be included.

[2]Note that this assumption can only be made because of the small values of the products of inertia for HORUS; the chosen reference vehicle. For Apollo-like vehicles this assumption does not hold as the products of inertia are much larger w.r.t the moments of inertia than for HORUS. See Mooij (2014a), pages 241-243, for the resulting changes in the results of this section.

- The nominal sideslip angle is zero.
- The asymmetric translational motion has no effect on the attitude kinematics, i.e., the trajectory is directed along the equator.

Using these assumptions, one gets a simplified set of equations that can be linearised (Mooij, 2014a). The result is a coupled set of linear differential equations which can be written in matrix form as:

$$\Delta\dot{\mathbf{x}} = \mathbf{A}\Delta\mathbf{x} + \mathbf{B}\Delta\mathbf{u} \tag{3-22}$$

where $\Delta\mathbf{x}$ is the $n \times 1$ state vector, $\Delta\mathbf{u}$ is the $m \times 1$ input vector, and $\mathbf{A}$ and $\mathbf{B}$ are the $n \times n$ state matrix and the $n \times m$ input matrix, respectively (Mooij, 2014a; Ogata, 2010). If one defines the state vector as:

$$\Delta\mathbf{x} = (\Delta V, \Delta\gamma, \Delta R, \Delta p, \Delta q, \Delta r, \Delta\alpha, \Delta\beta, \Delta\sigma)^T \tag{3-23}$$

where V is the speed of the object and γ is the flight-path angle. To complete the description of the state of vehicle an output equation is needed:

$$\Delta\mathbf{y} = \mathbf{C}\Delta\mathbf{x} + \mathbf{D}\Delta\mathbf{u} \tag{3-24}$$

where $\Delta\mathbf{y}$ is the $k \times 1$ output vector, and $\mathbf{C}$ and $\mathbf{D}$ are the $k \times n$ output and $k \times m$ transmission matrices, respectively (Mooij, 2014a; Ogata, 2010). In this report, it is assumed that the output vector is identical to the state vector, a selection of, or a combination of the state variables of the system. This means that $\mathbf{D}$ is a zero matrix, and Equation 3-24 can thus be written as:

$$\Delta\mathbf{y} = \mathbf{C}\Delta\mathbf{x} \tag{3-25}$$

This assumption is justified because the original controller of the chosen reference vehicle, HORUS, has the state of the vehicle as the "measured" state variables, see Section 4-3. This assumption would be false if one uses the control variables as input to an output feedback scheme, but this is not considered in this report.

For a winged re-entry vehicle it is logical to assume the following control vector $\Delta\mathbf{u}$, which can be written as (Mooij, 2014a):

$$\Delta\mathbf{u} = (\Delta\delta_e, \Delta\delta_a, \Delta\delta_r, \Delta M_{T,x}, \Delta M_{T,y}, \Delta M_{T,z})^T \tag{3-26}$$

with

δ_e = elevator deflection angle [rad]
δ_a = aileron deflection angle [rad]
δ_r = rudder deflection angle [rad]
$M_{T,x}$ = roll thruster moment [Nm]
$M_{T,y}$ = pitch thruster moment [Nm]
$M_{T,z}$ = yaw thruster moment [Nm]

The state vector differential equation can now be fully written out. For the HORUS-2B re-entry vehicle, the full state matrix and the input matrix are given in Appendix A.

When considering the dynamics of the attitude only, one can use a reduced form of the full linear differential equations. By setting $\Delta\dot{V} = \Delta\dot{\gamma} = \Delta\dot{R} = \Delta V = \Delta\gamma = \Delta R = 0$, one gets a reduced set of equations;

$$\begin{pmatrix} \Delta\dot{p} \\ \Delta\dot{q} \\ \Delta\dot{r} \\ \Delta\dot{\alpha} \\ \Delta\dot{\beta} \\ \Delta\dot{\sigma} \end{pmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & a_{p\beta} & 0 \\ 0 & 0 & 0 & a_{q\alpha} & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{r\beta} & 0 \\ 0 & a_{\alpha q} & 0 & a_{\alpha\alpha} & 0 & a_{\alpha\sigma} \\ a_{\beta p} & 0 & a_{\beta r} & 0 & a_{\beta\beta} & a_{\beta\sigma} \\ a_{\sigma p} & 0 & a_{\sigma r} & a_{\sigma\alpha} & a_{\sigma\beta} & a_{\sigma\sigma} \end{bmatrix} \begin{pmatrix} \Delta p \\ \Delta q \\ \Delta r \\ \Delta\alpha \\ \Delta\beta \\ \Delta\sigma \end{pmatrix} + \begin{bmatrix} 0 & b_{pa} & 0 & b_{px} & 0 & 0 \\ b_{qe} & 0 & 0 & 0 & b_{qy} & 0 \\ 0 & b_{ra} & b_{rr} & 0 & 0 & b_{rz} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \Delta\delta_e \\ \Delta\delta_a \\ \Delta\delta_r \\ \Delta M_{T,x} \\ \Delta M_{T,y} \\ \Delta M_{T,z} \end{pmatrix} \tag{3-27}$$

where the none-zero elements can be found in Appendix A.

It is useful to consider the required moment-fractions as the control output, and in a later stage to allocate them to the active actuators. In this manner, the controller architecture is independent of which actuators are active. In this case, Equation 3-27 can be changed to:

$$\begin{pmatrix} \Delta\dot{p} \\ \Delta\dot{q} \\ \Delta\dot{r} \\ \Delta\dot{\alpha} \\ \Delta\dot{\beta} \\ \Delta\dot{\sigma} \end{pmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & a_{p\beta} & 0 \\ 0 & 0 & 0 & a_{q\alpha} & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{r\beta} & 0 \\ 0 & a_{\alpha q} & 0 & a_{\alpha\alpha} & 0 & a_{\alpha\sigma} \\ a_{\beta p} & 0 & a_{\beta r} & 0 & a_{\beta\beta} & a_{\beta\sigma} \\ a_{\sigma p} & 0 & a_{\sigma r} & a_{\sigma\alpha} & a_{\sigma\beta} & a_{\sigma\sigma} \end{bmatrix} \begin{pmatrix} \Delta p \\ \Delta q \\ \Delta r \\ \Delta\alpha \\ \Delta\beta \\ \Delta\sigma \end{pmatrix} + \begin{bmatrix} b^\star_{px} & 0 & 0 \\ 0 & b^\star_{qy} & 0 \\ 0 & 0 & b^\star_{rz} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \eta_x \\ \eta_y \\ \eta_z \end{pmatrix} \tag{3-28}$$

where

$$b^\star_{px} = \frac{\mathcal{L}_{max}}{I_{xx}}, \quad b^\star_{qy} = \frac{M_{max}}{I_{yy}}, \quad b^\star_{rz} = \frac{N_{max}}{I_{zz}} \tag{3-29}$$

where $\mathcal{L}_{max}$, M_{max}, and N_{max} are interpreted as the maximum available moments around the roll-, pitch-, and yaw-axes, respectively. In this manner, η_x, η_y, and η_z are interpreted as the moment-fractions around the roll-, pitch-, and yaw-axes respectively, and are mathematically defined as:

$$\eta_x = \frac{\mathcal{L}_{cmd}}{\mathcal{L}_{max}}, \quad \eta_y = \frac{M_{cmd}}{M_{max}}, \quad \eta_z = \frac{N_{cmd}}{N_{max}} \tag{3-30}$$

where $\mathcal{L}_{cmd}$, M_{cmd}, and N_{cmd} are the commanded control moments (that is, moments caused only by the control actuators). It is noted that the maximum moments always have positive values, whereas the commanded moments can have value that are both positive and negative. This implies that $\eta \in [-1, 1]$.

It was shown by Mooij (1998) that the coupling between the longitudinal and lateral motion is not strong. From Equations 3-27 or 3-28 it can be seen that, when longitudinal and lateral motion are decoupled, that the time-derivative of the angle of attack couples only to the pitch rate. Hence, one can think of the longitudinal controller as a PD-controller. Similar arguments can be made to show that the lateral controller is also a PD-controller. One can introduce integral terms in

the linearised EoM by realizing that the time-derivative of the integral term for the longitudinal motion is simply the angle of attack itself, and thus a PID-controller is obtained. Similarly, two integral terms exist for the lateral motion. The time-derivatives of these integral terms are the sideslip angle and the bank angle. Adding these modifications to Equations 3-27 and 3-28, one obtains

$$\begin{pmatrix} \Delta\dot{p} \\ \Delta\dot{q} \\ \Delta\dot{r} \\ \Delta\dot{\alpha} \\ \Delta\dot{\beta} \\ \Delta\dot{\sigma} \\ \dot{I}_\alpha \\ \dot{I}_\beta \\ \dot{I}_\sigma \end{pmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & a_{p\beta} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_{q\alpha} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{r\beta} & 0 & 0 & 0 & 0 \\ 0 & a_{\alpha q} & 0 & a_{\alpha\alpha} & 0 & a_{\alpha\sigma} & 0 & 0 & 0 \\ a_{\beta p} & 0 & a_{\beta r} & 0 & a_{\beta\beta} & a_{\beta\sigma} & 0 & 0 & 0 \\ a_{\sigma p} & 0 & a_{\sigma r} & a_{\sigma\alpha} & a_{\sigma\beta} & a_{\sigma\sigma} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \Delta p \\ \Delta q \\ \Delta r \\ \Delta\alpha \\ \Delta\beta \\ \Delta\sigma \\ I_\alpha \\ I_\beta \\ I_\sigma \end{pmatrix} + \begin{bmatrix} 0 & b_{pa} & 0 & b_{px} & 0 & 0 \\ b_{qe} & 0 & 0 & 0 & b_{qy} & 0 \\ 0 & b_{ra} & b_{rr} & 0 & 0 & b_{rz} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \Delta\delta_e \\ \Delta\delta_a \\ \Delta\delta_r \\ \Delta M_{T,x} \\ \Delta M_{T,y} \\ \Delta M_{T,z} \end{pmatrix} \tag{3-31}$$

$$\begin{pmatrix} \Delta\dot{p} \\ \Delta\dot{q} \\ \Delta\dot{r} \\ \Delta\dot{\alpha} \\ \Delta\dot{\beta} \\ \Delta\dot{\sigma} \\ \dot{I}_\alpha \\ \dot{I}_\beta \\ \dot{I}_\sigma \end{pmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & a_{p\beta} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_{q\alpha} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{r\beta} & 0 & 0 & 0 & 0 \\ 0 & a_{\alpha q} & 0 & a_{\alpha\alpha} & 0 & a_{\alpha\sigma} & 0 & 0 & 0 \\ a_{\beta p} & 0 & a_{\beta r} & 0 & a_{\beta\beta} & a_{\beta\sigma} & 0 & 0 & 0 \\ a_{\sigma p} & 0 & a_{\sigma r} & a_{\sigma\alpha} & a_{\sigma\beta} & a_{\sigma\sigma} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \Delta p \\ \Delta q \\ \Delta r \\ \Delta\alpha \\ \Delta\beta \\ \Delta\sigma \\ I_\alpha \\ I_\beta \\ I_\sigma \end{pmatrix} + \begin{bmatrix} b^\star_{px} & 0 & 0 \\ 0 & b^\star_{qy} & 0 \\ 0 & 0 & b^\star_{rz} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \eta_x \\ \eta_y \\ \eta_z \end{pmatrix} \tag{3-32}$$

for the linear system with integral terms and moment-fraction with integral terms, respectively. The matrix coefficients can be found in Appendix A.

Chapter 4

Benchmark Controller

The task of a control system is to guarantee that the steering commands, obtained from the guidance system, are carried out such that the actual attitude equals the commanded attitude in a relatively short time interval. Classical control methodologies have been used extensively in the past and are still the *go-to* choice to design a control system. To compare knowledge-based control systems to classical control systems, one must know which classical control methodology is used for the reference vehicle, which is the purpose of the chapter. Section 4-1 will introduce the concept of feedback control, Section 4-2 will present an overview of optimal control theory. Section 4-3 will then conclude with an overview of the original attitude controller of the chosen reference vehicle. Section 4-4 will then conclude this chapter by presenting the changing the the benchmark controller when introducing the moment-fractions and integral terms.

4-1 Feedback control

To make the controller less sensitive to external disturbances, one can introduce the concept of feedback control. A control system with a feedback loop is also known as a closed-loop control system. This is schematically shown in Figure 4-1.

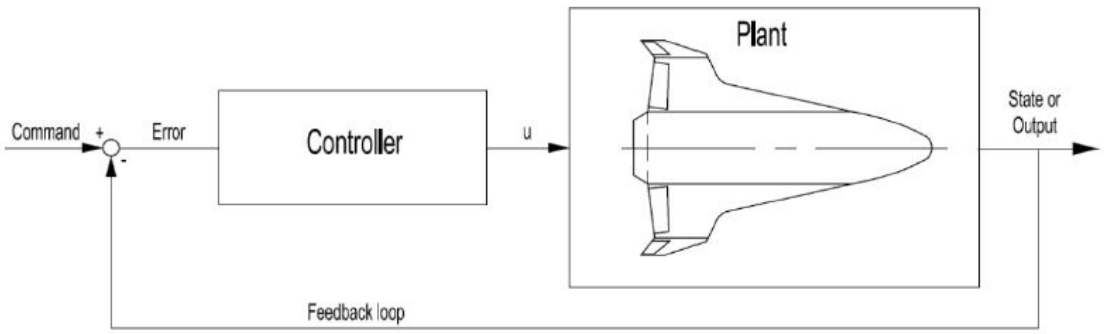


Figure 4-1: Schematic representation of a closed-loop control of an arbitrary plant by means of state or output feedback (Mooij, 2014a).

In Figure 4-1 it can be seen that a controller takes as an input the difference between the state or output of a system, and a commanded signal. It has as an output the control vector, which then serves as an input to the system.

If the equations of motion are described in state-space form, the state equation can be written in the form of Equation 3-22. If the state and input matrices **A** and **B** do not vary with time, then the system is called a Linear-Time-Invariant (LTI) system (Ogata, 2010). The topics discussed in this chapter will deal with this case only, therefore in the remainder of this chapter, it is assumed that the state-space representation of a system is a LTI system.

The open-loop behaviour of the system can be determined by calculation the eigenvalues and eigenvectors of the state matrix **A**. If the system has undesirable behaviour, one can introduce a feedback-loop to change the eigenvalues of the system, with which the behaviour of the closed-loop system can be tuned (Mooij, 2014a). This feedback-loop can either be state feedback or output feedback. In Section 3-7, it was reasoned that the output of the linear system was the state vector or a linear combination thereof. Hence, only state feedback is considered in this report, for which the control law is given by (Ogata, 2010);

$$\Delta \mathbf{u} = -\mathbf{K}\Delta \mathbf{x} \tag{4-1}$$

where **K** is known as the state feedback gain matrix, and controls the behaviour of the closed-loop system. (Mooij, 2014a; Ogata, 2010).

There are several methods to compute the gain matrix, for example, the techniques known as pole-placement (Ogata, 2010). However, using these techniques for high-order systems, such as those for re-entry dynamics, becomes tedious at best and therefore another method for determining the gain matrix is needed. A good method to be used is optimal control theory, which will be discussed in Section 4-2.

Before proceeding, it is useful to consider how the performance and operational characteristics of a control system are specified. Most commonly, they are specified to be the transient response to a unit step input. It is then common the express them using a delay time t_d, a rise time t_r, a peak time t_p, a maximum overshoot M_p, and a settling time t_s (Ogata, 2010). One can also define *performance metrics* with the integrated state deviation and integrated control effort. In the case that the state is represented by the angle of attack, α, the sideslip angle β, and the bank angle, σ, and the control variables of the reaction-control thrusters are represented by the roll, pitch, and yaw moments, $M_{T,x}, M_{T,y}$ and $M_{T,z}$, respectively, the performance metrics are defined by (Mooij, 2014a):

$$\sum_{\alpha_{err}} = \int_0^t \sqrt{(\alpha_c - \alpha_p)^2}\mathrm{d}t \quad \sum_{\beta_{err}} = \int_0^t \sqrt{(\beta_c - \beta_p)^2}\mathrm{d}t \quad \sum_{\sigma_{err}} = \int_0^t \sqrt{(\sigma_c - \sigma_p)^2}\mathrm{d}t \tag{4-2}$$

$$\sum_{x} = \int_0^t |M_{T,x}|\mathrm{d}t \qquad \sum_{y} = \int_0^t |M_{T,y}|\mathrm{d}t \qquad \sum_{z} = \int_0^t |M_{T,z}|\mathrm{d}t \tag{4-3}$$

where the subscripts "c" and "p" refer to the commanded and plant states, respectively. These metrics should preferably be as small as possible. It is noted that similar definitions exist for the performance metrics for the rotational rates and the aerodynamics control surface deflections.

Another measure of performance is the *robustness* of the system. Robustness is defined as "the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions." (IEEE, 1990). It means that the controller must not only work for the system or environment it has been designed for, but for a whole family of (similar)

systems and environments. If the controller is stable for a set of systems whose parameters deviate substantially from the nominal system, then the controller is said to have *robust stability*. To test the robustness of the system, one usually simulates a number of test cases with different error sources included, and with all dynamic, vehicle and environment models as accurately as possible. If in every simulation no constraints are violated, it can be said to have robust stability. Such tests are collectively known as *sensitivity analysis* (Mooij, 2014a).

4-2 Optimal Control Theory

A method commonly used to find the optimal feedback gain matrix is through the use of optimal control theory, where the feedback gain matrix is now computed as an optimization problem. Section 4-2-1 will present the general theory of the optimal control theory and Section 4-2-2 will present a specific case of optimal control, the LQR.

4-2-1 General Theory

In general, the optimal control problem can be stated as follows. Find the admissible function $\mathbf{u}(t)$ that minimize a cost function J, while the state variables $\mathbf{x}(t)$ satisfy a dynamic constraint along with the associated boundary conditions (Visser, 2014):

$$\min_{\mathbf{u}(t)\in U} J = \int_{t_0}^{t_f} \mathcal{L}[\mathbf{x}(t), \mathbf{u}(t), t]\mathrm{d}t \tag{4-4}$$

$$\text{subject to : } \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \tag{4-5}$$

where $\mathcal{L}$ is known as the Lagrangian specific to the problem that needs optimization. There exist many techniques to solve this optimization problem such as using a variational approach or numerical methods (Visser, 2014). However, in the next section a specific case of the optimal control theory will be highlighted.

4-2-2 Linear Quadratic Regulator

An advantage of the quadratic optimal control method over pole placement techniques that are commonly used to find the feedback gain matrix, is that the former provides a systematic way of computing the state feedback gain matrix. In this section the methods used to compute the state feedback gain matrix for a LTI system will be presented. The problem can be formulated as follows (Visser, 2014):

$$\begin{aligned} \min_{\mathbf{u}(t)\in U} J = \int_0^{\infty} \left(\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u}\right) \mathrm{d}t \\ \text{subject to : } \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \text{and : } \dot{\mathbf{u}} = -\mathbf{K}\mathbf{x} \end{aligned} \tag{4-6}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are positive-definite real symmetric matrices. Note that here $\mathbf{x}^T\mathbf{Q}\mathbf{x}$ determines the state deviation and $\mathbf{u}^T\mathbf{R}\mathbf{u}$ represents the control effort (Mooij, 2014a; Visser, 2014). An appropriate choice of the matrices $\mathbf{Q}$ and $\mathbf{R}$ must be made to obtain acceptable behaviour. For example, one may chose these matrices to be diagonal with (Bryson and Ho, 1975):

$$\mathbf{Q}_{ii} = \frac{1}{\Delta x_{i_{max}}^2}, \quad \mathbf{R}_{ii} = \frac{1}{\Delta u_{i_{max}}^2} \tag{4-7}$$

where $\Delta x_{i_{max}}$ is the maximum allowable amplitude of the i-th element of the state vector, and $\Delta u_{i_{max}}$ is the maximum allowable value of the j-th control variable. Choosing the matrices as shown above is known as *Bryson's rule* (Bryson and Ho, 1975). Substituting the dynamic constraints into the cost function of Equation 4-6 and solving the resulting equation, one obtains:

$$\mathbf{A^T P + PA - PBR^{-1}BP + Q = 0} \tag{4-8}$$

$$\mathbf{K = R^{-1}B^T P} \tag{4-9}$$

Equation 4-8 is known as the matrix Riccati equation and gives a matrix $\mathbf{P}$ as the solution, with which the optimal gain matrix can be computed using Equation 4-9. Solving the matrix Riccati equation is commonly done with standard algorithms available in toolboxes for MATLAB.

4-3 Attitude Controller of Reference Vehicle

The design of the original attitude control of the HORUS-2B is centred around a nominal trajectory. The equations of motion were linearised, obtaining Equation 3-27, and the longitudinal and lateral motions where decoupled, as it was shown by Mooij (1998) that the coupling between longitudinal and lateral motion was not strong.

The longitudinal controller consists of an inner and an outer loop. The inner loop takes care of longitudinal trim while the outer loop provides corrective control, *i.e.*, to make the actual attitude approach the commanded one in a finite time. The longitudinal trim is primarily taken care of by the body-flap of the vehicle, assisted by the elevons when required. For corrective longitudinal control, the elevators and pitch jets can be used, whereas the lateral corrective control is achieved by using the ailerons, the rudders, and the roll and yaw jets (Mooij, 1998, 2014a).

Mathematically, the trim algorithm is implemented as follows. First, the pitch moment of the vehicle, C_{m_0}, is computed from the on-board tables taking α_c and M_{nav} as input. Note that α_c is taken as input instead of α_{nav}, since the vehicle should be stable for α_c whereas α_{nav} might have an error, and that M_{nav} is the Mach number as given by the navigation system. Using inverse interpolation, the corresponding deflection angle of the body-flap, $\delta_{b_{trim}}$, is extracted from the aerodynamic tables. When the maximum body-flap deflection is not enough, the remaining moment coefficient is compensated for by the elevons with an elevon deflection angle, $\delta_{e_{trim}}$, which is similarly computed using inverse interpolation (Mooij, 1998).

When neglecting the lateral motion of the system by setting $\Delta\beta = \Delta\sigma = 0$ in Equation 3-27, the following approximation can be derived for the longitudinal dynamics:

$$\Delta\dot{q} = \frac{1}{I_{yy}} \left(\frac{\partial C_m}{\partial \alpha} \Delta\alpha + \frac{\partial C_m}{\partial \delta_e} \Delta\delta_e \right) \bar{q}_0 S_{ref} c_{ref} + \frac{\Delta M_{T,y}}{I_{yy}} \tag{4-10}$$

$$\Delta\dot{\alpha} = \Delta q - \frac{1}{mV_0} \frac{\partial C_L}{\partial \alpha} \bar{q}_0 S_{ref} \Delta\alpha \tag{4-11}$$

where $\bar{q}_0$ refers to the dynamic pressure at the point of linearisation. As control variables, the symmetric elevon deflection angle δ_e (elevator function) and the thruster moment about the pitch axis, $M_{T,y}$, are available. The activation of the different actuators can be found in Figure 2-2.

The state-feedback laws for longitudinal control are selected to be simple proportional laws (Mooij, 2014a):

$$\Delta\delta_e = -K_{eq}\Delta q - K_{e\alpha}\Delta\alpha \tag{4-12}$$

$$\Delta M_{T,y} = -K_{yq}\Delta q - K_{y\alpha}\Delta\alpha \tag{4-13}$$

where $\Delta q = q - q_c$, $\Delta\alpha \approx \alpha - \alpha_c$, and q_c and α_c are the commanded pitch rate and angle of attack, respectively. These control laws are linear combinations of the state variables, and can thus are of the right form for the LQR. Note that, due to the relation between α and q, the controller can be thought of as a PD controller. Writing the equations of motion of Equations 4-10 and 4-11, and the control laws of Equations 4-12 and 4-13 into matrix form, one can use the theory of LQR, as discussed in Section 4-2-2, to compute the gains by solving the Riccati equation, which is shown in Equation 4-8, after which the feedback gain matrix is computed with Equation 4-9. The weighting-matrices are then chosen to be (Mooij, 2014a):

$$\mathbf{Q} = \text{diag}\left\{\frac{1}{\Delta q_{max}^2}, \frac{1}{\Delta\alpha_{max}^2}\right\}, \quad \mathbf{R} = \text{diag}\left\{\frac{1}{\Delta\delta_{e_{max}}^2}, \frac{1}{\Delta M_{T,y_{max}}^2}\right\} \tag{4-14}$$

where the maximum allowed amplitudes of the state vector and control variables are different depending on which actuators are active. They are summarized in Table 4-1 for the longitudinal controller.

Table 4-1: Variations of the maximum allowed amplitudes of the state vector and control variables for longitudinal control depending on which actuators are active or Not Active (NA).

	Pitch thruster	Pitch thruster and elevator	Elevator
Δq_{max} [deg/s]	1.5	1.5	4
$\Delta\alpha_{max}$ [deg]	1	2.5	2
$\Delta\delta_{e_{max}}$ [deg]	NA	40	40
$\Delta M_{T,y_{max}}$ [Nm]	10,400	10,400	NA

It is noted that when either the elevator or pitch thruster is not used during a portion of the flight, the input vector $\mathbf{\Delta u}$ and the input matrix $\mathbf{B}$ are modified such that only the active actuators are included in the state equation. The feedback gain matrix then only includes gains for the active actuators. See Figure 2-2 in Section 2-1 for when which actuators are active.

By neglecting the longitudinal motion by setting $\Delta\alpha = 0$, the following set of equations can be derived for the lateral dynamics:

$$\Delta\dot{p} = \frac{1}{I_{xx}}\left(\frac{\partial C_l}{\partial\beta}\Delta\beta + \frac{\partial C_l}{\partial\delta_a}\Delta\delta_a\right)\bar{q_0}S_{ref}b_{ref} + \frac{\Delta M_{T,x}}{I_{xx}} \tag{4-15}$$

$$\Delta\dot{r} = \frac{1}{I_{zz}}\left(\frac{\partial C_n}{\partial\beta}\Delta\beta + \frac{\partial C_n}{\partial\delta_a}\Delta\delta_a + \frac{\partial C_n}{\partial\delta_r}\Delta\delta_r\right)\bar{q_0}S_{ref}b_{ref} + \frac{\Delta M_{T,z}}{I_{zz}} \tag{4-16}$$

$$\Delta\dot{\beta} = \sin\alpha_0\Delta p - \cos\alpha_0\Delta r - \frac{g_0}{V_0}\cos\gamma_0\cos\sigma_0\Delta\sigma \tag{4-17}$$

$$\Delta\dot{\sigma} = -\cos\alpha_0\Delta p - \sin\alpha_0\Delta r + \left(\frac{g_0}{V_0}\cos\gamma_0\cos\sigma_0 - \frac{L_0}{mV_0}\right)\Delta\beta + \frac{\tan\gamma_0}{mV_0}\cos\sigma_0 L_0\Delta\sigma \tag{4-18}$$

where the available control consists of the aileron, rudder, roll jets and yaw jets. The control laws for lateral control can be written as (Mooij, 2014a):

$$\Delta\mathbf{u} = \begin{pmatrix} \Delta\delta_a \\ \Delta\delta_r \\ \Delta M_{T,x} \\ \Delta M_{T,z} \end{pmatrix} = - \begin{bmatrix} K_{ap} & K_{ar} & K_{a\beta} & K_{a\sigma} \\ K_{rp} & K_{rr} & K_{r\beta} & K_{r\sigma} \\ K_{xp} & K_{xr} & K_{x\beta} & K_{x\sigma} \\ K_{zp} & K_{zr} & K_{z\beta} & K_{z\sigma} \end{bmatrix} \begin{pmatrix} \Delta p \\ \Delta r \\ \Delta\beta \\ \Delta\sigma \end{pmatrix} = -\mathbf{K}\Delta\mathbf{x} \tag{4-19}$$

Equation 4-19 has the right form to be used for the LQR. It is noted that the minus sign of Equation 4-19 is absorbed by the reverse order of the signs used to compute the state error. That is, $p_c - p = -\Delta p$. The weighting-matrices are then chosen to be (Mooij, 2014a):

$$\mathbf{Q} = \text{diag}\left\{ \frac{1}{\Delta p_{max}^2}, \frac{1}{\Delta r_{max}^2}, \frac{1}{\Delta\beta_{max}^2}, \frac{1}{\Delta\sigma_{max}^2} \right\} \tag{4-20}$$

$$\mathbf{R} = \text{diag}\left\{ \frac{1}{\Delta\delta_{a_{max}}^2}, \frac{1}{\Delta\delta_{r_{max}}^2}, \frac{1}{\Delta M_{T,x_{max}}^2}, \frac{1}{\Delta M_{T,z_{max}}^2} \right\} \tag{4-21}$$

Table 4-2: Variations of the maximum allowed amplitudes of the state vector and control variables for nominal lateral control depending on which actuators are active.

	Roll and yaw thruster	Roll and yaw thruster and ailerons	Roll and yaw thruster, aileron and rudder	Yaw thruster, rudder, and aileron
Δp_{max} [deg/s]	1.5	2.5	2.5	10
Δr_{max} [deg/s]	1.5	1.5	10	10
$\Delta\beta_{max}$ [deg]	1	1	2	2
$\Delta\sigma_{max}$ [deg]	4	4	5	10
$\Delta\delta_{a_{max}}$ [deg]	NA	40	40	40
$\Delta\delta_{r_{max}}$ [deg]	NA	NA	40	40
$\Delta M_{T,x_{max}}$ [Nm]	1,600	1,600	1,600	NA
$\Delta M_{T,z_{max}}$ [Nm]	7,600	7,600	7,600	7,600

where the maximum allowed amplitudes of the state vector and control variables are different depending on which actuators are active. Additionally, for large bank angle errors, which occur during bank-reversals, the maximum allowed amplitudes are modified to limit the induced sideslip angle due to large roll rates. For the nominal lateral controller, the maximum allowed amplitudes are summarized in Tables 4-2. For the bank-reversal amplitudes, only the maximum allowed amplitude $\Delta\sigma_{max}$ is changed in the phase were the yaw-thruster, the rudders, and the ailerons are active. It's value is changed to be 30 degrees. It is noted that care needs to be taken here whether the actuators are operating or not. To accomplish this, an identical method as was used for the longitudinal motion can be used to command the closed-loop system not to use an off-line actuator. Furthermore, when the bank angle error is less than two degrees, the gains resulting from the weight matrices of the nominal lateral controller are used. If the bank angle error is larger than 10 degrees, the gains resulting from the bank-reversal weight matrices are used. When the bank angle error is larger than two degrees, but less than 10 degrees, the gains are obtained by means of linear interpolation between the two sets of gains. In this manner the gains are scheduled as a function of the bank angle, and allows for reducing the induced sideslip angle due to large roll rates when the bank angle error is large, while simultaneously allowing for sharp responses when the bank angle error is small.

It is noted that the computation of the feedback gains is done at regular intervals, which allows for the adaptation of the control system to the rapidly changing flight environment and states.

Practically, this means that the gains are computed off-line for a selected number of points along the trajectory. These gains are then stored in on-board reference tables as a function of the dynamic pressure. In-flight computation of the actual gains is based on linear interpolation for the local value of the dynamic pressure. This scheme is also known as *gain scheduling* and is employed in the original control system of the HORUS-2B. It is noted that, in effect, a new controller is designed at multiple points along the reference trajectory, and in this manner one can take into account scheduled architecture changes.

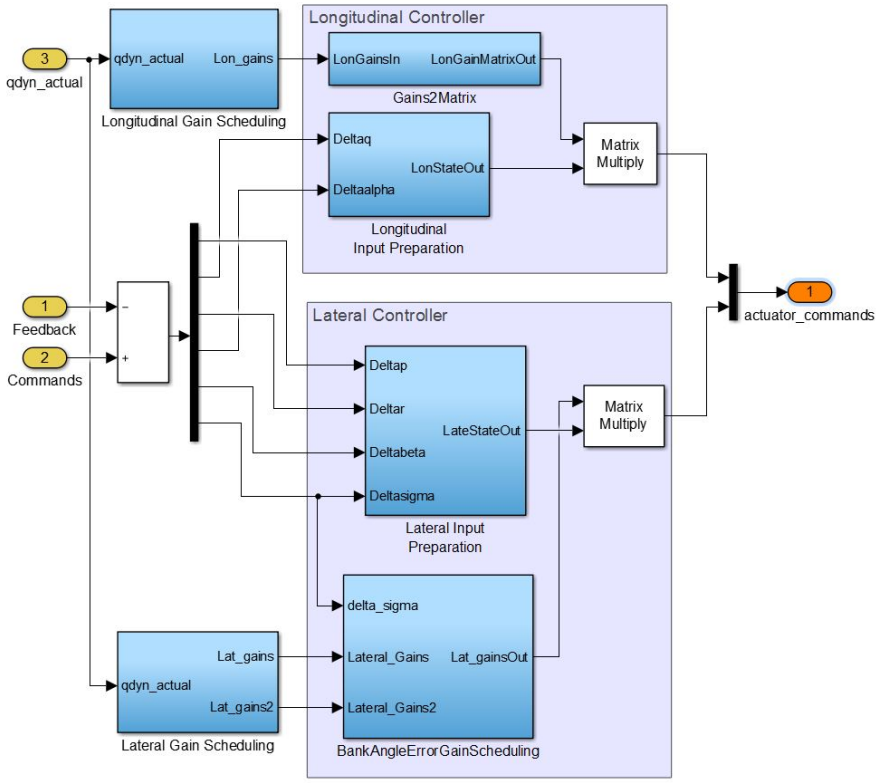


Figure 4-2: Software architecture of the benchmark controller.

The architecture of the benchmark controller is shown in Figure 4-2, and shows the flow of information through the controller. The gains are determined from the on-board reference tables and are constructed into matrices. Likewise, the longitudinal and lateral state variables are concatenated into vectors, which in turn are multiplied with the gain matrices to obtain the appropriate control action.

It is noted that the commanded rotational rates are chosen such that the aerodynamic rates $\dot{\alpha}$, $\dot{\beta}$, and $\dot{\sigma}$ are zero. The resulting commanded rotational rates can be computed by (Mooij, 2014a):

$$\begin{aligned} p_c &= \frac{g_0}{V_0}\cos\gamma_0 \sin\sigma_0 \sin\alpha_0 + \frac{L_0}{mV_0}\tan\gamma_0 \sin\sigma_0 \cos\alpha_0 \\ q_c &= \frac{L_0}{mV_0} - \frac{g_0}{V_0}\cos\gamma_0 \cos\sigma_0 \\ r_c &= -\frac{g_0}{V_0}\cos\gamma_0 \sin\sigma_0 \cos\alpha_0 + \frac{L_0}{mV_0}\tan\gamma_0 \sin\sigma_0 \sin\alpha_0 \end{aligned} \tag{4-22}$$

However, for the evaluation of the attitude controller at a specific point along the reference trajectory, the commanded rotational rates are set to zero. This is done because Equations 4-22 are related to the motion of the A-frame w.r.t. I-frame. These motions are not relevant at a specific point along the reference trajectory, and hence the commanded rotational rates are set to zero.

4-4 Modified Benchmark Controllers

For the benchmark controller, effectively, a new controller is designed at multiple points along the reference trajectory, and in this manner one can take into account scheduled architecture changes. Managing architecture changes in this manner is, however, not always possible for different types of controllers. Hence, it was found that it is useful to consider the required moment-fractions as the control output, instead of directly computing the actuator states. Additionally, it was mentioned that the original benchmark controller, as presented in Section 4-3, represented a PD-controller and could be modified to include an integral term, changing the controller to a full PID-controller. This section will present the details of the construction of the resulting controllers. Section 4-4-1 will present the controller obtained when the control outputs are the moment-fractions, and this controller will be referred to as the Moment-fraction Controller (MFC). Introducing integral terms into the linearised EoM allows for the creation of a controller that is different than the benchmark controller, and the resulting Linear Quadratic Integral Controller (LQIC) is discussed in Section 4-4-2. This section is then concluded by the design of a controller combining the MFC and the LQIC to create the Moment-fraction Integral Controller (MFIC). It is noted that the architecture of the modified benchmark controllers are identical to that of the benchmark controller, as shown in Figure 4-2, but the inputs are modified for the LQIC and the MFIC to include integral terms.

4-4-1 Moment-fraction Controller

Defining the control output to be the moment-fractions leads to different sets of linearised EoM for the longitudinal and lateral motion. For the longitudinal dynamics, Equation 4-10 is modified to:

$$\Delta\dot{q} = \frac{\bar{q}_0 S_{ref} c_{ref}}{I_{yy}} \frac{\partial C_m}{\partial \alpha} \Delta\alpha + \frac{M_{max}}{I_{yy}} \eta_y \tag{4-23}$$

The moment-fraction about the y-axis, η_y, is the only available control variable. It is noted that the activation modes of the actuators are not considered here, as it is the task of the actuator assignment algorithm (see Chapter 5) to correctly allocate the commanded moments to the active actuators.

Defining the state-feedback laws for the longitudinal controller to be analogous to the control laws for the benchmark longitudinal controller, the following state-feedback law is obtained:

$$\eta_y = -K_{yq}\Delta q - K_{y\alpha}\Delta\alpha \tag{4-24}$$

which is of the right form for the LQR. It is noted that the gains of Equation 4-24 are not identical to those found in Equation 4-13. The weighting-matrices are chosen in accordance to Bryson's rule and the control weighting-matrix $\mathbf{R}$ of Equation 4-14 is modified and given by:

$$\mathbf{R} = \frac{1}{\eta_{y,max}^2} \tag{4-25}$$

The maximum allowed amplitudes of the state vector and the control variables are summarized in Appendix B, Table B-1, for the moment-fraction longitudinal controller. It is noted that the amplitudes were chosen such that controller can stabilize an instantaneous change in the angle of attack of one degree. This is was done for all controllers discussed in this chapter.

While gain scheduling is no longer required to take the actuators modes into account, the effectiveness of the actuators change significantly during the flight. For example, 10% of the maximum available moment when only the pitch thrusters are active and the dynamic pressure is

low, produces a significantly smaller commanded moment than a 10% moment-fraction when only the elevator is active and the dynamic pressure is high, and thus, for the former case, a larger moment-fraction is required for identical state-errors. Hence, a reduced number of gain-scheduled parameters are still required for optimal control responses to state-errors.

For the lateral dynamics, Equations 4-15 and 4-16 are modified to:

$$\Delta\dot{p} = \frac{\bar{q}_0 S_{ref} b_{ref}}{I_{xx}} \frac{\partial C_l}{\partial \beta} \Delta\beta + \frac{\mathcal{L}_{max}}{I_{xx}} \eta_x \tag{4-26}$$

$$\Delta\dot{r} = \frac{\bar{q}_0 S_{ref} b_{ref}}{I_{zz}} \frac{\partial C_n}{\partial \beta} \Delta\beta + \frac{N_{max}}{I_{zz}} \eta_z \tag{4-27}$$

where the moment-fractions about the x-axis, η_x, and the z-axis, η_z, are the two available control variables. The state-feedback control laws can then be written as:

$$\Delta\mathbf{u} = \begin{pmatrix} \eta_x \\ \eta_z \end{pmatrix} = -\begin{bmatrix} K_{xp} & K_{xr} & K_{x\beta} & K_{x\sigma} \\ K_{zp} & K_{zr} & K_{z\beta} & K_{z\sigma} \end{bmatrix} \begin{pmatrix} \Delta p \\ \Delta r \\ \Delta\beta \\ \Delta\sigma \end{pmatrix} = -\mathbf{K}\Delta\mathbf{x} \tag{4-28}$$

The weighting-matrices are chosen in accordance to Bryson's law and the control weighting-matrix of Equation 4-21 is modified to:

$$\mathbf{R} = \text{diag}\left\{ \frac{1}{\eta_{x,max}^2}, \frac{1}{\eta_{z,max}^2} \right\} \tag{4-29}$$

Similar to the original benchmark controller of the previous section, for large bank angle errors the maximum allowed amplitudes are modified to limit the induced sideslip angle due to large roll rates. For the lateral controller, the maximum allowed amplitudes are summarized in Table B-2. It is noted that the amplitudes were chosen such that the controller can stabilize an instantaneous change in sideslip angle of one degree, and/or all bank manoeuvres encountered during the flight. This was done for all controllers discussed in this chapter.

4-4-2 Linear Quadratic Integral Controller

Including integral terms in the state vector leads to different sets of linearised EoM for the longitudinal and lateral motion. For the longitudinal dynamics, the dynamic equation describing the evolution of the integral term is added to Equations 4-10 and 4-11, and is given by:

$$\dot{I}_\alpha = \Delta\alpha \tag{4-30}$$

The state-feedback laws for the longitudinal controller of the benchmark controller are modified to include an additional term taking into account the integral term, and can be written as:

$$\Delta\delta_e = -K_{eq}\Delta q - K_{e\alpha}\Delta\alpha - K_{eI_\alpha} I_\alpha \tag{4-31}$$

$$\Delta M_{T,y} = -K_{yq}\Delta q - K_{y\alpha}\Delta\alpha - K_{yI_\alpha} I_\alpha \tag{4-32}$$

The state weighting-matrix of Equation 4-14 is modified to include the maximum allowed amplitude of the integral term and is given by:

$$\mathbf{Q} = \text{diag}\left\{ \frac{1}{\Delta q_{max}^2}, \frac{1}{\Delta \alpha_{max}^2}, \frac{1}{I_{\alpha,max}^2} \right\} \tag{4-33}$$

The maximum allowed amplitudes of the state vector and the control variables are summarized in Table B-3 for the longitudinal LQIC.

For the lateral dynamics, similar to the longitudinal dynamics, the dynamic equation describing the evolution of the integral terms of the sideslip angle and the bank-angle are added to Equations 4-15 to 4-18, and are given by:

$$\dot{I}_\beta = \Delta\beta, \qquad \dot{I}_\sigma = \Delta\sigma \tag{4-34}$$

The state-feedback control law of Equation 4-19 is modified to include linear combinations of the integral terms for each of the control variables. The state weighting-matrices of Equation 4-20 is modified to include the integral terms of the sideslip angle and the bank-angle, and is given by:

$$\mathbf{Q} = \text{diag}\left\{ \frac{1}{\Delta p_{max}^2}, \frac{1}{\Delta r_{max}^2}, \frac{1}{\Delta \beta_{max}^2}, \frac{1}{\Delta \sigma_{max}^2}, \frac{1}{I_{\beta,max}}, \frac{1}{I_{\sigma,max}} \right\} \tag{4-35}$$

where the maximum allowed amplitudes of the state vector and the control variables are different depending on which actuators are active. Additionally, for large bank-angle errors the maximum allowed amplitudes are modified to limit the induced sideslip angle due to large roll rates. For the nominal lateral controller, the maximum allowed amplitudes are summarized in Tables B-4.

4-4-3 Moment-fraction Integral Controller

Changing the control variables to the moment-fraction, as was discussed in Section 4-4-1, and introducing integral terms into the state vector, as was outlined in Section 4-4-2, simultaneously leads to the MFIC. Additionally, the resulting equations describing the longitudinal dynamics are the linear system of Equations 4-11, 4-23, and 4-30. The state-feedback control law of Equation 4-24 with an added linear term due to a contribution of the integral term. Furthermore, the state and control weighting matrices are identical to those of Equations 4-33 and 4-29, respectively. The maximum allowed amplitudes of the state vector and the control variables are summarized in Table B-5.

By neglecting the longitudinal motion in Equation 3-32, the set of equations that are derived for the lateral dynamics are identical to the combined sets of Equations 4-17 and 4-18, Equations 4-26 and 4-27, and Equations 4-34. The state-feedback control laws of Equation 4-28 is modified to include linear combinations of the integral terms for each of the control variables. Additionally, the state and control weighting matrices are identical to those of Equations 4-35 and 4-29, respectively. The maximum allowed amplitudes of the state vector and the control variables are summarized in Table B-6.

Chapter 5

Actuator Assignment

As mentioned in Section 3-7, to make the number of control outputs independent of the number of active actuators, it was chosen to have the control inputs of the (linearised) system be the moment-fractions about an axis. This necessitates that these moment-fractions have to be converted to the actuator commands that can be used by the vehicle, taking into account which actuators are active at any given instance during the flight. Wu et al. (2000) applied an actuator assignment algorithm to compute the control deflections and thruster moments from a set of moment-fractions for the X-38 re-entry vehicle. They linearised the aerodynamic model, and were able to determine the required control actions by first computing the thruster commands, after which the control surface deflections were computed by inverting the linear aerodynamic model, a process known as *aerodynamic inversion*. The combined moment contributions of the thrusters and control flaps complied with the set of commanded moments. However, because the moment coefficients are, in general, not linear functions, the linear approximation will create an error in the realized moment coefficient.

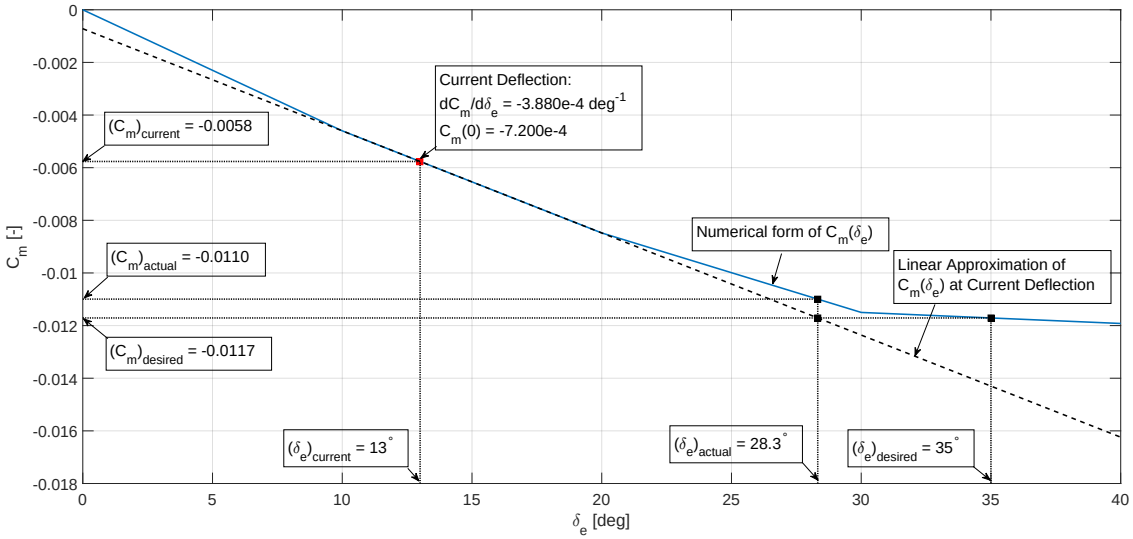


Figure 5-1: Graphical example of the error in the elevon contribution to the pitch moment coefficient when linearising the aerodynamic model, for $\alpha = 40^\circ$ and M = 18.

From Figure 5-1, it can be seen that an error occurs when computing the actual elevon deflection required to comply with the commanded pitch moment coefficient, if use is made of a linear aerodynamic model. It is found that an error of 6% occurs for the shown scenario. The error is largest when the difference between the derivatives of the pitch moment coefficient w.r.t. the elevon deflection of the current and the desired deflection is largest. Additionally, when the change in the pitch moment coefficient w.r.t the current deflection is small, then the error will also be small. It is noted that the linear approximation includes larger values for the pitch moment coefficient than the numerical data, which occurs when it reaches its maximum attainable value and the effectiveness of the elevon becomes low. Hence, the difference between desired and actual pitch moment coefficient is largest when the elevon approaches its saturation value.

The method employed by Wu et al. (2000) relied on the invertibility of the linear aerodynamic model, which was possible only because the number of control flaps were equal to the number of moment coefficients they affected. Durham (1993, 1994a,b, 2001) described an advanced bisecting edge-searching control allocation algorithm for distributing the effort of redundant control effectors. It allows one to find near-optimal solutions when there are more control flaps than moment coefficients. For example, Scalera (1999) applied this control allocation algorithm to the F-15 Advanced Control Technology for Integrated VEhicles (ACTIVE), which has twelve control effectors for the three moment coefficients. Similar to the method by Wu et al. (2000), it utilized a linearised aerodynamic model and, hence, is subject to the same linearisation error. It is noted that the X-38 also has seven control effectors; three thrusters, two elevons, and two rudders. However, by allocating the thrusters based on their availability, and realizing that the two rudders can, effectively, be described as a single rudder, Wu et al. (2000) circumvented the need for advanced control allocation algorithms. The reference vehicle used in this thesis, the HORUS-2B, possesses the same control actuators as the X-38. Therefore, a similar control allocation approach will be followed. However, a new algorithm will be used to compute the actuator commands corresponding to the moment-fractions as given by the controller. This new method will exploit the flatness property of the numerical aerodynamic database (see Section 2-3), and will not make use of a linearised aerodynamic model. Consequently, no linearisation error will occur.

This chapter will discuss the components of the new control allocation algorithm. Section 5-1 will show how the commanded moment coefficients can be obtained from the moment-fractions. The algorithm to compute the elevon deflections required to achieve commanded roll and pitch moment coefficients is presented in Section 5-2. From the computed elevon deflections and the commanded yaw moment coefficient, one can compute the commanded rudder deflections, which will be discussed in Section 5-3. It is possible that the combinations of commanded moment coefficients are not possible to be obtained by the aerodynamic actuators. In this case, one can use the thrusters to compensate for the difference in commanded and obtained moment coefficients, if the thrusters are available. These thruster corrections will be discussed in Section 5-4. The flow diagram of the actuator assignment algorithm can be seen in Figure 5-2. It is noted that the algorithm presented in this chapter will be verified in Section 9-2-2.

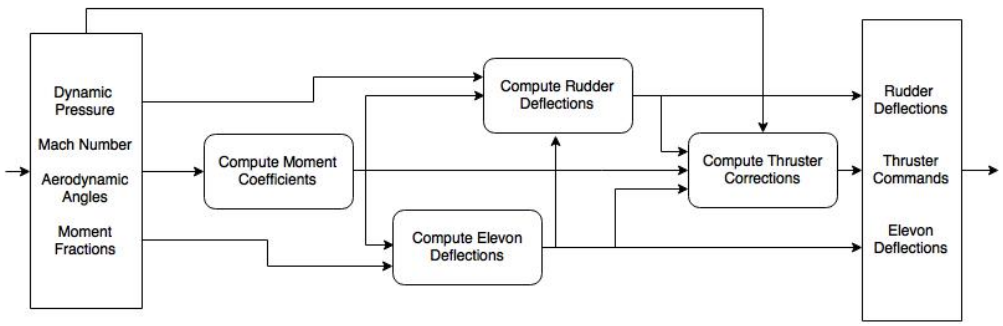


Figure 5-2: Flow diagram of the actuator assignment algorithm, including expected input and output parameters.

5-1 Commanded Moment Coefficients

One can compute the commanded moment coefficients from the moment-fractions using their definition. The moment-fractions are the percentage of the maximum available moment about an axis of the vehicle. Therefore, to compute the commanded moments, one only needs to compute the maximum available moments. However, these moments are dependent on the flight parameters and on which actuators are active at any given time during the flight. It is known from Figure 2-2 when which actuators are active, hence, given the flight conditions, one can determine the maximum available moments. It is noted that the commanded moment coefficients are the contributions to the total moment coefficients due to the elevon and rudder deflections.

Before the maximum available moment can be computed, however, one first needs to determine the maximum achievable moment coefficients. For the roll- and pitch-axes, the positive and negative maximum achievable moment coefficients as a function of the left and right elevons, given a Mach number and angle of attack, are not equal to each other in general. Furthermore, it is guaranteed that the positive maximum coefficient is positive and the negative maximum coefficient is negative, however, it is not known which absolute value of the two is the larger one. Since a positive value is required for the maximum moment coefficient one should choose between the two maximum moment coefficients. Alternatively, one could use the sign of the commanded moment-fraction to determine which one to select. However, this may lead to fluctuations when the moment-fractions are close to zero, which may lead to an unstable system and is thus avoided. If one chooses the bigger absolute moment coefficient one makes an overestimate of the available moment in one direction, and if one chooses the smaller absolute moment coefficient one underestimates it in the other direction. If one chooses the former one may encounter early elevon saturation, whereas if one chooses the latter one never makes full use of the elevon range in one direction. Realizing that the elevons are also used for trim stability if the body-flap is saturated, one should try to prevent elevon saturation as much as possible. It is therefore chosen to select the minimum of the two maximum moment coefficients.

For the yaw-axis, a similar argument can be made as for the roll- and pitch-axes with one difference. Since the rudders can only move outward, and the moment generated by a deflection of the left rudder works against the moment generated by a deflection of the right rudder, one chooses a maximum yaw moment coefficient for which one or both of the rudder deflections are zero, to ensure efficient use of the rudders. This distinction is a formal one, however, since it is guaranteed that the maximum positive and the maximum negative yaw moment coefficient is given when one of the rudder deflections is zero. Therefore, the commanded yaw moment coefficient can be computed in a manner identical as for the roll and pitch moment coefficients.

Based on the above arguments, one can determine the equations to compute the maximum achievable moment coefficients, which are given by:

$$C_{l_{max}} = \min\left(\max\left(C_{l_\delta}\right), -\min\left(C_{l_\delta}\right)\right) \tag{5-1}$$

$$C_{m_{max}} = \min\left(\max\left(C_{m_\delta}\right), -\min\left(C_{m_\delta}\right)\right) \tag{5-2}$$

$$C_{n_{max}} = \min\left(\max\left(C_{n_\delta}\right), -\min\left(C_{n_\delta}\right)\right) \tag{5-3}$$

Using these maximum moment coefficients, one can compute the maximum moments:

$$\mathcal{L}_{max} = \begin{cases} T_{x_{max}} & \text{if } 0 \leq \bar{q} < 100 \\ \mathcal{L}_{aero_{max}} + T_{x_{max}} & \text{if } 100 \leq \bar{q} < 500 \\ \mathcal{L}_{aero_{max}} & \text{if } \bar{q} \geq 500 \end{cases} \tag{5-4}$$

$$M_{max} = \begin{cases} T_{y_{max}} & \text{if } 0 \leq \bar{q} < 100 \\ M_{aero_{max}} + T_{y_{max}} & \text{if } 100 \leq \bar{q} < 1000 \\ M_{aero_{max}} & \text{if } \bar{q} \geq 1000 \end{cases} \tag{5-5}$$

$$N_{max} = \begin{cases} T_{z_{max}} & \text{if } 0 \leq \bar{q} < 150 \text{ and } M \geq 1 \\ N_{aero_{max}} + T_{z_{max}} & \text{if } \bar{q} \geq 150 \text{ and } M \geq 1 \\ N_{aero_{max}} & \text{if } \bar{q} \geq 150 \text{ and } M < 1 \end{cases} \tag{5-6}$$

where

$$\mathcal{L}_{aero_{max}} = C_{l_{max}} \bar{q} S_{ref} b_{ref} \tag{5-7}$$
$$M_{aero_{max}} = C_{m_{max}} \bar{q} S_{ref} c_{ref} \tag{5-8}$$
$$N_{aero_{max}} = C_{n_{max}} \bar{q} S_{ref} b_{ref} \tag{5-9}$$

One can then continue to compute the commanded moments as required by the control system as:

$$\mathcal{L}_{cmd} = \eta_x \mathcal{L}_{max} \tag{5-10}$$
$$M_{cmd} = \eta_y M_{max} \tag{5-11}$$
$$N_{cmd} = \eta_z N_{max} \tag{5-12}$$

The commanded moments then need to be separated into a part provided by the aerodynamic actuators and the thrusters, depending on which actuators are active. The thrusters are only used when the aerodynamic actuators cannot provide the total commanded moment. Then Equations 5-13 through 5-18 can be used to compute the moments assigned to the different types of actuators.

$$\mathcal{L}_{aero} = \text{sign}(\eta_x) \min(\|\mathcal{L}_{cmd}\|, \mathcal{L}_{aero_{max}}) \tag{5-13}$$
$$M_{aero} = \text{sign}(\eta_y) \min(\|M_{cmd}\|, M_{aero_{max}}) \tag{5-14}$$
$$N_{aero} = \text{sign}(\eta_z) \min(\|N_{cmd}\|, N_{aero_{max}}) \tag{5-15}$$
$$T_x = \mathcal{L}_{cmd} - \mathcal{L}_{aero} \tag{5-16}$$
$$T_y = M_{cmd} - M_{aero} \tag{5-17}$$
$$T_z = N_{cmd} - N_{aero} \tag{5-18}$$

Finally one can compute the commanded moment coefficients from the commanded aerodynamic moments using:

$$C_{l_{cmd}} = \frac{\mathcal{L}_{aero}}{\bar{q} S_{ref} b_{ref}} \tag{5-19}$$
$$C_{m_{cmd}} = \frac{M_{aero}}{\bar{q} S_{ref} c_{ref}} \tag{5-20}$$
$$C_{n_{cmd}} = \frac{N_{aero}}{\bar{q} S_{ref} b_{ref}} \tag{5-21}$$

At the end of the block one thus has the commanded roll, pitch, and yaw moment coefficients and the corresponding thruster-moments about each axis. The commanded moment coefficients can now be used to compute the elevon and rudder deflections, which will be the subject of Sections 5-2 and 5-3, respectively.

5-2 Elevon Assignment

The left and right elevons should be chosen such that both the commanded roll and pitch moment coefficients are realized. It is known from the numerical database of the reference vehicle how the roll and pitch moment coefficients vary w.r.t. the left and right elevons, given an angle of attack and Mach number. Each of the moment coefficients can be represented as a surface in three-dimensional space with two independent variables, the left and right elevons, as is shown in Figure 5-3.

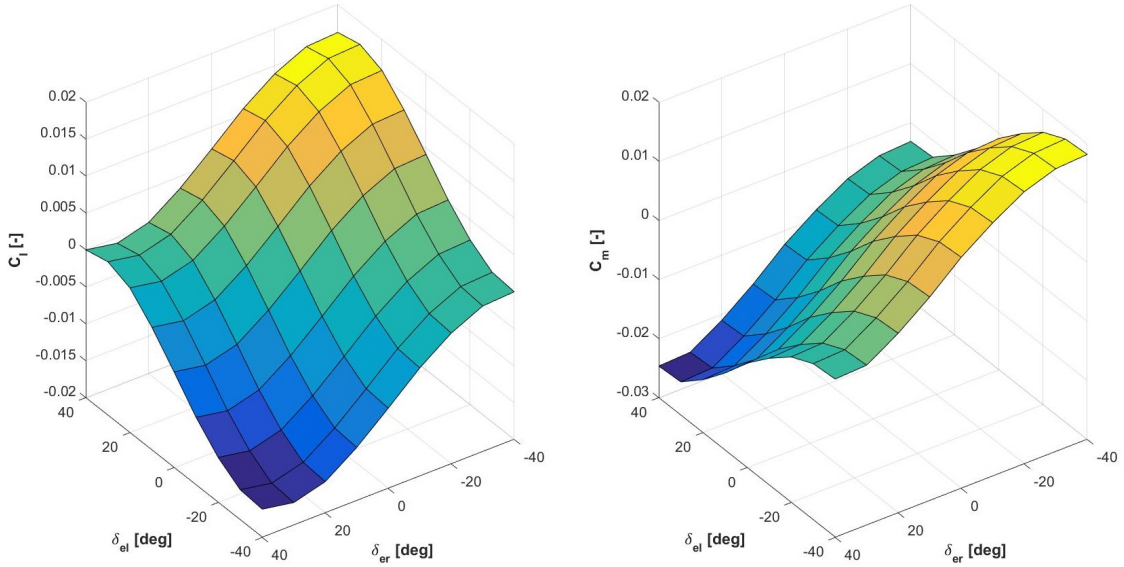


Figure 5-3: Variations of roll and pitch moment coefficients (left and right, respectively) as function of the left and right elevon. Computed as an angle of attack of 40 degrees and a Mach number of 17.

Numerically, the roll and pitch moment coefficients, at a given Mach number and angle of attack, are represented by $N \times N$ matrices. The values for the left and right elevon deflections at which the moment coefficients are evaluated are identical for both the roll and pitch moment coefficients. This property will be exploited later.

When the commanded moment coefficients are subtracted from the surfaces, the points intersecting the plane for which $C_{m_\delta} - C_{m_{cmd}}$ and $C_{l_\delta} - C_{l_{cmd}}$ (the subtracted surfaces) are zero (the XY-plane) gives the possible set of solutions for which one of the commanded moment coefficients is realized. Similarly, the points where both the subtracted surfaces intersect each other and the XY-plane gives the solutions of the left and right elevon deflections for which both the commanded roll and pitch moment coefficients are realized. Mathematically, one can state that the solution space δ is given by:

$$\delta = \underset{\delta_{er},\delta_{el}}{\arg} \left(C_{l_\delta} - C_{l_{cmd}}\right) \cap \left(C_{m_\delta} - C_{m_{cmd}}\right) \cap \left(\bar{XY}\right) \tag{5-22}$$

where C_{l_δ} and C_{m_δ} represent the surfaces of roll and pitch moment coefficients as function of the left and right elevons, $\bar{XY}$ represents the XY-plane and the symbol $\cap$ denotes the intersection operator of two sets. Equation 5-22 should be read as; the solution space is equal to the values of δ_{er} and δ_{el} for which $C_{l_\delta} - C_{l_{cmd}}$, $C_{m_\delta} - C_{m_{cmd}}$, and $\bar{XY}$ intersect each other. It is noted that, theoretically, multiple values of the left and right elevon can be found that realize the commanded

roll and pitch moment coefficients. If this is the case, the algorithm picks the pair of left and right elevon deflections closest to their current values.

The algorithm to determine the solution space δ can be broken down into several steps, taking advantage of the discrete nature of the surfaces, which allows for analysing each surface segment sequentially. First, one needs to find where the three surfaces intersect each other. Secondly, wherever the surfaces intersect, one has to find the set of points at the boundaries of each surface segment where the two segments intersect. Lastly, from the obtained intersection points, one can obtain the values of the elevon deflections for which both the commanded moment coefficients are realized. The first step will be discussed in Section 5-2-1, the second in Section 5-2-2, and the last step in Section 5-2-3. It is, however, always possible that no solution can be found that realizes both commanded moment coefficients. Section 5-2-4 discusses this scenario.

5-2-1 Surface Intersection

Due to the discrete nature of the surfaces, it can be broken down into a collection of surface segments. Hence, to determine where the surfaces intersect, on simply has to find which surfaces segments intersect. These surface segments are flat planes in three-dimensional space. To create an algorithm to determine if two flat surface segments intersect it is instructive to first discuss the intersection of two lines in two-dimensional space. From the obtained insight, one can easily generalize to higher dimensions.

Figure 5-4 shows the intersection of two lines in two-dimensional space. Also shown are the rectangles created by the outermost points of each line. The question of whether two lines are intersecting is now transformed into a question of whether the two rectangles intersect. It is noted that the two rectangles can intersect but the lines do not, as is shown in Figure 5-5. Hence, there is no guarantee that the lines intersect when the rectangles created by the lines do intersect. Therefore, this part of the algorithm only determines likely candidates where line segments may intersect. However, if the two lines consist of multiple line segments, all possible pairs of line segments need to be evaluated to determine whether they intersect and using this method one can eliminate multiple pairs that require no further evaluation further into the algorithm, significantly increasing the speed of the algorithm. As will be seen in Section 5-2-2, one can easily eliminate the cases where the rectangles do intersect, but the line segments do not. The property of intersecting rectangles when their defining line segments do not will be referred to as *phantom intersection* in the remainder of this section.

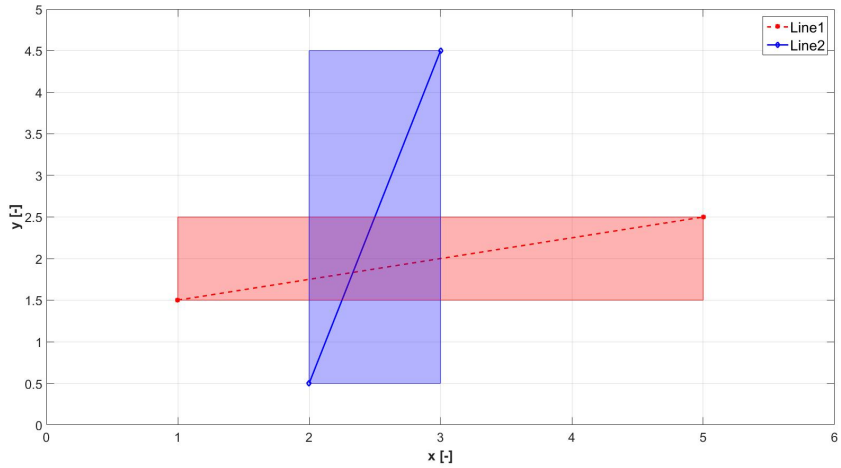


Figure 5-4: Example of two-dimensional line intersection.

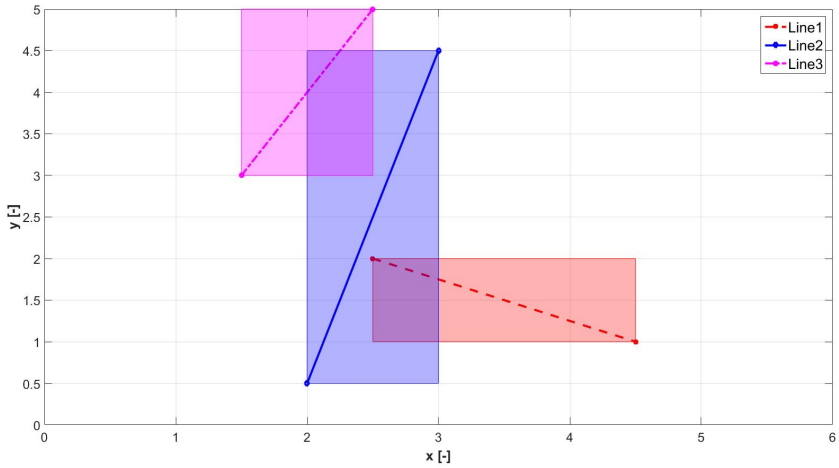


Figure 5-5: Examples of phantom intersection of two-dimensional line segments.

Mathematically, one can determine whether the two rectangles intersect using Equation 5-23.

$$\begin{aligned} I = &(\min(x_1) \leq \max(x_2)) \ \textbf{and} \ (\max(x_1) \geq \min(x_2)) \\ &\textbf{and} \ (\min(y_1) \leq \max(y_2)) \ \textbf{and} \ (\max(y_1) \geq \min(y_2)) \end{aligned} \tag{5-23}$$

where x_i and y_i are the x- and y-values of the i^{th} line segment, respectively, and the **and**-operator is the logical and-operator. When two rectangles intersect, the intersection logical I will have a value of one, otherwise it will have a value of zero.

It is instructive to verify Equation 5-23 by example, using Figure 5-4 as a visual guide. Equation 5-23 states that the smallest x-value of line 1 should be smaller than the largest x-value of line 2. It can be seen that this is true. Additionally, the largest x-value of line 1 should be larger than the smallest x-value of line 2, which is also true. Using the same reasoning one finds that all four conditions of Equation 5-23 are true, hence the two rectangles intersect and the intersection logical I will have a value of one. Because of the logical and-operator, only one of the conditions has to be false for there to be no intersection.

The extension to flat planes in three-dimensional space is straightforward. Instead of checking whether two rectangles intersect, one checks whether two rectangular cuboids intersect. Then for two intersecting plane segments, Equation 5-23 changes to:

$$\begin{aligned} I = &(\min(x_1) \leq \max(x_2)) \ \textbf{and} \ (\max(x_1) \geq \min(x_2)) \\ &\textbf{and} \ (\min(y_1) \leq \max(y_2)) \ \textbf{and} \ (\max(y_1) \geq \min(y_2)) \\ &\textbf{and} \ (\min(z_1) \leq \max(z_2)) \ \textbf{and} \ (\max(z_1) \geq \min(z_2)) \end{aligned} \tag{5-24}$$

where all of the variables have similar meaning as in Equation 5-23. An example of two intersecting plane segments and the resulting rectangular cuboids can be seen in Figure 5-6. It is noted that all of the previously mentioned properties of the intersecting line segments also apply to the intersecting plane segments. However, the property of phantom intersection only applies to cuboids with non-zero height, width, and length. If any of these dimensions have a value of zero, it is guaranteed that the two plane segments intersect. This is the case for an intersecting cuboid with a rectangular plane of the XY-plane.

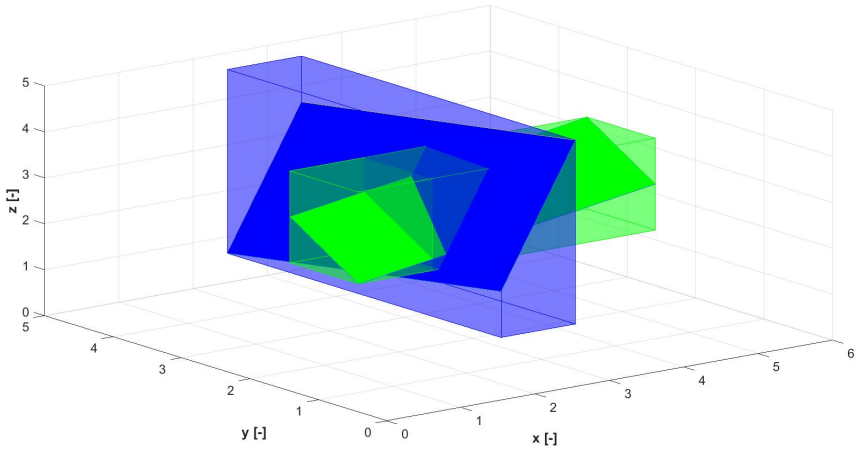


Figure 5-6: Example of plane segments intersecting in three-dimensional space and the resulting rectangular cuboids used to check whether the two plane segments intersect.

The addition of multiple line segments is straightforward. Let $I_{i,j}$ be the intersection logical of line segments i and j. Then for three intersecting line segments, Equation 5-23 changes to:

$$I = I_{1,2} \textbf{ and } I_{1,3} \textbf{ and } I_{2,3} \tag{5-25}$$

Intuitively, Equation 5-25 says that, in order for three line segments to intersect, the first line segment must intersect with the second, the first line segment must intersect with the third, and the second line segment must intersect with third. This line of reasoning can be extended to an arbitrary number of line segments. It is noted that $I_{i,j} = I_{j,i}$, $I_{i,i} = 1$, and for N line segments, there are $N(N-1)/2$ combinations that need to be evaluated. Additionally, Equation 5-25 also applies for intersecting plane segments.

It is now known how to determine whether two plane segments intersect. What remains is to determine which plane segments of two or more surfaces intersect. However, the number of plane segments pairs that have to be checked for two intersecting surfaces increase as $((N_1-1)\times(M_1-1))\times((N_2-1)\times(M_2-1))$, where N_i and M_i are the number of rows and columns of the matrices representing the surfaces, respectively, which increases rapidly with increasing number of plane segments. For example, for two surfaces with square matrices with a size of nine, the number of plane segment pairs that need to be evaluated equals 4096. However, as was mentioned previously, the values for the left and right elevon deflections at which the moment coefficients are evaluated are identical for both the roll and pitch moment coefficients. This means that the first two rows of Equation 5-24 are always satisfied if the same elements[1] are chosen for both matrices, and are never satisfied when different elements are chosen. Hence, only the pairs with identical matrix elements have to evaluated, reducing the number of evaluations to $N \times M$. Additionally, since the first two rows of Equation 5-24 are always satisfied, only the last row has to be evaluated.

At this point in the algorithm it is known which plane segments of the subtracted surfaces and the XY-plane may potentially intersect each other. These plane segments of the subtracted surfaces will be referred to as the *intersecting plane segments* of the subtracted surfaces in the rest of the section. Determining the set of points at the boundaries of each of the intersecting plane segments is the topic of the next subsection. If no plane segments of the subtracted surfaces and the XY-plane intersect (phantom intersections or otherwise), the algorithm will continue to the part of the algorithm discussed in Section 5-2-4.

[1]The same elements of two matrices are picked when for $A(i_1, j_1)$ and $B(i_2, j_2)$, $i_1 = i_2$ and $j_1 = j_2$

5-2-2 Segment Intersection

The intersecting plane segments of the subtracted surfaces are guaranteed to cross the XY-plane due to the fact that the phantom intersection property does not hold for the intersection with a plane segment of the XY-plane. This is due to having a value of zero for the height of the resulting cuboid. However, while it is guaranteed that both subtracted surfaces intersect with the XY-plane, is not guaranteed that both subtracted surfaces also intersect each other and the XY-plane in a single point that lies on all three plane segments. Hence, phantom intersections may still occur. Furthermore, since the plane segments are flat, it is sufficient to know the points where the boundaries of the plane segment intersects each other to know the intersection line of the two plane segments. Additionally, since the values for the left and right elevon deflections at which the moment coefficients are evaluated are identical for both the roll and pitch moment coefficients, it is guaranteed that the boundaries of one plane segment will only cross boundaries of another plane segment. An example of this is shown in Figure 5-7. In essence, the z-values of both plane segments are evaluated for the same four pairs of x- and y-values. This guarantees that the boundary of one plane segments will only intersect a boundary of the second plane segment. It is noted that, since the intersecting plane segments are guaranteed to intersect the XY-plane, one only has to determine the intersecting points of the plane segments of the subtracted surfaces.

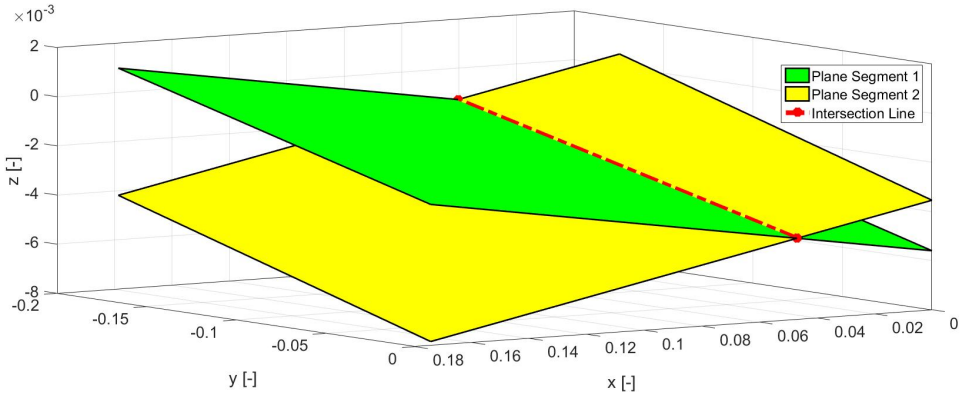


Figure 5-7: Two plane segments intersecting solely at the boundaries due to all four vertices of both plane segments having the same x- and y-values.

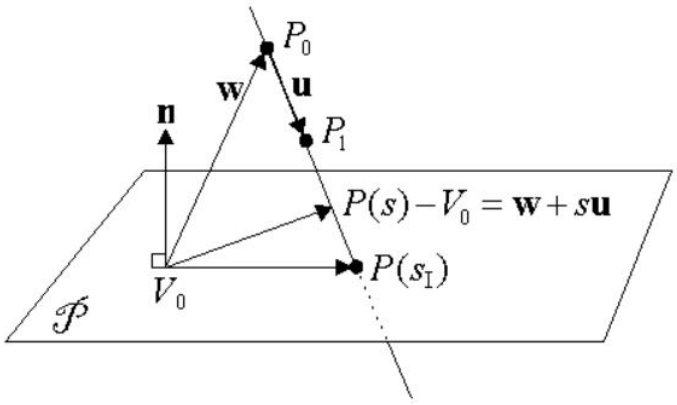


Figure 5-8: Visual representation of a line crossing a plane in three-dimensional space (Hughes et al., 2013).

To determine whether the line segment corresponding to a plane segment boundary crosses another plane segment, one has to check if and where on the line it crosses the plane segment. Multiple methods exist to accomplish this. In three-dimensional space, a line $\mathbf{L}$ can be parametrised as:

$$\mathbf{P}(s) = \mathbf{P_0} + s\,(\mathbf{P_1} - \mathbf{P_0}) \tag{5-26}$$

and the plane $\mathcal{P}$ is given by an arbitrary point $\mathbf{V_0}$ on the plane an a normal vector $\mathbf{n}$ (Hughes et al., 2013). The problem can then be visualized as shown in Figure 5-8. The line then intersects the plane for a parameter value as computed by:

$$s_I = \frac{(\mathbf{V_0} - \mathbf{P_0}) \cdot \mathbf{n}}{(\mathbf{P_1} - \mathbf{P_0}) \cdot \mathbf{n}} \tag{5-27}$$

Since the line $\mathbf{L}$ is a finite line segment from $\mathbf{P_0}$ to $\mathbf{P_1}$, a valid solution is only obtained when $0 \leq s_I \leq 1$. If this condition is satisfied the line segment intersects the plane segment. It is noted that, since the plane segment is also finite, one also has to check whether the finite line segment intersects the finite plane segment. However, it is guaranteed that if the line segment intersects the finite plane-segment, it does so on the boundaries of the plane segment, and therefore any valid intersection point lies on the plane segment and thus this condition does not have to be checked. This also implies that the algorithm only has to check the boundaries of one plane segment intersecting with the other plane segment, because identical solution would be found when one computes the boundary intersection points for the boundaries of the second plane segment intersecting with the first plane segment.

Repeating the previous calculations for all four boundaries of a plane segment which intersect with another plane segment, one obtains the boundary intersection points. It is guaranteed that two such points exist for each pair of flat plane segments that intersect (Hughes et al., 2013). A straight line between these two boundary intersection points then defines the intersection line of the two plane segments.

In case of phantom intersections where two plane segments of the subtracted surfaces do not intersect each other but both do intersect the XY-plane, no valid solutions will be obtained from Equation 5-27. Therefore, these phantom intersections are easily identified and can be eliminated from the algorithm at this point. Additionally, it is possible that phantom intersections occur when two plane segments of the subtracted surfaces do intersect each other and the XY-plane, but do not do so in a single point that lies on all three plane segments. Such cases are difficult to detect without performing calculations more computationally expensive than those of Equation 5-27 (Hughes et al., 2013). Therefore, it is decided to leave these type of phantom intersections in the algorithm as their presence does not interfere with the remainder of the algorithm. The result will be that more points will be computed where two plane segments of the subtracted surfaces intersect.

At this point in the algorithm, the intersection line of the plane segments when the subtracted surfaces and XY-plane intersect each other is known. However, it is not guaranteed that this intersection line crosses the XY-plane. The following section discuss the approach to identify whether a solution exists and if so, to compute the solution. If no solution exists, the algorithm will continue to the part of the algorithm discussed in Section 5-2-4.

5-2-3 Deflection Computation

To determine the values of the left and right elevon deflections, one needs to find when the intersection line crosses the XY-plane. Therefore, one can make use of the same algorithm as was used in Section 5-2-2 to determine when a line segment crosses a plane segment. Furthermore, since the intersection line is a piece-wise linear line in three-dimensional space, one can easily

determine which line segment crosses the XY-plane, eliminating any intersection line segments that were the by-product of phantom intersections. Taking into account that the plane segment that the intersection line crosses is the XY-plane, for which $\mathbf{n} = (0, 0, 1)^T$, Equation 5-27 can be simplified to:

$$s_I = \frac{V_{0_z} - P_{0_z}}{P_{1_z} - P_{0_z}} \tag{5-28}$$

Inserting the result of Equation 5-28 into Equation 5-26 then gives the left and right elevon deflections for which the intersection line crosses the XY-plane and, by extension, where the subtracted surfaces of the roll and pitch moment coefficients cross each other and the XY-plane in a single point on all three surfaces. The obtained solution thus realizes both the commanded roll and pitch moment coefficients simultaneously.

At this point in the algorithm, one has obtained the required left and right elevon deflection for which both moment coefficients are realized simultaneously. It is, however, possible that the intersection line does not cross the XY-plane and is thus solely the by-product of phantom intersections and hence, no solutions exist. When this occurs, the algorithm will continue with the part of the algorithm discussed in the next section. It is noted that, if solutions were found, the algorithm outputs the found solution and continues with the rest of the simulation.

5-2-4 The Curious Case of No Solution

If at any point in the elevon assignment algorithm it became clear that no solutions for the left and right elevon deflections could be found for which both the roll and pitch moment coefficients were realized simultaneously, then an alternative solution must be found for the left and right elevon deflections which are then outputted to the simulation.

The subtracted surfaces may also be interpreted as the error surfaces of the commanded roll and pitch moment coefficients. The elevon deflection are then chosen such that the sum of the absolute values of both errors is minimized. This is expressed mathematically as:

$$[\delta_{el}, \delta_{er}] = \underset{\delta_{er}, \delta_{el}}{\arg\min} \left(\| C_{l_\delta} - C_{l_{cmd}} \| + \| C_{m_\delta} - C_{m_{cmd}} \| \right) \tag{5-29}$$

Choosing the elevon deflections in this way ensures that the resulting moment coefficients are both as close to the commanded moment coefficients as is allowed by the system constraints.

At this point in the algorithm, one has computed the left and right elevon deflections such that either both the commanded moment coefficients are realized, or both are realized as far as is allowed by the system constrained.

5-3 Rudder Assignment

The rudder deflections should be chosen such that commanded yaw moment coefficient is realized. However, from the numerical aerodynamic database, it is known that the elevon deflections also cause a yaw moment, which may either reduce the commanded yaw moment coefficient, or increase it. In either case, the rudder deflections are chosen such that the combined effect the elevons, which are given at this point in the algorithm, and the rudders realize the commanded yaw moment coefficient. The yaw moment coefficient contribution due to the rudders alone can be computed as:

$$C_{n_{cmd,corr}} = C_{n_{cmd}} - \Delta C_{n_{e,l}} - \Delta C_{n_{e,r}} \tag{5-30}$$

From the aerodynamic database the yaw moment coefficient as a function of the left and right rudder, for a given angle of attack, sideslip angle and Mach number, is known. The yaw moment coefficients can be represented as a surface in three-dimensional space, with two independent variables, the left and right rudder. However, to make efficient use of the rudders, one should choose rudder deflections such that one rudder has a zero deflection, depending on the sign of $C_{n_{cmd,corr}}$. A non-zero deflection for the left rudder should be chosen when $C_{n_{cmd,corr}}$ is negative, and a non-zero deflection for the right rudder when the it is positive. When $C_{n_{cmd,corr}}$ is zero, both rudders should have zero deflection.

Depending on the sign of $C_{n_{cmd,corr}}$, it is now known which rudder should have zero deflection. The remaining rudder deflection must be chosen such that it realises the corrected commanded yaw moment coefficient. From the aerodynamic database it is known that the yaw moment coefficient as function of a rudder deflection is a piece-wise linear line. Linear interpolation can then be used to compute the rudder deflection that realizes the corrected commanded yaw moment coefficient. It is noted that the interpretation of which rudder corresponds to the computed rudder deflection depends on the sign of the corrected commanded yaw moment coefficient.

5-4 Thruster Corrections

In the previous sections, it has been explained how to compute the elevon and rudder deflections, and the thruster values given a set of commanded moment-fractions at any given instance during the flight. Because of the limited range of the elevon and rudder deflections, it is possible that they saturate before the commanded moment coefficient can be realized. Additionally, it is possible that there is no solution to the combined elevon deflections, such that both the roll and pitch moment coefficients are realized simultaneously. In these cases, an error exists between the commanded and the actual, realized moment. If the thrusters are still active (and not saturated), however, one can use them to compensate for any error that may have occurred. It is noted that no compensation has to occur for the roll- and pitch-axes if the elevons are not active, and no compensation for the yaw-axis when the rudder is not active.

If the thruster of a specific axis is not active, no compensation occurs for that particular axis. If the thrusters are active however, one can first compute the corrective thruster coefficient using Equations 5-31 - 5-33. Comparison to Equations 5-4 - 5-6 shows that the thrusters only compute the corrective thrusters coefficients when both the elevons and roll and pitch thrusters are active for the roll- and pitch-axes, and when both the rudders and the yaw thruster are active for the yaw-axis.

$$C_{T_{x,corr}} = \begin{cases} 0 & \text{if } 0 \leq \bar{q} < 100 \\ C_{T_x} & \text{if } 100 \leq \bar{q} < 500 \\ 0 & \text{if } \bar{q} \geq 500 \end{cases} \tag{5-31}$$

$$C_{T_{y,corr}} = \begin{cases} 0 & \text{if } 0 \leq \bar{q} < 100 \\ C_{T_y} & \text{if } 100 \leq \bar{q} < 1000 \\ 0 & \text{if } \bar{q} \geq 1000 \end{cases} \tag{5-32}$$

$$C_{T_{z,corr}} = \begin{cases} 0 & \text{if } 0 \leq \bar{q} < 150 \text{ and } M \geq 1 \\ C_{T_z} & \text{if } \bar{q} \geq 150 \text{ and } M \geq 1 \\ 0 & \text{if } \bar{q} \geq 150 \text{ and } M < 1 \end{cases} \tag{5-33}$$

where the thruster coefficients are computed as follows:

$$C_{T_x} = C_{l_{cmd}} - \Delta C_{l_{e,l}} - \Delta C_{l_{e,r}} \tag{5-34}$$
$$C_{T_y} = C_{m_{cmd}} - \Delta C_{m_{e,l}} - \Delta C_{m_{e,r}} \tag{5-35}$$
$$C_{T_z} = C_{n_{cmd}} - \Delta C_{n_{r,l}} - \Delta C_{n_{e,l}} - \Delta C_{n_{e,r}} - \Delta C_{n_{r,r}} - \left[\Delta \left(\frac{\partial C_n}{\partial \beta} \right)_{r,l} + \Delta \left(\frac{\partial C_n}{\partial \beta} \right)_{r,r} \right] \beta \tag{5-36}$$

From the corrective thruster coefficients one can compute the new commanded thruster moment as the sum of the corrective thruster moments and the original commanded thruster moments computed in Equations 5-16, 5-17, and 5-18. The equations for the new commanded thruster moments are given by:

$$T_{x,cmd} = C_{T_{x,corr}} \bar{q} S_{ref} b_{ref} + T_x \tag{5-37}$$
$$T_{y,cmd} = C_{T_{y,corr}} \bar{q} S_{ref} c_{ref} + T_y \tag{5-38}$$
$$T_{z,cmd} = C_{T_{z,corr}} \bar{q} S_{ref} b_{ref} + T_z \tag{5-39}$$

It is noted that the methods outlined by Engelen (2000) and Durham (1993, 1994a,b, 2001) both include constraints on the rates of the control actuators, while the method presented in this chapter does not. This was not implemented because the goal of this chapter was to demonstrate an actuator allocation algorithm that did not require the use of a linearised aerodynamic model. However, it is possible to implement rate constraints on the actuators by limiting the search-space of the variations of the roll and pitch moment coefficients around the current deflections of the left and right elevons.

Chapter 6

Neural Network Controller

Neural networks are motivated by imitating the human reasoning processes where the relation between input and output is not explicitly given, but is "coded" in a network and its parameters. Additionally, it was shown by Cybenko (1989) that neural networks can approximate any function to any desired degree of accuracy, given enough hidden nodes. This section will describe the inner working of neural networks with Section 6-1 introducing the network structure and Section 6-2 will explain the chosen "learning" method. For further applications, the output of the neural network as a response to an input is required. Hence, Section 6-3 will present the algorithm used to compute the output of a general neural network. This chapter is concluded by Section 6-4 discussing the application of neural networks to an attitude controller of an entry vehicle. It is noted that verification of the neural net decoder will occur in Section 9-2-3.

6-1 Network Structure

Artificial Neural Networks (ANNs) consist of interconnected neurons (nodes) which are usually arranged in several *layers*. This arrangement is referred to as the *architecture* or *structure* of the neural network. Networks with several layers are called *multi-layered* networks, as opposed to *single-layered* networks that have only one layer. Within and among the layers, networks can be interconnected in two ways (Babuska, 2010):

- *Feedforward networks*: Neurons are arranged in several layers in which information flows in only one direction, as is shown in Figure 6-1.
- *Recurrent networks*: Neurons are arranged in one or more layers and feedback is introduced either internally in the neurons, to the other neurons in the same layer or to neurons in the preceding layer. An example is shown in Figure 6-2.

The learning process in biological neural networks is based on the change of the interconnection strength between neurons. The learning process for a neural network is some kind of optimization strategy whose aim is to minimize the difference between the desired and actual behaviour of the neural network by changing the weights between each node within the network, where the weights are the ANN version of the interconnection strength between biological neurons (Babuska, 2010).

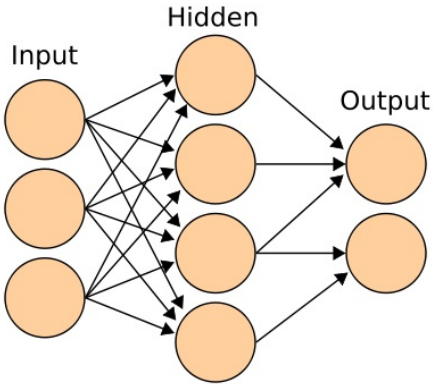


Figure 6-1: Feedforward neural network (Babuska, 2010).

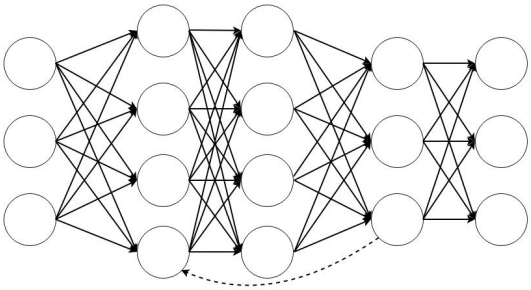


Figure 6-2: Neural network with a single recurrent connection. The dashed connection indicates a recurrent connection.

Each neuron in the hidden layer(s) outputs a so-called *activation value*. The activation value is computed using the weighted sum of the inputs. The process of the hidden layer neuron is graphically represented in Figure 6-3 and can be computed as follows:

$$net_j = \sum_{i=1}^{n} w_{ij} x_i \tag{6-1}$$

where net_j is the j^{th} net input, x_i is the i^{th} input, and w_{ij} is the weight of the connection from the i^{th} node to the j^{th} neuron. The activation function, φ, then maps net_j into a certain interval, such as $[0, 1]$ or $[-1, 1]$. This is denoted as:

$$o_j = \varphi(net_j) \tag{6-2}$$

where o_j is the neuron activation level. Several activation functions can be used, such as the threshold, sigmoidal, and tangent hyperbolic functions (Babuska, 2010). In this report, a logistic function is used to map the input to an interval of [-1,1], as is shown by:

$$\varphi(net_j) = \frac{2}{1 + \exp(-4.9 net_j)} - 1 \tag{6-3}$$

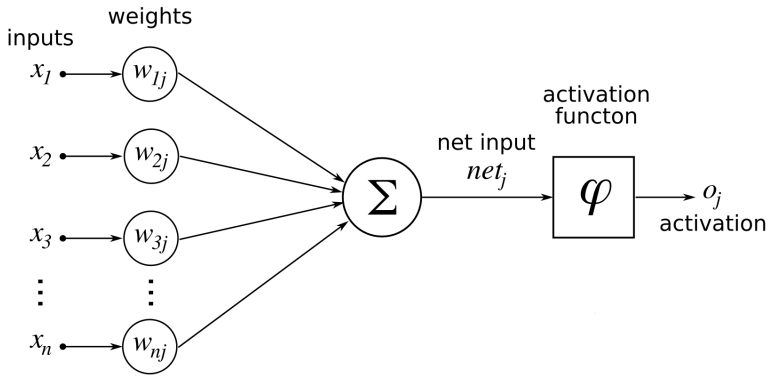


Figure 6-3: Artificial Neuron (Babuska, 2010).

6-2 Neuro-evolution

Stanley and Miikulainen (2002) showed that the NEAT methodology is a powerful method of optimizing the structure and connection weights of an ANN simultaneously and, hence, will be used to find an optimal NNC for the attitude control of an entry vehicle. The NEAT methodology consist of several stages. The first being the population generation, then the genetic encoding. The third stage is the tracking of the genes through historical markings, which allows mating and mutations, after which the individuals are put into species, a process known as speciation, to protect innovation by letting an individual mate only with another individual within its own species. The flow diagram of the NEAT algorithm is shown in Figure 6-4. Each of these stages will be discussed separately.

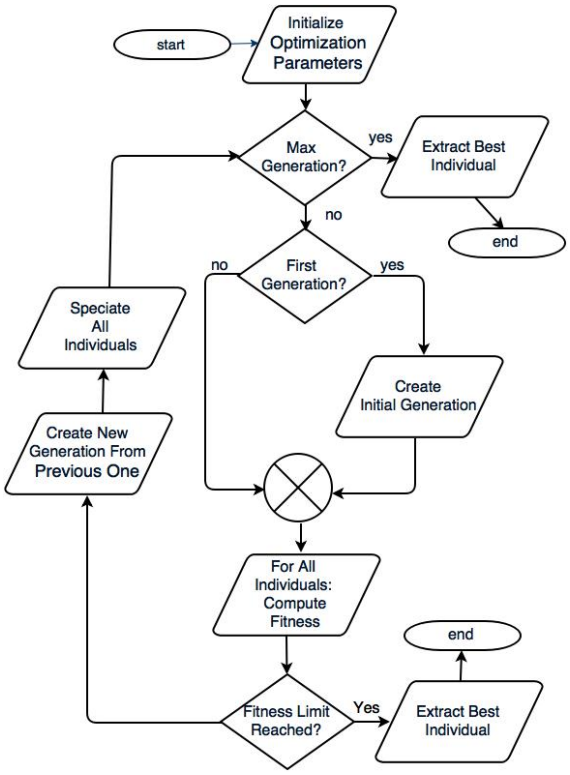


Figure 6-4: Flow diagram of the NEAT optimization algorithm, including stop criteria.

There are three structures that continuously flow through the algorithm. The first of these structures contains all individuals inside a generation, has information to what species they belong to, and keeps track of their fitness. The second structure is the *innovation record*, which keeps track of the innovations that have occurred during evolution and is continuously updated each generation. The last structure contains the optimization parameters, which determines the exact behaviour of the NEAT algorithm. It is noted that all information in this section is taken from the paper by Stanley and Miikulainen (2002) presenting the method, unless stated otherwise.

6-2-1 Population Generation

The first step of the NEAT optimization algorithm is to generate an initial population. This first generation is generated with minimal complexity, that is, without any hidden nodes and recurrent connections. This ensures that the optimal solution will have minimal complexity. Each individual will be initialized with a different set of randomly generated connection weights.

At the end of each cycle through the NEAT optimization algorithm, a new generation needs to be generated from the current generation. Several operations are performed to ensure that the best qualities of the previous generation are transferred to the new generation, as shown in the flow diagram of Figure 6-5. These operations act on each species that are present in the population. The first operation is to determine the number of offspring assigned to each species within the population, which is related to the total number of species, the population size, and the total fitness of the species. The second operation is known as *elitism*, which directly transfers the best performing individuals to the new generation. This ensures that the fitness cannot decrease. An exception is made when the species of an individual eligible for elitism is allotted only a single offspring. When this occurs, the eligible individual is not copied to the new generation, and the species dies off. Furthermore, such an occurrence is the only way for the maximum fitness of the population to decrease. The final operation kills off the worst performing individuals of each species, to ensure that their bad "traits" are removed from the gene-pool. It is noted that the individuals that were transferred from the previous generation to the new one by means of elitism, are part of the total number of offspring allotted to each species.

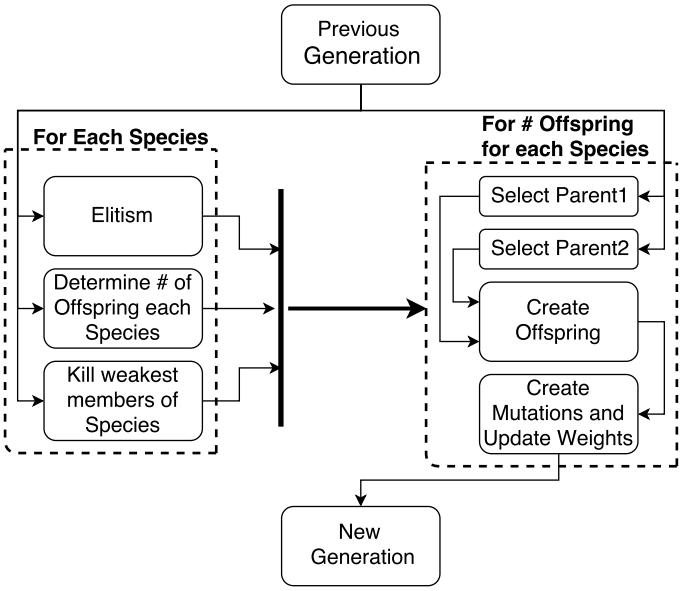


Figure 6-5: Flow diagram of the process that generates a new population from the previous one.

After the species operations have occurred, offspring are created for each species until all of them have a total number of individuals equal to their allotted number of offspring. This process starts by randomly selecting two parents from the same species found in the previous generation. The process of creating an offspring of two randomly selected individuals, which may have different architectures, is outlined in Section 6-2-3. Furthermore, mutations are applied to the newly created offspring, which allows for the creation of new architecture innovations. The types and process of mutations are discussed in Section 6-2-2. Additionally, for each offspring, a randomly selected subset of their connection weights are updated by a uniform random distribution centred about the current value of the connection weight. Once all species are assigned their allotted number of offspring, the new generation is fed to the NEAT optimization algorithm of Figure 6-4 and optimization process continues.

6-2-2 Genetic Encoding

The structure of the neural network is encoded into *genomes*, each of which has two arrays of information about the nodes and the connections of the neural network. The first array contains a

list of *node genes*. Each node gene contains information of the unique identifier of that node and what type of node it is. The second array contains a list of *connection genes*. Each connection gene specifies the *from-node*, the *to-node*, and the weight of the connection. Additionally, the array provides information on whether or not the connection gene is expressed (an enable bit), and an *innovation number*, which allows finding corresponding genes. Figure 6-6 shows how one can translate a genome to the physical structure of the neural net. It is noted that the connection and node genes are stored as matrices in numerical environments. Their definition, however, remains identical to the one described above. The genetic encoding scheme applied by the NEAT algorithm is designed such that identical genes of two different genomes can be easily lined up during mating.

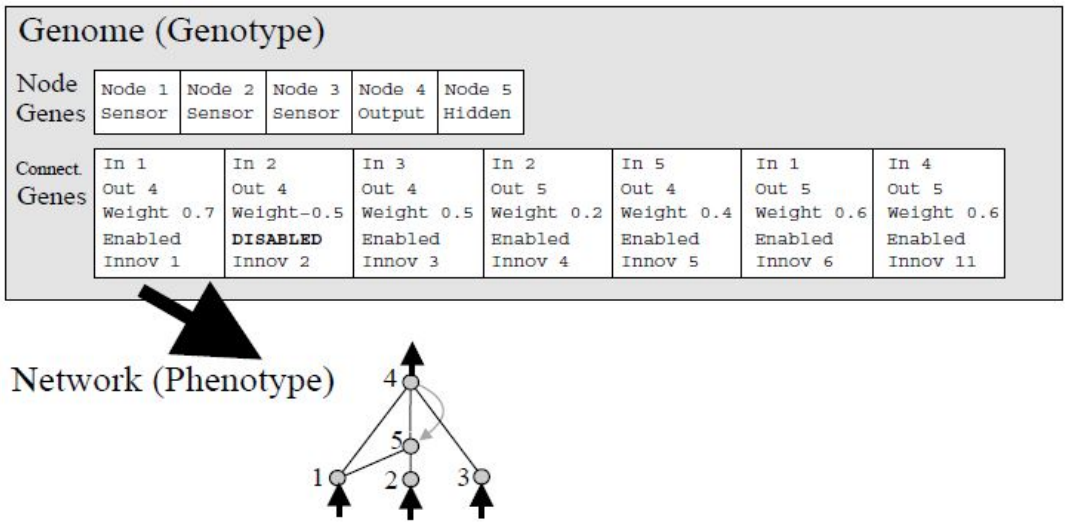


Figure 6-6: Genotype to phenotype mapping (Stanley and Miikulainen, 2002).

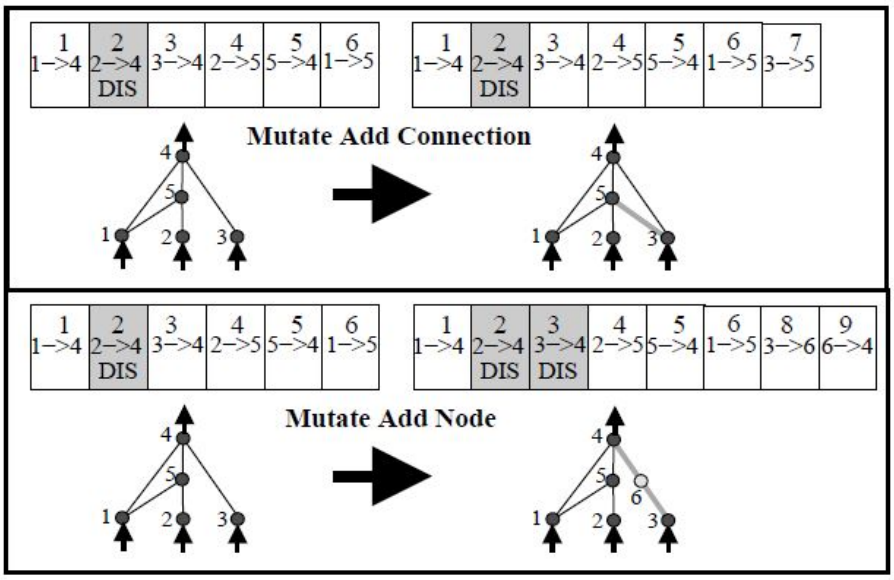


Figure 6-7: The two types of structural mutation in NEAT. Both types, adding a connection and adding a node, are illustrated with the connection genes of a networks shown above their phenotype. The top number in each genome is the *innovation number* of that gene. New genes are assigned new increasingly higher numbers. (Stanley and Miikulainen, 2002).

Mutations in NEAT can change both the connection weights and the network structure. Connection weights can mutate such that their parameter values either change or not, at the start of a new generation. Mutations in the structure of network can occur in two ways. Both mutations, however, expand the size of the genome by adding one or more genes. The first structural mutation is the *add connection* mutation, which adds a single new connections gene with a random weight, connecting two previously unconnected nodes. The second structural mutation is the *add node* mutation, which adds a node such that an existing connection is split and the newly added node is placed where the split connection used to be. Additionally, two new connections are added to the genome, connecting the new node with the nodes that were originally connected by the split connection. The new connections that connects *to* the new node receives a weight of one, and the connection connecting *from* the new node receives the same weight of the split connection. Figure 6-7 shows how the connection genes change due to the mutations.

6-2-3 Tracking Genes through Historical Markings

In the previous section, it was mentioned that mutations add one or more genes. This implies that the genomes will gradually get larger, which will result in genomes of varying sizes at any given generation, sometimes with different connections at the same point. To allow these differently sized genomes to cross over and mate, the innovation number is used to determine which genes are identical in any two genomes. If two genes have an identical innovation number, then both genes must represent the same structure, since both genes are derived from the same ancestral gene. Because of this, the innovation numbers are also called *historical markings* (Stanley and Miikulainen, 2002). Thus, all information that the NEAT algorithm needs to determine which genes are identical, is the innovation number.

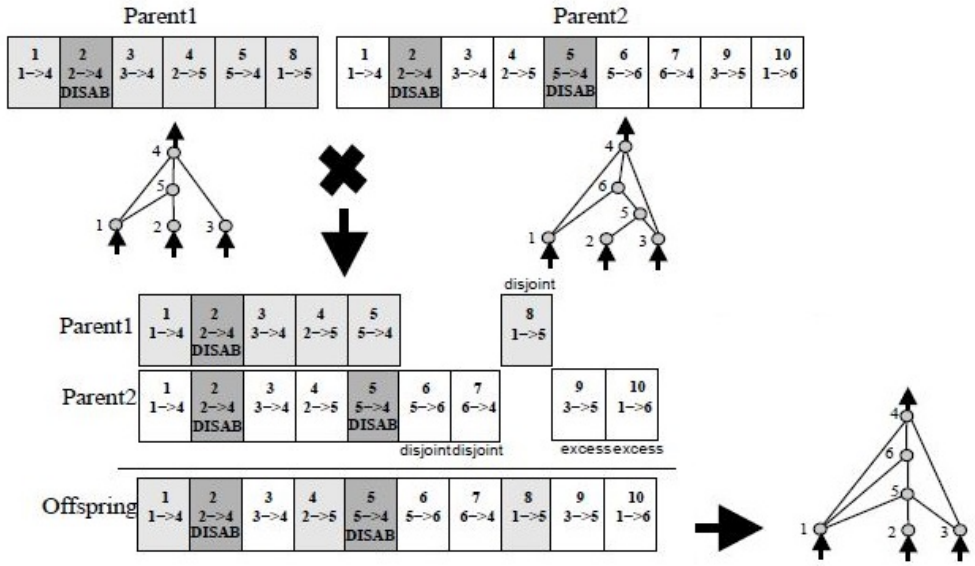


Figure 6-8: Graphical representation of matching up genomes for different network structures using the innovation number. Here, both parents have equal fitness (Stanley and Miikulainen, 2002).

When a structural mutation adds a new gene, a *global innovation number* is incremented and assigned as the innovation number of that gene. This implies that the innovation numbers also keeps track of the chronological appearance of every gene within the population. Furthermore, when two genomes mate, the genes of the offspring will have the same innovation numbers as the corresponding genes of its parents. This means that innovations numbers never change during

evolution. It is noted that, if a structural mutation produces a gene that already occurs within the population, then the innovation number of this new structural mutation will be set equal to the innovation number of this original gene.

When genes cross over, the genes of each genome is lined up with the corresponding gene of the other genome with the same innovation numbers; a process called *matching genes*. Genes that do not line up are *disjoint* or *excess*, depending on whether the innovation number is less than or higher than the highest innovation number of the other parent's genome. Physically, disjoint and excess genes of a parent represent structure that is not found in the other parent's genome. Matching genes are randomly chosen from either parent when creating the offspring, and all excess or disjoint genes are always included from the parent with the highest fitness. Figure 6-8 shows a graphical representation of the cross over process.

6-2-4 Protecting Innovation through Speciation

By adding new genes to the population by means of structural mutations, and allowing genomes representing different structures to produce offspring, a population of diverse structures can occur. However, the statistical odds that adding random nodes and connections decreasing the fitness of a network is high, hence recently changed genomes have a low probability of surviving more than one generation. Additionally, the innovations which the structural mutations represent may be of vital importance to solve the task in the long run. For these reasons the innovations are protected by a process known as *speciation* (Stanley and Miikulainen, 2002).

Speciating the population ensures that individuals compete primarily within their own "genetic neighbourhood" instead of with the complete population. This way, structural innovations are protected in their "neighbourhood" where they have time to optimize their structure through competition within their genetic neighbourhood. Thus, individuals with similar structures are put in the same species, with the number of disjoint and excess genes between pairs of genomes being a natural measure of the similarity and compatibility. Therefore, the compatibility distance δ of a pair of genomes can be expressed as a linear combination of the number of excess, E, and disjoint, D, genes, as well as the average weight difference of matching genes, $\bar{W}$, including disabled genes. This definition can be written in equation form as:

$$\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \bar{W} \tag{6-4}$$

where the coefficients c_1, c_2 and c_3 allow for adjustments to the relative importance of the three factors, and the factor N represents the number of genes in the larger genome, which normalizes for genome size.

The distance measure δ allows for speciating using a compatibility threshold δ_t.Every generation, each genome is placed into a species, and a record containing a list of species is maintained. A random genome inside the species from the previous generation is used to represent each species. If a genome G in the current generation is compatible with the representative genome of a species, then G is placed into that species. This implies that the different species will not overlap. If G is not compatible with any existing species, a new species is created with G as its representative genome (Stanley and Miikulainen, 2002).

The reproduction mechanics used is known as *explicit fitness sharing*, where individuals in the same species must share the fitness of their species. This can be seen as if there is only a limited amount of resources (fitness) available to a species, and thus the individuals within a species must share their resources. Hence, the number of individuals within a species cannot become large, even if the individual genomes perform well. In this manner, it is unlikely that one species will dominate the entire population, which is a requirement for speciated evolution to work. The adjusted fitness f'_i for individual i is calculated according to its distance δ from every other individual j in the population:

$$f_i' = \frac{f_i}{\sum_{j=1}^{n} \text{sh}(\delta(i,j))}, \quad \text{sh}(\delta) = \begin{cases} 0, & \text{if } \delta > \delta_t \\ 1, & \text{otherwise} \end{cases} \tag{6-5}$$

where f_i is the fitness of the ith genome. The sharing function sh is set to 0 when distance δ is above the threshold δ_t, otherwise it is set to 1. Hence, the denominator reduces to the number of individuals within the species of individual i.

The number of offspring assigned to each species is proportional to the sum of the adjusted fitness f_i' of its members. Reproduction occurs by first eliminating the weakest members from the population, a process known as *elitism*. The offspring of the remaining individuals in each species then replace the entire population. It is noted that, if the fitness of the entire population does not increase after more than 20 generations, only the offspring of two best performing species are used to replace the population. This approach allows for refocusing the search into the most promising areas.

6-3 Neural Net Decoder

The structure of the neural network is encoded in its genome. For further applications, however, the output of the neural network in response to an input is required. This section will present the steps involved in computing the output of a general neural network, involving multiple layers and (possibly) recurrent connections. The first part of the algorithm, as will be discussed in Section 6-3-1, identifies recurrent connections (if present), and adds this information to the connection genes. After the recurrent connections have been identified, each node can be assigned a layer number. This part of the algorithm will be presented in Section 6-3-2. Section 6-3-3 will then conclude this section by discussing the method used to compute the outputs of the neural network.

6-3-1 Recurrent Connection Identification

Recurrent connections within a neural network can be thought of as creating loops in the network. Hence, identifying which connections, if any, are recurrent connections, requires the identification of loops in the network. The structure of neural networks are equivalent to the mathematical objects known as *graphs*. Graph theory is a well studied discipline in mathematics and hence many theorems and algorithms exist concerning the various properties of graphs. A specific algorithm, known as Kruskal's algorithm, will be used to identify the *minimal-spanning-tree*, which is defined to be the graph with no loops (Kruskal, 1956).

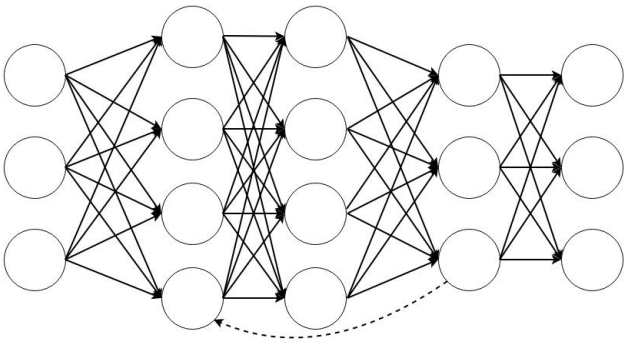


Figure 6-9: Neural network with a single recurrent connection. The dashed connection indicates a recurrent connection.

It is instructive to work trough an example while each component of the algorithm is discussed and explained. Figure 6-9 shows the neural network that will be used in this section as the example network. It is noted that it contains a single recurrent connection. Furthermore, the total number of connections is referred to as the *complexity* of the network.

The first step of the algorithm is to (temporarily) eliminate any nodes that either have no connection to or from another node. Additionally, all connections to and from these nodes will be removed. This is then repeated until this operation does not reduce the complexity of the network. The unconnected nodes are removed from any further analysis to reduce the changes of finding erroneous solutions. Figures 6-10 and 6-11 show the resulting networks after the unconnected nodes operations has been executed one and two time(s), respectively. It is noted that after two operations, no more unconnected nodes are present in the reduced network.

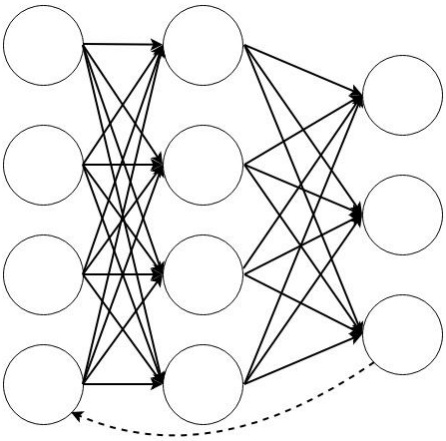


Figure 6-10: Remaining elements of the neural network after a single operation of removing unconnected nodes and connections.

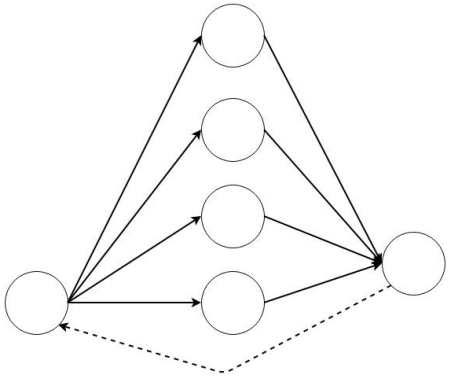


Figure 6-11: Remaining elements of the neural network after two operations of removing unconnected nodes and connections. No more unconnected nodes exist at this point.

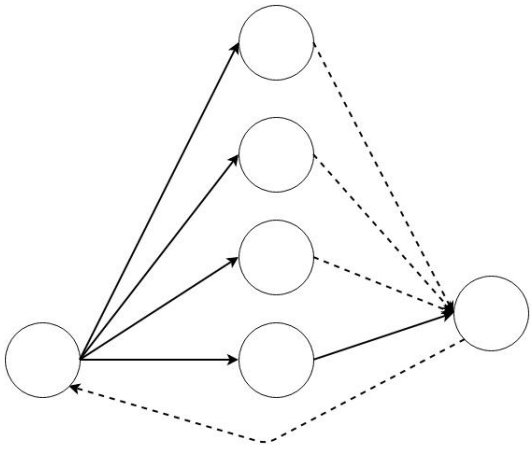


Figure 6-12: Neural network structure with the dashed lines indicating possible recurrent connections.

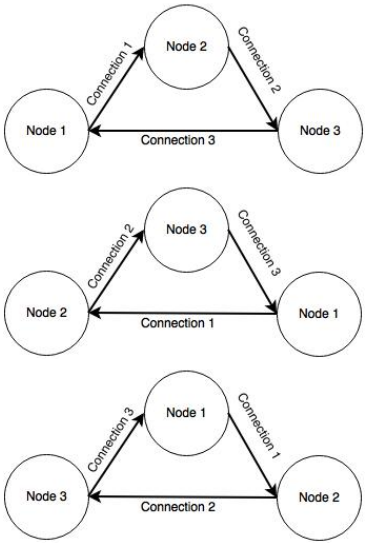


Figure 6-13: Three different orientations of a loop of the same network.

From Figure 6-11 it can be seen that there exist four loops within the network. Making use of Kruskal's algorithm will, however, only find possible candidates of which connections are to be removed to have a minimal-spanning-tree. It does this by starting a new graph with the first occurring connection, and at each step adding new connections to the graph only if this connection does not create a loop within the resulting graph (Kruskal, 1956). Hence, depending on where the algorithm starts, the minimal-spanning-tree may have selected multiple connections to be loop-forming connections. For example, as shown in Figure 6-12, it has selected four connections to be loop-forming connections.

It is important to realize, that at this point in the algorithm, all selected loop-forming connections are equally valid recurrent connections. The remainder of the problem lies, however, in the fact that not all of those connections need to be identified as recurrent connections. Additionally, one should not interpret connections moving from the right to the left as recurrent connections, since the orientation of the nodes are not relevant to the algorithm. This is illustrated in Figure 6-13, where three identical network loops are shown, but orientated differently. It implies that any of the three connections of Figure 6-13 can be seen as recurrent connections. Hence, of the selected loop-forming connections, one does not have to determine which is the "true" recurrent connection. What needs to be determined is the connection which, if identified as a recurrent connection, removes as many of the loops as possible. It is noted that this argument fails if the recurrent connection originates from the output, since the output nodes are supposed to be the last nodes in the network and thus their position cannot be shuffled around. Hence, connections originating from output nodes are always considered recurrent connections and eliminated from any further analysis in the algorithm.

The algorithm continues with selecting one of the loop-forming connections, temporarily removing it, and remove as many of the unconnected nodes and connections using the same method as was discussed above. If the complexity of the network is reduced to zero, then no further loops are present in the network and the selected loop-forming connection is identified as the recurrent connection. Likewise, if the complexity has a non-zero value, additional loops are present in the network. The algorithm repeats this process for all loop-making connections that were identified previously. The loop-making connection that gives the largest reduction in complexity is chosen as a recurrent connection. If after the first identification of a recurrent connection more loops still exist in the network, the algorithm is repeated until no more loops are present in the network.

It is noted that the recurrent connection that removes as many of the loops as possible should not be selected as the starting point of Kruskal's algorithm. Fortunately, in the numerical environment, the connection-genes-matrix stores the connections in order of occurrence in the neuro-evolution process. Because the feed-forward connections occur before any recurrent connection can develop/mutate, a feed-forward connection will be selected by Kruskal's algorithm, eliminating any chance that Kruskal's algorithm starts on the "true" recurrent connection. Furthermore, an identifier is added to each connection-gene to indicate whether a connection is a recurrent connection (a value of one), or not (a value of zero).

6-3-2 Layer Assignment

At this point in the algorithm, it is known which connections are identified as recurrent connections, it is possible to determine in which layer each node belongs. In this context, the input and bias nodes are defined to exist in the first layer, whereas the output nodes exist in the last layer. The nodes that only have connections to the input and bias nodes in the first layer, are defined to exist in the second layer. Nodes that have connections from the first and second layer, are said to exist in the third layer. In summary, a node is defined to exist in the i^{th} layer, if it only has connections originating from the previous layers.

It is noted that it is important in this part of the algorithm to ignore connections that have been identified as recurrent connections, since the loops they create sends this algorithm into an infinite loop. This can be visualized with the help of Figure 6-14 and explained by an example.

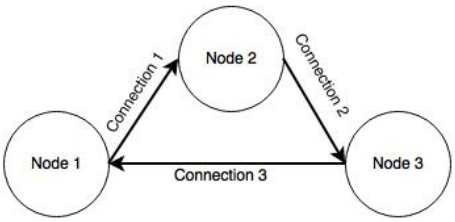


Figure 6-14: Example network where connection 3 has been identified as the recurrent connection.

Assume that Node 1 has connections only from the first layer and is connected from Node 3 by a recurrent connection. If the recurrent connection is not ignored, Node 1 cannot be defined to exist in the second layer, because it does not meet the requirement to exist in that layer. Furthermore, the algorithm determines that Node 2 should be placed in a layer after Node 1, and Node 3 should be placed in a layer after Node 2, but before Node 1, due to its connection to Node 1. This creates a paradox, and hence the algorithm descends into an infinite loop. However, if the recurrent connections is ignored, Node 1 can be positively identified to exist in the second layer, and Node 3 only has to be placed in a layer after Node 2, which is possible. Hence, the recurrent connections should be ignored by this part of the algorithm. It is noted that an identifier is added to each node-gene to indicate in which layer the node belongs.

6-3-3 Output Computation

Now that the recurrent connections have been identified and all nodes have been placed in a specific layer, it is relatively straightforward to compute the neural network's output. For each layer, the node inputs and outputs are sequentially computed in accordance with Equations 6-1 and 6-2, respectively, using the activation function of Equation 6-3. If a node is connected to a recurrent connection, the value for the node-output of the node from which the recurrent connection originates is taken of the previous time-step. In this manner, the neural network can form memory structures. It is noted that recurrent connections do necessitate an initial value for the node-outputs.

6-4 Application to Attitude Control

Sections 6-1 through 6-3 discussed the theory about the inner workings and the evolution of neural networks. However, one must relate this theory to the problem at hand; the attitude controller of an entry vehicle. It is known that, given sufficient time and information, a neural network can approximate any mathematical function (Cybenko, 1989). This means that the above theory can be applied relatively easily to the attitude controller of an entry vehicle, as the optimization will ensure that the neural network controls the attitude of the entry vehicle as specified by the user, given enough time for the controller to learn the task.

Furthermore, while the internal structure of the neural network is determined through the NEAT algorithm, the inputs and outputs of the network are determined beforehand. For the attitude control problem in this report, the inputs are the aerodynamic angles and the body-axis rotational rates describing the rotational state of the vehicle. The outputs of the neural network are the moment-fractions. Additionally, one can include integral terms of the aerodynamic angles as additional inputs. The software implementing the NEAT algorithm is obtained from the website of Stanley[1]. The flow diagram of the NEAT optimization algorithm shown in Figure 6-4 corresponds with the obtained software.

[1]K. Stanley, The NeuroEvolution of Augmenting Topologies (NEAT) Users Page, http://www.cs.ucf.edu/~kstanley/neat.html [Retrieved 19-10-2016], 2016

To apply the NEAT algorithm to the optimization of the attitude controller of an entry vehicle, it is only necessary to modify the number of inputs and outputs in a file describing the initial population settings, and to modify the module that computes the fitness of a specific neural net, which is computed in a number of steps, as shown in Figure 6-15. The first step is to identify the recurrent connections that may exist in a network. The method to do so is outlined in Section 6-3-1. After the recurrent connections have been identified, each node within the network is assigned a layer number, as discussed in Section 6-3-2. The neural network now contains additional information about the recurrent connections and layers in the network. The matrices describing the neural network are inputs to the simulation. The NNC module implemented in the simulator only performs the task of computing the output of the network in response to an input, which is described in Section 6-3-3, and is completely contained within a single function.

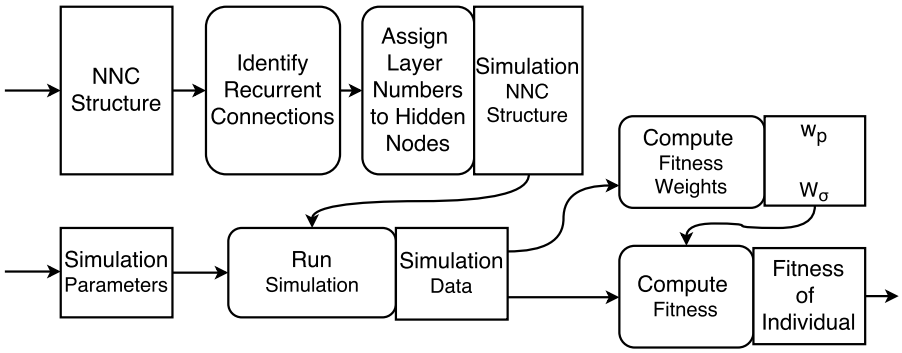


Figure 6-15: Flow diagram of the fitness computation during optimization of an NNC.

The fitness of the NNC is defined as the weighted sum of the integrated state deviation and control effort, which are defined similar to Equations 4-2 and 4-3. However, the state-errors are modified to include a weight, to ensure that specific values of the state vector are penalized. This is necessary to ensure that the controller will not exceed the vehicle constraints, as mentioned in Section 2-2-2. The weight function is defined to be:

$$w(x; lim_{low}, lim_{up}) = \begin{cases} 1 & \text{if } |x| < lim_{low} \\ \frac{lim_{up} - lim_{low}}{lim_{up} - |x|} & \text{if } lim_{low} \leq |x| < lim_{up} \\ \infty & \text{if } |x| \geq lim_{up} \end{cases} \tag{6-6}$$

where lim_{low} and lim_{up} represent the lower and upper limits in-between which the weight function is not unity. If the state has a value larger than the upper limit, the weight becomes infinite, having the physical meaning that the current controller does not satisfy the specified constraints. It is noted that the upper limit is also an asymptote. Including the weight function in the performance metrics of Equation 4-2, defines the performance metrics for each state variable:

$$\begin{aligned} \sum_p &= \int_0^t w_p \sqrt{p^2}\mathrm{d}t & \sum_q &= \int_0^t w_q \sqrt{q^2}\mathrm{d}t & \sum_r &= \int_0^t w_r \sqrt{r^2}\mathrm{d}t \\ \sum_{\alpha_{err}} &= \int_0^t w_\alpha \sqrt{(\alpha_c - \alpha_p)^2}\mathrm{d}t & \sum_{\beta_{err}} &= \int_0^t w_\beta \sqrt{(\beta_c - \beta_p)^2}\mathrm{d}t & \sum_{\sigma_{err}} &= \int_0^t w_\sigma \sqrt{(\sigma_c - \sigma_p)^2}\mathrm{d}t \end{aligned} \tag{6-7}$$

where the weights of the performance metrics are functions of the state-errors at each time-step. The lower and upper limits of the weight-functions are given in Table 6-1. It is chosen to set the limits for the weight-function of the angle of attack and bank angle to infinity, because they are

the commanded control variables obtained from the guidance system, and thus the performance metrics of the angle of attack and bank angle errors will be large compared to the remaining performance metrics of the state variables. The limits of the weight-function for the sideslip angle are based on the linearisation assumption of the aerodynamic model w.r.t. the sideslip angle. In this manner, the optimization process avoids solutions for which a large sideslip angle occurs and the assumption is violated. The limits of the weight-functions for the rotational rates are based on the rate-limit of the roll rate, as shown in Table 2-7. It is noted that, in practice, only stable solutions are found for which all rates stayed below ten degrees per second.

Table 6-1: Upper and lower limits for the weight functions of the state variables.

	w_p	w_q	w_r	w_α	w_β	w_σ
lim_{low} [deg]	10	10	10	∞	1	∞
lim_{up} [deg]	20	20	20	∞	5	∞

Additionally, performance metrics of the actuators can be defined, which encode the integrated control effort of the vehicle, and are given by Equation 4-3, restated here for convenience.

$$\sum_{\delta_e} = \int_0^t |\delta_e| \mathrm{d}t \qquad \sum_{\delta_a} = \int_0^t |\delta_a| \mathrm{d}t \qquad \sum_{\delta_r} = \int_0^t |\delta_r| \mathrm{d}t \tag{6-8}$$

$$\sum_{x} = \int_0^t |M_{T,x}| \mathrm{d}t \qquad \sum_{y} = \int_0^t |M_{T,y}| \mathrm{d}t \qquad \sum_{z} = \int_0^t |M_{T,z}| \mathrm{d}t \tag{6-9}$$

Defining the fitness of an individual genome to be inversely proportional to the weighted sum of the performances metrics of the state variables and the control effort, the fitness can then be explicitly written as:

$$f_i = \frac{10^3}{\mathbf{\Sigma}_{state}^T \mathbf{Q} \mathbf{\Sigma}_{state} + \mathbf{\Sigma}_{control}^T \mathbf{R} \mathbf{\Sigma}_{control} + 1} \tag{6-10}$$

$$\mathbf{\Sigma}_{state} = \left(\sum_p, \sum_q, \sum_r \sum_{\alpha_{err}}, \sum_{\beta_{err}}, \sum_{\sigma_{err}} \right)^T, \quad \mathbf{\Sigma}_{control} = \left(\sum_{\delta_e}, \sum_{\delta_a}, \sum_{\delta_r} \sum_x, \sum_y, \sum_z \right)^T \tag{6-11}$$

The matrices, $\mathbf{Q}$ and $\mathbf{R}$, can be freely specified in accordance with one's needs. A natural choice for these matrices would be analogous to Bryson's rule. The diagonal of the matrix would represent the maximum allowable integrated state deviation and control effort:

$$\mathbf{Q} = \mathrm{diag} \left\{ \frac{1}{\Delta \bar{p}^2}, \frac{1}{\Delta \bar{q}^2}, \frac{1}{\Delta \bar{r}^2}, \frac{1}{\Delta \bar{\alpha}^2}, \frac{1}{\Delta \bar{\beta}^2}, \frac{1}{\Delta \bar{\sigma}^2} \right\} \tag{6-12}$$

$$\mathbf{R} = \mathrm{diag} \left\{ \frac{1}{\bar{\delta}_e^2}, \frac{1}{\bar{\delta}_a^2}, \frac{1}{\bar{\delta}_r^2}, \frac{1}{\bar{M}_{T_x}^2}, \frac{1}{\bar{M}_{T_y}^2}, \frac{1}{\bar{M}_{T_z}^2} \right\} \tag{6-13}$$

with

$$\Delta\bar{p} = \int_0^t p_{max}\mathrm{d}t \qquad \Delta\bar{q} = \int_0^t q_{max}\mathrm{d}t \qquad \Delta\bar{r} = \int_0^t r_{max}\mathrm{d}t \tag{6-14}$$

$$\Delta\bar{\alpha} = \int_0^t \Delta\alpha_{max}\mathrm{d}t \qquad \Delta\bar{\beta} = \int_0^t \Delta\beta_{max}\mathrm{d}t \qquad \Delta\bar{\sigma} = \int_0^t \Delta\sigma_{max}\mathrm{d}t \tag{6-15}$$

$$\bar{\delta_a} = \int_0^t \delta_{a_{max}}\mathrm{d}t \qquad \bar{\delta_e} = \int_0^t \delta_{e_{max}}\mathrm{d}t \qquad \bar{\delta_r} = \int_0^t \delta_{r_{max}}\mathrm{d}t \tag{6-16}$$

$$\bar{M}_{T_x} = \int_0^t M_{T_{x_{max}}}\mathrm{d}t \qquad \bar{M}_{T_y} = \int_0^t M_{T_{y_{max}}}\mathrm{d}t \qquad \bar{M}_{T_z} = \int_0^t M_{T_{z_{max}}}\mathrm{d}t \tag{6-17}$$

where p_{max}, q_{max}, and r_{max} are the maximum allowable rotational rate deviations, and $\Delta\bar{p}$, $\Delta\bar{q}$, and $\Delta\bar{r}$ represent the integrated maximum allowable rotational rate deviations. The amplitudes are given in Table 6-2. It is noted that, in general, these parameters are all functions of time, however, they are known from the reference vehicle system constraints and control modes. Additionally, the fitness of a genome can only be computed once a simulation has been done, as it requires the evolution of the state and control variables as a function of time.

Table 6-2: Upper and lower limits for the weight functions of the state variables.

State Amplitudes	Values	Control Amplitudes	Values
p_{max}	5 [deg]	$\delta_{a_{max}}$	40 [deg]
q_{max}	5 [deg]	$\delta_{e_{max}}$	40 [deg]
r_{max}	5 [deg]	$\delta_{r_{max}}$	40 [deg]
$\Delta\alpha_{max}$	2 [deg]	$M_{T_{x_{max}}}$	1,600 [Nm]
$\Delta\beta_{max}$	1 [deg]	$M_{T_{y_{max}}}$	10,400 [Nm]
$\Delta\sigma_{max}$	5 [deg]	$M_{T_{z_{max}}}$	7,600 [Nm]

To simplify the optimization process of the NNC, it is separated into the sequential optimization of the controllers for the individual pitch-, roll-, and yaw-axes. The fitness of each of these controllers is computed as discussed above. First, the controller for the pitch-axis is optimized such that a one degree disturbance in the angle of attack is eliminated with the highest fitness for the controller. The pitch-axis controller takes the pitch rate and angle of attack as inputs and outputs the moment-fraction about the pitch-axis. During the optimization of the pitch-controller, the roll- and yaw-controllers are the controllers of the MFC performing the same task. After an optimal controller for the pitch-axis was found, the controller for the yaw-axis is optimized such that a one degree disturbance in the sideslip angle is eliminated with the highest fitness for the controller. The yaw-axis controller takes the roll and yaw rate, and the sideslip and bank angles as inputs, and outputs the moment-fraction about the yaw-axis. During the optimization of the controller for the yaw-axis, the previously found optimal NNC for the pitch-axis, and the roll-axis controller of the MFC will be used. Lastly, the NNC for the roll-axis is optimized such that it could perform one of the bank-reversals of Figure 2-3, with the other two controllers being the previously found NNCs for the pitch- and yaw-axes. The roll-axis controller takes the same inputs as the yaw-axis controller, but outputs the moment-fraction about the roll-axis. An identical optimization procedure will be followed for the optimization of the Neural Network Integral Controller (NNIC), with the integral terms as additional inputs.

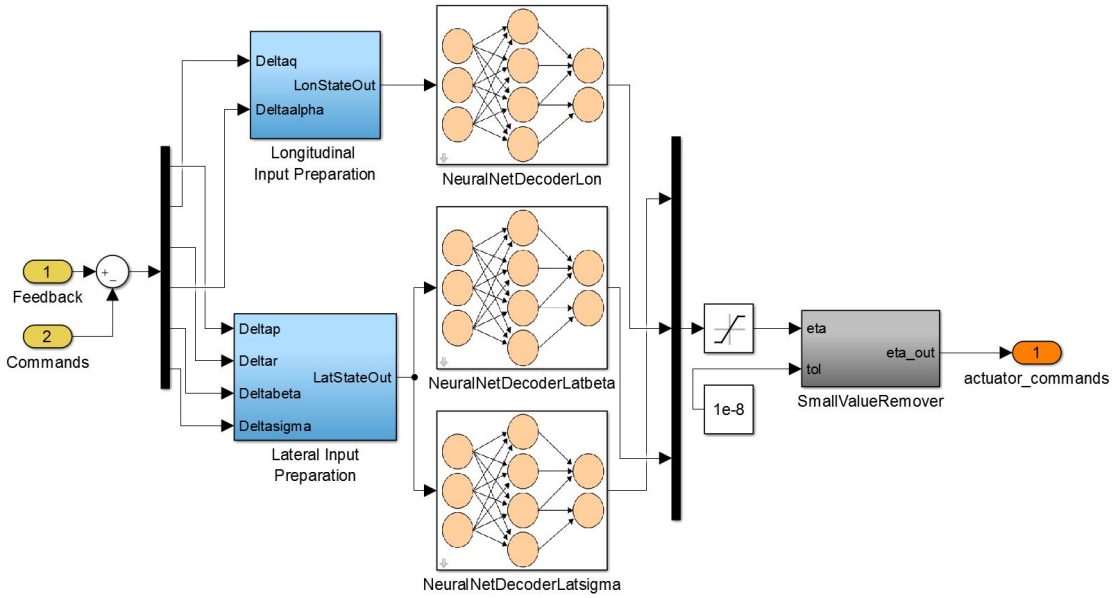


Figure 6-16: Software architecture of a general NNC.

The software architecture of the NNC and the NNIC are shown in Figure 6-16. For the NNC, the inputs are the errors in the rotational state of the vehicle, and each neural net decoder directly outputs a moment-fraction corresponding to its specific axis. For the NNIC, the inputs additionally include the integral terms. A functionality is added to remove moment-fractions smaller than 10^{-8}, such that the control system is not sensitive to noise or numerical errors. It is noted that, it is possible to construct a single neural network from the three separate ones for each axis. In the Software Architecture Diagram (SAD) shown in Figure 6-16, the three controllers are left separate, to indicate that they are trained individually. Furthermore, the numerical values of the integral terms are computed using a discrete integral, with a sample frequency of 20 Hz and a default gain of one.

The NNC obtained from the optimization will be optimized for a single point along the reference trajectory. In practice, the obtained controller will perform as desired around a neighbourhood of points around the optimized point. Finding an optimal NNC for multiple points along the reference trajectory and scheduling the network weights as a function of dynamic pressure will give an optimal NNC along the entire nominal trajectory. Alternatively, one could allow the dynamic pressure to be an additional input to the network, and let the optimization algorithm determine how to process this additional information. However, part of the the goal of this report was to show if neural networks can control the attitude of a reference vehicle at given points along the reference trajectory, and hence looking at the performance along the entire reference trajectory is beyond the scope of this report.

Chapter 7

Fuzzy Logic Control

Controllers such as fuzzy logic and neuro-fuzzy controllers apply fuzzy logic within their system. Similar to neural networks, fuzzy systems are universal approximators and are able to approximate any function to any degree of accuracy (Wang, 1992). This section will present and discuss the various components that enable fuzzy logic to be used to design an attitude controller. Section 7-1 will present the concept of membership functions within fuzzy logic, and Section 7-2 will discuss the concept of a rule-base and model dependent inference. It is noted that all information in these sections is taken from Babuska (2010), unless stated otherwise. Section 7-3 will conclude this chapter by discussing the practical consideration of applying a FLC to the attitude control of an entry vehicle.

7-1 Membership Functions

In ordinary set theory, an element either fully belongs to a set or is fully excluded from it. In mathematical notation, this can be denoted as:

$$\mu_A(x) = \begin{cases} 1 & \text{iff } x \in A, \\ 0 & \text{iff } x \notin A. \end{cases} \tag{7-1}$$

where $\mu_A(x)$ is the membership of x of a classical set A, which is a subset of the *universe* X. μ_A is known as a *membership function*. While this strict definition is useful in pure mathematics and many sciences, it is not as useful in many real-life and engineering problems, where it is not necessarily clear whether an element belongs to a set or not. A clear example is the question of whether something is expensive. For a poor person something might be expensive whereas for a rich person the same object might be cheap. This is where fuzzy sets come in: sets of which the membership has grades in the unit interval $[0, 1]$, instead of being either zero or one.

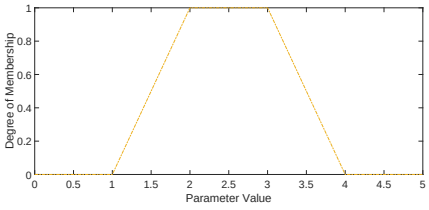
A fuzzy set is a set with graded membership in the real interval: $\mu_A(x) \in [0, 1]$. Hence, an element can belong to a fuzzy set with a certain degree. As such, fuzzy sets can be used for mathematical representations of vague concepts, such as *low temperature* or *expensive car*. If the value of the membership function, called the *degree of membership*, equals one, then the element belongs completely to the fuzzy set. If it equals zero, the element does not belong to the fuzzy set. If the membership is between zero and one, the element is a partial member of the fuzzy set. This can be put in mathematical term as:

$$\mu_A(x) = \begin{cases} 1 & \text{if } x \text{ is a full member of A,} \\ \in [0,1] & \text{if } x \text{ is a partial member of A,} \\ 0 & \text{if } x \text{ is not a member of A.} \end{cases} \tag{7-2}$$

There are several ways to define a fuzzy set. Through an analytic description of its membership function is the most common and most intuitive method and is therefore the only method that will be explained in what follows (Babuska, 2010). These analytical definitions can be thought of to describe the shape of the membership functions, and are named after the shape they describe.

- *Trapezoidal* membership function:

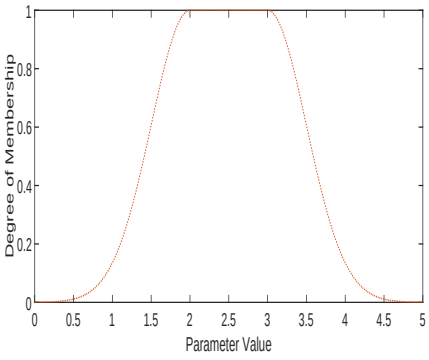
$$\mu(x; a, b, c, d) = \max\left(0, \min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right)\right)$$



  where a, b, c and d are the coordinates of the trapezoid's apexes. When $b = c$, a *triangular* membership function is obtained. The trapezoidal membership function is shown in the figure to the right.

- *Piece-wise exponential* membership function:

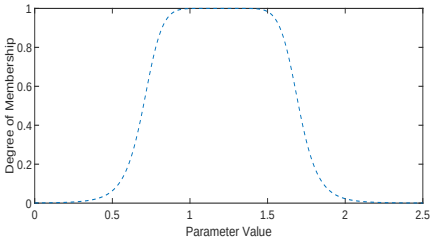
$$\mu(x; c_l, c_r, w_l, w_r) = \begin{cases} \exp\left(-\left(\frac{x-c_l}{2w_l}\right)^2\right) & \text{if } x \leq c_l, \\ \exp\left(-\left(\frac{x-c_r}{2w_r}\right)^2\right) & \text{if } x \geq c_r, \\ 1 & \text{otherwise.} \end{cases}$$



  where c_l and c_r are the left and right shoulder, respectively, and w_l, w_r are the left and right width, respectively. For $c_l = c_r$ and $w_l = w_r$ the Gaussian membership function is obtained. The piece-wise exponential membership function is shown in the figure to the right.

- *Bell-shaped* membership function:

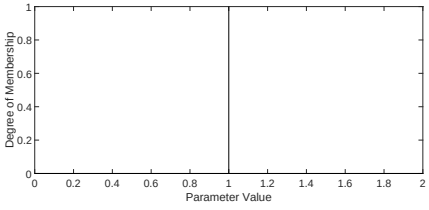
$$\mu(x; a, b, c) = \frac{1}{1 + \left\|\frac{x-c}{a}\right\|^{2b}}$$



  where the parameters a and b are usually positive and the parameter c locates the centre of the curve. The bell-shaped membership function is shown in the figure to the right.

- *Singleton* membership function

$$\mu(x) = \begin{cases} 1 & \text{if } x = x_0, \\ 0 & \text{otherwise.} \end{cases}$$



  The singleton membership function is the fuzzy set representation of a number and is quite similar to the delta-function, and is shown in the figure to the right.

The shape of a membership function depends on the notion the set is intended to describe and on the particular application involved. For example, when describing the notion of "is approximately", one would choose to use a membership function centred around the value to be approximated. It is noted that the membership functions mentioned above are the ones most commonly used for control applications (Babuska, 2010).

7-2 Rule Base and Inference

In fuzzy logic, the behaviour of the system is encoded as a collection of if-then *rules* with fuzzy predicates in the following general form:

If *antecedent proposition* **then** *consequent proposition.*

Fuzzy propositions are statements like "x is big", where "big" is a *linguistic label*, defined by a fuzzy set on the universe of the variable x. The antecedent proposition is always a fuzzy proposition of the type "x is A", where x is a linguistic variable and A is linguistic label. Depending on the particular structure of the consequent proposition, three main types of models exist (Babuska, 2010):

- *Linguistic fuzzy model*, where both the antecedent and consequent are fuzzy propositions.
- *Fuzzy relational model*, which can be regarded as a generalization of the linguistic model, allowing one particular antecedent proposition to be associated with several different consequent propositions via a fuzzy relation.
- *Takagi-Sugeno (TS) fuzzy model*, where the consequent is a crisp function of the antecedent variables rather than a fuzzy proposition.

The most commonly used model, is a special case of the linguistic fuzzy model, known as the Mamdani model. For the Mamdani model, the rule base is usually written as:

$$\mathcal{R}_i\text{: } \textbf{If } x_1 \text{ is } A_{i1} \textbf{ and } x_2 \text{ is } A_{i2} \textbf{ and } \dots \textbf{and } x_p \text{ is } A_{ip}, \textbf{ then } \begin{cases} y_1 \text{ is } B_{i1}, \\ y_2 \text{ is } B_{i2}, \\ \dots, \\ y_r \text{ is } B_{ir}. \end{cases}$$

where p is the number of of input variables and r the number of output variables. It is noted that the output of the Mamdani model is a fuzzy set. The algorithm used to determine this output fuzzy set is represented graphically in Figure 7-1. Before continuing it is useful to present the different operators one can apply to a fuzzy set or a pair of fuzzy sets. For the Mamdani model, the **and**-operator is the minimum operator, and the **or**-operator is the maximum operator. It is noted that the **and**- and **or**-operators, in equation form, are often denoted as (Babuska, 2010):

$$\mu_C(x) = \mu_A(x) \wedge \mu_B(x) \tag{7-3}$$
$$\mu_C(x) = \mu_A(x) \vee \mu_B(x) \tag{7-4}$$

It is noted that the **and**-operator is used to determine the degree of fulfilment of each rule, and the **or**-operator to aggregate the individual rules to obtain a single fuzzy set as output for each output variable (Babuska, 2010).

From Figure 7-1 one can see a Mamdani model with two inputs, two rules, and a single output. In this example both inputs are crisp (numerical) values, however, the same algorithm is used when the inputs are fuzzy sets. First, the degrees of fulfilment, also known as “rule strengths”, are determined by using the **and**-operator. Note that the degree of fulfilment is always a numerical value for the Mamdani algorithm. Second, the **and**-operator is used to determine the output fuzzy set of each rule from the degree of fulfilment and consequent fuzzy set of that rule. Lastly, the **or**-operator is used to compute the aggregated output fuzzy set of the rule base from the individual

output fuzzy sets of each rule. For the Mamdani model, the **or**-operator is the *max*-operator by definition (Babuska, 2010). Mathematically, the algorithm can be written as:

Algorithm 1 Mamdani inference (Babuska, 2010)

1: Compute the degree of fulfilment for each rule by: $\beta_i = \max[\mu_{A'}(\mathbf{x}) \wedge \mu_{A_i}(\mathbf{x})]$, $1 \leq i \leq K$. Note that for a singleton set the equation for β_i simplifies to $\beta_i = \mu_{A_i}(\mathbf{x}_0)$.
2: Derive the output fuzzy sets B'_i: $\mu_{B'_i}(\mathbf{y}) = \beta_i \wedge \mu_{B_i}(\mathbf{y})$, $1 \leq i \leq K$.
3: Aggregate the output fuzzy sets B'_i: $\mu_{B'}(\mathbf{y}) = \max \mu_{B'_i}(\mathbf{y})$.

In Algorithm 1, $\mu_{AA'}(\mathbf{x})$ represents the fuzzy input, $\mu_{A_i}(\mathbf{x})$ represents the fuzzy antecedent, β_i the degree of fulfilment of rule i, $\mu_{B_i}(\mathbf{y})$ the consequent fuzzy set, $\mu_{B'_i}(\mathbf{y})$ the output for each rule, and $\mu_{B'}(\mathbf{y})$ the aggregated fuzzy output set.

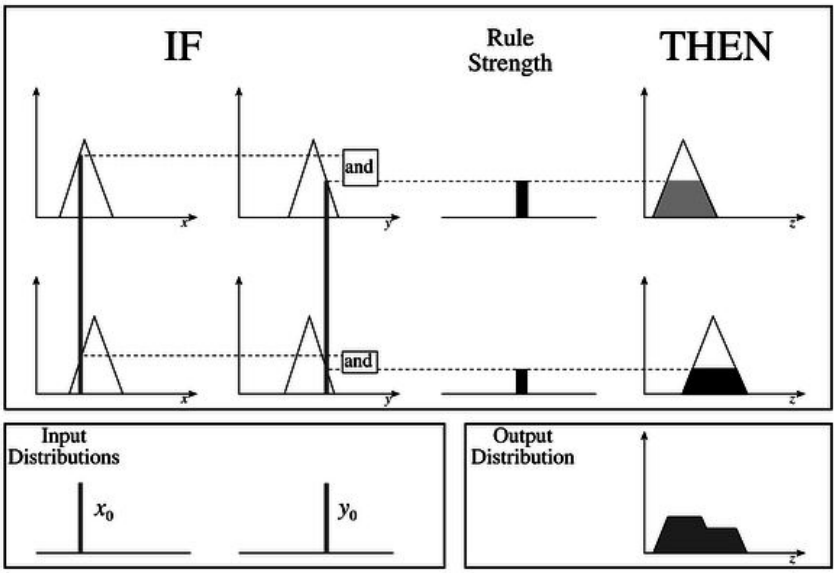


Figure 7-1: Graphical representation of fuzzy inference for the Mamdani model. In this example, the **and**-operator is given to be the *min*-operator (Babuska, 2010).

As mentioned, the result of the fuzzy inference for the Mamdani model is a output fuzzy set B'. If a numerical output value is required, the output fuzzy set must be defuzzified. Defuzzification is a transformation that replaces a fuzzy set by a single numerical value representative of that set (Babuska, 2010). The most commonly used method is the Centre of Gravity (CoG) method. The CoG method calculates numerically the y-coordinate of the centre of gravity of the fuzzy set B':

$$y = \text{cog}(B') = \frac{\sum_{j=1}^{F} \mu_{B'}(y_j) y_j}{\sum_{j=1}^{F} \mu_{B'}(y_j)} \tag{7-5}$$

where F is the number of elements y_j in its universe. The COG method is used with the Mamdani inference, as it provides interpolation between the consequents, in proportion to the height of the individual consequent sets. This is necessary, since the Mamdani inference method does not interpolate by itself (Babuska, 2010).

7-3 Application to Attitude Control

In the previous sections, the theory behind the fuzzy logic controller was presented and discussed. The theoretical story was, however, told for a general controller. This section will present how the theoretical basis of fuzzy logic can be applied to construct an attitude controller for an entry vehicle. It should be mentioned that the number of rules an FLC can have is the total product of the number of membership functions for each input. For example, if each input to the controller has nine membership functions, this means that the lateral controller will have $9^4 = 6,561$ rules, or $9^6 = 531,441$ rules for the integral controllers, for each output. This means that it is practically impossible to construct a rulebase for each output that depends in a fully non-linear way on the input variables, in a single fuzzy rulebase. To combat this problem, it is decided to construct a fuzzy rulebase analogous to an LQR. However, instead of the gains being scheduled only as a function of dynamic pressure, the gains are now also scheduled as a function of the state variable corresponding to that gain parameter. For example, K_{xb}, and K_{zs} are now also scheduled as a function of sideslip angle and bank angle, respectively. This type of fuzzy logic control is known as *supervisory control* (Babuska, 2010). An added benefit of this approach is that the designer can create physical graphs that allows one to obtain insight into the behaviour of the system, even when the outputs are also scheduled as a function of dynamic pressure. It is noted that the interpretation of the membership functions is the most intuitive when triangular membership functions are used, and hence it chosen to use this type of membership functions for the FLC in this report (Babuska, 2010). However, the methodologies discussed in this section can be generalized to the different types of membership functions mentioned in Section 7-1.

Membership Function Construction

To construct the membership functions for a variable in a consistent and systematic way means one must encode the parameters of the membership functions as a set of numbers. For the triangular membership functions, a logical choice for these set of numbers would be the location of its peaks. If one identifies the peak of triangular membership function with the exact location of where another membership function becomes zero, and another stops to be zero, then the entire set of membership functions is fully determined by set of numbers R_i, representing its peaks, as is shown in Figure 7-2. Additionally, one is often aware of the operational range of the variables, and hence one can determine the *fixed* values for R_{min}, and R_{max}. This implies that the optimization algorithm only has to look for values between the two fixed values, reducing its search space.

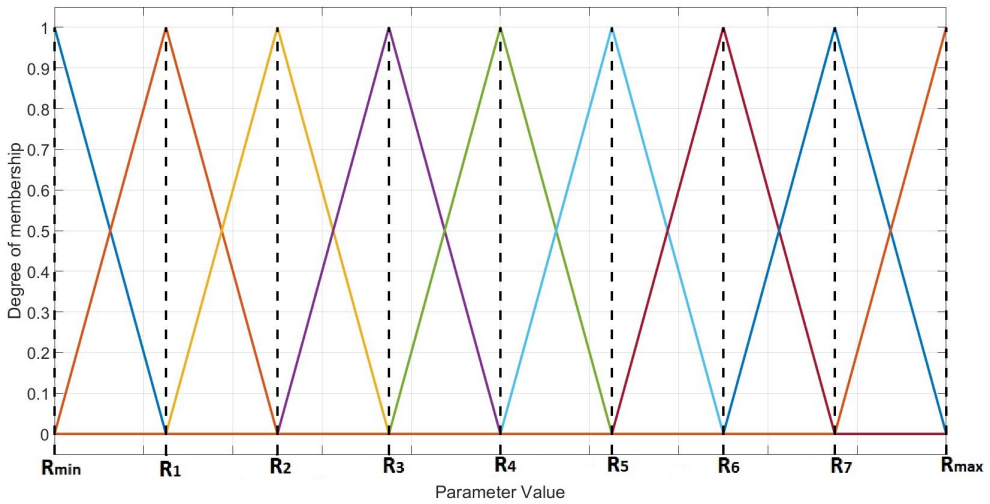


Figure 7-2: Description of the triangular membership functions of a variable in terms of its peaks locations R_i.

The sets of numbers defining the set of membership functions for a variable can then be concatenated and be seen as the vector that needs to be optimized for optimal performance of the control system. Furthermore, the values of each peak are not independent of the remaining values, that is, the values are constrained. For example, the value for R_i should be higher than the value for R_{i-1}, but lower than the value for R_{i+1}. Combining these conditions gives Equation 7-6.

$$\begin{bmatrix} -1 & 0 & 0 & \dots & 0 & 0 \\ 1 & -1 & 0 & \dots & 0 & 0 \\ 0 & 1 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & -1 \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix} \begin{bmatrix} R_1 \\ R_2 \\ R_3 \\ \vdots \\ R_{N-1} \\ R_N \end{bmatrix} \leq \begin{bmatrix} -R_{min} \\ 0 \\ 0 \\ \vdots \\ 0 \\ R_{max} \end{bmatrix}$$
$$\mathbf{A}_{mf}\mathbf{x}_{mf} \leq \mathbf{b}_{mf} \tag{7-6}$$

For clarity, it is mentioned that all the diagonal values of $\mathbf{A}_{mf}$ are equal to -1, and all the values of the sub-diagonal below the diagonal are equal to 1. Furthermore, $\mathbf{A}_{mf} \in \mathbb{R}^{(N+1)\times N}$, where N is the number of parameters, which is two less than the number of membership functions. It is noted that Equation 7-6 is valid for a single variable. To solve for multiple variables, one can apply:

$$\begin{bmatrix} \mathbf{A}_{mf,1} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{mf,2} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{A}_{mf,p} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{mf,1} \\ \mathbf{x}_{mf,2} \\ \vdots \\ \mathbf{x}_{mf,p} \end{bmatrix} \leq \begin{bmatrix} \mathbf{b}_{mf,1} \\ \mathbf{b}_{mf,2} \\ \vdots \\ \mathbf{b}_{mf,p} \end{bmatrix}$$
$$\mathbf{A}_{flc}\mathbf{x}_{flc} \leq \mathbf{b}_{flc} \tag{7-7}$$

where p is the number of variables of the fuzzy logic controller (inputs and outputs).

Optimization and Initial Conditions

To find the optimal structure of the membership functions, the set of numbers representing these membership functions need to be optimized. This is done with a pattern-search algorithm to search the grid of possible solutions. A genetic algorithm can be used, however, due to large number of parameters that need to be optimized, it was found to have trouble finding an optimal solution. In fact, the description of the genetic algorithm function of MATLAB advised to use the pattern-search algorithm instead, if there are no integer constraints (which there are not). The flow diagram describing the optimization process is shown in Figure 7-3. The pattern-search algorithm is a build-in function of MATLAB, and requires as inputs a function describing the computation of fitness, and a function describing any constraints. In the case of linear constraints, such as those of Equation 7-7, the pattern-search function requires the input of the $\mathbf{A}_{flc}$ matrix and the $\mathbf{b}_{flc}$ vector. The iteration procedure occurs internally, and the details of it are not considered here, that is, the function is seen as a black-box.

The initial FLC is generated such that it approximates the tuned MFC. The range of the universe of a variable was selected differently for the input and outputs. For the inputs, the range was selected such that it corresponded to the range of values encountered during a nominal bank-reversal. For the outputs, the range of the gains was selected to be $\pm 10\%$ of the gains as were computed for the MFC. The membership functions of the initial FLC are distributed evenly over the universe of the appropriate variable. Additionally, it was found experimentally that nine membership functions for all variables provided a good balance between optimization behaviour

and control performance. The range parameters are summarized in Tables 7-1 and 7-2 for the longitudinal and lateral FLCs, respectively. A similar procedure is followed for the initial Fuzzy Logic Integral Controller (FLIC). It is noted that the membership functions and rulebases can be defined and constructed using build-in commands within MATLAB's fuzzy logic toolbox, and are stored as a text file (a so-called .fis file).

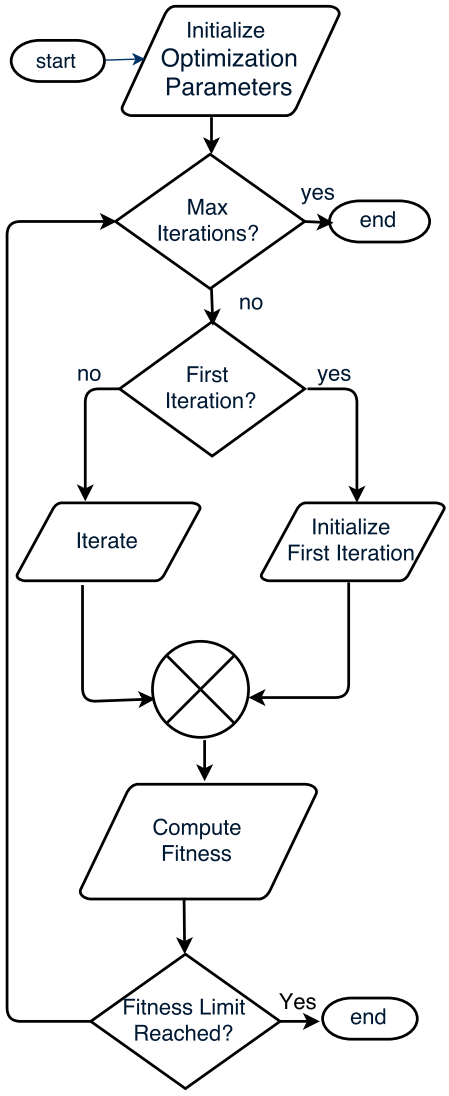


Figure 7-3: Flow diagram of the pattern-search optimization algorithm, including stop criteria.

Table 7-1: Overview of initial values for the ranges of the variables for the longitudinal FLCs.

Parameters	Range without integral terms	Range with integral terms
Δq_{max} [deg/s]	[0,1]	[0,1]
$\Delta \alpha_{max}$ [deg]	[0,1]	[0,1]
$I_{\alpha,max}$ [deg·s]	-	[0,1]
K_{yq} [s/deg]	[35,45]	[50,60]
K_{ya} [1/deg]	[35,45]	[90,100]
K_{yia} [1/(deg·s)]	-	[25,30]

Table 7-2: Overview of the initial values for the ranges of the variables for the lateral FLCs.

Parameter	Range without integral terms	Range with integral terms
Δp_{max} [deg/s]	[0,10]	[0,5]
Δr_{max} [deg/s]	[0,5]	[0,5]
$\Delta \beta_{max}$ [deg]	[0,1]	[0,1]
$\Delta \sigma_{max}$ [deg]	[0,20]	[0,6]
$I_{\beta,max}$ [deg·s]	-	[0,1]
$I_{\sigma,max}$ [deg·s]	-	[0,5]
K_{xp} [s/deg]	[6,8]	[6,8]
K_{xr} [s/deg]	[-5,-3]	[-18,-13]
K_{xb} [1/deg]	[20,22]	[95,105]
K_{xs} [1/deg]	[-6,-4]	[-2,-1]
K_{xib} [1/(deg·s)]	-	[135,145]
K_{xis} [1/(deg·s)]	-	[-0.6,-0.2]
K_{zp} [s/deg]	[-0.7,-0.2]	[-1.6,-1]
K_{zr} [s/deg]	[10,20]	[15,22]
K_{zb} [1/deg]	[-35,-20]	[-60,-50]
K_{zs} [1/deg]	[-4,-1]	[-6,-2]
K_{zib} [1/(deg·s)]	[-65,-55]	[-65,-55]
K_{zis} [1/(deg·s)]	[-0.6,-0.4]	[-0.6,-0.4]

The fitness of a particular choice of parameters of the membership functions is computed with the same method as was used for the computation of the fitness of an NNC, which was presented and discussed in Section 6-4. The flow diagram corresponding the fitness computation is shown in Figure 7-4. The output of the FLCs is computed using a build-in Simulink block that is also part of the fuzzy logic toolbox, and requires only the .fis file describing the FLC, as input. The Simulink block performs the fuzzy inference as described in Section 7-2. Similar to the previous controllers in the report, the inputs to the FLC are the errors in the rotational states (with or without integral terms) of the vehicle. However, the outputs of the FLC are the gains, which,

when multiplied with the state of the vehicle, give the moment-fractions. The resulting controller architecture for the FLCs is shown in Figure 7-5. It is noted that, for the FLIC, the rotational state vector is modified to include the integral terms.

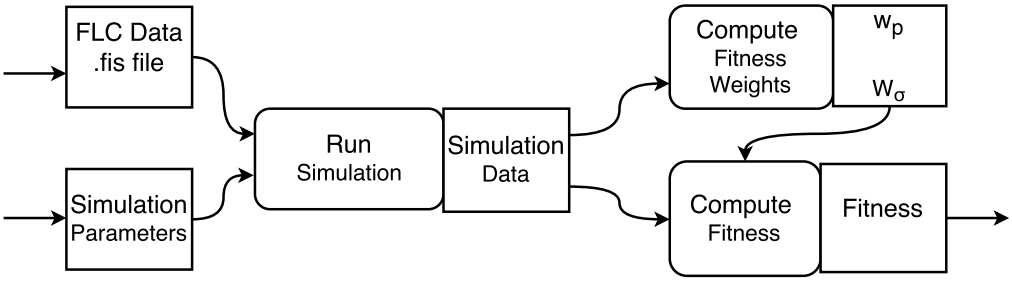


Figure 7-4: Flow diagram of the fitness computation during optimization of an FLC.

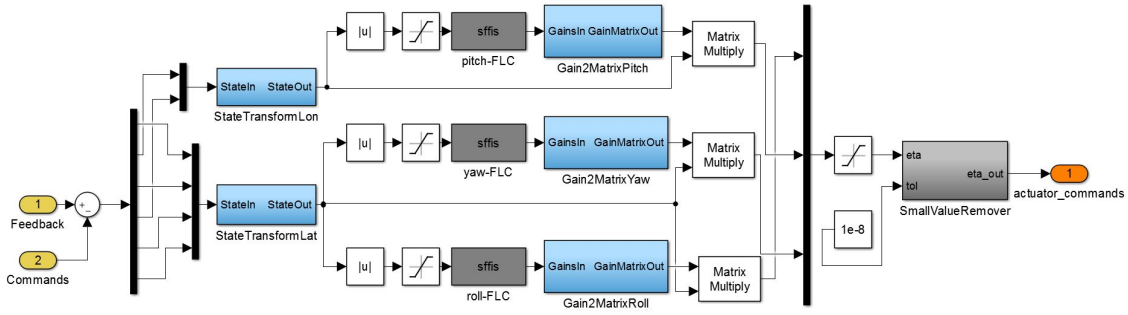


Figure 7-5: Software Architecture of a general FLC.

The optimization process of the FLC separated into the sequential optimization of the controllers for the individual pitch-, roll-, and yaw-axes in an identical fashion as was done for the NNC. Therefore, the optimization procedure discussed for the NNC is repeated for each of the three axes for both the FLC and the FLIC. It is noted that, it is possible to derive a single FLC from the three separate controllers for each axis. In the SAD shown in Figure 7-5, the three controllers are left separate, to indicate that they are trained individually.

It is noted that an FLC is designed only for a single point along the reference trajectory. To design a controller that is valid along the entire reference trajectory, one can optimize an FLC along multiple points of the reference trajectory and interpolate the optimal FLCs, by scheduling the obtained membership functions as a function of both a state variable and the dynamic pressure. This takes advantage of the internal interpolation ability of the Mamdani controller. It is noted that, because each output is a function of only two variables, a state variable and the dynamic pressure, one can still have physical insight into the behaviour of the system by means of visual graphs. However, the design of an FLC that performs optimally along the entire reference trajectory is beyond the scope of this report.

Chapter 8

Model Predictive Control

Model Predictive Control has developed from a basic multi-variable process control technology to a methodology that enables operation of processes within well defined operation constraints, and is considered by many researchers as one of the most promising methods in control engineering (Allgöwer et al., 1999; Bequette, 1991; Garcia et al., 1989; Mayne et al., 2000). MPC makes use of an explicit model for the dynamics. However, some assumptions are made on the model which are as follows (van den Boom, 2013):

- An LTI model
- Discrete-time
- Causal
- Finite Order
- State space description

The model of the dynamics described in Equation 3-27 satisfies four out of five of the assumptions. The only assumption that is not satisfied is the discrete-time assumption. It is, however, always possible to transform a continuous time LTI model to a discrete-time LTI model using, for example, a *zero-order hold transformation* (van den Boom, 2013). Furthermore, it is assumed that the output of the system is known perfectly, that is, there is no Zero-Mean White Noise (ZMWN) addition to the state- and output-signal. Additionally, it is assumed that there are no known disturbances acting on the system. It is noted that Equation 3-27 is, technically, not time-invariant. However, in practice, the linearised system varies slowly with time and, hence, the time-invariant assumption is justified.

To determine the optimal control action, MPC minimizes a performance index subject to constraints. To facilitate the use of various such performance indices, the control problem is often written in a standard formulation, known as the Standard Predictive Control Problem (SPCP). Section 8-1 will introduce the notations of the SPCP. Section 8-2 will then introduce the chosen performance index and how this relates to the SPCP performance index. Similarly, Section 8-3 will present the chosen process model and relates this to the standard model. Because MPC relies on making predictions of the future states, Section 8-4 will present the method used to make predictions of the future states. Section 8-5 will introduce the main reason for the popularity of MPCs; the constraint-handling. Section 8-6 will then show how to find the solution to the fully constrained SPCP as defined in previous sections, and Section 8-7 will discuss the recommended settings for the tunable parameters. Section 8-8 will conclude this chapter by discussing the practical consideration of applying a MPCs to the attitude control of an entry vehicle.

8-1 Standard Predictive Control Problem

In predictive control various types of performance indices can be used. It is, however, useful to have a single, standard performance index to work with when solving the control problem. Therefore, a standard performance index for the SPCP is defined. The performance index of the SPCP can be written as (van den Boom, 2013):

$$J(\mathbf{v},k)=\sum_{j=0}^{N-1}\hat{\mathbf{z}}^T(k+j|k)\mathbf{\Gamma}(j)\hat{\mathbf{z}}(k+j|k) \tag{8-1}$$

where $\hat{\mathbf{z}}(k+j|k)$ is the prediction of $\mathbf{z}(k+j)$ at time k and $\mathbf{\Gamma}(j)$ is a diagonal selection matrix with ones and zeros on the diagonal. This can also be written as:

$$J(\mathbf{v},k)=\tilde{\mathbf{z}}^T(k)\bar{\mathbf{\Gamma}}\tilde{\mathbf{z}}(k) \tag{8-2}$$

with

$$\tilde{\mathbf{z}}(k)=\begin{bmatrix}\hat{\mathbf{z}}(k|k)\\ \hat{\mathbf{z}}(k+1|k)\\ \vdots\\ \tilde{\mathbf{z}}(k+N-1|k)\end{bmatrix}\quad,\quad \bar{\mathbf{\Gamma}}=\begin{bmatrix}\mathbf{\Gamma}(0) & \mathbf{0} & \dots & \mathbf{0}\\ \mathbf{0} & \mathbf{\Gamma}(1) & & \mathbf{0}\\ \vdots & & \ddots & \vdots\\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{\Gamma}(N-1)\end{bmatrix} \tag{8-3}$$

where $\tilde{\mathbf{z}}(k)$ is the concatenated vector of the predicted performance signals $\hat{\mathbf{z}}(k+j|k)$ and $\bar{\mathbf{\Gamma}}$ is the block diagonal matrix of $\mathbf{\Gamma}(j)$. How to compute the prediction signals $\hat{\mathbf{z}}(k+j|k)$ will be explained in Section 8-4.

Besides a standard performance index, also a standard model description is used. This model includes the state signal $\mathbf{x}$, the input signal $\mathbf{v}$, the output signal $\mathbf{y}$, and the performance signal $\mathbf{z}$ and they are related as follows (van den Boom, 2013):

$$\begin{aligned}\mathbf{x}(k+1)&=\mathbf{A}\mathbf{x}(k)+\mathbf{B}\mathbf{v}(k)\\ \mathbf{y}(k)&=\mathbf{C}_1\mathbf{x}(k)\\ \mathbf{z}(k)&=\mathbf{C}_2\mathbf{x}(k)+\mathbf{D}_{23}\mathbf{v}(k)\end{aligned} \tag{8-4}$$

where the matrices $\mathbf{A},\mathbf{B},\mathbf{C}_1,\mathbf{C}_2$, and $\mathbf{D}_{23}$ are constant matrices for each time-step. The SPCP is thus an LTI-model. The physical meaning and values of those matrices will be discussed in the following sections.

8-2 Linear Quadratic Predictive Control Performance Index

In the LQPC method, the performance index is based on the control and state signals (Garcia et al., 1989):

$$J(\mathbf{u},k)=\sum_{j=N_m}^{N}\hat{\mathbf{x}}(k+j|k)^T\mathbf{Q}\hat{\mathbf{x}}(k+j|k)+\sum_{j=1}^{N}\mathbf{u}^T(k+j-1)^T\mathbf{R}\mathbf{v}^T(k+j-1) \tag{8-5}$$

where $\mathbf{x}(k)$ is the state signal, $\mathbf{v}(k)$ is the process control signal, which can be either the input-signal or the increment input-signal, N_m is the minimum cost-horizon, N is the prediction horizon, $\mathbf{Q}$ is the state weighting-matrix, and $\mathbf{R}$ is the weighting on the control signal. Furthermore, $\hat{\mathbf{x}}(k+j|k)$ is the prediction of $\mathbf{x}(k+j)$, based on knowledge up to time k, and $\delta\mathbf{u}(k)=0$ for $j\geq N_c$, where

N_c is the control horizon. It is noted that the LQPC performance index of Equation 8-5 can be transformed into the standard performance index of Equation 8-1 by choosing:

$$\mathbf{z}(k) = \begin{bmatrix} \mathbf{Q}^{1/2}\mathbf{x}(k+1) \\ \mathbf{R}^{1/2}\mathbf{v}(k) \end{bmatrix} \tag{8-6}$$

$$\mathbf{\Gamma}(j) = \begin{cases} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} & \text{for} \quad 0 \leq j < N_m - 1 \\ \\ \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} & \text{for} \quad N_m - 1 \leq j \leq N - 1 \end{cases} \tag{8-7}$$

where $\mathbf{I}$ is the identity matrix.

It is interesting to note that the performance index of Equation 8-5 is equivalent to the cost-functional of the finite-horizon, discrete-time LQR (Strejc, 1981). This implies that when the prediction horizon is moved to infinity and the sample time of the discrete system approaches zero, then an unconstrained linear quadratic predictive controller is equivalent to an infinite horizon, continuous-time linear quadratic controller, which were chosen for the benchmark controllers.

8-3 Predictive Control Process Model

In standard MPC, the *present* states are used to predict the states. However, it was shown by Ghahramani and Towhidkhah (2009) that including both present and previous states can improve the robustness of the resulting controller. The system used by Ghahramani and Towhidkhah (2009), however, was a relatively simple robot, and while Dehkordi et al. (2013) showed similar improvements of robustness for a satellite, an entry vehicle is still a more complex system. It is therefore decided to investigate whether improved robustness also occurs for entry vehicles, and what its effects might be on the performance of the controller. Hence, a process model using only the present states for predictions, known as the Input-Output (IO)-model, and a process model using both the present and previous states for predictions, known as the Increment-Input-Output (IIO)-model, will be used to design a model-predictive controller. Sections 8-3-1 and 8-3-2 will discuss the IO-model and the IIO-model, respectively.

8-3-1 Input-Output Model

Determining the physical meaning of the matrices of Equation 8-4 can be done by rewriting the state-space model of the reference vehicle into the SPCP format. First, the continuous-time state-space equation must be transformed to its discrete counterpart. Because of the assumption of LTI system, the continuous-time state matrix, $\mathbf{A}$, and the continuous-time input matrix , $\mathbf{B}$, of Equation 3-22 are assumed to be constant and, hence, the zero-order hold transformation can be used to determine the discrete system. A clever trick can be used to compute the discrete state matrix, $\mathbf{A}_d$, and the discrete input matrix, $\mathbf{B}_d$ in one step: (DeCarlo, 1989)

$$\begin{aligned} e^{\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} T_s} &= \begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \\ \mathbf{A}_d &= \mathbf{M}_{11} \\ \mathbf{B}_d &= \mathbf{M}_{12} \end{aligned} \tag{8-8}$$

where T_s is the sample time. It is noted that the discrete version of the output matrix, $\mathbf{C}_d$, is equal to its continuous counterpart. It is noted that, within the MATLAB programming language, a standard command exists to compute the discrete counterpart of a continuous state-space system.

Applying Equation 8-8 to the continuous state-space system of Equations 3-22 and 3-25 gives, omitting the Δ-notation for simplicity:

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A}_d\mathbf{x}(k) + \mathbf{B}_d\mathbf{u}(k) \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) \end{aligned} \tag{8-9}$$

Equation 8-9 is already in right form for the SPCP. What remains is to determine the matrices for performance signal $\mathbf{z}(k)$. Substituting Equation 8-9 into Equation 8-6 gives (van den Boom, 2013):

$$\begin{aligned} \mathbf{z}(k) &= \begin{bmatrix} \mathbf{Q}^{1/2}\mathbf{x}(k+1) \\ \mathbf{R}^{1/2}\mathbf{u}(k) \end{bmatrix} = \begin{bmatrix} \mathbf{Q}^{1/2}\mathbf{A}_d \\ \mathbf{0} \end{bmatrix} \mathbf{x}(k) + \begin{bmatrix} \mathbf{Q}^{1/2}\mathbf{B}_d \\ \mathbf{R}^{1/2} \end{bmatrix} \mathbf{u}(k) \\ &= \mathbf{C}_2\mathbf{x}(k) + \mathbf{D}_{23}\mathbf{v}(k) \end{aligned} \tag{8-10}$$

where $\mathbf{v}(k) = \mathbf{u}(k)$.

8-3-2 Increment-Input-Output Model

The matrices of the SPCP are defined differently when one uses a IIO-model. The input-increment is defined as shown in Equation 8-11. It is noted that the lower-case δ is defined to be the increment of a variable, while the upper-case Δ is used to describe the difference between the actual value of the variable and its value around which the EoM were linearised, as was used in Section 3-7. This distinction in notation is made to avoid any confusion on what type of variable is meant.

$$\delta\mathbf{u}(k) = \mathbf{u}(k) - \mathbf{u}(k-1) \tag{8-11}$$

Using the increments of the input signal implies that the model needs to keep track of the actual values of the input signal by integrating the increments of the inputs. The discrete one-step-ahead prediction of the states as a function of the current and previous states can be derived as follows:

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A}_d\mathbf{x}(k) + \mathbf{B}_d\mathbf{u}(k) \\ &= \mathbf{A}_d\mathbf{x}(k) + \mathbf{B}_d\mathbf{u}(k) + \mathbf{x}(k) - (\mathbf{A}_d\mathbf{x}(k-1) + \mathbf{B}_d\mathbf{u}(k-1)) \\ &= (\mathbf{A}_d + \mathbf{I})\,\mathbf{x}(k) - \mathbf{A}_d\mathbf{x}(k-1) + \mathbf{B}_d\delta\mathbf{u}(k) \end{aligned} \tag{8-12}$$

While the transformation is not explicitly found in the literature, one can easily derive the required transformation to bring Equation 8-12 to the standard form of Equation 8-4. Defining a new state as:

$$\underline{\mathbf{x}}(k) = \begin{bmatrix} \mathbf{x}(k-1) \\ \mathbf{x}(k) \end{bmatrix} \tag{8-13}$$

from which one can easily derive:

$$\begin{aligned}\underline{\mathbf{x}}(k+1) &= \begin{bmatrix}\mathbf{x}(k)\\ \mathbf{x}(k+1)\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\\ -\mathbf{A}_d & \mathbf{A}_d+\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{x}(k-1)\\ \mathbf{x}(k)\end{bmatrix} + \begin{bmatrix}\mathbf{0}\\ \mathbf{B}_d\end{bmatrix}\delta\mathbf{u}(k)\\ &= \underline{\mathbf{A}}\underline{\mathbf{x}}(k) + \underline{\mathbf{B}}\mathbf{v}(k)\end{aligned} \tag{8-14}$$

where $\mathbf{v}(k) = \delta\mathbf{u}(k)$. The corresponding output signal can be extracted as:

$$\mathbf{y}(k) = \begin{bmatrix}\mathbf{C} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{x}(k)\\ \mathbf{x}(k-1)\end{bmatrix} = \underline{\mathbf{C}}_1\underline{\mathbf{x}}(k) \tag{8-15}$$

Additionally, the performance signal z must be found to make the SPCP complete. Substituting Equation 8-12 into Equation 8-6 gives:

$$\begin{aligned}\mathbf{z}(k) &= \begin{bmatrix}\mathbf{Q}^{1/2}\mathbf{x}(k+1)\\ \mathbf{R}^{1/2}\delta\mathbf{u}(k)\end{bmatrix} = \begin{bmatrix}-\mathbf{Q}^{1/2}\mathbf{A}_d & \mathbf{Q}^{1/2}\left(\mathbf{A}_d+\mathbf{I}\right)\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{x}(k-1)\\ \mathbf{x}(k)\end{bmatrix} + \begin{bmatrix}\mathbf{Q}^{1/2}\mathbf{B}_d\\ \mathbf{R}^{1/2}\end{bmatrix}\delta\mathbf{u}(k)\\ &= \underline{\mathbf{C}}_2\underline{\mathbf{x}}(k) + \underline{\mathbf{D}}_{23}\mathbf{v}(k)\end{aligned} \tag{8-16}$$

8-4 Predictions

The control law in predictive control is, no surprise, based on predictions. At each time constant k, all signals over the horizon N are considered. To do so, predictions of the performance signal $\mathbf{z}(k+j)$ need to be made for $j = 1, 2, \ldots, N$. These predictions, denoted by $\hat{\mathbf{z}}(k+j|k)$, are based on the model of Equation 8-4, using information given at time k and future values of the control signal $\mathbf{v}(k+j|k)$. At time instant k a signal vector $\tilde{\mathbf{z}}(k)$ with the prediction signals $\hat{\mathbf{z}}(k+j|k)$ is defined. Additionally, the input signal vector $\tilde{\mathbf{v}}(k)$ with the future control signal $\mathbf{v}(k+j|k)$ is defined. In equation form, they are given as follows (van den Boom, 2013):

$$\tilde{\mathbf{z}}(k) = \begin{bmatrix}\hat{\mathbf{z}}(k|k)\\ \hat{\mathbf{z}}(k+1|k)\\ \vdots\\ \hat{\mathbf{z}}(k+N-1|k)\end{bmatrix}, \quad \tilde{\mathbf{v}}(k) = \begin{bmatrix}\mathbf{v}(k|k)\\ \mathbf{v}(k+1|k)\\ \vdots\\ \mathbf{v}(k+N-1|k)\end{bmatrix} \tag{8-17}$$

The concatenated performance signal $\tilde{\mathbf{z}}(k)$ can be computed as:

$$\tilde{\mathbf{z}}(k) = \tilde{\mathbf{C}}_2\mathbf{x}(k) + \tilde{\mathbf{D}}_{23}\tilde{\mathbf{v}}(k) \tag{8-18}$$

with

$$\tilde{\mathbf{C}}_\mathbf{2} = \begin{bmatrix}\mathbf{C}_2\\ \mathbf{C}_2\mathbf{A}\\ \mathbf{C}_2\mathbf{A}^2\\ \vdots\\ \mathbf{C}_2\mathbf{A}^N\end{bmatrix}, \qquad \tilde{\mathbf{D}}_{23} = \begin{bmatrix}\mathbf{D}_{23} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0}\\ \mathbf{C}_2\mathbf{B}_3 & \mathbf{D}_{23} & \ddots & \vdots & \vdots\\ \mathbf{C}_2\mathbf{A}\mathbf{B}_3 & \mathbf{C}_2\mathbf{B}_3 & \ddots & \mathbf{0} & \mathbf{0}\\ \vdots & & \ddots & \mathbf{D}_{23} & \mathbf{0}\\ \mathbf{C}_2\mathbf{A}^{N-1}\mathbf{B}_3 & \ldots & \ldots & \mathbf{C}_2\mathbf{B}_3 & \mathbf{D}_{23}\end{bmatrix}$$

8-5 Constraint Handling

In many practical situations, there will be signal constraints. These constraints on control, state and output can be motivated by safety, environmental constraints and system limitations. There are two types of constraints; *equality* and *inequality* constraints, which will be treated separately in the following.

8-5-1 Equality Constraints

In the SPCP framework, the equality constraints are defined to be linear combinations of the state and the input vector. In this report, the only equality constraint considered is the *control horizon*, which only involves the inputs. Hence, the equality constraints can be written as:

$$\tilde{\boldsymbol{\phi}}(k) = \tilde{\mathbf{D}}_{33}\tilde{\mathbf{v}}(k) = \mathbf{0}, \quad \tilde{\boldsymbol{\phi}}(k) = \begin{bmatrix} \mathbf{S}_1\hat{\boldsymbol{\phi}}(k+1|k) \\ \mathbf{S}_2\hat{\boldsymbol{\phi}}(k+2|k) \\ \vdots \\ \mathbf{S}_N\hat{\boldsymbol{\phi}}(k+N|k) \end{bmatrix} \tag{8-19}$$

where $\tilde{\boldsymbol{\phi}}$ is the stacked vector containing all predicted constraints $\hat{\boldsymbol{\phi}}$, and the matrix $\mathbf{S}_j$ is used to indicate that certain equality constraints are only active for specific j.

The control horizon constraint for the IO-model (with $\mathbf{v}(k) = \mathbf{u}(k)$) is given by $\mathbf{v}(k+N_c+j|k) = \mathbf{v}(k+N_c-1|k)$ for $j = 0, \ldots, N-N_c-1$, which is equivalent to (van den Boom, 2013):

$$\begin{aligned} \begin{bmatrix} \hat{\boldsymbol{\phi}}(k+N_c|k) \\ \hat{\boldsymbol{\phi}}(k+N_c+1|k) \\ \vdots \\ \hat{\boldsymbol{\phi}}(k+N-1|k) \end{bmatrix} &= \begin{bmatrix} \mathbf{0} & \ldots & \mathbf{0} & -\mathbf{I} & \mathbf{I} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \ldots & \mathbf{0} & -\mathbf{I} & \mathbf{0} & \mathbf{I} & \ddots & \vdots \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \ldots & \mathbf{0} & -\mathbf{I} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{v}(k|k) \\ \vdots \\ \mathbf{v}(k+N_c-2|k) \\ \mathbf{v}(k+N_c-1|k) \\ \mathbf{v}(k+N_c|k) \\ \mathbf{v}(k+N_c+1|k) \\ \vdots \\ \mathbf{v}(k+N-1|k) \end{bmatrix} \\ &= \tilde{\mathbf{D}}_{33}\tilde{\mathbf{v}}(k) = \mathbf{0} \end{aligned} \tag{8-20}$$

The control horizon constraint for the IIO-model (with $\mathbf{v}(k) = \delta\mathbf{u}(k)$) is given by $\mathbf{v}(k+N_c+j|k) = \mathbf{0}$ for $j = 0, \ldots, N-N_c-1$, which changes the definition of $\tilde{\mathbf{D}}_{33}$ when compared to Equation 8-20, as is shown in Equation 8-21 (van den Boom, 2013).

$$\begin{aligned} \begin{bmatrix} \hat{\boldsymbol{\phi}}(k+N_c|k) \\ \hat{\boldsymbol{\phi}}(k+N_c+1|k) \\ \vdots \\ \hat{\boldsymbol{\phi}}(k+N-1|k) \end{bmatrix} &= \begin{bmatrix} \mathbf{0} & \ldots & \mathbf{0} & \mathbf{I} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} & \mathbf{I} & \ddots & \vdots \\ \vdots & & \vdots & \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{v}(k|k) \\ \vdots \\ \mathbf{v}(k+N_c-1|k) \\ \mathbf{v}(k+N_c|k) \\ \mathbf{v}(k+N_c+1|k) \\ \vdots \\ \mathbf{v}(k+N-1|k) \end{bmatrix} \\ &= \tilde{\mathbf{D}}_{33}\tilde{\mathbf{v}}(k) = \mathbf{0} \end{aligned} \tag{8-21}$$

With the definitions of $\tilde{\mathbf{D}}_{33}$ as found in Equations 8-20 and 8-21, the control horizon constraint has been translated into the standard form of Equation 8-19. It is noted that $\tilde{\mathbf{D}}_{33} \in \mathbb{R}^{Nm \times (N-N_c)m}$, where m is the number of input signals. This implies that, when the control horizon is equal to the prediction horizon, there is no equality constraint. If the control horizon is chosen small compared to the prediction horizon, the matrix $\tilde{\mathbf{D}}_{33}$ will be relatively large. This size-dependence will be referred to when discussing the solution to the SPCP in Section 8-6. Additionally, the definition and size of $\tilde{\mathbf{D}}_{33}$ implies that it will always have full row rank, which means the SPCP can be solved (van den Boom, 2013).

8-5-2 Inequality Constraints

In the SPCP framework, the inequality constraints are defined as (van den Boom, 2013):

$$\tilde{\boldsymbol{\psi}}(k) = \tilde{\mathbf{C}}_4 \mathbf{x}(k) + \tilde{\mathbf{D}}_{43} \tilde{\mathbf{v}}(k) \leq \tilde{\boldsymbol{\Psi}}(k) \tag{8-22}$$

where $\tilde{\boldsymbol{\psi}}$ and $\tilde{\boldsymbol{\Psi}}$ are the stacked vectors containing all predictions of the inequality constraints, $\hat{\boldsymbol{\psi}}$, and the upper boundaries of the inequality constraints, $\hat{\boldsymbol{\Psi}}$, i.e.:

$$\tilde{\boldsymbol{\psi}}(k) = \begin{bmatrix} \hat{\boldsymbol{\psi}}(k|k) \\ \hat{\boldsymbol{\psi}}(k+1|k) \\ \vdots \\ \hat{\boldsymbol{\psi}}(k+N|k) \end{bmatrix}, \qquad \tilde{\boldsymbol{\Psi}}(k) = \begin{bmatrix} \hat{\boldsymbol{\Psi}}(k|k) \\ \hat{\boldsymbol{\Psi}}(k+1|k) \\ \vdots \\ \hat{\boldsymbol{\Psi}}(k+N|k) \end{bmatrix} \tag{8-23}$$

while only upper bounds are considered in Equation 8-22, it is always possible to transform lower bounds into upper bounds by setting $-\mathbf{z}(k) \leq -\mathbf{z}_{min}$.

As was the case for the equality constraints, the definitions of the matrices for the inequality constraints are different, depending on whether an IO-model or an IIO-model is used. However, this difference is only present if one wants to have constraints on the increments of the state- or input-signals for the IO-model, or on the increment of the state-signal and on the input-signal for the IIO-model. Because the state vector of Equation 3-23 already contains derivatives of the attitude of the reference vehicle, inequality constraints on the increments of the state vector will not be considered. How to implement inequality constraints of the input-signal and its increments for both models will be derived in the remainder of this section by example.

Inequality Constraints for IO-model

Suppose one has an IO-model with the following constraints:

$$\begin{aligned} \mathbf{x}_{min}(k) \leq & \mathbf{S}_{\mathbf{x}} \mathbf{x}(k) \leq \mathbf{x}_{max}(k) \\ \mathbf{u}_{min}(k) \leq & \mathbf{S}_{\mathbf{u}} \mathbf{u}(k) \leq \mathbf{u}_{max}(k) \\ \delta\mathbf{u}_{min}(k) \leq & \mathbf{S}_{\delta\mathbf{u}} \delta\mathbf{u}(k) \leq \delta\mathbf{u}_{max}(k) \end{aligned} \tag{8-24}$$

where $\mathbf{S}_{\mathbf{x}}, \mathbf{S}_{\mathbf{u}}$, and $\mathbf{S}_{\delta\mathbf{u}}$ represent the selection matrices of the state-, input-, and increment input-signals, respectively. Transforming the lower-bounds into upper-bounds, substituting Equation 8-11 into Equation 8-24, and concatenating the system of equations into matrices, one obtains:

$$\begin{bmatrix} \mathbf{S_x} \\ -\mathbf{S_x} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \mathbf{x}(k) + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{S_u} \\ -\mathbf{S_u} \\ \mathbf{S}_{\delta\mathbf{u}} \\ -\mathbf{S}_{\delta\mathbf{u}} \end{bmatrix} \mathbf{u}(k) + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ -\mathbf{S}_{\delta\mathbf{u}} \\ \mathbf{S}_{\delta\mathbf{u}} \end{bmatrix} \mathbf{u}(k-1) \leq \begin{bmatrix} \mathbf{x}_{max} \\ -\mathbf{x}_{min} \\ \mathbf{u}_{max} \\ -\mathbf{u}_{min} \\ \delta\mathbf{u}_{max} \\ -\delta\mathbf{u}_{min} \end{bmatrix} \tag{8-25}$$

The inequality of Equation 8-25 can be rewritten using newly defined matrices, as is shown in Equation 8-26.

$$\boldsymbol{\psi}(k) = \mathbf{C}_4\mathbf{x}(k) + \mathbf{D}_{43}\mathbf{u}(k) + \mathbf{D}_{43}^{\dagger}\mathbf{u}(k-1) \leq \boldsymbol{\Psi}(k) \tag{8-26}$$

The predictions of the inequality constraints can be derived by substitutions of Equation 8-4 into Equation 8-26, as is shown in Equation 8-27.

$$\begin{aligned}
\hat{\boldsymbol{\psi}}(k+1|k) &= \mathbf{C}_4\mathbf{x}(k+1) + \mathbf{D}_{43}\mathbf{u}(k+1) + \mathbf{D}_{43}^{\dagger}\mathbf{u}(k) \leq \hat{\boldsymbol{\Psi}}(k+1|k) \\
&= \mathbf{C}_4\mathbf{A}\mathbf{x}(k) + \left(\mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger}\right)\mathbf{u}(k) + \mathbf{D}_{43}\mathbf{u}(k+1) \leq \hat{\boldsymbol{\Psi}}(k+1|k) \\
\hat{\boldsymbol{\psi}}(k+2|k) &= \mathbf{C}_4\mathbf{A}\mathbf{x}(k+1) + \left(\mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger}\right)\mathbf{u}(k+1) + \mathbf{D}_{43}\mathbf{u}(k+2) \leq \hat{\boldsymbol{\Psi}}(k+2|k) \\
&= \mathbf{C}_4\mathbf{A}^2\mathbf{x}(k) + \mathbf{C}_4\mathbf{A}\mathbf{B}\mathbf{u}(k) + \left(\mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger}\right)\mathbf{u}(k+1) + \mathbf{D}_{43}\mathbf{u}(k+2) \leq \hat{\boldsymbol{\Psi}}(k+2|k) \\
&\vdots \\
\hat{\boldsymbol{\psi}}(k+j|k) &= \mathbf{C}_4\mathbf{A}^j\mathbf{x}(k) + \sum_{i=1}^{j}\mathbf{C}_4\mathbf{A}^{i-1}\mathbf{B}\mathbf{u}(k+j-i|k) + \mathbf{D}_{43}\mathbf{u}(k+j|k) + \\
&\quad + \mathbf{D}_{43}^{\dagger}\mathbf{u}(k+j-1|k) \leq \hat{\boldsymbol{\Psi}}(k+j|k)
\end{aligned} \tag{8-27}$$

Equation 8-27 can be compactly written into the format of Equation 8-22, as is shown in Equation 8-28.

$$\begin{aligned}
\begin{bmatrix} \hat{\boldsymbol{\psi}}(k|k) \\ \hat{\boldsymbol{\psi}}(k+1|k) \\ \vdots \\ \hat{\boldsymbol{\psi}}(k+N|k) \end{bmatrix} &= \begin{bmatrix} \mathbf{C}_4 \\ \mathbf{C}_4\mathbf{A} \\ \mathbf{C}_4\mathbf{A}^2 \\ \vdots \\ \mathbf{C}_4\mathbf{A}^N \end{bmatrix} \mathbf{x}(k) + \begin{bmatrix} \mathbf{D}_{43} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger} & \mathbf{D}_{43} & \ddots & \vdots & \vdots \\ \mathbf{C}_4\mathbf{A}\mathbf{B} & \mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger} & \ddots & \mathbf{0} & \mathbf{0} \\ \vdots & & \ddots & \mathbf{D}_{43} & \mathbf{0} \\ \mathbf{C}_4\mathbf{A}^{N-1}\mathbf{B} & \dots & \dots & \mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger} & \mathbf{D}_{43} \end{bmatrix} \tilde{\mathbf{v}}(k) \\
&\leq \begin{bmatrix} \hat{\boldsymbol{\Psi}}(k|k) - \mathbf{D}_{43}^{\dagger}\mathbf{u}(k-1) \\ \hat{\boldsymbol{\Psi}}(k+1|k) \\ \vdots \\ \hat{\boldsymbol{\Psi}}(k+N|k) \end{bmatrix} \\
\tilde{\boldsymbol{\psi}}(k) &= \tilde{\mathbf{C}}_4\mathbf{x}(k) + \tilde{\mathbf{D}}_{43}\tilde{\mathbf{v}}(k) \leq \tilde{\boldsymbol{\Psi}}(k+N|k)
\end{aligned} \tag{8-28}$$

The last line of Equation 8-28 shows that the set of inequality constraints, as defined by Equation 8-24, have been rewritten into that standard format of Equation 8-22.

Inequality Constraints for IIO-model

Suppose one has an IIO-model with the constraints identical to those of Equation 8-24. Noting that $\mathbf{u}(k) = \mathbf{u}(k-1) + \delta\mathbf{u}(k)$, one can rewrite and concatenate the system of constraints to obtain Equation 8-29.

$$\begin{bmatrix} \mathbf{S_x} \\ -\mathbf{S_x} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \mathbf{x}(k) + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{S_u} \\ -\mathbf{S_u} \\ \mathbf{S}_{\delta\mathbf{u}} \\ -\mathbf{S}_{\delta\mathbf{u}} \end{bmatrix} \delta\mathbf{u}(k) + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{S_u} \\ -\mathbf{S_u} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \mathbf{u}(k-1) \leq \begin{bmatrix} \mathbf{x}_{max} \\ -\mathbf{x}_{min} \\ \mathbf{u}_{max} \\ -\mathbf{u}_{min} \\ \delta\mathbf{u}_{max} \\ -\delta\mathbf{u}_{min} \end{bmatrix} \tag{8-29}$$

The inequality of Equation 8-29 can be rewritten using newly defined matrices, as is shown in Equation 8-30.

$$\boldsymbol{\psi}(k) = \mathbf{C}_4\mathbf{x}(k) + \mathbf{D}_{43}\delta\mathbf{u}(k) + \mathbf{D}_{43}^{\dagger}\mathbf{u}(k-1) \leq \boldsymbol{\Psi}(k) \tag{8-30}$$

The predictions of the inequality constraints can be derived by substitutions of Equation 8-4 into Equation 8-30, as is shown in Equation 8-31.

$$\begin{aligned}
\hat{\boldsymbol{\psi}}(k+1|k) &= \mathbf{C}_4\mathbf{x}(k+1) + \mathbf{D}_{43}\delta\mathbf{u}(k+1|k) + \mathbf{D}_{43}^{\dagger}\mathbf{u}(k) \leq \hat{\boldsymbol{\Psi}}(k+1|k) \\
&= \mathbf{C}_4\mathbf{A}\mathbf{x}(k) + \left(\mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger}\right)\delta\mathbf{u}(k|k) + \mathbf{D}_{43}\delta\mathbf{u}(k+1|k) + \mathbf{D}_{43}^{\dagger}\mathbf{u}(k-1) \leq \hat{\boldsymbol{\Psi}}(k+1|k) \\
\hat{\boldsymbol{\psi}}(k+2|k) &= \mathbf{C}_4\mathbf{A}\mathbf{x}(k+1) + \left(\mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger}\right)\delta\mathbf{u}(k+1|k) + \mathbf{D}_{43}\delta\mathbf{u}(k+2|k) + \mathbf{D}_{43}^{\dagger}\mathbf{u}(k) \leq \hat{\boldsymbol{\Psi}}(k+2|k) \\
&= \mathbf{C}_4\mathbf{A}^2\mathbf{x}(k) + \left(\mathbf{C}_4\mathbf{A}\mathbf{B} + \mathbf{D}_{43}^{\dagger}\right)\delta\mathbf{u}(k|k) + \left(\mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger}\right)\delta\mathbf{u}(k+1|k) + \mathbf{D}_{43}\delta\mathbf{u}(k+2|k) + \\
&\quad + \mathbf{D}_{43}^{\dagger}\mathbf{u}(k-1) \leq \hat{\boldsymbol{\Psi}}(k+2|k) \\
&\vdots \\
\hat{\boldsymbol{\psi}}(k+j|k) &= \mathbf{C}_4\mathbf{A}^j\mathbf{x}(k) + \sum_{i=1}^{j}\left(\mathbf{C}_4\mathbf{A}^{i-1}\mathbf{B} + \mathbf{D}_{43}^{\dagger}\right)\delta\mathbf{u}(k+j-i|k) + \mathbf{D}_{43}\delta\mathbf{u}(k+j|k) + \\
&\quad + \mathbf{D}_{43}^{\dagger}\mathbf{u}(k-1) \leq \hat{\boldsymbol{\Psi}}(k+j|k)
\end{aligned} \tag{8-31}$$

Equation 8-31 can be compactly written into the format of Equation 8-22, as is shown in Equation 8-32.

$$\begin{aligned}
\begin{bmatrix} \hat{\boldsymbol{\psi}}(k|k) \\ \hat{\boldsymbol{\psi}}(k+1|k) \\ \vdots \\ \hat{\boldsymbol{\psi}}(k+N|k) \end{bmatrix} &= \begin{bmatrix} \mathbf{C}_4 \\ \mathbf{C}_4\mathbf{A} \\ \mathbf{C}_4\mathbf{A}^2 \\ \vdots \\ \mathbf{C}_4\mathbf{A}^N \end{bmatrix} \mathbf{x}(k) + \begin{bmatrix} \mathbf{D}_{43} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger} & \mathbf{D}_{43} & \ddots & \vdots & \vdots \\ \mathbf{C}_4\mathbf{A}\mathbf{B} + \mathbf{D}_{43}^{\dagger} & \mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger} & \ddots & \mathbf{0} & \mathbf{0} \\ \vdots & & \ddots & \mathbf{D}_{43} & \mathbf{0} \\ \mathbf{C}_4\mathbf{A}^{N-1}\mathbf{B} + \mathbf{D}_{43}^{\dagger} & \dots & \dots & \mathbf{C}_4\mathbf{B} + \mathbf{D}_{43}^{\dagger} & \mathbf{D}_{43} \end{bmatrix} \tilde{\mathbf{v}}(k) \\
&\leq \begin{bmatrix} \hat{\boldsymbol{\Psi}}(k|k) - \mathbf{D}_{43}^{\dagger}\mathbf{u}(k-1) \\ \hat{\boldsymbol{\Psi}}(k+1|k) - \mathbf{D}_{43}^{\dagger}\mathbf{u}(k-1) \\ \vdots \\ \hat{\boldsymbol{\Psi}}(k+N|k) - \mathbf{D}_{43}^{\dagger}\mathbf{u}(k-1) \end{bmatrix} \\
\tilde{\boldsymbol{\psi}}(k) &= \tilde{\mathbf{C}}_4\mathbf{x}(k) + \tilde{\mathbf{D}}_{43}\tilde{\mathbf{v}}(k) \leq \tilde{\boldsymbol{\Psi}}(k+N|k)
\end{aligned} \tag{8-32}$$

The last line of Equation 8-32 shows that the set of inequality constraints, as defined by Equation 8-24, have been rewritten into that standard format of Equation 8-22.

8-6 Solution the Standard Predictive Control Problem

Up to now, this chapter has presented the steps needed to construct the SPCP. It was found that the SPCP is defined by the following state-space realization (van den Boom, 2013):

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{v}(k) \\ \mathbf{y}(k) &= \mathbf{C}_1\mathbf{x}(k) \\ \mathbf{z}(k) &= \mathbf{C}_2\mathbf{x}(k) + \mathbf{D}_{23}\mathbf{v}(k) \end{aligned} \tag{8-33}$$

The first line of the Equation 8-33 can be given either by Equation 8-9 or 8-12, representing the IO- and IIO-model, respectively. Both process models are derived from the discretization of the linearised equations of motion. The optimal control action is then determined by minimizing the performance index:

$$J(\mathbf{v}, k) = \tilde{\mathbf{z}}^T(k)\bar{\mathbf{\Gamma}}\tilde{\mathbf{z}}(k)$$

subject to the following constraints:

$$\begin{aligned} \tilde{\boldsymbol{\phi}}(k) &= \tilde{\mathbf{D}}_{33}\tilde{\mathbf{v}}(k) = \mathbf{0} \\ \tilde{\boldsymbol{\psi}}(k) &= \tilde{\mathbf{C}}_4 x(k) + \tilde{\mathbf{D}}_{43}\tilde{\mathbf{v}}(k) \leq \tilde{\mathbf{\Psi}}(k) \end{aligned}$$

Define:

$$\begin{aligned} \mathbf{A}_\psi &= \tilde{\mathbf{D}}_{43}\tilde{\mathbf{D}}_{33}^{r\perp} \\ \mathbf{b}_\psi(k) &= \left(\tilde{\mathbf{C}}_4 - \tilde{\mathbf{D}}_{43}\tilde{\mathbf{D}}_{33}^{r\perp}\mathbf{\Xi}\tilde{\mathbf{C}}_2\right)\mathbf{x}(k|k) - \tilde{\mathbf{\Psi}}(k) \\ \mathbf{H} &= 2\left(\tilde{\mathbf{D}}_{33}^{r\perp}\right)^T\tilde{\mathbf{D}}_{23}^T\bar{\mathbf{\Gamma}}\tilde{\mathbf{D}}_{23}\tilde{\mathbf{D}}_{33}^{r\perp} \\ \mathbf{F} &= \mathbf{E}_v\left(\tilde{\mathbf{D}}_{33}^{r\perp}\mathbf{\Xi}\tilde{\mathbf{C}}_2\right) \\ \mathbf{D}_\mu &= \mathbf{E}_v\tilde{\mathbf{D}}_{33}^{r\perp} \\ \mathbf{\Xi} &= \left(\left(\tilde{\mathbf{D}}_{33}^{r\perp}\right)^T\tilde{\mathbf{D}}_{23}^T\bar{\mathbf{\Gamma}}\tilde{\mathbf{D}}_{23}\tilde{\mathbf{D}}_{33}^{r\perp}\right)^{-1}\left(\tilde{\mathbf{D}}_{33}^{r\perp}\right)^T\tilde{\mathbf{D}}_{23}^T\bar{\mathbf{\Gamma}} \\ \mathbf{E}_v &= [\ \mathbf{I}\ \ \mathbf{0}\ \ \ldots\ \ \mathbf{0}\ ] \end{aligned}$$

where $\tilde{\mathbf{D}}_{33}^{r\perp}$ is the right complement of $\tilde{\mathbf{D}}_{33}$ determined by computing the singular value decomposition of $\tilde{\mathbf{D}}_{33}$ which, by construction has more columns than rows, and defining the right complement as:

$$\tilde{\mathbf{D}}_{33} = \mathbf{U}\begin{bmatrix}\mathbf{\Sigma} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{V}_1^T \\ \mathbf{V}_2^T\end{bmatrix}, \quad \tilde{\mathbf{D}}_{33}^{r\perp} = \mathbf{V}_2 \ \in \mathbb{R}^{Nm\times N_c m}$$

It is noted that the newly defined matrices have no direct physical interpretation and are simply used to simplify the notation in the remainder of this section. The optimal control law that optimizes the constrained SPCP is given by (van den Boom, 2013):

$$\mathbf{v}(k) = -\mathbf{F}\mathbf{x}(k|k) + \mathbf{D}_\mu\tilde{\boldsymbol{\mu}}_I(k) \tag{8-34}$$

where $\tilde{\boldsymbol{\mu}}_I(k)$ is the solution to the quadratic programming problem

$$\begin{aligned} &\min_{\tilde{\boldsymbol{\mu}}_I(k)} \frac{1}{2}\tilde{\boldsymbol{\mu}}_I^T(k)\mathbf{H}\tilde{\boldsymbol{\mu}}_I(k) \\ &\text{s.t.} \quad \mathbf{A}_\psi\tilde{\boldsymbol{\mu}}_I(k) + \mathbf{b}_\psi(k) \leq \mathbf{0} \end{aligned} \tag{8-35}$$

The control law of the MPC is shown in Equation 8-34. It can be seen that it is comprised of two parts. The first part, the matrix $\mathbf{F}$, serves as a gain matrix similar to those of the benchmark controller. However, the computation of $\mathbf{F}$ is based on the minimization of the performance index, subject to the control horizon constraint of Equation 8-20 for the IO-model, and Equation 8-21 for the IIO-model. The second part determines the *restoring* control action such that the inequality constraints of Equation 8-28 for the IO-model, and Equation 8-32 for the IIO-model, are always satisfied. These inequality constraints encode the maximum values of the state variables and the maximum values and rates of the control actuators. It is noted that the control law is non-linear, and cannot be expressed in terms of a linear combination of the state and input signals.

The SAD describing the flow of information required to compute the solution to the SPCP is shown in Figure 8-1. It can be seen that the computation of the solution to the SPCP is broken down into several modules. The names of these modules directly relate to their function as described in this chapter. It is noted that an identifier is used to indicate whether an IO-model or an IIO-model is given as input. This identifier tells the "Compute Control Horizon Constraint", the "LQPC to SPCP", and the "Compute Inequality Constraints" modules to take into account the differences between the two models. However, only a small number of lines of code are different between the use of the two models, and thus the architecture of the algorithm remains identical. This would not be the case if the problem was not first transformed to the SPCP format, and this example shows the power behind this methodology. Additionally, the identifier tells the "Add" module to integrate the increment actuator commands. Therefore, the MPC algorithm outputs directly the actuator commands for the MPC^{bm} and the MPC^{lqic}, and outputs the moment-fractions for the MPC^{mfc} and the MPC^{mfic}.

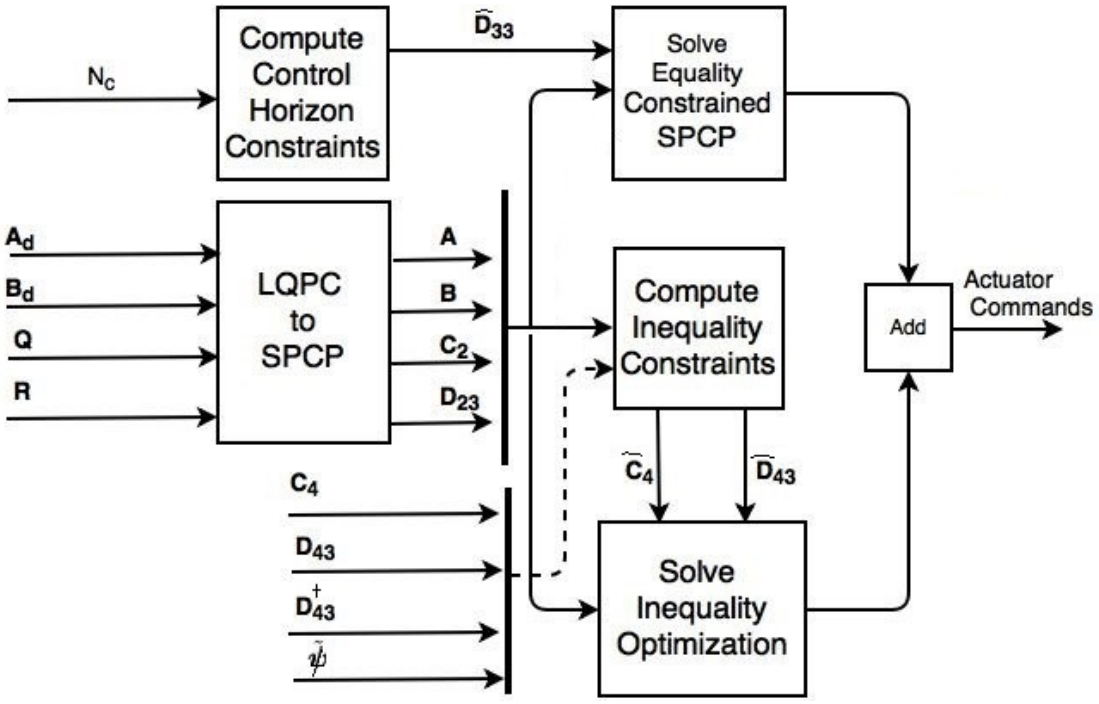


Figure 8-1: Software architecture diagram of MPC control law algorithm.

Furthermore, the "Solve Equality Constrained SPCP" module computes the matrix $\mathbf{F}$ and multiplies it with the state vector to obtain the first part of Equation 8-34. The "Solve Inequality Optimization" computes the solution to the optimization problem of Equation 8-35. It then multiplies $\tilde{\boldsymbol{\mu}}_I(k)$ with $\mathbf{D}_\mu$ to obtain the second part of Equation 8-34.

For the numerical computation of $\boldsymbol{\Xi}$, the inverse of $\left(\tilde{\mathbf{D}}_{33}^{r\perp}\right)^T \tilde{\mathbf{D}}_{23}^T \bar{\boldsymbol{\Gamma}} \tilde{\mathbf{D}}_{23} \tilde{\mathbf{D}}_{33}^{r\perp}$ must be computed, which may be badly conditioned (singular) and, hence, should be avoided. Because $\bar{\Gamma}$ was defined by Equation 8-3 to be a diagonal matrix with either ones or zeros on the diagonal, it implies that $\bar{\boldsymbol{\Gamma}} = \bar{\boldsymbol{\Gamma}}^2$. Furthermore, it is possible to replace the inversion of general matrix with the inversion of a diagonal matrix by applying a singular value decomposition on $\mathbf{M} = \bar{\boldsymbol{\Gamma}} \tilde{\mathbf{D}}_{23} \tilde{\mathbf{D}}_{33}^{r\perp} \in \mathbb{R}^{N(n+m)\times N_c m}$. Defining the singular value decomposition of $\mathbf{M}$ to be:

$$\mathbf{M} = \begin{bmatrix} \mathbf{U}_1 & \mathbf{U}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Sigma} \\ \mathbf{0} \end{bmatrix} \mathbf{V}^T = \mathbf{U}_1 \boldsymbol{\Sigma} \mathbf{V}^T \tag{8-36}$$

where $\mathbf{U}_1^T \mathbf{U}_1 = \mathbf{I}$, $\mathbf{V}$ is an orthonormal matrix, and $\boldsymbol{\Sigma}$ is a diagonal matrix containing the (real) singular values. Using the definition of the singular value decomposition as shown in Equation 8-36, one can simplify the computation of $\boldsymbol{\Xi}$ as:

$$\begin{aligned}
\boldsymbol{\Xi} &= \left(\left(\tilde{\mathbf{D}}_{33}^{r\perp}\right)^T \tilde{\mathbf{D}}_{23}^T \bar{\boldsymbol{\Gamma}} \tilde{\mathbf{D}}_{23} \tilde{\mathbf{D}}_{33}^{r\perp} \right)^{-1} \left(\tilde{\mathbf{D}}_{33}^{r\perp}\right)^T \tilde{\mathbf{D}}_{23}^T \bar{\boldsymbol{\Gamma}} \\
&= \left(\left(\bar{\boldsymbol{\Gamma}} \tilde{\mathbf{D}}_{23} \tilde{\mathbf{D}}_{33}^{r\perp}\right)^T \bar{\boldsymbol{\Gamma}} \tilde{\mathbf{D}}_{23} \tilde{\mathbf{D}}_{33}^{r\perp} \right)^{-1} \left(\bar{\boldsymbol{\Gamma}} \tilde{\mathbf{D}}_{23} \tilde{\mathbf{D}}_{33}^{r\perp}\right)^T \\
&= \left(\left(\mathbf{U}_1 \boldsymbol{\Sigma} \mathbf{V}^T\right)^T \mathbf{U}_1 \boldsymbol{\Sigma} \mathbf{V}^T \right)^{-1} \left(\mathbf{U}_1 \boldsymbol{\Sigma} \mathbf{V}^T\right)^T \\
&= \left(\mathbf{V} \boldsymbol{\Sigma} \mathbf{U}_1^T \mathbf{U}_1 \boldsymbol{\Sigma} \mathbf{V}^T\right)^{-1} \mathbf{V} \boldsymbol{\Sigma} \mathbf{U}_1^T \\
&= \left(\mathbf{V} \boldsymbol{\Sigma}^2 \mathbf{V}^T\right)^{-1} \mathbf{V} \boldsymbol{\Sigma} \mathbf{U}_1^T \\
\boldsymbol{\Xi} &= \mathbf{V} \boldsymbol{\Sigma}^{-1} \mathbf{U}_1^T
\end{aligned} \tag{8-37}$$

Equation 8-37 shows that the calculation of the inverse of a general matrix is replaced by the inversion of a diagonal matrix, which is trivial to compute. It is noted that, once $\mathbf{M}$ has been computed, one can easily compute $\mathbf{H}$ by observing that $\mathbf{H} = 2\mathbf{M}^T\mathbf{M} \in \mathbb{R}^{N_c m \times N_c m}$. This prevents one from performing unnecessary matrix multiplications, increasing the overall computational speed of the controller. Additionally, from the size of $\mathbf{H}$, one can deduce that a small control horizon reduces the size of the optimization vector $\tilde{\boldsymbol{\mu}}_I(k)$, hence speeding up the optimization algorithm (van den Boom, 2013).

It is theoretically possible that at any given time, the constraints will be violated. This might be due to unforeseen disturbances or numerical round-off errors, for example. One can introduce the concept of soft-constraints which, in addition to the standard performance index, also penalizes constraint violations (Ricker et al., 1988; Zheng and Morari, 1995). Equation 8-35 can be modified to include a *violation parameter* $\boldsymbol{\alpha}$, without changing its functional form. The resulting optimization problem is given by:

$$\begin{aligned}
&\min_{\tilde{\boldsymbol{\mu}}_I(k)} \frac{1}{2} \begin{bmatrix} \tilde{\boldsymbol{\mu}}_I^T(k) \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} \mathbf{H} & \mathbf{0} \\ \mathbf{0} & c\mathbf{I} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{\mu}}_I^T(k) \\ \boldsymbol{\alpha} \end{bmatrix} \\
&\text{s.t.} \quad \begin{bmatrix} \mathbf{A}_\psi & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{ineq} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{\mu}}_I^T(k) \\ \boldsymbol{\alpha} \end{bmatrix} + \begin{bmatrix} \mathbf{b}_\psi(k) \\ \mathbf{0} \end{bmatrix} \leq \mathbf{0}
\end{aligned} \tag{8-38}$$

where $c \gg 1$, and $\mathbf{R}_{ineq} = \operatorname{diag}\{r_1, r_2, \ldots, r_M\}$ with $r_i > 0$ for $i = 1, \ldots, N_c m$. In this way, violations of the constraints are allowed, but at the cost of a higher performance index.

It is worth mentioning that the "Solve Inequality Optimization" module of Figure 8-1 computes a solution without soft-constraints first. Only if this does not provide a solution, will the algorithm look for a solution with soft-constraints. This is done in this manner because, most of the time, a solution can be found without the use of the soft-constraint, and adding the soft-constraints adds degrees of freedom to the problem, thereby requiring more time to obtain an optimal solution. Therefore, looking for a solution without the soft-constraints will decrease the computational effort of the algorithm.

8-7 Tuning

For the LQPC-model, one can find three parameters and two matrices that need to be tuned to optimize the MPC. These parameters are the summation parameters, the minimum cost horizon, N_m, the control horizon, N_c, and the prediction horizon N, and the matrices are the state and control weighting-matrices $\mathbf{Q}$ and $\mathbf{R}$, respectively. According to Soeterboek (1992), the following initial settings are recommended for summation parameters:

$$\begin{aligned} N_m &= 1 + d \\ N_c &= n \\ N &= \text{int}(\alpha_N t_s), \text{ when well-damped} \\ N &= \text{int}(\beta_N \omega_s / \omega_b), \text{ when badly damped or unstable} \end{aligned}$$

where d is the dead-time of the system, n is the dimension of the system, $\alpha_N \in [1.5, 2]$, $\beta_N \in [4, 25]$, t_s is the 5% settling time of the discrete-time uncontrolled process, ω_s is the sampling frequency, and ω_b is the bandwidth of the of the continuous time uncontrolled process. For badly-damped or unstable systems, increasing N often increases the stability and performance of the closed loop. It is noted that these recommended settings are a good *starting* point for further tuning, with the optimal setting of the parameters being problem specific.

The state and control weighting-matrices $\mathbf{Q}$ and $\mathbf{R}$ for model-predictive controllers have a similar meaning to those found for the benchmark controllers, as found in Section 4-3 and Chapter 4-4, and may be scheduled as a function of dynamic pressure for optimal behaviour at any given time during the flight. It is noted that, while the above mentioned parameters are recommended for all problems, the effects of changing the parameters are highly problem specific. In the case that the basic rules-of-thumb are not sufficient to produce the required behaviour of the closed loop, one can always find the optimal parameters through optimization techniques (van den Boom, 2013).

8-8 Application to Attitude Control

Sections 8-1 through 8-7 discussed the theory about the inner workings of the MPC. Given the linearised state-space model of HORUS, one can derive the standard model using the methods outlined in Section 8-3. Similarly, the matrices for the equality and inequality constraints can be derived from the system constraints, as given in Section 2-2. However, once these matrices have been derived and given as input to the software, the controller is ready to be used, because the calculation of the optimal control action occurs on-line. This is different than for the NNCs and the FLCs, where optimization occurs by training the controllers. Tuning the MPC can influence its performance, however, this is a relatively simple task that does not require an optimization algorithm of its own, and thus can be done by hand.

One can design two model-predictive controllers analogous to each of the four benchmark controllers; one with an IO-model and another with an IIO-model. The MPC analogues of the

benchmark controller, MFC, LQIC, and MFIC will be distinguished from one another by a superscript. The MPC analogue of the benchmark controller will be written as MPC^{bm}, and the MPC analogues of the MFC, LQIC, and the MFIC will be written as MPC^{mfc}, MPC^{lqic}, and MPC^{mfic}, respectively. Distinction between a MPC with an IO-model and an IIO-model will be made by a subscript of the model abbreviation.

It is noted that, similar to the previously designed controllers, the model-predictive controllers will be split into a longitudinal and lateral controller. Besides the decoupled nature of longitudinal and lateral motion, their uncontrolled behaviour is also different. Longitudinal motion consists of periodic stable motion, whereas the lateral motion contains unstable periodic, and stable aperiodic behaviours. This implies that the controllers will have different values for the summation parameters. It was found experimentally that, for all longitudinal controllers, a prediction horizon of 20 steps at a sample-frequency of 20 Hz was sufficient to obtain optimal control performance. In fact, a higher value of the prediction horizon did not improve the performance. Additionally, if was found that the control horizon acted like a derivative term of a PID-controller for increasingly small values. Hence is was chosen to set the control horizon equal to the number of states of the longitudinal system. Similarly, it was found that, for all lateral controllers, a prediction horizon of 35 steps at a sample-frequency of 20 Hz was sufficient to obtain optimal control performance with a control horizon of four steps. Because the linearised state-space model of HORUS does not contain dead-time, it implies that the minimum-cost horizon can be set equal to one, which in turn implies that $\bar{\mathbf{\Gamma}}$ reduces to an identity matrix. The summation parameter choices are summarized in Table 8-1.

Table 8-1: Overview of MPC summation parameter choices.

	Longitudinal	**Lateral**
N_m	1	1
N_c	2	4
N	20	35

Additionally, constraints are put on the rotational rates and on the sideslip angle. No constraints are put on the angle of attack and the bank angle errors, because those are the steering variables. The longitudinal state-signal selection matrices, required for Equations 8-25 and 8-29, are given by:

$$(\mathbf{S_x})_{lon}^{BM/MFC} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad (\mathbf{S_x})_{lon}^{LQIC/MFIC} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{8-39}$$

The lateral state-signal selection matrices are given by:

$$(\mathbf{S_x})_{lat}^{BM/MFC} = \begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{0}_{1x1} \\ \mathbf{0}_{1x3} & \mathbf{0}_{1x1} \end{bmatrix}, \quad (\mathbf{S_x})_{lat}^{LQIC/MFIC} = \begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & \mathbf{0}_{3x3} \end{bmatrix} \tag{8-40}$$

Furthermore, constraints are put on the maximum control surface deflections and thruster moments, and on the maximum control surface deflection rates. For all MPCs, the input-signal selection matrix $\mathbf{S_u}$ is given by:

$$\mathbf{S_u} = \mathbf{I}_{m \times m} \tag{8-41}$$

where m is the number of actuators that are active for the MPC^{bm} and MPC^{lqic}, and $m = 1$ and $m = 3$ for the longitudinal and lateral parts, respectively, of the MPC^{mfc} and the MPC^{mfic}. The

increment input-signal selection matrices for the MPC$^{\text{bm}}$ and the MPC$^{\text{lqic}}$ are dependent on the number of actuators active. The longitudinal and lateral increment input-signal selection matrices are, respectively, given by:

$$(\mathbf{S}_{\delta\mathbf{u}})_{lon}^{BM/LQIC} = \begin{cases} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} & \text{if } 100 \leq \bar{q} < 1000 \\ 1 & \text{if } 0 \leq \bar{q} < 100 \text{ or } \bar{q} \geq 1000 \end{cases} \tag{8-42}$$

$$(\mathbf{S}_{\delta\mathbf{u}})_{lat}^{BM/LQIC} = \begin{cases} \mathbf{0}_{2x2} & \text{if } 0 \leq \bar{q} < 100 \\ \begin{bmatrix} 1 & \mathbf{0}_{1x2} \\ \mathbf{0}_{2x2} & \mathbf{0}_{2x2} \end{bmatrix} & \text{if } 100 \leq \bar{q} < 150 \\ \begin{bmatrix} \mathbf{I}_{2x2} & \mathbf{0}_{2x2} \\ \mathbf{0}_{2x2} & \mathbf{0}_{2x2} \end{bmatrix} & \text{if } 150 \leq \bar{q} < 500 \\ \begin{bmatrix} \mathbf{I}_{2x2} & \mathbf{0}_{2x1} \\ \mathbf{0}_{1x2} & \mathbf{0}_{1x2} \end{bmatrix} & \text{if } \bar{q} \geq 500 \end{cases} \tag{8-43}$$

Because the MPC$^{\text{mfc}}$ and the MPC$^{\text{mfic}}$ output moment-fractions, the increment input-signal selection matrices do not depend on the dynamic pressure and are given by:

$$(\mathbf{S}_{\delta\mathbf{u}})_{lon}^{MFC/MFIC} = 1, \quad (\mathbf{S}_{\delta\mathbf{u}})_{lat}^{MFC/MFIC} = \mathbf{I}_{2\times 2} \tag{8-44}$$

The selection matrices are used to construct the matrices $\mathbf{C}_4$, $\mathbf{D}_{43}$, and $\mathbf{D}_{43}^{\dagger}$. These matrices are direct inputs to the MPC algorithm. Furthermore, the maximum control surface deflections and thrusters moments are set equal to maximum values as shown in Section 2-2-2, Table 2-5. The thrusters have an infinite control rate, whereas the control surfaces have a control rate of 15 deg/s. The maximum value of the moment-fractions are naturally constrained to be equal to one, whereas their maximum rate are set equal to 0.1. This value corresponds to the maximum moment-fraction change that would be possible with the rate constraints of the control actuators. The constraint parameters are summarized in Table 8-2, and are used to construct the constraint vector $\boldsymbol{\psi}$, which is a direct input to the MPC algorithm.

Table 8-2: Overview of MPC constraint parameter values.

Parameter	Value	Parameter	Value
p [deg/s]	15	η [-]	1
q [deg/s]	5	$\dot{\eta}$ [1/s]	0.1
r [deg/s]	15	$\dot{\delta}$ [deg/s]	15
β [deg]	1	$\dot{T}$ [Nm/s]	∞

Instead of defining the parameters of the weight matrices when certain actuators are active, one can also define these parameters at certain locations along the reference trajectory, and schedule them as a function of dynamic pressure, while still keeping track of when which actuators are active. Because the same actuators can be active over a wide range of dynamic pressures, having the same parameter settings for that entire range means one has to compromise and design a controller that works for the entire range of dynamic pressures. Actively scheduling the parameters allows the designer to optimize the controller for many points along the reference trajectory, which improves overall performance. This is possible within MPC, because the control system computes the control action for each time-step. For each time-step, the controller will take the weighting-matrices as

inputs, and thus they too can change between time-steps. Hence, actively scheduling the weighting-matrices as a function of dynamic pressure is possible and will improve the performance of the overall controller and therefore it is chosen to use this approach, instead of using the weighting-matrices scheduling methodology that was used for the benchmark controllers.

For both process models, the MPC analogues of the benchmark controller, the MFC, the LQIC, and the MFIC, the state and control weighting-matrices, $\mathbf{Q}$ and $\mathbf{R}$ respectively, are defined identical to the controllers they are analogous to. The parameter values of the IO-model MPCs for all of the study-cases, as presented in Section 2-2-1, are shown in Tables B-11, B-13, B-15, and B-17 for the $\text{MPC}_{IO}^{\text{bm}}$, $\text{MPC}_{IO}^{\text{mfc}}$, $\text{MPC}_{IO}^{\text{lqic}}$, and $\text{MPC}_{IO}^{\text{mfic}}$, respectively. Similarly, for the IIO-model MPCs, the parameters values for all the study-cases are shown in Tables B-12, B-14, B-16, and B-18 for the $\text{MPC}_{IIO}^{\text{bm}}$, $\text{MPC}_{IIO}^{\text{mfc}}$, $\text{MPC}_{IIO}^{\text{lqic}}$, and $\text{MPC}_{IIO}^{\text{mfic}}$, respectively. It is noted that the definition of the control weighting parameters for the IIO-model are related to the increment control actuators.

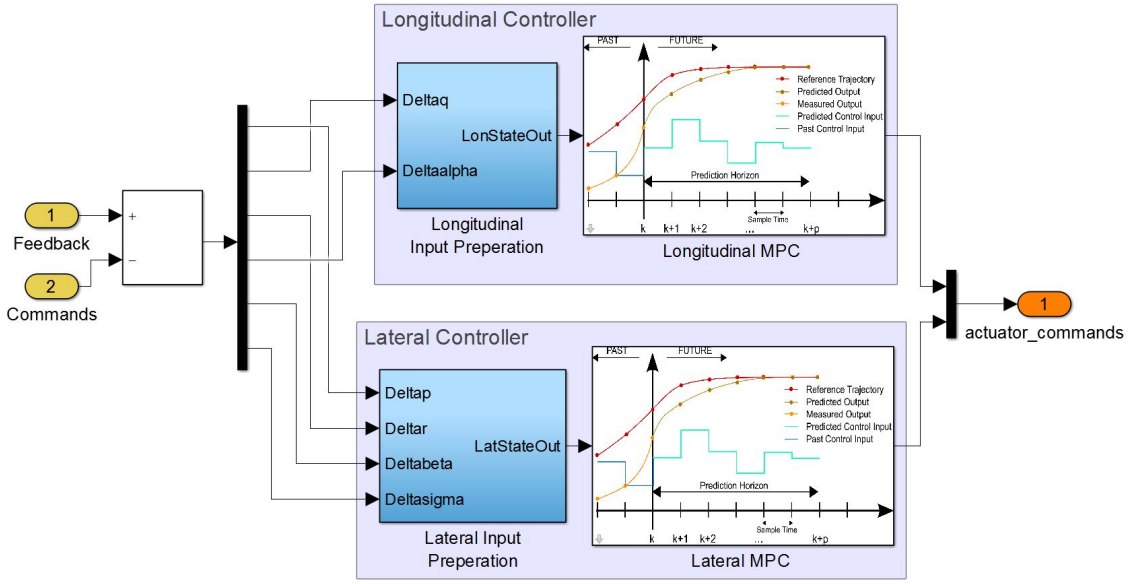


Figure 8-2: Software architecture of a general MPC.

Figure 8-2 shows the SAD of the general implementation of a MPC. The details of the "Input Preparation" block differs for the four different types of MPCs. For the MPC^{lqic} and the MPC^{mfic} the inputs are modified to include to integral terms, whereas for the MPC^{bm} and the MPC^{mfc} the inputs are left unchanged. The raw control outputs are then fed into an actuator allocation algorithm. For the MPC^{bm} and the MPC^{lqic} the raw control outputs are the elevator, aileron, and double-sided rudder deflections, which need to be transformed into the two elevon and rudder deflections by means of Equation 2-1. For the MPC^{mfc} and the MPC^{mfic} the raw control outputs are the moment-fractions, which need to be transformed to the elevon and rudder deflections with the actuator allocation algorithm of Chapter 5. It is noted that, at the architecture level shown in Figure 8-2, no difference between the IO- and IIO-model is apparent.

The MPC methodology discussed in this chapter was numerically implemented according to Figure 8-1 and, hence, needs to be verified for correct implementation. Section 9-2-3 will present the verification of the complete MPC algorithm.

Chapter 9

Software Architecture and Verification

To analyse the performance of the control systems discussed in the previous chapters, simulations must be performed. For this task, one requires the use of a numerical environment which simulates the motion of the reference vehicle within its environment. The TU Delft Astrodynamics Toolbox (TUDAT)[1], a set of C++ software libraries, is commonly used as a numerical environment to simulate various astrodynamics applications. However, MATLAB/Simulink is the most common language used for designing control systems due to the availability of a large number of toolboxes related to control systems design. Additionally, the Generic Guidance, Navigation, and Control Simulator (GGNCSim) library, developed by Dutch Space and described by Mooij and Wijnands (2002) and Mooij and Ellenbroek (2011), and made for Simulink, is available. Therefore, the numerical simulation environment will be build within Simulink. Section 9-1 will present the software architecture of the simulation environment.

When custom software is written, or different software packages are integrated together, one must ensure that the software behaves as intended. Hence, one must *verify* the custom-made software. The GGNCSim library contains verified building blocks and hence it was preferred to use as many of the components within this library to simplify the verification process. The verification process is discussed and presented in Section 9-2.

9-1 Software Architecture

SADs represent the flow of information and data between the different *modules* of the software. Each module has an *interface* that describes the inputs and outputs of the specific module, which allows a designer to easily understand and replace a module. Additionally, because a module can be thought of as a black-box that performs a specific task, it allows for abstraction of the software at the highest level. Furthermore, modularity of the simulation environment allows one to re-use as much as possible of the software, which decreases design and verification effort.

Extensive use is made of modular components for the simulation environment of the reference vehicle and its environment. The top-level SAD is shown in Figure 9-1. The simulation environment is constructed from five different modules. The "MissionEventManager" module sends the

[1] Delft University of Technology, TUDAT Overview Page, http://tudat.tudelft.nl/projects/tudat [Retrieved 21-12-2106], 2016

mission specific commands to the "Trim_And_GuidanceSystem" module. From this module the attitude and trim commands are fed into the "Horus_Controller" module, which determines a set of actuator commands. The "VehicleSimulator" module then executes the actuator commands and computes the new states of the reference vehicle. These vehicle states are then transformed by the "NavigationSystem" module into a different state vector which is used by the previous modules to compute the appropriate actions.

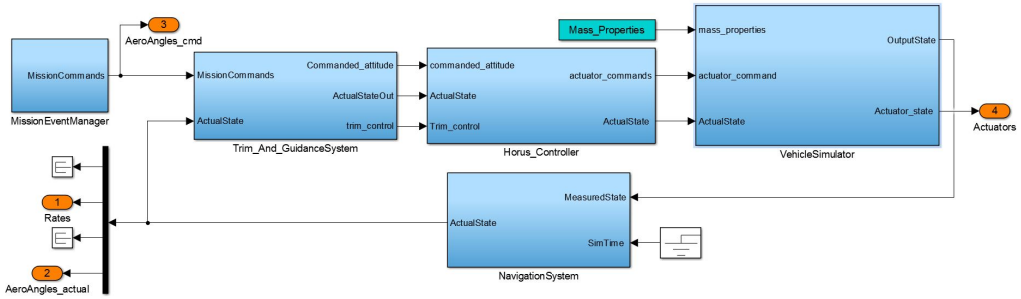


Figure 9-1: Top-level architecture of simulator.

The different modules of the simulation environment will be presented in the remainder of this section. Sections 9-1-1, 9-1-2, 9-1-3, and 9-1-4 will discuss the SADs of the vehicle simulator, guidance and trim, navigation, and controller modules, respectively. The "MissionEventManager" module will not be discussed in a separate section, because, in this report, it only outputs a set of commanded aerodynamic angles, which are directly imported from the MATLAB workspace.

9-1-1 Vehicle Simulator

The "VehicleSimulator" module computes the forces and moments that act on the vehicle, and updates the state of the vehicle for a new time-step. The SAD of the vehicle simulator is shown in Figure 9-2. The forces that act on the vehicle are the gravitational and aerodynamic forces. It is noted that, when simulating rotational motion only, the forces are not taken into account, as they act on the CoM of the vehicle. The moments that act on the vehicle about the CoM are the aerodynamic moments. Furthermore, because the forces are computed w.r.t. the B-frame, they must be transformed to the I-frame, which is done using the quaternion encoding the orientation of the B-frame w.r.t. the I-frame, which is part of the state vector outputted by the "equations of motion and integrator" module.

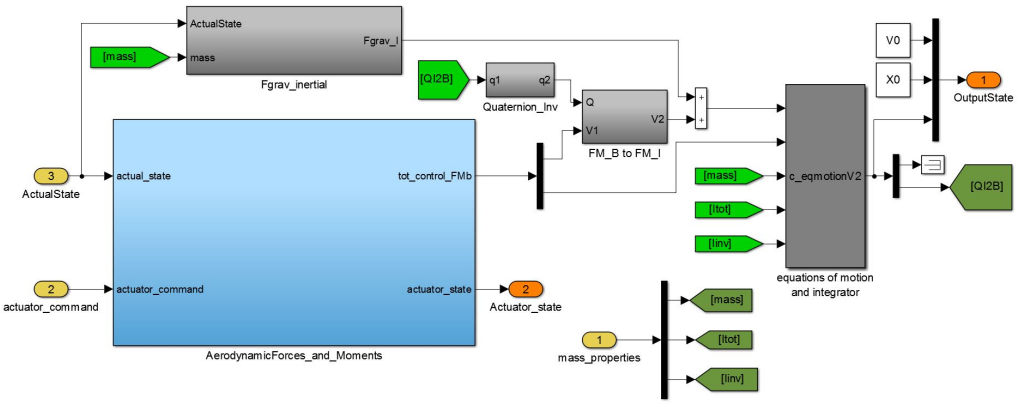


Figure 9-2: Software architecture of the vehicle simulator.

Figure 9-3 shows the "AerodynamicForces_and_Moments" module of Figure 9-2. The computation of the forces and moments that act on the vehicle are split into the contributions of the surface deflections and the contributions due to the state of the vehicle. Additionally, the "Actuators" module takes the rate limitations of the actuators into account. It is noted that the vehicle simulator is part of the GGNCSim library and, hence, does not require verification.

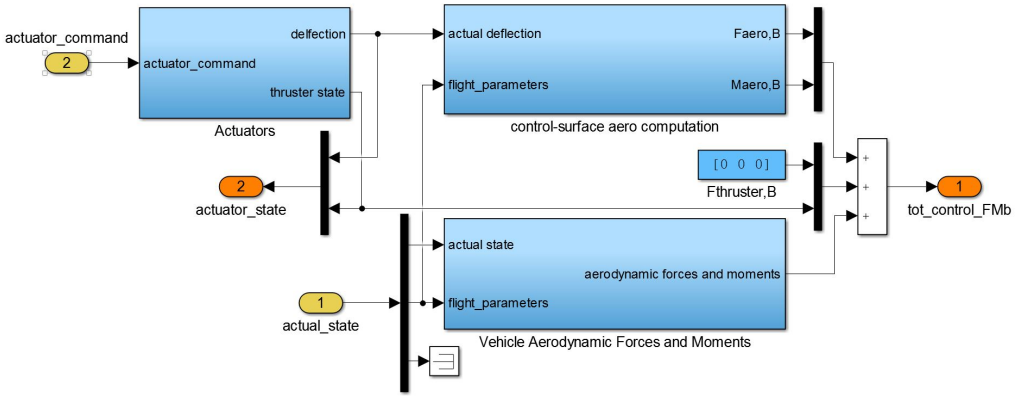


Figure 9-3: Software architecture of the aerodynamic forces and moments calculations.

9-1-2 Guidance and Trim

The "Trim_And_GuidanceSystem" module transforms the missions commands and vehicle state to a set of commanded attitudes, and computes the body-flap and elevon deflections required to maintain trim stability. However, because in this report only rotational motion about the CoM of the reference vehicle is considered, no guidance system has been implemented. The SAD of the "Trim_And_GuidanceSystem" module is shown in Figure 9-4. It is noted that this module is part of the GGNCSim library and, hence, requires no verification.

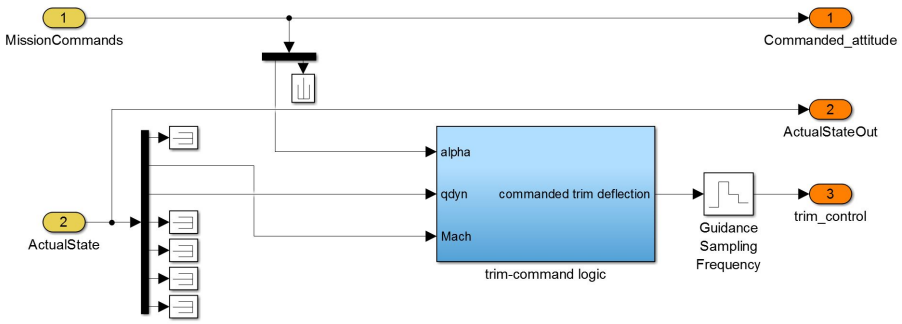


Figure 9-4: Software architecture of the guidance and trim subsystem

9-1-3 Navigation

In this report, no sensor measurements are considered and, hence, the state of the vehicle as outputted by the "VehicleSimulator" module is identical to the measures state. If one wants to add models of sensors to simulate a more realistic scenario, the "NavigationSystem" module is the location to add these systems. Furthermore, the state outputted by the vehicle simulator encoded the rotational orientation of the vehicle as a quaternion, which must be transform to a set of aerodynamic angles, which are used by the guidance, trim, and control systems. Additionally, the

navigation system module is used as a central place to compute derived quantities of the states (dynamic pressure, Mach Number, altitude, etc.) to prevent computing the same quantities in different locations, which increases the chances for errors to occur. The SAD of the "Navigation-System" module is shown in Figure 9-5. It is noted that this is the location within the navigation system to account for wind in the atmosphere. This report, however, did not consider this and hence the wind velocities are set equal to zero.

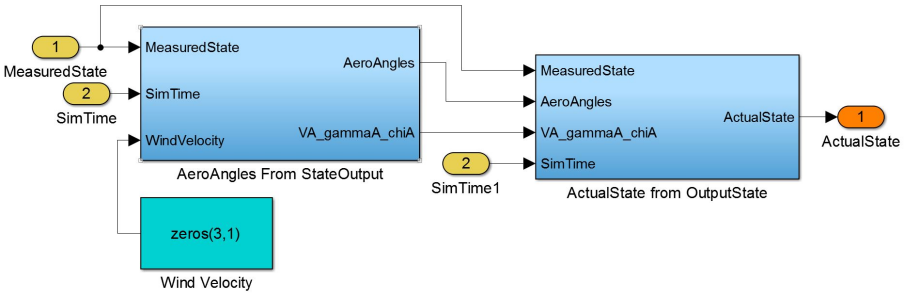


Figure 9-5: Software architecture of the navigation subsystem

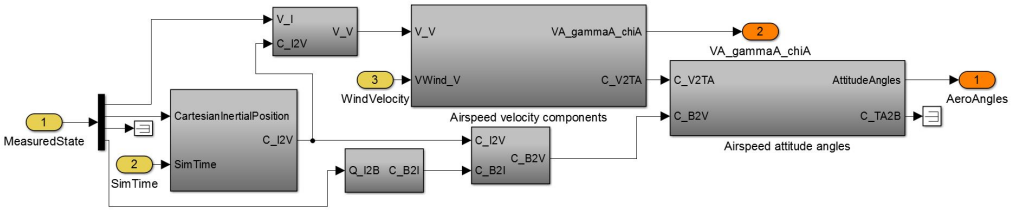


Figure 9-6: Software architecture of the aerodynamic angle computation subsystem within the navigation system

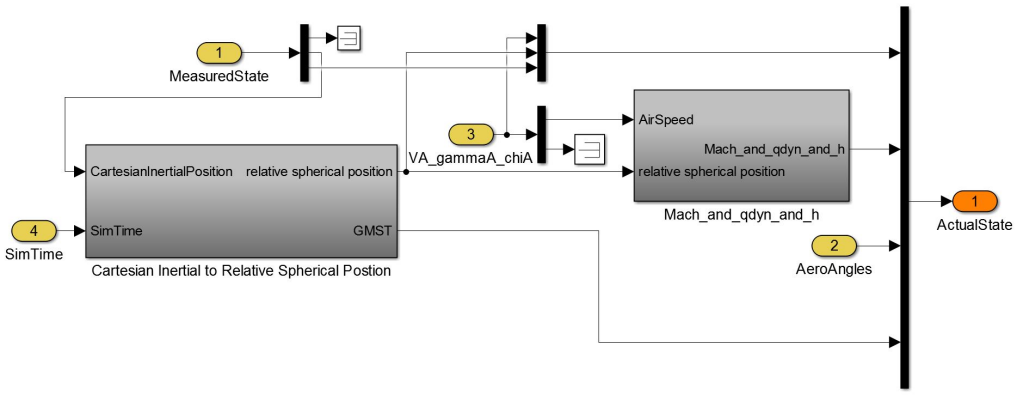


Figure 9-7: Software architecture of the actual state extraction subsystem within the navigation system

The "NavigationSystem" module is split into two subsystems. The first subsystem is the "AeroAngles From StateOutput" module and computes the aerodynamic angles from the attitude quaternion, and the SAD of this module is shown in Figure 9-6. The second subsystem is the "ActualState From OutputState" module which computes the derived state quantities and concatenates the aerodynamic angles, Mach number, dynamic pressure, altitude, relative spherical position,

aerodynamic spherical velocity, and the Greenwich Mean Sidereal Time (GMST) into a new "ActualState" state vector. The SAD of this subsystem is shown in Figure 9-7. It is noted that the "NavigationSystem" module is fully constructed from components found in the GGNCSim library and, hence, no components require verification. An integration verification test will, however, be performed in Section 9-2-1.

9-1-4 Control Laws

Similar to the previous architectures, the control subsystem is also build in modular components, as shown in Figure 9-8. The "Input Preparation" block extracts the commanded and the vehicle's rotational states from the inputs. The "Controller" block contains the actual control laws, and outputs a set of actuator commands. However, these actuator commands need to be transformed into a set of surface deflections for the elevons and rudders, and a set of thruster moments. The implementation of this "ActuatorAssignment" block is depended on the chosen controller. Two cases can be identified. The first case is used when the controller outputs aileron, elevator, and rudder deflections, and a set of thruster moments. In this case, the aileron and elevator deflections must be combined to obtain two elevon deflections, for which Equation 2-1 can be used. The second case is used when the controller outputs a set of moment-fractions. In this case, the actuator assignment algorithm of Chapter 5 is used to obtain a the elevon and rudder deflections, and the thruster moments. This actuator assignment algorithm will be verified in Section 9-2-2.

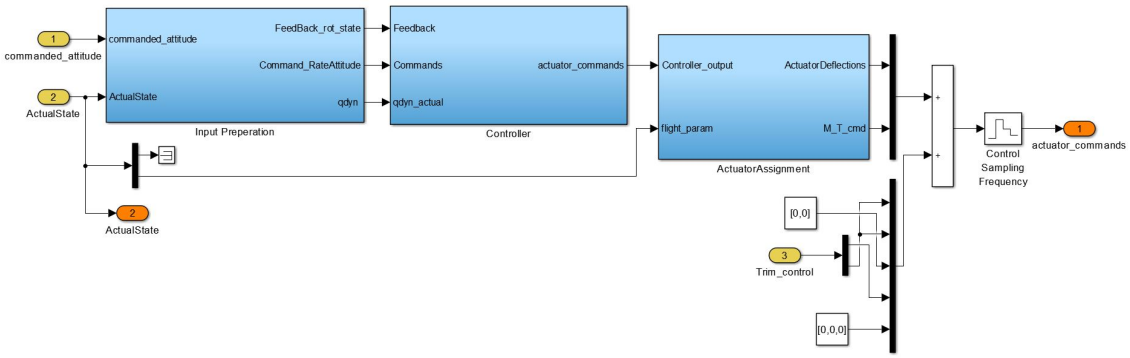


Figure 9-8: General software architecture for the control algorithm subsystem.

After the set of surface deflections and moment thrusters have been computed by the controller and the actuator assignment algorithm, the values of the surface deflections needed to maintain trim stability are added to these actuator commands. It is noted that the set of surface deflections also includes the deflection angle of the body-flap, which is primarily used to maintain trim stability. It is, however, possible that the elevons need to assist the body-flap.

The first controller discussed in the report was the benchmark controller, for which the SAD were shown in Figure 4-2. The SADs for the modified benchmark controllers have a similar structure, albeit with a combination of different inputs and outputs. For all benchmark controllers, the computation of the actuator commands from the scheduled gains is a trivial matrix multiplication and, hence, will not be discussed in Section 9-2.

For the NNCs, it is recalled that the identification of recurrent connections and assigning layers to the nodes is done off-line, and the information is stored within the numerical structure of the network. This implies that, the process of computing an output within the simulator of a general network is presented in Section 6-3-3. Furthermore, the NEAT optimization algorithm was obtained directly from the website of Stanley[2]. The algorithm and its SAD are discussed in

[2]K. Stanley, The NeuroEvolution of Augmenting Topologies (NEAT) Users Page, http://www.cs.ucf.edu/~kstanley/neat.html [Retrieved 19-10-2016], 2016

Section 6-2. It is noted that the decoding of the neural network structure is completely contained within a single CMEX[3] S-function[4] to improve performance over MATLAB scripts. The SAD describing the implementation of the "Controller" block for the NNCs is shown in Figure 6-16. While computing the output of a network is a relatively simple computation, it is crucial that it behaves as intended and, hence, its verification process will be discussed in Section 9-2-3.

Furthermore, for the FLCs, the membership functions and rulebases can be defined and constructed using build-in commands within MATLAB's fuzzy logic toolbox, stored as a text file (a so-called .fis file), and the output of the FLCs can be computed using a build-in Simulink block that is also part of the fuzzy logic toolbox. Likewise, the pattern-search algorithm used to optimize the membership functions of the FLCs is a build-in function of MATLAB. The SAD describing the implementation of the "Controller" block for the FLCs is shown in Figure 7-5. Because the design process of the FLCs only involves build-in components of MATLAB/Simulink, there is no need to verify them.

The last type of controller that was discussed, were the MPCs. The MPC algorithm to compute the optimal control action was discussed in Section 8-6 and the software implementation was custom-made. The SAD describing the implementation of the "Controller" block for the MPCs is shown in Figure 8-2. The complete MPC algorithm will be verified in Section 9-2-3.

9-2 Verification

To verify that the custom made software behaves as intended, one must execute a verification process to increase the confidence of the correct implementation of the software. Unfortunately, it is beyond the scope of this thesis to report on the verification of every aspect and component of every algorithm within the simulation environment. It is sufficient to say that such unit-tests have been performed, and it was concluded that all these components have been implemented correctly. However, integration tests of the major modules of the top-level simulation environment do need be completed to increase the confidence that all these modules have been implemented and integrated correctly.

In the previous section, it was identified which modules required integration tests and, additionally, which control law algorithms needed further verification. To this end, this section has been divided into three different subsections. Subsection 9-2-1 will verify for the correct integration of the navigation system. It is very important that the calculation of the aerodynamic angles from the attitude quaternion works as intended, as any error will propagate to the other modules. Hence, the attention of this subsection will be focused on the computation of the aerodynamic angles from the attitude quaternion. Subsection 9-2-2 will verify the correct implementation of the actuator assignment algorithm of Chapter 5. The correct implementation of this algorithm is of great importance, because errors between commanded moment-fractions and the corresponding actuator deflections will lead to oscillatory or unstable behaviour of the reference vehicle. Subsection 9-2-3 concludes this section with the integration tests of the control law algorithm that required further verification, as was established by the previous section.

9-2-1 Navigation System

The computation of the aerodynamic angles from the attitude quaternion relies on the combination of the ability of the quaternion to be transformed to an rotation matrix, and Equation 9-1. Using a fixed set of flight parameters defining $\mathbf{C_{T,V}}$ and $\mathbf{C_{V,I}}$, one can compute $\mathbf{C_{T,B}}$ from $\mathbf{C_{I,B}}$, and vice versa (Mooij, 1997). It is noted that the GGNCSim library contains a block for transforming

[3] A CMEX file allows for calling functions written in C/C++ by MATLAB.

[4] An S-function is a computer language description of a Simulink block. It allows for creating Simulink blocks with custom functionalities from the command line.

from the quaternion $\mathbf{Q_{I,B}}$ to $\mathbf{C_{I,B}}$ and vice versa. Additionally, MATLAB's aerospace toolbox contains functions for the same functionality.

$$\mathbf{C_{T,B}} = \mathbf{C_x}(\sigma)\mathbf{C_z}(\beta)\mathbf{C_y}(-\alpha) = \mathbf{C_{T,V}}\mathbf{C_{V,I}}\mathbf{C_{I,B}} \tag{9-1}$$

The verification process involves computing the aerodynamic angles from the quaternion with the simulator module, and an independent MATLAB function performing the opposite task. If one chains the output of this function to the input of the simulator module, then the input to the MATLAB function should be identical to the output of the simulator module. Because the simulator module and MATLAB function are constructed from independent components, the confidence that the simulator module works as intended is increased. Table 9-1 gives the flight parameters used for the verification process, and Table 9-2 gives the comparison of the inputs and outputs the MATLAB function and the simulator module.

From Table 9-2 one can see that the Simulink module converts a quaternion back to the aerodynamic angles that was used to create the quaternion. It is noted that the Simulink module is accurate up to the floating-point error of the numerical environment.

Table 9-1: Flight parameters for the verification process of the navigation system.

Parameter	**Value**
GMST [rad]	4.8288
$\mathbf{V}_I$ [m/s]	(585.8915; -506.5924; -195.6546)
$\mathbf{X}_I$ [10^6·m]	(-4.6853; -4.3329; 0.5538)

Table 9-2: Input and Output data of the MATLAB function for the verification process of the navigation system.

Input MATLAB Function [deg]	**Output MATLAB Function & Input Simulator Module** [-]	**Output Simulator Module** [deg]
(0; 0; 0)	(-0.6466; 0.3312; -0.1537; 0.6698)	(-6.3611; 2.5110; -6.8980)·10^{-15}
(10; 10; 10)	(-0.7115; 0.3318; -0.2399; 0.5710)	(10; 10; 10)
(50; 30; 100)	(-0.9609; 0.0621; -0.2645; -0.0537)	(50; 30; 100)
(0; 0; 179)	(0.6754; -0.1566; -0.3298 ;0.6407)	(1.1132·10^{-14}; -6.2120·10^{-15}; 179)
(0; 0; -179)	(0.6642; -0.1508; -0.3325; 0.6524)	(1.1132·10^{-14}; -2.9818·10^{-16}; -179)

The "ActualState from OutputState" module within the navigation system performs a set of trivial calculations, hence, it is sufficient to say that it has been verified for correct implementation. Therefore, one can conclude that the "NavigationSystem" module has been successfully verified.

9-2-2 Actuator Assignment

To verify the actuator assignment algorithm of Chapter 5, one can input a set of moment-fractions and flight conditions, compute the corresponding actuator commands, and use those to compute the resulting moments using the "control-surface aero computation" module of Figure 9-3. The maximum moments corresponding to the set of flight conditions can be computed using the methods of Section 5-1. This procedure allows one to verify that one computes exactly what is desired. The flight parameters used in the verification process are given in Table 9-3, and Table 9-3 gives the comparison between the input and output moment-fractions.

Table 9-3: Flight parameters for the verification process of the navigation system.

Parameter	Value
Mach Number [-]	18
Dynamic Pressure [Pa]	5500
Angle of Attack [deg]	30
Sideslip angle [deg]	0

Table 9-4: Overview of input and output moment-fractions for the verification process of the actuator assignment algorithm of Chapter 5.

Input	Output
(0; 0; 0)	(0; 0; 0)
(0.2; 0; 0)	(0.2; 0; 0)
(0.2449; 0.2280; 0.3232)	(0.2449; 0.2280; 0.3232)
(0.1071; 0.4485; 0.8638)	(0.1071; 0.4485; 0.8638)
(0.6117; 0.5895; 0.1464)	(0.3895; 0.5000; 0.1464)
(-0.5117; -0.4895; -0.1464)	(-0.5117; -0.4895; 0.3256)

From Table 9-4 it can be seen that all but the last of the examples have matching input and output values. In the second-to-last scenario, the moment-fractions of the roll- and pitch-axis are chosen such that the corresponding commanded roll- and pitch-moment coefficients cannot be satisfied simultaneously. Figure 9-9 shows the graphical representation of the problem. Additionally, the last scenario depicts a situation where the elevon deflections corresponding with the moment-fractions of the roll- and pitch-axis induce a yaw moment which the combined effort of the rudder and yaw thrusters cannot compensate for.

From Figure 9-9 it can be seen that there exist no solution were the two subtracted surfaces intersect each other and the XY-plane in a single location. Hence, it is expected that the input and output moment-fraction will not match. In this scenario, the left and right elevons are chosen such that the combined errors of the subtracted surfaces are minimized. It is confirmed that this is indeed the case. It is noted that, when all moment-fractions can be realized, the solution is accurate up to floating-point errors.

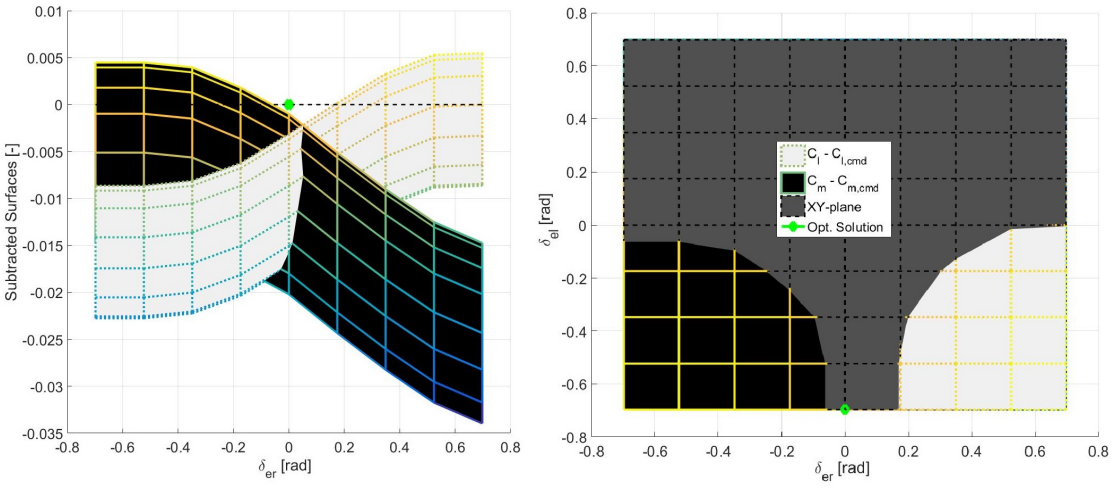


Figure 9-9: Graphical representation of the last verification example for the actuator assignment example. The left figure shows the two-dimensional view perpendicular to the XY-plane. The right figure shows the same figure rotated 90 degrees about the δ_{er}-axis.

From the above paragraphs, it can be concluded that actuator assignment algorithm works as intended, with the deviations between the inputs and outputs fully explainable based on the physical nature of the system. Hence, it can be concluded that the actuator assignment algorithm of Chapter 5 has been successfully verified.

9-2-3 Control law

In Section 9-1-4 it was identified that the output computation of a general neural network and the MPC CMEX S-functions required further verification. For both controllers, the verification process will be a check between an independent MATLAB code and the corresponding Simulink blocks. If both solutions are equal, then one has increased confidence of correct implementation. First, the output computation of general neural network will be discussed, after which the verified process of the MPC will be presented.

Neural Network Controller

To verify that the CMEX S-function correctly computes the output of a general network, one must test the algorithm with networks of different structural architecture. Each network will have two inputs, one output, and all connection weights are set to unity. The first network will be the trivial network; no hidden nodes and connections from the input nodes directly to the output node. The second network will contain an additional hidden node which is connected from both input nodes and connects to the output node. The last network structure that will be tested is identical to that of the second network, but has an additional *recurrent* connection from the output node to the hidden node. All three networks are shown graphically in Figure 9-10.

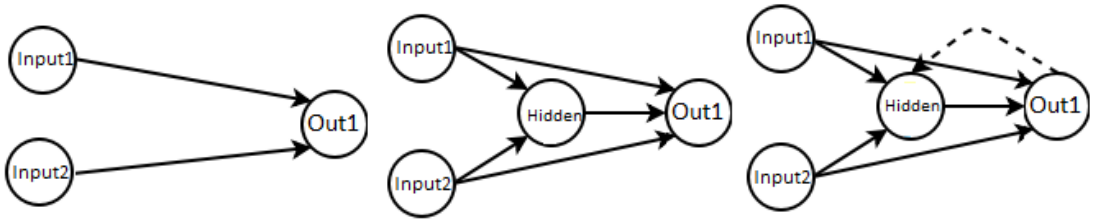


Figure 9-10: Graphical representation of the three networks that are used for the verification process of the output computation of a general network. The dashed connection represents a recurrent connection. From left to right, they are referred to as the trivial, hidden, and recurrent networks, respectively.

It is expected that the first two network always give the same answer for the same input. However, the third network, due to the recurrent connection, "remembers" its previous states and, hence, the output will have a different value for each time the network is evaluated, given a fixed input. This feature is an essential property of recurrent networks. and, therefore, it is invalid to compute the output of the recurrent network for only one time-step.

Table 9-5: Comparison of CMEX S-function and independent MATLAB code outputs of for the trivial and hidden networks.

	Input: [0; 0]	
Network	**CMEX S-function**	**MATLAB code**
Trivial	0	0
Hidden	0	0
	Input: [0.03211; 0.00691]	
Network	**CMEX S-function**	**MATLAB code**
Trivial	0.0953	0.0953
Hidden	0.3177	0.3177

Table 9-6: Comparison of CMEX S-function and independent MATLAB code outputs of the recurrent network.

	Input: [0; 0]		
Network	**Time-step 1**	**Time-step 2**	**Time-step 3**
CMEX S-function	0	0	0
MATLAB code	0	0	0
	Input: [0.03211; 0.00691]		
Network	**Time-step 1**	**Time-step 2**	**Time-step 3**
CMEX S-function	0.3177	0.9487	0.9868
MATLAB code	0.3177	0.9487	0.9868

From Tables 9-5 and 9-6 one can see that the CMEX S-function outputs are identical to that of the independent MATLAB code. Further inspection of the data reveals that the outputs are identical up to the floating-point error of the numerical environment. Additional tests with different inputs and connection weights have been performed, however, the conclusion remains unchanged. Therefore, it is decided not to include these tests in this report, for the sake of brevity. It is interesting to note that, the output of the first time-step of the recurrent network is identical to the (constant) output of the hidden network. This is due to the fact that, in the first time-step, the recurrent connection has no memories yet, which therefore does not contribute to the output of the network. Because the recurrent network is identical to the hidden network up to a recurrent connection, it is not unexpected that the two outputs are identical.

From the above paragraphs, it can be concluded that the computation of the output of a general network is implemented as intended and hence, one can conclude that the CMEX S-function performing these calculations has been successfully verified.

Model Predictive Controller

Similar to the verification process of the neural network decoder, the MPC algorithm will be verified by comparison with an independent MATLAB code. However, instead of comparing the outputs at different moments in time, it provides more insight to compare the responses of a system when it is controlled by the different implementations of the same MPC.

The to-be-controlled system is defined as a discrete state-space IO-model, with the state and input matrices as shown in Equation 9-2. The output of the model is set equal to its state. The state and control weighting-matrices are given by Equation 9-3.

$$\mathbf{A}_d = \begin{bmatrix} 0.9996 & -0.0177 \\ 0.050 & 0.9995 \end{bmatrix}, \qquad \mathbf{B}_d = \begin{bmatrix} -0.0344 \\ -0.0009 \end{bmatrix} \tag{9-2}$$

$$\mathbf{Q} = \operatorname{diag}\{205.1754; 3, 282.8064\}, \quad \mathbf{R} = 2.0158 \tag{9-3}$$

Furthermore, the inequality constraint matrices are given by:

$$\mathbf{C}_4 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ -1 & 0 \\ 0 & 0 \\ \mathbf{0}_{4\times1} & \mathbf{0}_{4\times1} \end{bmatrix}, \quad \mathbf{D}_{43} = \begin{bmatrix} \mathbf{0}_{4\times1} \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix}, \quad \mathbf{D}_{43}^{\dagger} = \begin{bmatrix} \mathbf{0}_{6\times1} \\ 1 \\ -1 \end{bmatrix}, \quad \boldsymbol{\phi} = \begin{bmatrix} 5^\circ \\ 0 \\ 5^\circ \\ 0 \\ 40^\circ \\ 40^\circ \\ 15^\circ \\ 15^\circ \end{bmatrix} \tag{9-4}$$

Additionally, the prediction horizon, N, is set equal to 20 time-steps, the control horizon, N_c, is set equal to 2 time-steps, the sample frequency is set equal to 20 Hz, the constraint violation cost, c, is set equal to one thousand and all diagonal values of $\mathbf{R}_{ineq}$ are set equal to 0.01.

Figure 9-11 shows the system responses for both the CMEX S-function and MATLAB implementation of the MPC. It shows that one cannot distinguish between the two implementations with a purely graphical representation. Further investigation of the raw data shows that both implementations provide system responses that, up to a floating-point error, are identical. Furthermore, if the discrete state-space model is interpreted as an IIO-model, the responses of the system of both implementations are similarly identical. However, when one compares the responses between the IO- and IIO-models with the same implementation, they will not be the same. This difference is expected, because the definition of the control weighting-matrix has changed. Although the different models produce different responses, the difference is small for the system tested here, and cannot be seen by a graphical representation.

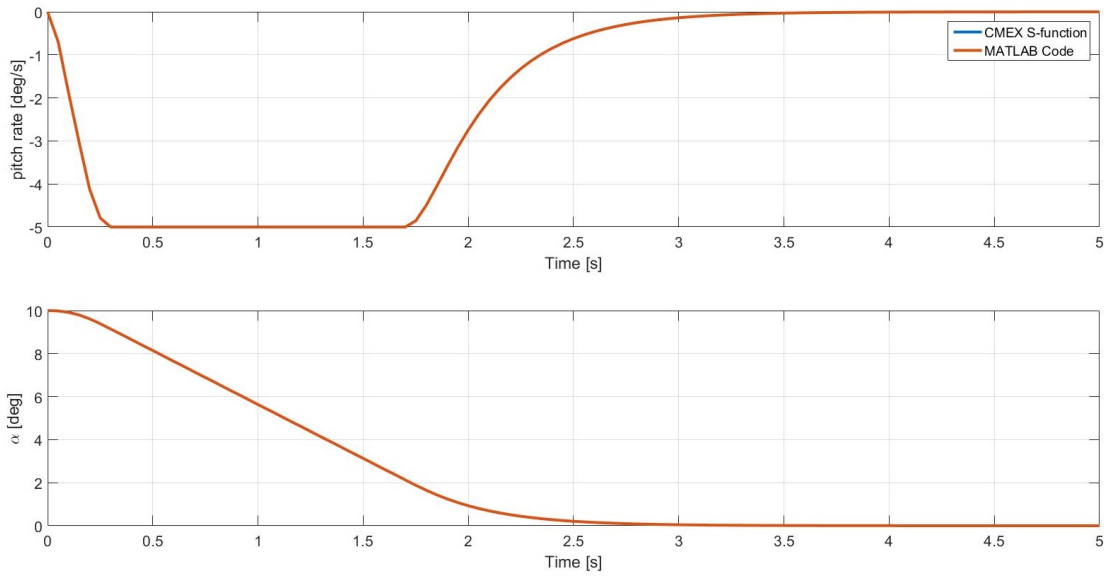


Figure 9-11: System response comparison of the CMEX S-function and MATLAB code tasked to eliminate an 10 degree error in the angle of attack. Note that one cannot distinguish between the two lines.

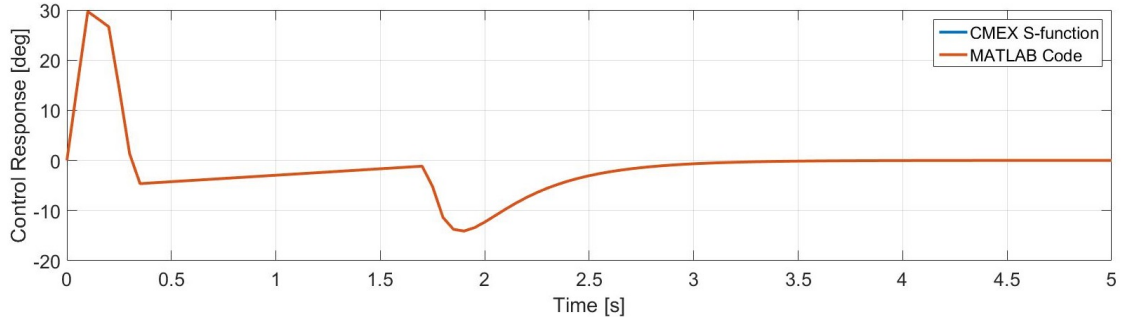


Figure 9-12: Control response comparison of the CMEX S-function and the MATLAB code. Note that one cannot distinguish between the two lines.

It is interesting to note that the pitch rate in Figure 9-11 remains constant for over a second. This is because there was a constraint that limited the absolute value of the pitch rate to be less than five degrees. This demonstrates that the inequality constraints are taken into account when

computing the optimal control action. The control response can be seen in Figure 9-12. It can be seen that the rate limit constraints the control response to be 15 degrees per time-step.

From the above paragraphs, it has become clear that the CMEX S-function and MATLAB implementation of the MPC produce identical results. Furthermore, it was proven by example that inequality constraints are taken into account when the control algorithm computes the optimal control action. Because both implementations are identical and behave as expected, one can conclude that the CMEX S-function implementation of a MPC is successfully verified.

Chapter 10

Results and Discussion

In the previous chapters the theoretical framework for the reference vehicle, the environment it interacts with, and the attitude controllers and their set of numerical parameters were discussed. Furthermore, the SADs of the numerical simulator were presented, and the components of the simulator were verified. All pieces of the puzzle have now been assembled and it is now possible to look at the puzzle in its entirety, that is, to perform simulations from which answers to the research questions can be derived.

This thesis focuses on the rotational motion of the reference vehicle during bank-reversals. It is during these manoeuvres that the attitude controllers can best be tested, because, at these moments, the demand on the control system will be the highest. However, because 16 different controllers were designed in this thesis, it is not practical to analyse all of them for each of the cases of interest (see Section 2-2-1). Additionally, it is not practical to compare all combinations of the designed controllers, because each combination generates a lot of data that needs to be analysed. For these reasons, a selection of sets of controllers was made that are comparable to each other, and the comparative analyses of these sets will be shown in this chapter. Additionally, the main conclusions of the remaining combinations will be presented.

The point along the reference trajectory at which the performance of the controllers are evaluated corresponds to the last bank-reversal of the reference trajectory, which is denoted as case five in Figure 2-3. The initial conditions and parameters of this point are shown in Table 10-1. It is noted that the rate of change of the bank angle is limited to 15 deg/s, to ensure a maximum roll and yaw rate for which the vehicle remains stable during the bank-reversal and to ensure it does not exceed its vehicle constraints.

Table 10-1: Overview of the initial conditions and parameters of the bank-reversal of case five.

Parameter	Value
Dynamic Pressure [Pa]	8,387.00
Mach Number [-]	2.66
Cmd. angle of attack [deg]	17.06
Initial Bank angle [deg]	-55.13
Final Bank angle [deg]	55.13

For each set of controllers, a comparative analysis will be made of their *nominal behaviour*, which will include the behaviour of the state variables, the control surfaces, and the thrusters. Additionally, to obtain insight into the robustness of the controller, a sensitivity analysis will be performed. To obtain meaningful conclusions of the sensitivity analysis, one thousand simulations will be performed, for each of which the aerodynamic moment coefficients will be randomly varied between

$\pm 10\%$ of their nominal value. The Random Number Generator (RNG) is initialized with the same seed at the start of every simulation batch. Then, for each simulation, the performance metrics of Equations 4-2 and 4-3 for the state variables and the control actuators are computed. The spread of the resulting distribution of the performance metrics will give an indication of the robustness of the control system.

It is possible to obtain a small value for the performance metric of a control actuator if it oscillates rapidly with a small amplitude. However, in this case, the *effective control effort* is still large, as the system must perform a lot of work to oscillate the control actuator. For this reason, it is desirable to investigate the distribution of the oscillatory behaviour of the control actuators for the changing system parameters of the sensitivity analysis.

The layout of the chapter follows the comparative analysis of the chosen sets of controllers. In Section 10-1, the methodology used to obtain a number indicative of oscillatory behaviour is presented. In Section 10-2, a comparative analysis of the benchmark controllers will be made. Section 10-3 will present the comparative analysis of the MFIC and MFC with the NNCs and FLCs with and without integral terms, respectively. A comparative analysis of the benchmark controllers with the IO- and IIO-models of the MPC analogous to each of the benchmark controllers will be made in Section 10-4. This chapter will then be concluded in Section 10-5 with a comparative analysis of the best controllers previously presented and discussed. It is noted that for each comparative analysis, the focus will be on the behaviour of the newly introduced controllers when compared to previously discussed controllers. For clarity, Table 10-2 summarises the abbreviations used for the different controllers in this chapter.

Table 10-2: Summary of the commonly used abbreviations in this chapter.

Abbreviation	Full Name
LQIC	Linear Quadratic Integral Controller
MFC	Moment-fraction Controller
MFIC	Moment-fraction Integral Controller
NNC	Neural Network Controller
NNIC	Neural Network Integral Controller
FLC	Fuzzy Logic Controller
FLIC	Fuzzy Logic Integral Controller
MPC	Model Predictive Control
IO	Input-Output
IIO	Increment-Input-Output

Additionally, the MPCs analogues of the benchmark controllers are distinguished by a superscript. For example, the MPC analogue to the original benchmark controller, as described in Section 4-3, is denoted by MPC^{bm}. A subscript refers to either the IO-model or the IIO-model.

10-1 Oscillation Test

For the sensitivity analysis, it is desired to have a measure for the oscillatory behaviour of the control responses. Due to the lack of available literature on this topic, it was necessary to create a methodology to investigate the oscillatory behaviour.

In essence, one wants to detect the perturbation of a smooth signal. However, only the perturbed signal is provided, which must therefore be smoothed. A natural way to smooth a signal is to use a *moving mean*. Computing the standard deviation for the same sample as was used to compute the moving mean gives an indication of the *local* oscillatory behaviour of the perturbed signal.

Equations 10-1 and 10-2 are used to compute the moving mean and standard deviation.

$$\mu_{mov}(k+N/2) = \frac{1}{N}\sum_{j=k}^{k+N-1} x(j) \tag{10-1}$$

$$\sigma_{mov}(k+N/2) = \sqrt{\frac{1}{N-1}\sum_{j=k}^{k+N-1} \left(x(j) - \mu_{mov}(k+N/2)\right)^2} \tag{10-2}$$

where N should be chosen such that it is sensitive up to medium/high frequency oscillations, which means it should be chosen small. Experience has shown that N should be chosen less than 0.5% of the number of data points for good behaviour.

A cumulative standard deviation is defined as the integral of the moving standard deviation. If one takes the integral over the entire interval of the function, one obtains a single number, which is a *global* indicator of the oscillatory behaviour of the curve. Furthermore, if one computes the integral for each time-step, one obtains the cumulative standard deviation curve shown on the bottom-right of Figure 10-1.

Figure 10-1 shows a graphical example of the oscillation test described in the above paragraphs. In the left figure, one can see a sinusoidal curve sequentially perturbed by white-noise, low frequency oscillations, and high frequency oscillations. It can be seen that the moving mean "filters" the white-noise, resulting in large values for the moving standard deviation, as shown in the top-right figure of Figure 10-1. However, physically, the white-noise implies rapid oscillations of the control actuators, and hence a large deviation is desirable. Because a large moving standard deviation gives a large value to the cumulative standard deviation, one can conclude from the *global* cumulative standard deviation that the control actuators oscillate rapidly.

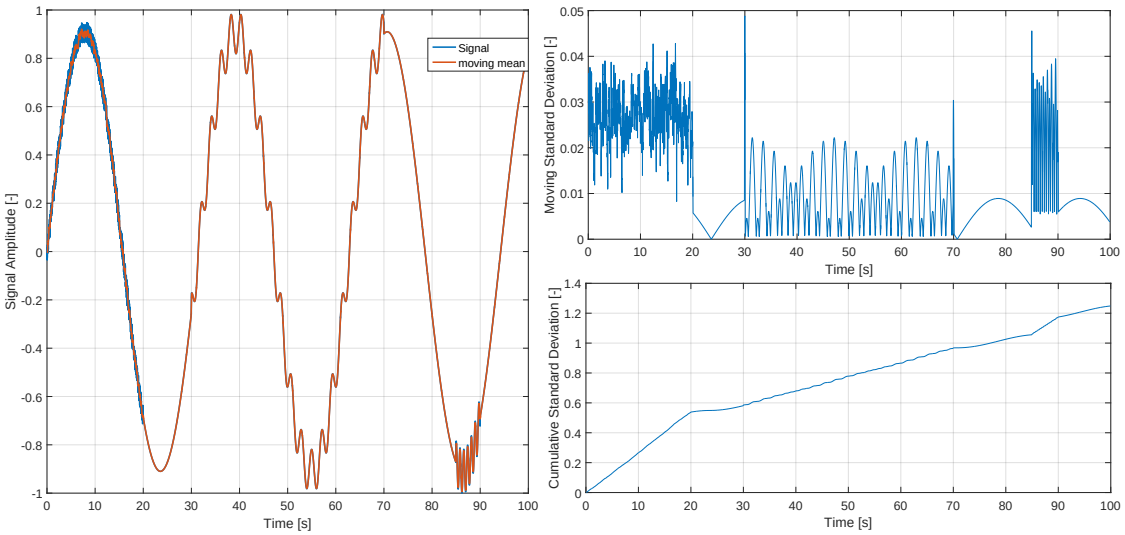


Figure 10-1: Graphical example of the behaviour of the moving mean, moving standard deviation, and cumulative standard deviation.

It is noted that the *global* oscillation indicator does not distinguish between rapid oscillations for a short time, or slow oscillation for longer periods. For information about the *local* behaviour, one has to look at the behaviour of the moving standard deviation. Additionally, to limit the sensitivity of the oscillatory measure to changes in amplitude, the perturbed signal has been scaled such that the largest absolute value of the signal is equal to one.

10-2 Benchmark Controllers

In Chapter 4, four different benchmark controllers were presented. Before comparing them to KBCSs, it is useful to compare them against each other to observe the effects of introducing moment-fractions and integral terms to the original controller of the reference vehicle. For clarity, the comparative analysis has been split in two parts. The first part compares the benchmark controller with the LQIC and the MFIC, and the second part compares the benchmark controller with the MFC and the MFIC.

The implementation effort completely comes from the off-line computation of the gains. However, a build-in function in MATLAB computes the optimal gain matrix when given an state and input matrices and weight-matrices, and, hence, the effort is transferred to computing the state and input matrices, and selecting the weight-matrices parameters. For the former task, once a function has been constructed that computes the state and input matrices, given a linearisation point, this becomes a routine task. Therefore, the remaining effort lies in the selection of the weight-matrices parameters, which have to be selected to obtain the desired responses. This is, however, a straightforward tasks, which can be completed by simple trail-and-error methods, or by an optimization algorithm.

Benchmark vs LQIC vs MFIC

Figure 10-2 shows the response of the state variables for the bank-reversal of the fifth case, as shown in Figure 2-3, for the first part of the comparative analysis. From the behaviour of the bank angle, it can be seen that the LQIC and the MFIC achieve the commanded bank angle quicker than the original benchmark controller, at the price of a small overshoot. This overshoot is characteristic of integral controllers and hence, is expected. Additionally, from the behaviour of the angle of attack and the sideslip angle, it can clearly be seen that the integral terms eliminate the state-errors during the phase were the rotational rates are constant, in contrast to the benchmark response. The rotational rates are constant during this phase because the error in the roll and yaw rates compensate the error in the bank angle.

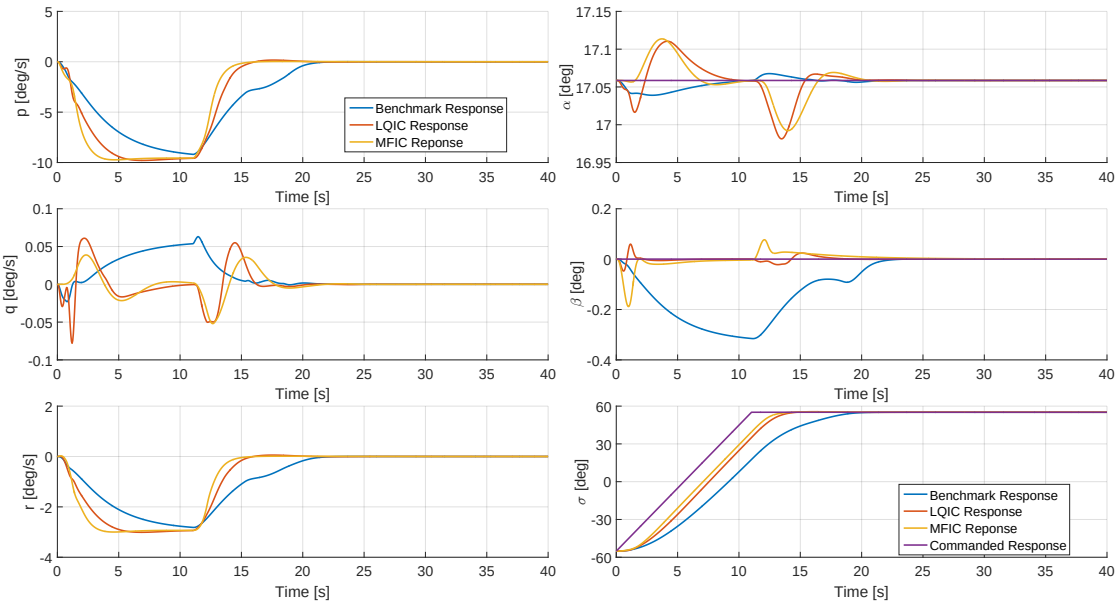


Figure 10-2: State response comparison between the benchmark controller, the LQIC, and the MFIC, for the bank-reversal of case five.

Furthermore, from the behaviour of the roll and yaw rates, it can be seen that the MFIC initially accelerates, but after three seconds settles to a constant rate. Closer inspection reveals that the the roll and yaw rates overshoot the constant rate, and decelerates before reaching the constant rate. This behaviour occurs due to the combination of the specific tuning and the finite reaction rate of the aerodynamic control surfaces. Due to the finite control rate, the positions of the elevons cannot change as quickly as the control laws command. Therefore, the vehicle still accelerates beyond the point for which the errors balance each other to obtain the constant rates. Hence, the vehicle needs to decelerate to the point of error balance.

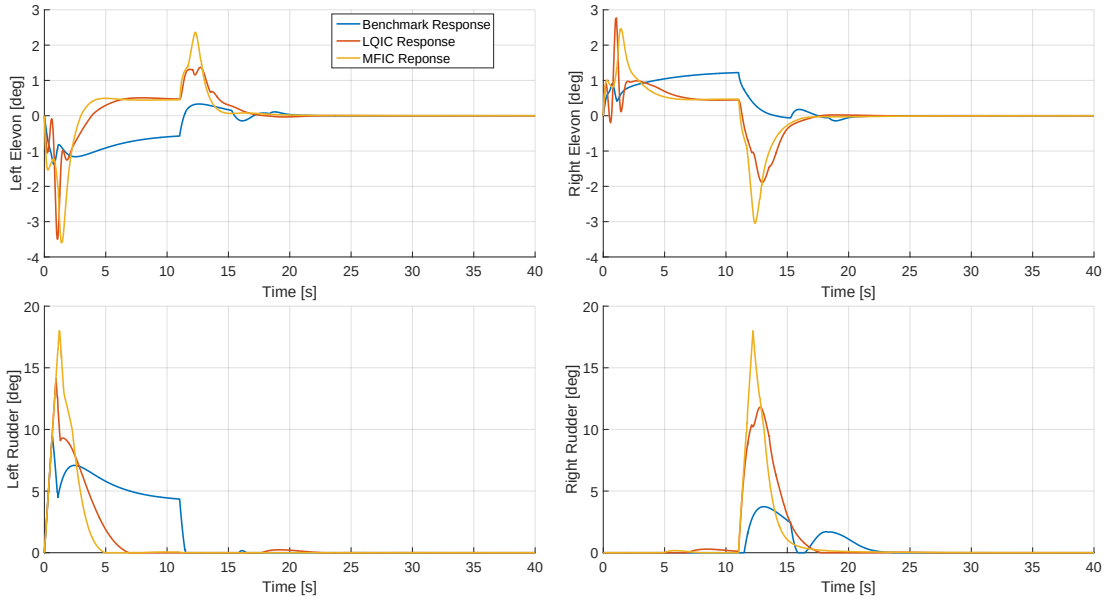


Figure 10-3: Control surface response comparison between the benchmark controller, the LQIC, and the MFIC, for the bank-reversal of case five.

The nominal behaviour of the control surfaces corresponding to Figure 10-2 is shown in Figure 10-3. It can be seen that the better performance of the state responses come at the price of a higher control effort, compared to that of the benchmark controller. Additionally, both the LQIC and the MFIC, have sharper control responses than the benchmark controller. It is noted that the sharpness of the peaks is limited by the finite reaction rate of the control surfaces.

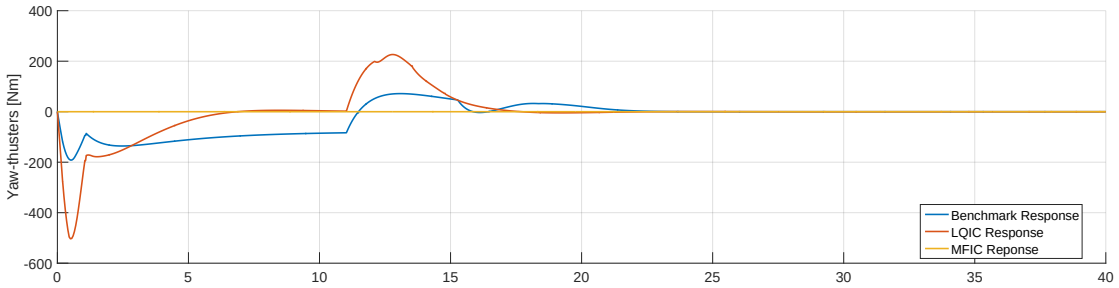


Figure 10-4: Thruster response comparison between the benchmark controller, the LQIC, and the MFIC, for the bank-reversal of case five.

In addition to the aerodynamic control surfaces, the reference vehicle can also use its reaction thrusters to generate control moments. However, during the bank-reversal of the fifth case considered here, only the yaw thrusters are still active. Figure 10-4 shows the response of the yaw

thrusters. It can be seen that the benchmark controller and the LQIC use their yaw-thrusters in addition to the control surfaces. The MFIC, however, does not. The reason for this is that the benchmark controller and the LQIC *directly* compute the commanded surface deflection and thruster commands, whereas the MFIC computes the commanded moment-fraction. The advantage of the latter is that, when the control surfaces are not saturated, the actuator assignment algorithm will not use the thrusters, thereby saving fuel. The use of the actuator assignment algorithm allows the vehicle to be smarter in its use of the actuators and available fuel, at the cost of increased complexity and power usage of the control surfaces. Because the MFIC does not use its yaw thrusters, it is not unexpected that the control effort of the control surfaces is higher for the MFIC, compared to that of the benchmark controller and the LQIC. It is noted that, if a controller makes use of the actuator assignment algorithm of Chapter 5, then the yaw thrusters will not be used for the scenario considered in this chapter.

Varying the moment coefficients of the reference vehicle allows for an analysis on the sensitivity of the control system to varying system parameters. If the resulting responses are not very sensitive to these varying system parameters, one can say that the system has *robust stability*. From the sensitivity analysis of the benchmark controller, the LQIC, and the MFIC, and comparing the variations of the integrated errors of the state variables, the control effort, and the control surface fluctuations, an indication of the systems robustness can be obtained. The distributions of the integrated errors of the state variables are shown in Figure 10-5. For the integrated error of the bank angle, the MFIC has smaller values for all of its variations compared to that of the LQIC, and both have smaller values for all variations compared to the benchmark controller. Additionally, the distribution width of the MFIC is smaller than that of the LQIC, and both have smaller widths than that of the benchmark controller. For the integrated error of the sideslip angle, a similar observation can be made, however, the roles of the LQIC and the MFIC have been reversed; the LQIC has smaller values for all system variations and a smaller distribution width compared to that of the MFIC. Furthermore, it can be seen that the distribution of the integrated error of the angle of attack for the LQIC and the MFIC are contained within the same interval, with the LQIC having half the distribution width than that of the MFIC. The distribution width of the benchmark controller for the integrated error of the angle of attack is eight times as large as that for the MFIC.

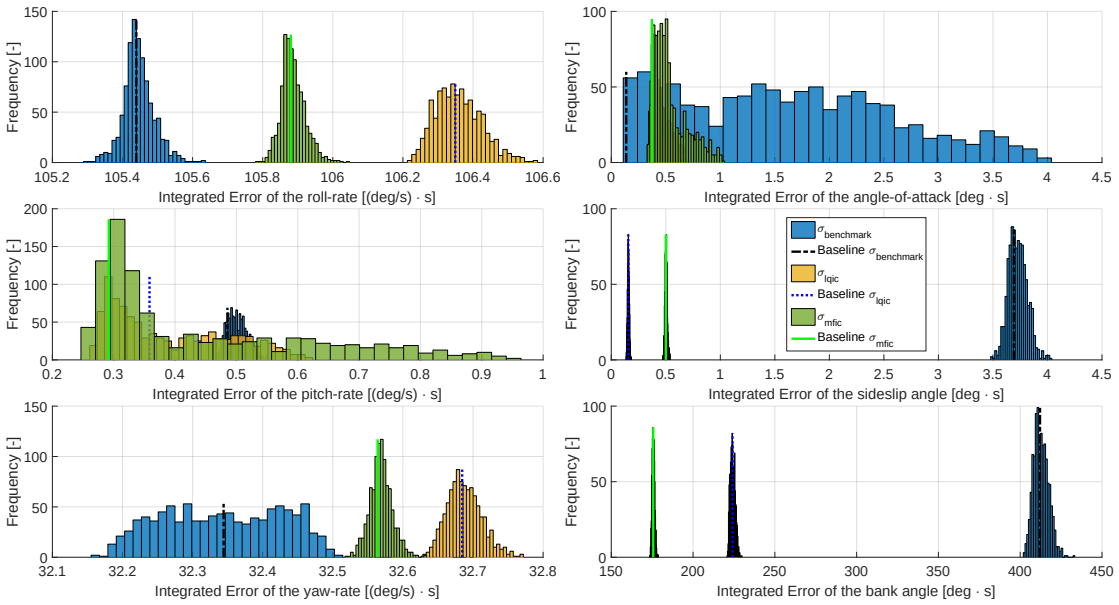


Figure 10-5: Comparative analysis of the sensitivity of the state variables of the benchmark controller, the LQIC, and the MFIC, for the bank-reversal of case five.

Additionally, the benchmark controller has smaller values for all system variations for the integrated error of the roll and yaw rates when compared to those of the MFIC, and both have smaller values for all variations when compared to those of the LQIC. Furthermore, for the integrated error of the pitch rate, the distribution width of the LQIC is half that of the MFIC, but are contained within the same interval. The distribution width of both controllers are larger than those of the benchmark controller. This can be explained by the fact that the benchmark controller is not tuned as tightly as the LQIC and the MFIC, and is therefore less sensitive to system variations. Further inspection reveals that the larger pitch rate distribution width of the MFIC is due a too tightly tuned longitudinal controller. It is noted that the LQIC and the MFIC have a smaller baseline value for the integrated pitch rate error than the benchmark controller.

The sensitivity of the control effort due to system variations is analysed similar to what was done for the state variables, and the resulting distributions are shown in Figure 10-6. The distributions of the left and right elevons for the LQIC and the MFIC are similar, but the distribution of the latter is shifted to higher values of the control effort, due to its stronger responses. The distribution width of the benchmark controller is larger than that of the other two controllers, but otherwise overlap. For the control effort of the rudders, the distribution widths are similar for all three controllers. However, for the left rudder, the benchmark requires the highest control effort, and for the right rudder, it requires the least control effort. This is not unexpected, because Figure 10-3 clearly shows that the benchmark controller uses its left rudder more than its right rudder. Curiously, for the left and right rudders, the baseline control effort of the MFIC is lower than that of the LQIC, even though the latter used its yaw thrusters in addition to its rudders. This occurs because the MFIC responds faster, with corresponding sharper control responses, than the LQIC. Therefore, even though the MFIC has higher peaks for the rudder control responses, the rudders spend less time away from its zero deflection, and, hence, the integrated control effort is lower than that of the LQIC.

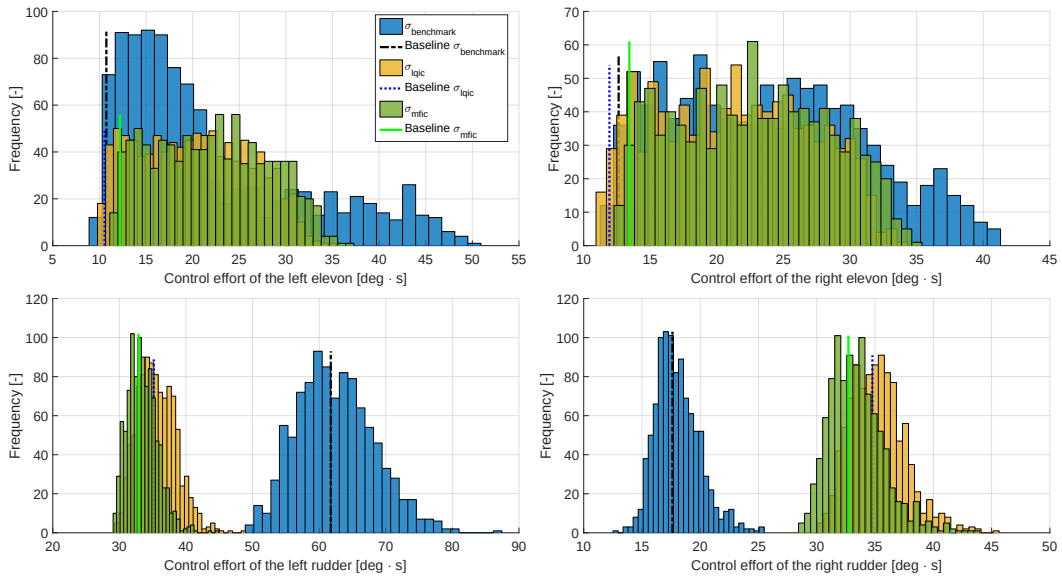


Figure 10-6: Comparative analysis of the sensitivity of the control effort of the benchmark controller, the LQIC, and the MFIC, for the bank-reversal of case five.

In addition to the control effort of the control surfaces, the changes in the oscillatory behaviour due to system changes can also be studied. From Figure 10-7, it can be seen that the oscillatory behaviour of the MFIC for the elevons is less sensitive to system changes than the benchmark controller and the LQIC. This is because, even though the MFIC has sharper control responses than the other two controllers, it has a smoother control response overall, which, therefore, makes

the control fluctuations less sensitive to system variations. For the left rudder, the distribution widths are similar for the three controllers. However, even though the baseline fluctuation value of the MFIC is lower than that of the LQIC, the peak of both distributions occur for the same fluctuation value. Further inspection reveals that the system variations cause the sideslip angle to vary which primarily causes more work for the left rudder of the MFIC, which simultaneously increases its fluctuation values. For the right rudder, the distribution width of the MFIC is larger than those of the benchmark controller and the LQIC. However, for all system variations, the fluctuation values of the MFIC are smaller than those of the LQIC, and both have (much) smaller values than those of the benchmark controller. This is because of the relative smoothness of the control responses of the different controllers.

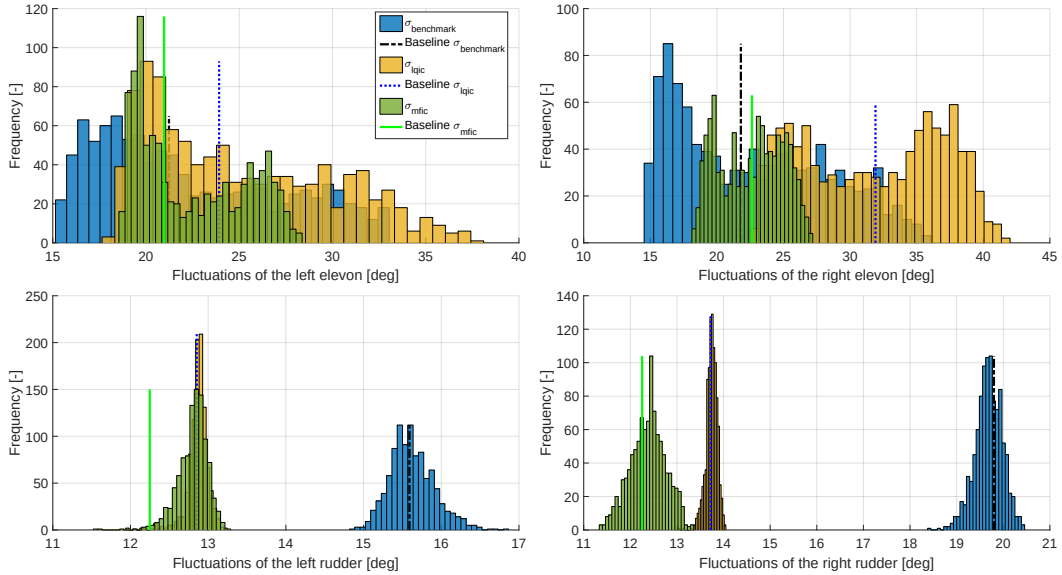


Figure 10-7: Comparative analysis of the sensitivity of the control fluctuations of the benchmark controller, the LQIC, and the MFIC, for the bank-reversal of case five.

Furthermore, for the fluctuations of the elevons, the benchmark controller and the LQIC have similar distribution widths, which is twice as large as that of the MFIC. This is due to the smoother elevon responses of the MFIC compared to those of the other two controllers. It is noted that the baseline elevon fluctuation values of the benchmark controller and the MFIC are similar, and are less than those of the LQIC.

Benchmark vs MFC vs MFIC

The second part of this section will present and discuss the comparative analysis of the benchmark controller, the MFC, and the MFIC. Figure 10-8 shows the response of the state variables. From the behaviour of the bank angle, it can be seen that the MFC eliminates the error quicker than the benchmark controller, but slower than the MFIC. Additionally, similar to the benchmark controller, the MFC has no overshoot. Furthermore, from the behaviour of the sideslip angle, it can be seen that the MFC induces a large sideslip angle, the effect of which is seen in the roll and yaw rates, which maintains a constant value and becomes positive, respectively. When the sideslip angle decreases, the roll and yaw rates start to increase. However, this induces another increase in the sideslip angle, to which the roll and yaw rates respond in a similar fashion as before. It is noted that, during the constant rate phase, the angle of attack and the sideslip angle maintain a constant non-zero error, unlike the MFIC. This clearly demonstrates the effect of introducing integral terms of the aerodynamic angles.

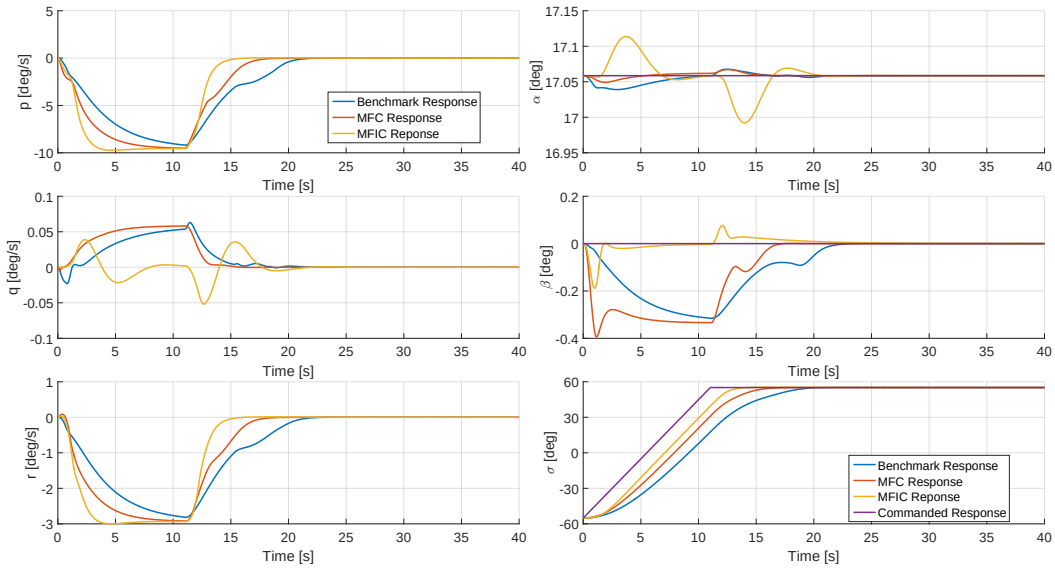


Figure 10-8: State response comparison between the benchmark controller, the MFC, and the MFIC, for the bank-reversal of case five.

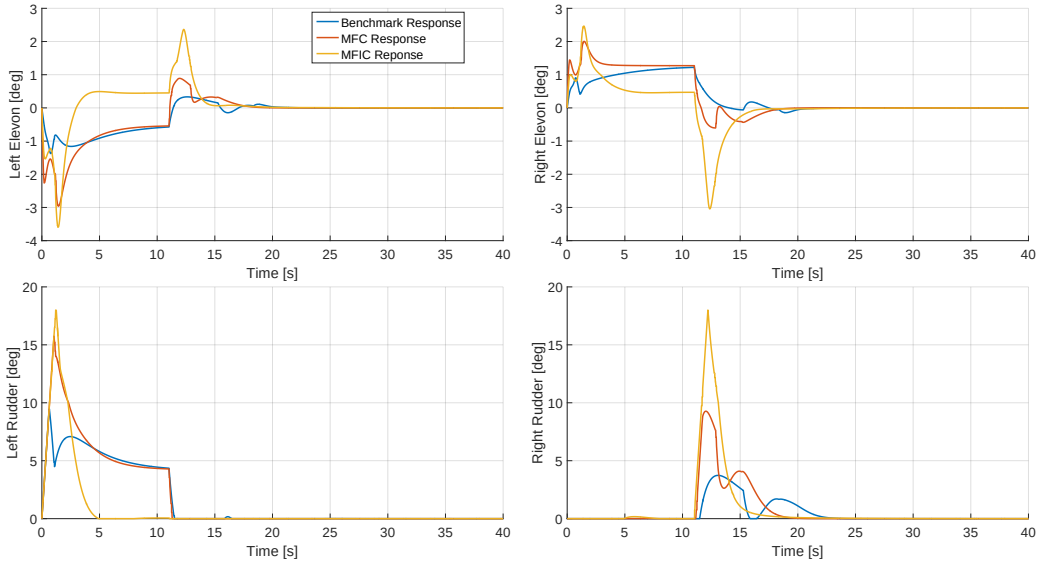


Figure 10-9: Control surface response comparison between the benchmark controller, the MFC, and the MFIC, for the bank-reversal of case five.

The nominal behaviour of the control surfaces corresponding to Figure 10-8 is shown in Figure 10-9. It can be seen the control surfaces of the MFC respond less strongly than the MFIC, but more strongly than the benchmark controller. Additionally, during the constant-rate phase, the left rudder maintains a non-zero deflection, unlike the MFC, due to the non-zero sideslip angle. Hence, it is expected that the control effort of the left rudder will be larger for the MFC than for the MFIC, and comparable to that of the benchmark controller. Furthermore, the elevons of the MFC fluctuate similar to those of the MFIC. The rudders of the MFC, however, fluctuates more than that of the MFIC, especially to right rudder, which is due to the deceleration of the vehicle. Therefore, it is expected that the right rudder of the MFC will be assigned larger values for its oscillatory behaviour than its MFIC counterpart. It is noted that the yaw thrusters of the MFC and MFIC are not used.

From the comparative analysis of the sensitivity of the state variables to changes in the system parameters, as shown in Figure 10-10, it can be seen that, for the MFC, the angle of attack and sideslip angle are equally sensitive to varying system parameters as the benchmark controller, and both are more sensitive than the MFIC. The angle of attack distribution of the MFC is nearly identical to that of the benchmark controller. For the sideslip angle, the integrated error of the MFC is (much) larger than those for the MFIC, for all system variations, clearly demonstrating the effect of the integral terms. For the bank angle, the MFC is more sensitive to system variations than the MFIC, and the integrated error is larger for all system variations. Furthermore, the distributions of the integrated errors of the roll and yaw rates for the MFC and the benchmark controller are comparable, with the distribution width of the roll rate of the MFC being smaller than that of the benchmark controller. The distribution of the pitch rate of the MFC has similar width to that of the benchmark controller, but has larger values, for all system variations.

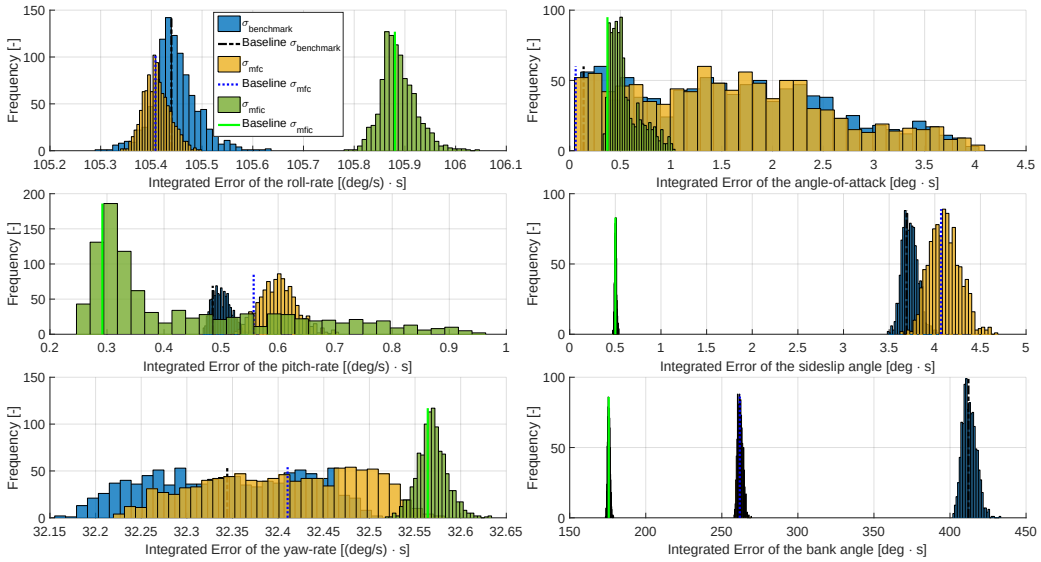


Figure 10-10: Comparative analysis of the sensitivity of the state variables of the benchmark controller, the MFC, and the MFIC, for the bank-reversal of case five.

The comparative analysis of the control effort distributions is shown in Figure 10-11. It can be seen that the elevon control effort distributions of the MFC are similar to that of the benchmark controller, but shifted to higher values for the control effort. This statement is also true for the left rudder distribution of the MFC. Furthermore, the right rudder distribution of the MFC has a similar distribution width to that of the MFIC. This implies that the introduction of integral terms do not affect the sensitivity to system variations. Furthermore, for almost all variations, the right rudder control effort of the MFC is lower than those for the MFIC.

Figure 10-12 shows the control fluctuation distributions. It can be seen that, for the left elevon distribution, the MFC is similar to that of the MFIC, but is shifted to lower values, implying the left elevon of the MFC experiences less fluctuations than that of the MFIC, which can be explained by the fact that the left elevon response of the MFC return relatively slowly to a zero deflection, whereas the MFIC does so with sharp responses, thereby increasing the fluctuation value assigned to it. Furthermore, it is found that the distributions of the right elevon of the MFC and the benchmark controller are nearly identical. The distributions of the left rudder for the MFC and the MFIC are similar, but the former is assigned lower fluctuation values. This is as expected, because from Figure 10-9, it can be seen that the MFC has a smoother control response compared to that of the MFIC. Furthermore, it can be seen that the right rudder control fluctuations of the MFC is assigned larger values than those of the MFIC, which can be explained by the oscillations in the right rudder control response, as shown in Figure 10-9. The rudder distribution widths all similar for all three controllers.

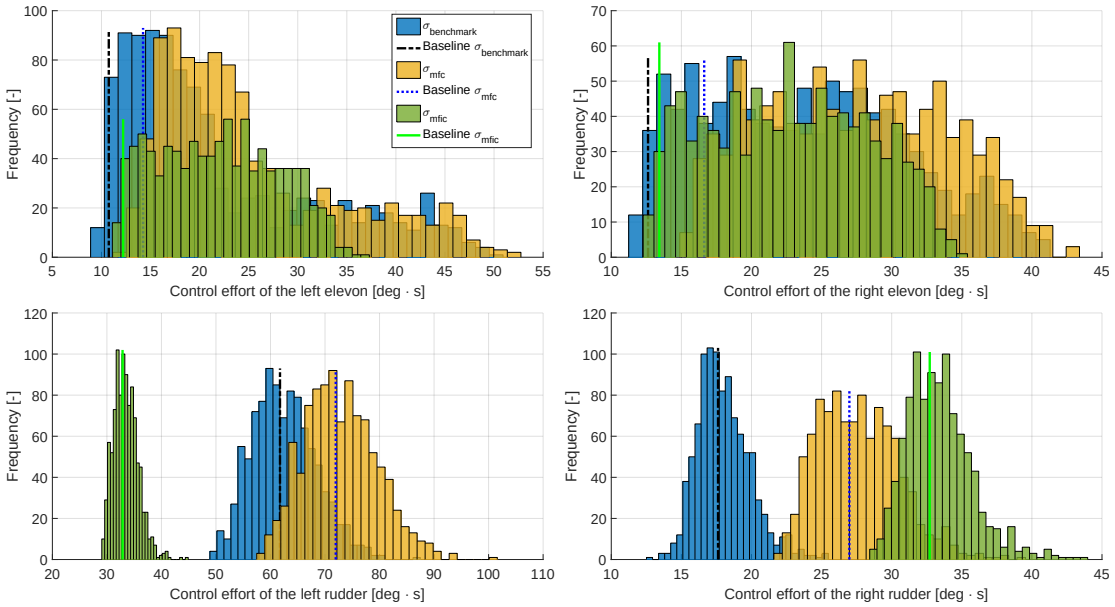


Figure 10-11: Comparative analysis of the sensitivity of the control effort of the benchmark controller, the MFC, and the MFIC, for the bank-reversal of case five.

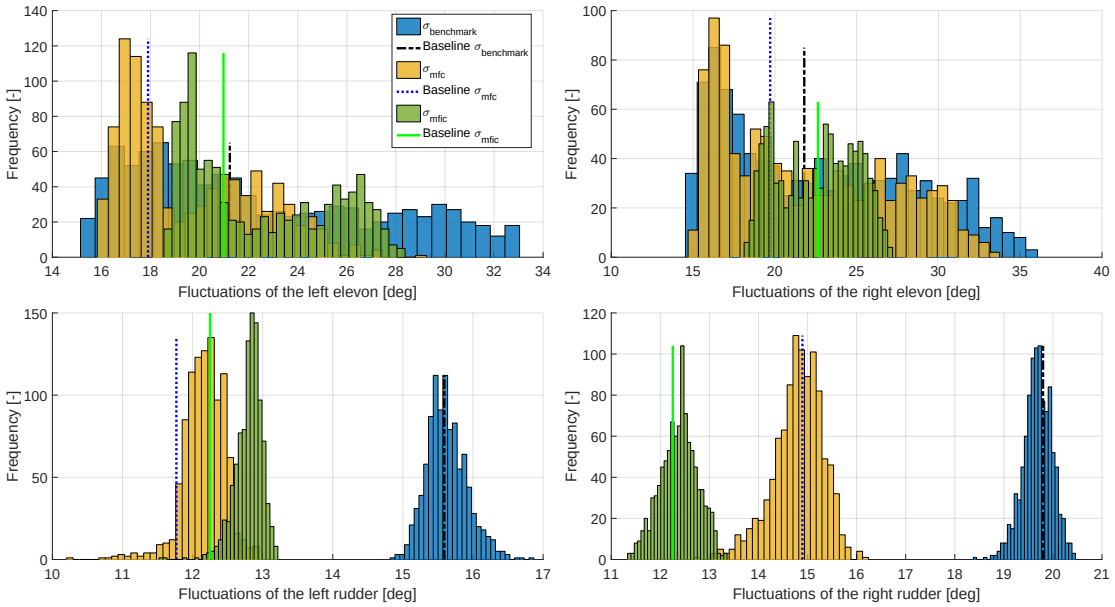


Figure 10-12: Comparative analysis of the sensitivity of the control fluctuations of the benchmark controller, the MFC, and the MFIC, for the bank-reversal of case five.

10-3 Neural Network and Fuzzy Logic Controllers

The NNCs and the FLCs with and without integral terms are analogous to the MFIC and the MFC, respectively. Therefore, both the NNC and the FLC will be compared to each other and their analogous (modified) benchmark controller. However, before the comparative analysis can be performed, the NNCs and the FLCs need to be optimized first.

Optimization Results

The optimal neural network structure was found to contain no hidden nodes, and was fully connected from the input layer to the output layer. The connection weights of the final network are shown in Table B-7 for the network structure with (NNIC) and without (NNC) integral terms, for the optimal control action during the bank-reversal of case five. Apart from a non-linear output node, the resulting networks are similar to the benchmark controllers, with the optimal gains now determined with the NEAT algorithm. This implies that, although the optimization algorithm is capable of generating more complex networks, a (quasi-) linear controller with proper connection weights/gains is more optimal than a complex network with hidden nodes and recurrent connections. However, it does not imply that more complex structures with a higher performance cannot be found with more optimization effort.

The FLCs were optimized using a build-in pattern-search algorithm within MATLAB, with the initial conditions as described in Section 7-3. The resulting membership parameters describing the peaks of each triangular membership function for the FLCs with (FLIC) and without (FLC) integral terms are shown in Tables B-9 and B-10, respectively, for the optimal control action during the bank-reversal of case five. It is noted that, for some variables, the membership functions are clustered together. This implies that the controller prefers the output values corresponding to the clustered inputs, or more degrees of freedom are required for the outputs for the corresponding clustered inputs to obtain optimal control behaviour. Further optimization based on these optimization results shown in the two tables can improve the control performance by finding the optimal ranges of all input and output variables.

Theoretically, NNCs and FLCs have great potential, and are able to approximate any function to any desired degree of accuracy. However, this potential comes at the cost of large degrees of freedom when the controllers needs to be optimized, thereby drastically increasing the implementation effort compared to the benchmark controllers and the MPCs. The optimization of the NNCs and FLCs required simulations that took several hours for each axis of a controller. Additionally, the optimization algorithm contained a large number of parameters that changes the behaviour of the optimization algorithm (default options were chosen in this thesis). The large simulations times imply that it is not practical to do a large number of iterations with different optimization parameters, complicating the design process for the optimal controller.

Comparative Analysis Without Integral Terms

Now that the optimal solutions for the NNCs and FLCs are determined, a comparative analysis can be made. First, the MFC will be compared to the NNC and the FLC without integral terms. Figure 10-13 shows the response of the state variables. From the behaviour of the bank angle, it can be seen that both the NNC and the FLC eliminate the bank angle error quicker than the MFC, with the FLC slightly quicker than the NNC. The quicker responses of the bank angle comes paired with sharper responses of the roll and yaw rates. For the NNC, the roll and yaw rates have no "overshoots", unlike for the FLC. The origin for these overshoots is identical to those of found for the MFIC. The induced sideslip angle for the FLC behaves similar to that of the MFC, but with higher peaks corresponding to the overshoots of the roll and yaw rates, whereas the NNC obtains positive values. Because the NNC does not have overshoots for the roll and yaw rates, the sideslip angle does not oscillate during the first three seconds of the simulation.

Additionally, it can be seen that, like the sideslip angle, the angle of attack of the NNC grows larger than its commanded value, whereas for the FLC it grows smaller. The opposite is true for the behaviour of the pitch rate. This occurs because the NNC and the FLC have learned to find an optimal point of error balance for which they can control the vehicle while executing the bank-reversal. What happens for the NNC, is that the strong response to a small error in the sideslip angle over corrects and turns it positive, at which point the combined positive sideslip angle and bank angle error combine, together with the roll and yaw rates, into an error balance. Similar behaviour explains the angle of attack response of the NNC. Furthermore, it can be seen that the

NNC has a larger positive error than the FLC has a negative error. It is therefore expected that the nominal value of the integrated error of that angle of attack for the former is larger than that of the latter. Furthermore, for the FLC, it can be seen that the pitch rate oscillates at the start of the simulation and at the exact time when the bank angle reaches its commanded value. The oscillations are a direct consequence of the oscillations shown in the elevons, as shown in Figure 10-14, which in turn occur due the too tightly tuned lateral FLC.

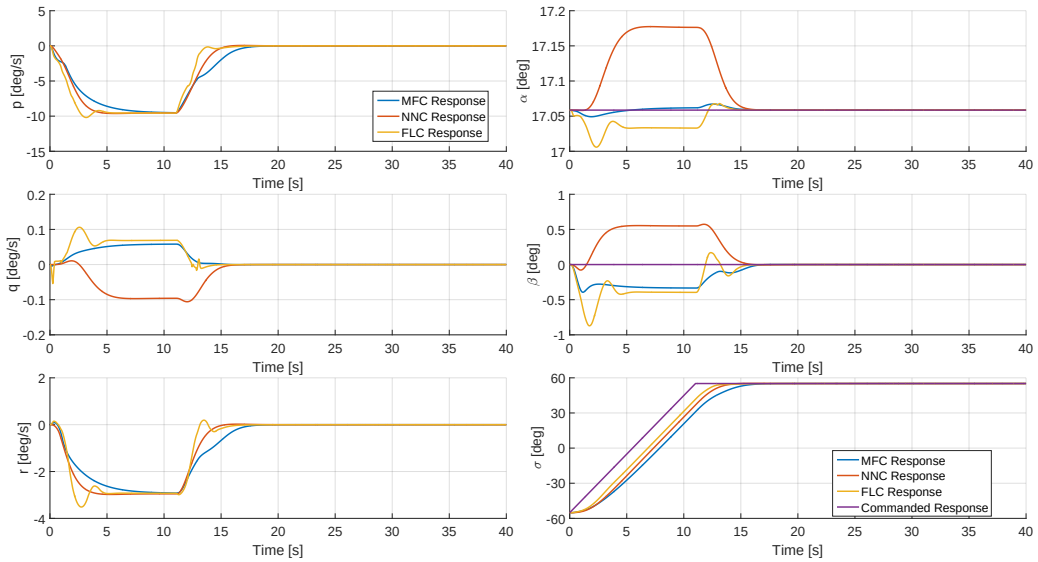


Figure 10-13: State response comparison between the MFC, the NNC, and the FLC, for the bank-reversal of case five.

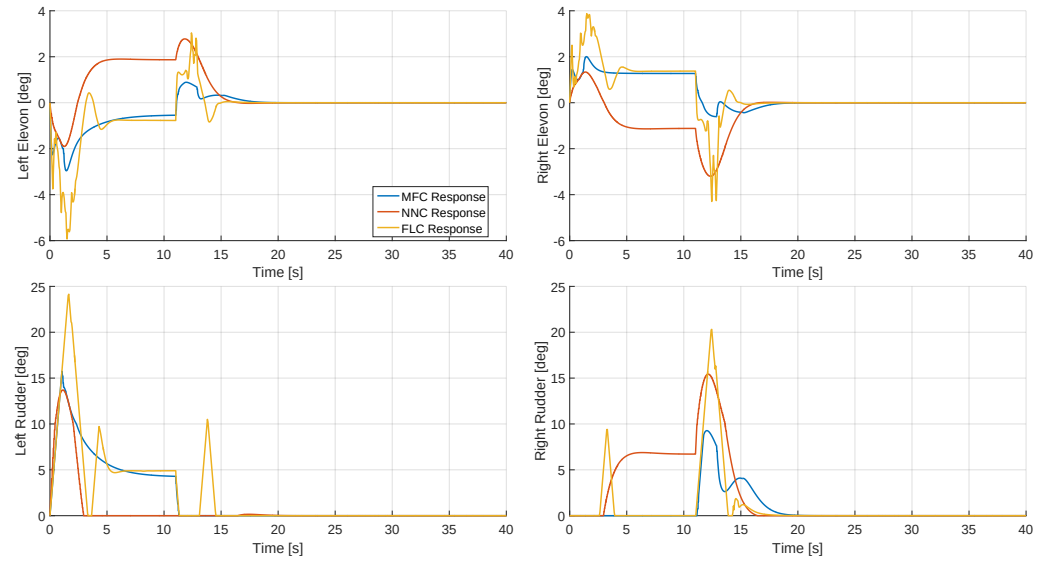


Figure 10-14: Control surface response comparison between the MFC, the NNC, and the FLC, for the bank-reversal of case five.

The nominal control response corresponding to the nominal motion of Figure 10-13, is shown in Figure 10-14. It shows that the behaviour of the control surfaces of the NNC move smoothly, with no high frequency oscillations. This occurs because of the non-linear activation function of the neural network. Because the controller needs to be stable for small state errors, for which

the activation function is approximately linear, it requires larger errors for an identical control response, which explains why the angle of attack and sideslip angle errors are larger for the NNC than for the other two controllers. Additionally, because the errors rise slowly, the feedback control response will also slowly settle to the constant rate phase. Because of the lack of oscillations in the control response for the NNC, it is expected that the fluctuation are small compared to that of the MFC. On the other hand, the control surfaces of the FLC shows high frequency oscillations in the elevons. Further inspection reveals that the fluctuations are mostly due to too tightly tuned gains of the roll and yaw rates and the bank angle for the roll-axis control. Rapid fluctuations of the variable gains of the longitudinal controller due to induced changes in the angle of attack accounts for the remaining fluctuations. Adjusting the range of the membership functions of these variables should be able to eliminate the fluctuations in the control surfaces of the FLC.

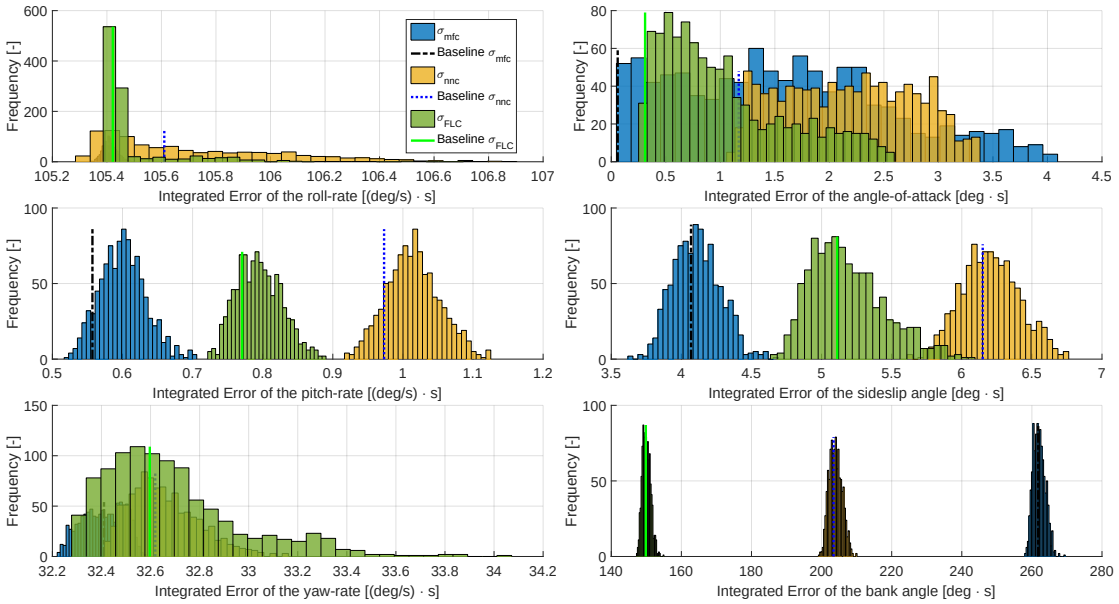


Figure 10-15: Comparative analysis of the sensitivity of the state variables of the MFC, the NNC, and the FLC, for the bank-reversal of case five.

For the comparative analysis of the sensitivity of the controllers to system variations, it is found that the aerodynamic angle distributions for the NNC and the FLC, as shown in Figure 10-15, have similar widths compared to each other. Additionally, for the angle of attack, the FLC has a lower baseline integrated error than the NNC, but the MFC has a lower baseline integrated error than the FLC, which is as expected from the angle of attack response shown in Figure 10-13. Furthermore, the angle of attack distribution of the MFC has a larger width than the NNC and the FLC. Because a small change in the system parameters will induce a relatively large change in the angle of attack response, the resulting distribution widths are relatively large. For the sideslip angle and the bank angle, the FLC has a lower baseline integrated error than the NNC. Additionally, for the sideslip angle, the baseline values for the NNC and the FLC are higher than that of the MFC, and for the bank angle, the MFC has the largest baseline value, as is expected from their nominal responses.

For the integrated error of the roll rate, the baseline value of the FLC is comparable to that of the MFC, but the former has outliers, widening its overall distribution. If these outliers were removed, the resulting distribution almost completely overlaps with that of the MFC. The distribution width of the NNC is ten times as large as that of the MFC, and its baseline value is larger than that of the FLC and the MFC. Further inspection reveals that the outliers of the FLC are due to the too tightly tuned gains of the bank angle controller. The large distribution width of the NNC is due to it being more responsive to small errors, which cause oscillations in the responses, and,

hence, increase in the integrated errors. For the yaw rate, the baseline value of the FLC is nearly identical to that of the NNC, but the former has a larger distribution width. This is due to the oscillations that occur before settling to a constant rate phase. The distribution width of the NNC is larger than that of the MFC, due to the same reason as for the roll rate. For the distributions of the pitch rate, all three controllers have similar width. The NNC has a larger baseline value than the FLC and both have larger baseline values than the MFC, which is as expected from the nominal response. For all system variations, the distributions do not overlap.

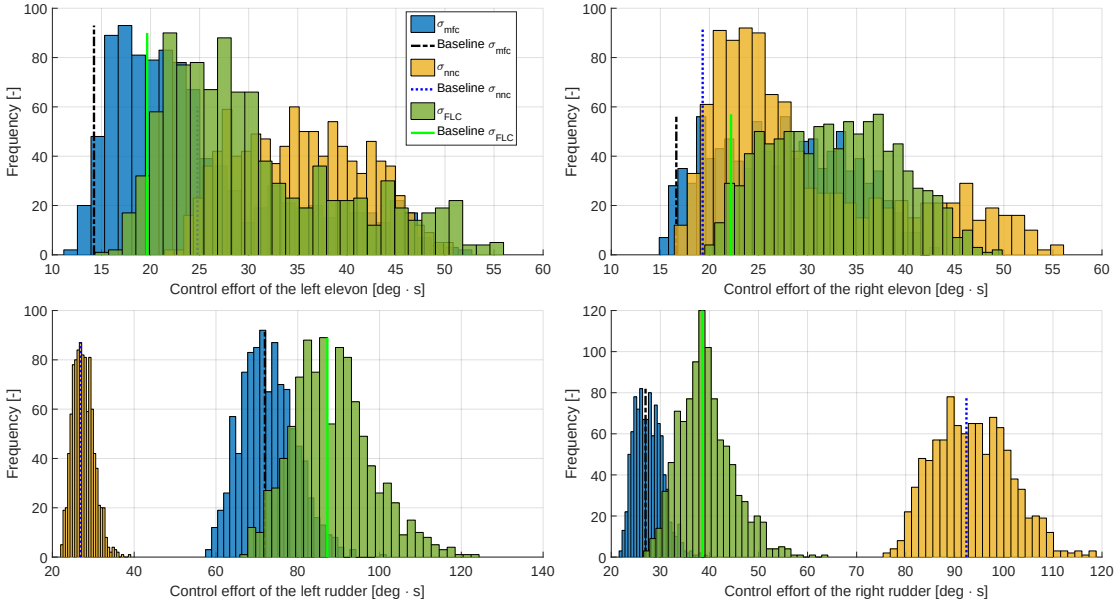


Figure 10-16: Comparative analysis of the sensitivity of the control effort of the MFC, the NNC, and the FLC, for the bank-reversal of case five.

For the distributions of the elevon control effort, it can be seen from Figure 10-16 that the FLC has similar distribution widths as that of the MFC, but is shifted to higher values. Furthermore, for the left elevon and rudder, the distribution widths of the NNC are smaller than those of the FLC and the MFC, whereas for the right elevon and rudder, the opposite is true. For the rudders, this occurs because the NNC predominantly uses its right rudder, which results in a larger increase in control effort for small system variations than for the left rudder. This effect is augmented by the fact that the MFC and the FLC predominantly use their left rudder, resulting in the same increase in control effort, which causes them to be more sensitive to system variations. Still, the right rudder distribution width of the FLC is as large as that of the NNC, due to the oscillating sideslip angle response, which is sensitive to system variations. The sensitive sideslip angle oscillations directly translate to the sensitivity of both rudders of the FLC. Because the sign of the angle of attack error of the NNC is the opposite to that of the MFC and the FLC, system variations will impact the right elevon more than the left elevon of the NNC, and vice versa for the MFC and the FLC.

For all control surfaces, the baseline fluctuation values of the FLC, as shown in Figure 10-17, are larger than those of both the MFC, and the NNC is assigned the lowest fluctuation values, as was expected based upon Figure 10-14. Similarly, it can clearly be seen that the rudder distribution widths of the FLC are larger than those of the MFC and the NNC, which is due to the oscillating sideslip response that is sensitive to system variations. Furthermore, while the elevon fluctuation values for the FLC are the largest, they are not more sensitive to system variations than the MFC and the NNC. This demonstrates that, while the FLC shows larger state and control fluctuations, system variations do not destabilize the vehicle and the controller can still control it.

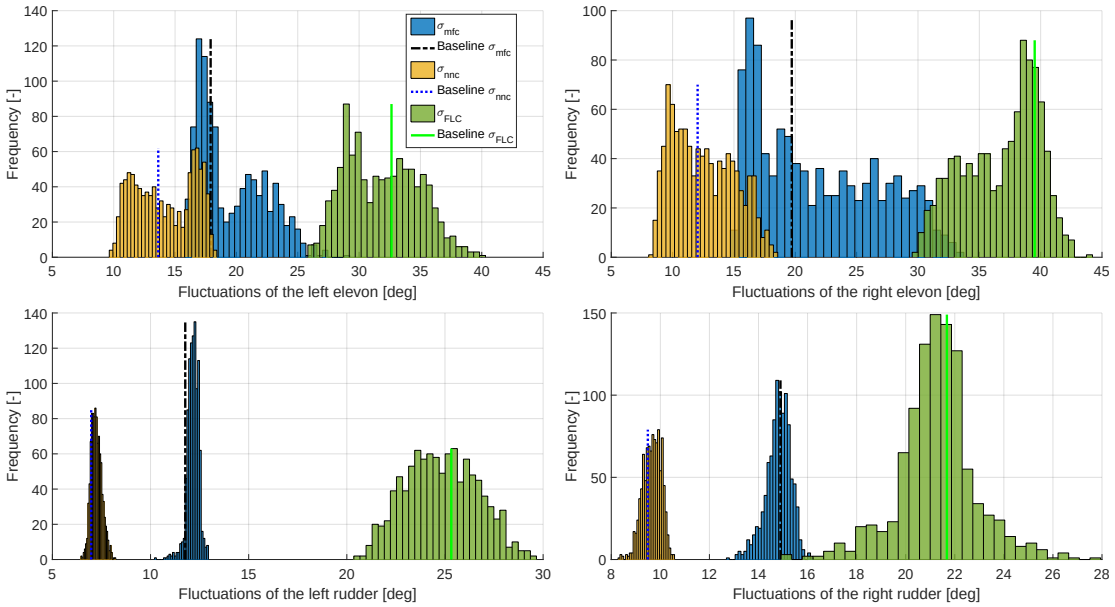


Figure 10-17: Comparative analysis of the sensitivity of the control fluctuations of the MFC, the NNC, and the FLC, for the bank-reversal of case five.

Comparative Analysis With Integral Terms

Similar to the benchmark controller and the MFC, it is possible to add integral terms to the input variables of an NNC and FLC, to obtain an NNIC and FLIC. The benchmark controller analogous to these controllers is the MFIC, which will be used for the comparative analysis. Figure 10-18 shows the state responses, from which it can be seen that the FLIC has trouble eliminating the steady-state error of the angle of attack. This is not unexpected, because, compared to the MFIC, the gains of the corresponding integral term of the FLIC is small (see Table B-9).

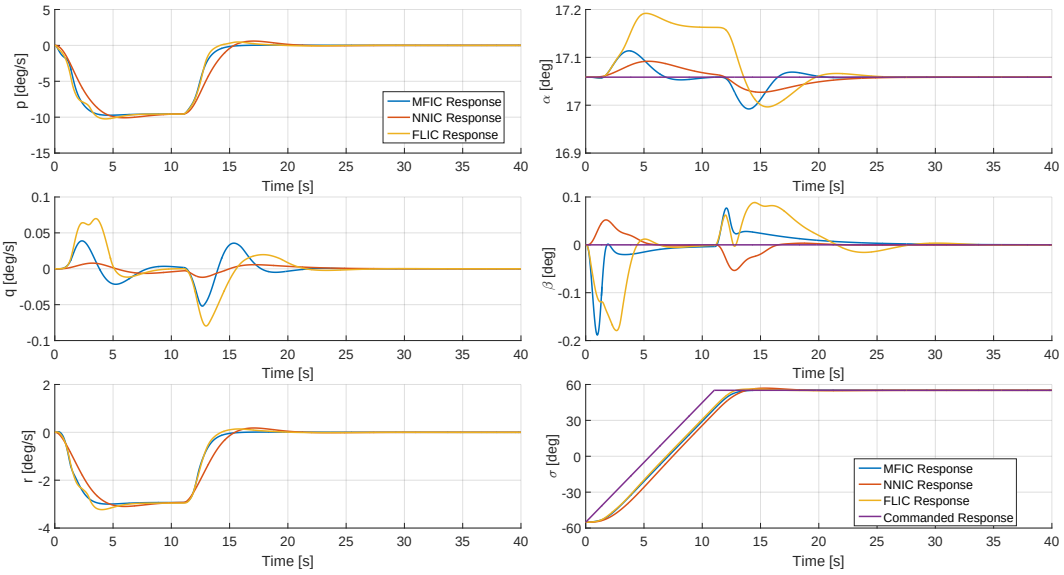


Figure 10-18: State response comparison between the MFIC, the NNIC, and the FLIC, for the bank-reversal of case five.

From the response of the bank angle it can be seen that the NNIC responds the slowest, which is caused completely by a large gain value of the bank controller for the sideslip angle, which limits the moment-fraction about the roll-axis. The slower response also prevents the vehicle from slowing down once it has completed its bank-reversal, and, hence, causes an overshoot. Furthermore, the FLIC responds faster than the MFIC, and has a larger overshoot, which takes longer to eliminate than that of the MFIC, which is completely attributed to the larger gain values of the lateral integral terms of the FLC. Additionally, it can be seen from the angle of attack and sideslip angle responses, that the oscillations in those variables are smaller in amplitude for the NNIC than those for the MFIC and the FLIC, which happens because of the slower response, which in turn induces less error in the angle of attack and the sideslip angle. It is noted that, while the NNIC has a larger overshoot than the FLIC, both have a nearly identical settling times.

From the responses of the roll and yaw rate, it can be seen that all three controllers enter and exit the constant rate phase with an overshoot. This occurs because the integral terms have accumulated values of a particular sign, and needs to remove those accumulated values by contributions of the opposite sign, which causes the overshoot. Additionally, oscillations occur in the roll and yaw rates when they approach the constant rate phase. These oscillations occurs due to the increasing value of the sideslip angle, which causes the point of error balance to shift and assigns values for the moment-fractions that decrease or increase the vehicle's acceleration. For the pitch rate, the oscillation amplitudes of the NNIC are smaller than those of the MFIC and the FLIC, due to the decreased induced error.

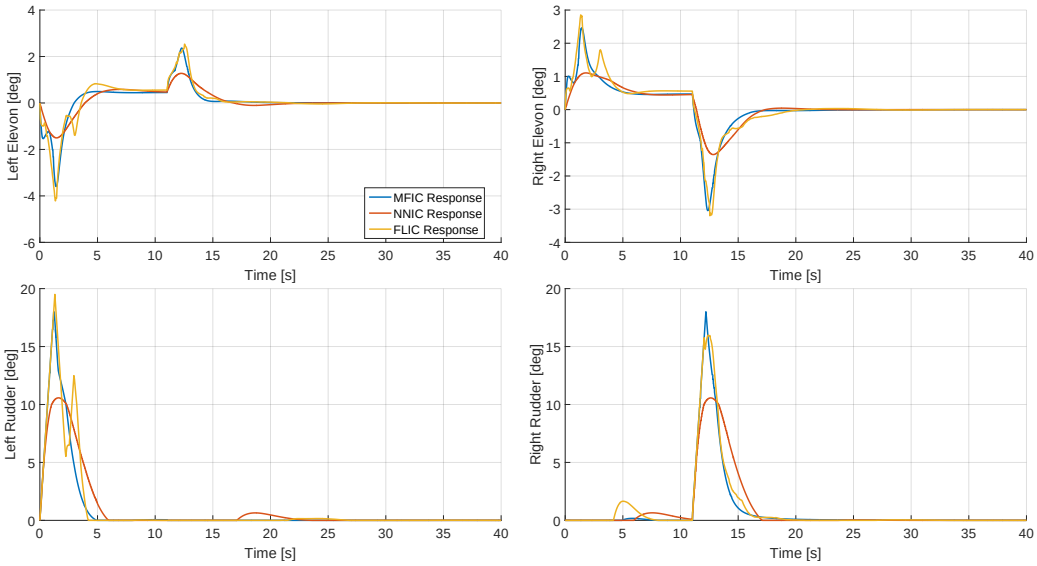


Figure 10-19: Control surface response comparison between the MFIC, the NNIC, and the FLIC, for the bank-reversal of case five.

The control response corresponding to the state response of Figure 10-18 is shown in Figure 10-19. It can be seen that, unlike the FLC, the control surfaces of the FLIC does not contain high frequency oscillations, except for those in response to the induced sideslip angle. Furthermore, it can be seen that the MFIC and the FLIC have sharper responses than the NNIC and, therefore, it is expected that the latter will have smaller fluctuation values than the former two. The smoothness of the control response of the NNIC is identical in origin to those of its NNC counterpart.

The distributions of all integrated state errors are shown in Figure 10-20. It can be seen that the FLIC has lower integrated error for the bank angle than the MFIC and the NNIC, as was expected based upon Figure 10-18. Furthermore, it can be seen that the angle of attack responses of the FLIC are sensitive to the system variations. Further inspection reveals that this is due to rapidly

varying longitudinal gain values in response to large pitch rate and angle of attack errors, which causes the longitudinal FLC to be sensitive to system variations. For the sideslip angle, the FLIC has larger a larger distribution width than the MFIC and the NNIC. Further inspection reveals that this occurs because the yaw controller has trouble eliminating the sideslip angle, which causes it to oscillate. This oscillation is sensitive to system variations, because they can either reduce the oscillation, but causing the remaining error to take longer to dissipate, or can increase the oscillations when the controller reacts too strongly. In either case, the integrated error of the sideslip angle responses change significantly, making it sensitive to system variations.

For the roll and yaw rate distributions of the NNIC, the large width can be explained by the nature of the bank angle overshoot, which is sensitive to changes in the moment coefficients of the system, and may increase or decrease the overshoot. The change in overshoot causes a corresponding increase or decrease in the roll and yaw rates which, due to the fact that they are proportional to the time-derivative of the bank angle, are sensitive to small changes in the overshoot. Because the overshoot of the MFIC and the FLIC is less than that of the NNIC, they are less affected by the system variations. The larger distribution width of the yaw rate for the FLIC is caused by the oscillations in its response due to the increasing sideslip angle. For the pitch rate, the distribution width for the FLIC is directly related to the sensitivity of the corresponding angle of attack.

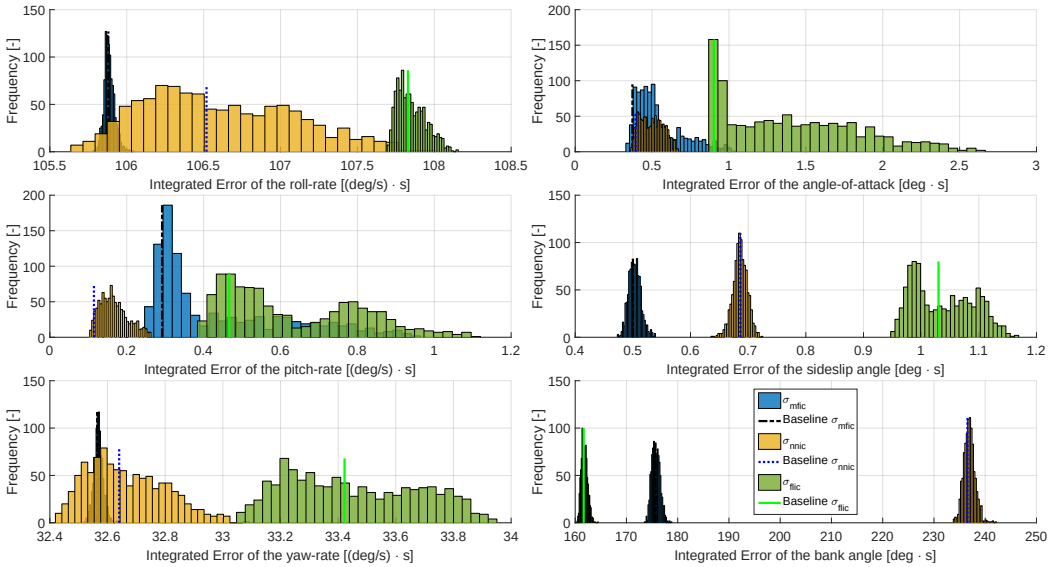


Figure 10-20: Comparative analysis of the sensitivity of the state variables of the MFIC, the NNIC, and the FLIC, for the bank-reversal of case five.

For the sensitivity of the control effort of all control surfaces, the distribution widths and values are similar for the NNIC and the MFIC, as shown in Figure 10-21. This is a curious observation, because the nominal control responses for both controllers are different. Further inspection reveals the added control effort by the sharper responses of the MFIC is equal to that of the slower control responses by the NNIC. Furthermore, the rudder distribution widths of all three controllers are also similar, but the control effort is shifted to higher values for the FLIC, which is as expected from Figure 10-19. Additionally, the elevon distribution widths of the FLIC are larger than those of the other two controllers. This is caused by the elevon oscillations that were caused as a response to the increasing sideslip angle.

For all control surfaces, the nominal values of the fluctuations of the FLIC are larger than those of the NNIC and the MFIC. For both rudders, this statement remains true for all system variations. For the elevons, the fluctuation values of the NNIC are smaller than both the FLIC and the MFIC, for almost all system variations. Additionally, for the fluctuations of the elevons, the distribution

widths of all three controllers are similar. For the left rudder, the distribution width of the NNIC and the FLIC are similar, and are larger than that of the MFIC. For both, the increased sensitivity is caused by the effect of the system variations on the overshoot of the bank angle, which causes more induced sideslip angle oscillations that are corrected by the rudders and cause increasing rudder oscillations. This analysis is also true for the right rudder of all three controllers.

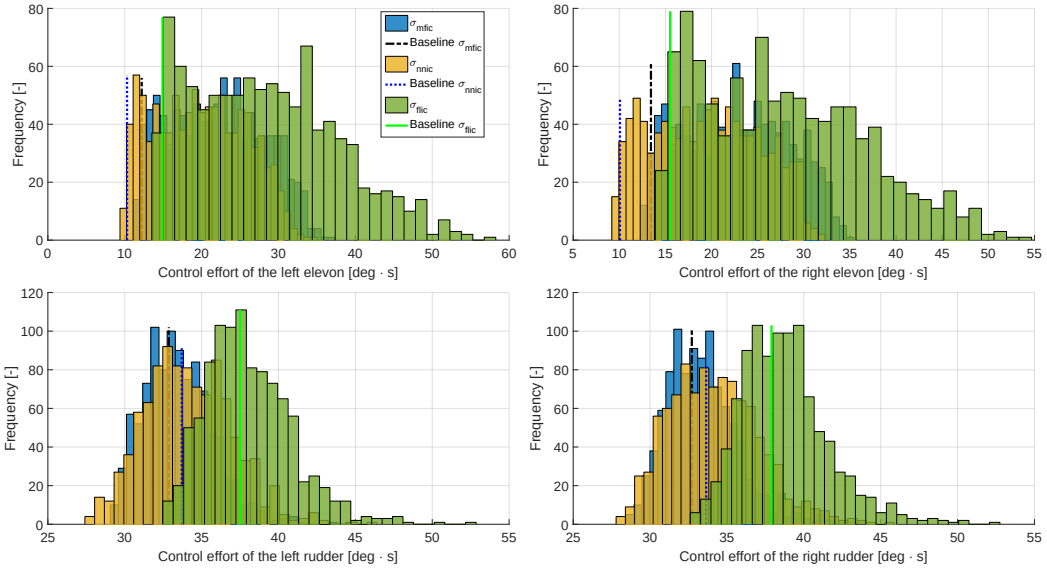


Figure 10-21: Comparative analysis of the sensitivity of the control effort of the MFIC, the NNIC, and the FLIC, for the bank-reversal of case five.

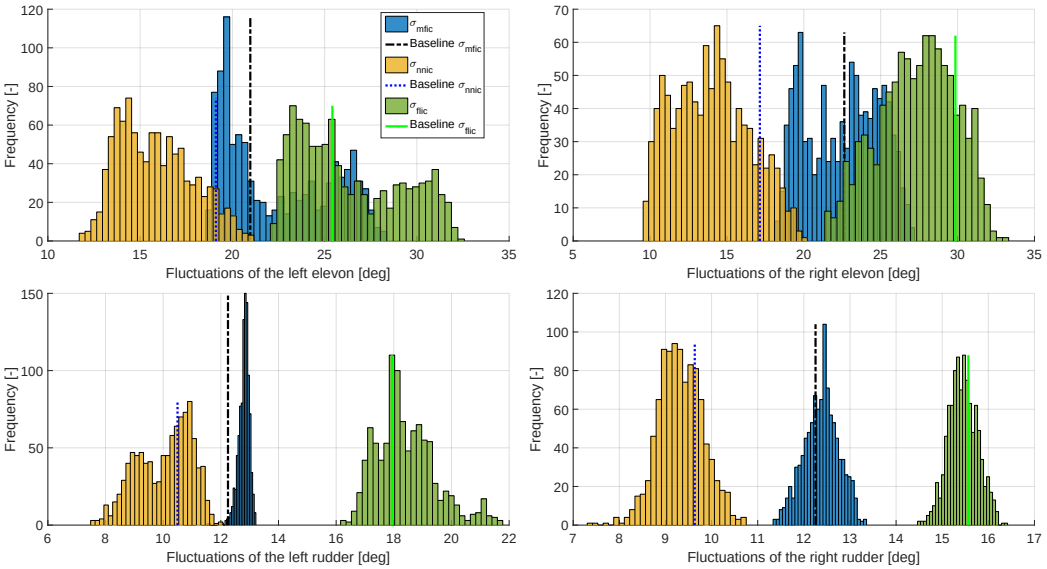


Figure 10-22: Comparative analysis of the sensitivity of the control fluctuations of the MFIC, the NNIC, and the FLIC, for the bank-reversal of case five.

It is noted that, for all aerodynamic angles, the controllers *with* integral terms performed better than their counterparts without integral terms. This statement is true for both the distribution widths and their nominal integrated errors. This statement is also true for the integrated error of the pitch rate. For the roll and yaw rate, the FLIC has the highest nominal value for all system variations, but the distributions of all other controllers overlap. Of the KBCSs discussed in this section, the NNIC has the highest overall performance and robustness.

10-4 Model Predictive Controllers

The last group of knowledge-based controllers are the MPCs. In Chapter 8, a total of eight MPCs were defined, paired in groups of two, with each group analogous to one of the benchmark controllers. However, instead of comparing all four groups to its analogous benchmark controller, only the groups analogous to the MFC and the MFIC will be comparatively analysed in this section. These two groups showed faster responses with less control fluctuations than the other two groups, and are generally less sensitive to variations in the system parameters. Coincidentally, the analysed NNCs and FLCs are also analogous to the MFC and the MFIC. In this section, first the responses of the MPCs analogous to the MFC will be discussed, after which the results of the MPCs analogous to the MFIC will be presented.

In terms of implementation effort, the MPCs required additional effort to define the matrices relating to the constraints. Additionally, four summation parameters required tuning in addition to the state and control weighting-matrices. However, these matrices have straightforward definitions and the (integer) values of the summation parameters are easily found through iterations. Hence, the implementation effort of an MPC increases only marginally compared to that of the benchmark controllers.

Comparison of MPCs Analogous to the MFC

Figure 10-23 shows the comparative analysis of the responses of the state variables of the MFC and the MPCs analogous to it. It can be seen that the bank angle responses of the MPCs are quicker than that of the MFC. Further inspection reveals that the IO-model MPC is marginally quicker than its IIO-model counterpart. Additionally, it can be seen from the response of the angle of attack that both MPCs settle to a constant error during the constant rate phase, with the IIO-model having a smaller absolute error during the same phase.

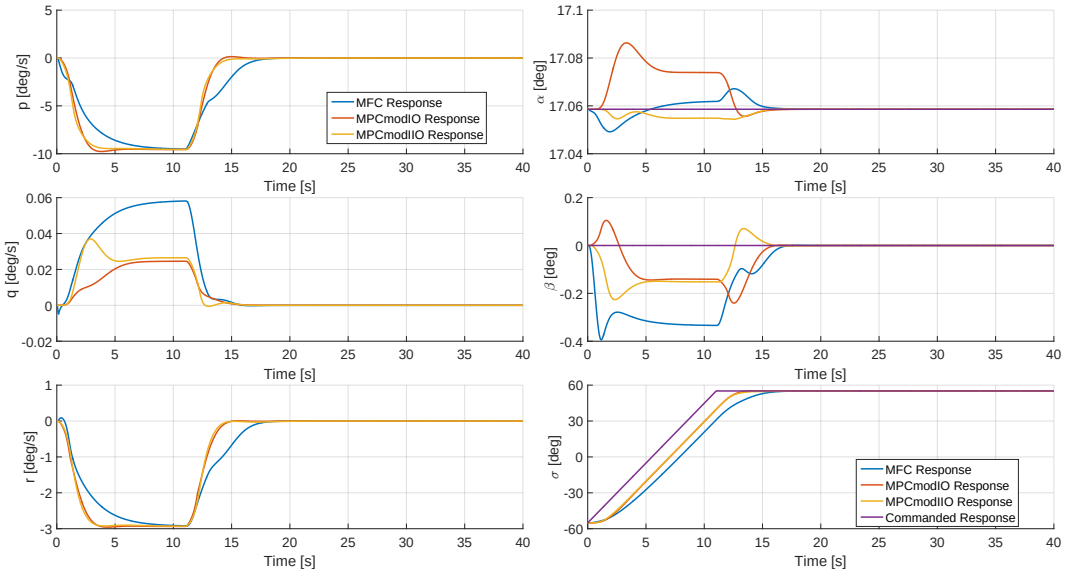


Figure 10-23: State response comparison between the MFC, the MPC_{IO}^{mfc}, and the MPC_{IIO}^{mfc}, for the bank-reversal of case five.

Additionally, it can be seen that similar behaviour is observed for the sideslip angle response. The constant aerodynamic angles and rates are because of the error balance the controller maintains for optimal performance. Furthermore, the response of the roll and yaw rates of the MPC_{IO}^{mfc} show a small oscillation between acceleration and deceleration, which is not shown in the responses of the MFC and the MPC_{IIO}^{mfc}. The origin of this behaviour is purely due to the finite deflection rates, which limit the rate at which the vehicle can change its accelerations.

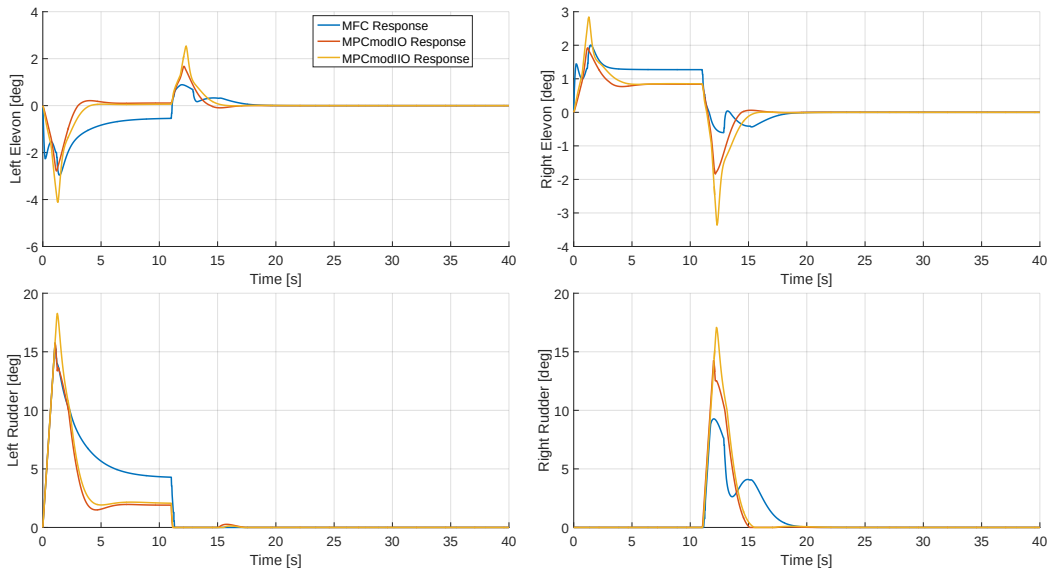


Figure 10-24: Control response comparison between the MFC, the MPC$_{IO}^{mfc}$, and the MPC$_{IIO}^{mfc}$, for the bank-reversal of case five.

Figure 10-24 shows the nominal control responses corresponding to the state responses shown in Figure 10-23. It can be seen that the control responses of both MPCs oscillate less than the responses of the MFC. Additionally, it can be seen that the IIO-model requires more control effort than its IO-model counterpart. The difference in control responses between the two MPCs can be explained by the fact that the IIO-model minimizes a performance index that is dependent on the moment-fraction rates. Because the performance index sums the moment-fraction rates over several time-steps, it effectively performs a discrete integration, with the result being the actual moment-fractions. On the other hand, the IO-model computes the performance index with a sum over the moment-fractions, thereby computing the areas of the moment-fraction curves, which grow more quickly than the moment-fractions themselves. Hence, the IO-model has a stronger contribution of the control actions to the performance index than the IIO-model, and therefore avoids the higher control values that the IIO-model reaches.

The distribution widths of all aerodynamic angles of the MPC$_{IIO}^{mfc}$ are smaller than those of its IO-model counterpart, and the widths of both controllers are smaller than those of the MFC, as shown in Figure 10-25. In fact, the distribution width for the IIO-model is so small, that the distributions are barely visible in Figure 10-29. For the sideslip angle, the distribution width of the IIO-model is ten times smaller than that of the IO-model, whereas for the bank angle, the distribution width of the IIO-model is 30 times smaller than that of its IO-model counterpart. For the angle of attack, the difference in distribution width of the IIO-model is 50 times smaller than that of the IO-model. Ghahramani and Towhidkhah (2009) claimed that using the IIO-model would improve the robustness of the controller and, hence, decrease the widths of the distributions, and this has clearly been demonstrated to be true.

For the integrated error of the roll rate, all three controllers have similar nominal values. However, due to outliers of the MPC$_{IIO}^{mfc}$, its distribution width is larger than that of the MFC. Further inspection reveals that the outliers occur because of a too tightly tuned lateral controller for the MPC$_{IIO}^{mfc}$. Still, the distribution width of the MPC$_{IO}^{mfc}$ is larger than those of both other controllers. This is because of the roll rate overshoots of the MPC$_{IO}^{mfc}$, which are effected by the system variations, and do not occur in the MFC and the MPC$_{IIO}^{mfc}$. For the distribution of the integrated error of the yaw rate, the width of the MPC$_{IIO}^{mfc}$ is smaller than that of the MPC$_{IO}^{mfc}$, and both have smaller widths than that of the MFC. Similar to the roll rate, the yaw rate distribution of the MPC$_{IIO}^{mfc}$ has outliers due to the too tightly tuned lateral controller.

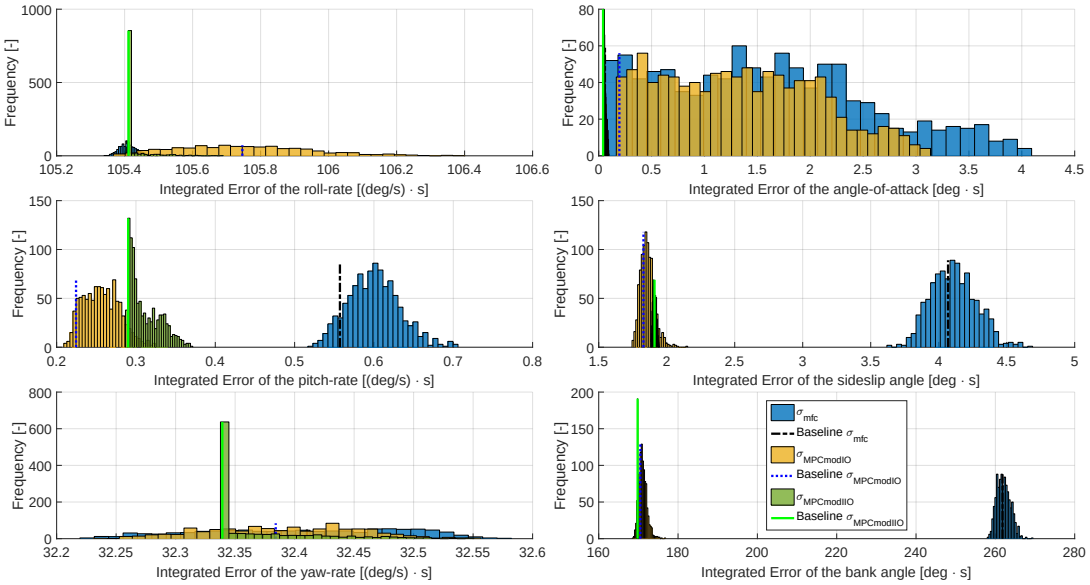


Figure 10-25: Comparative analysis of the sensitivity of the state variables of the MFC, the MPC_{IO}^{mfc}, and the MPC_{IIO}^{mfc}, for the bank-reversal of case five.

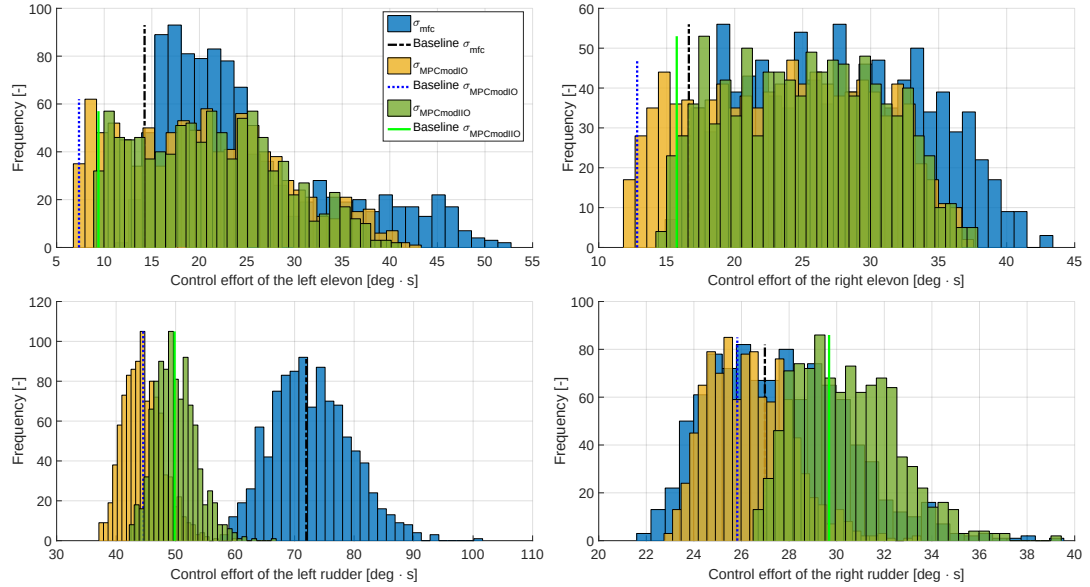


Figure 10-26: Comparative analysis of the sensitivity of the control effort of the MFC, the MPC_{IO}^{mfc}, and the MPC_{IIO}^{mfc}, for the bank-reversal of case five.

The distributions of the control effort for the MPCs are similar in width and values, as shown in Figure 10-26. Additionally, for all control surfaces, the nominal values of the MPCs are smaller than those of the MFC, but the distributions do overlap. Furthermore, the control effort distributions of the MPC_{IIO}^{mfc} are shifted to higher values than the MPC_{IO}^{mfc}, which is expected based upon the nominal control responses of Figure 10-24.

As shown in Figure 10-27, the elevon fluctuation distributions for both MPCs are similar, with the nominal values for the IIO-model higher than those of the IO-model, which is expected based upon its stronger control responses, as shown in Figure 10-24. The elevon fluctuation distributions

widths of both MPCs are similar to each other, and less than that of the MFC. Additionally, the width of the left rudder fluctuation distribution of the MPC_{IIO}^{mfc} is larger than that of its IO-model counterpart, whereas for the right rudder, both MPCs have similar distribution widths which are smaller than that of the MFC.

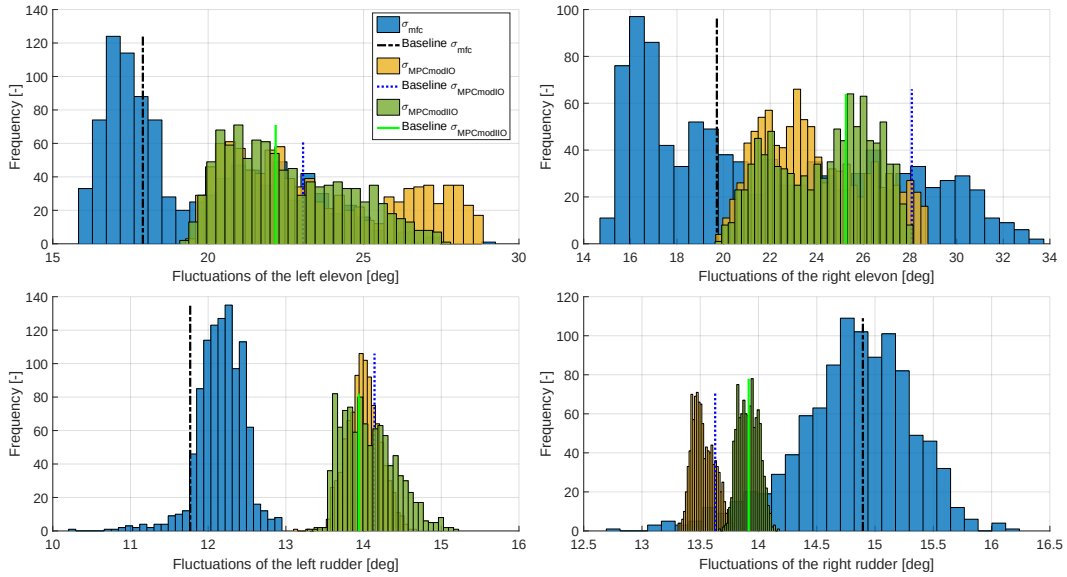


Figure 10-27: Comparative analysis of the sensitivity of the control fluctuations of MFC, the MPC_{IO}^{mfc}, and the MPC_{IIO}^{mfc}, for the bank-reversal of case five.

Comparison of MPCs Analogous to the MFIC

The state responses of the MPCs analogous to the MFIC is shown in Figure 10-28. It can be seen that the bank angle response of the MFIC and the MPCs are nearly identical. Further inspection reveals that the IIO-model is marginally faster than the MFIC and the MPC_{IO}^{mfic}, and contains no overshoot, unlike the other two controllers. The MPC_{IO}^{mfic} has a larger overshoot than the MFIC, but has a lower settling time. From the sideslip angle response, it can be seen that the MPC_{IO}^{mfic} oscillates similar to the MFIC, but do so with smaller amplitudes. This is because the former responds slightly slower than the latter, and, hence, the lateral MPC_{IO}^{mfic} is better able to eliminate the sideslip angle. For the MPC_{IIO}^{mfic} the sideslip angle oscillates more than the other two controllers, which is due to it responding more strongly, because of the IIO-model assigns larger control deflections than the other controllers. This is similarly reflected by the oscillations in the roll and yaw rates of the MPC_{IIO}^{mfic}. From the angle of attack response, the MPC_{IO}^{mfic} has a larger amplitude than its IIO-model counterpart, and both have a smaller amplitude than the MFIC. Additionally, the MPC_{IIO}^{mfic} eliminates the error faster than the other two controllers. The differences in amplitudes are directly related to the difference in the responses of the roll and yaw rates. This is because, even though the coupling between the longitudinal and lateral dynamics is not strong, a coupling does exist for a non-zero sideslip angle (see Equation A-7)

The pitch rate response shows that the larger amplitudes of the angle of attack response of the MPC_{IIO}^{mfic} corresponds with larger amplitudes of the corresponding pitch rate. Additionally, the angle of attack response of the MPC_{IO}^{mfic} shows similar fluctuating behaviour similar to that of the MFIC, but does so with smaller amplitudes. For the roll and yaw rates, the MPC_{IO}^{mfic} shows similar overshoot as the MFIC, with the MPC_{IIO}^{mfic} showing an overshoot, the cause of which has been explained in the previous paragraph. Overshoots in the roll and yaw rates suggests more fluctuations in the control surfaces to facilitate the oscillations between acceleration and deceleration.

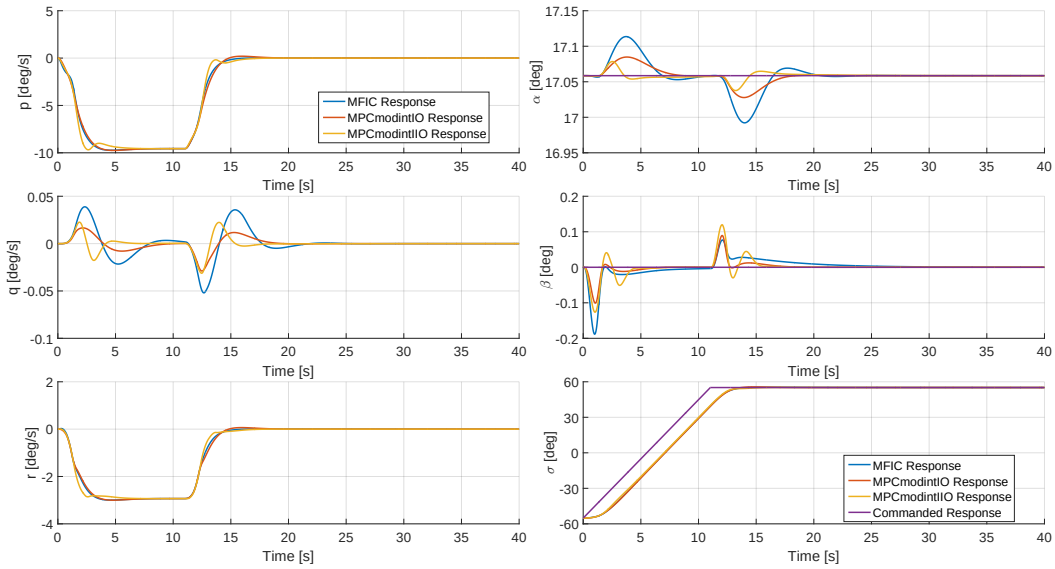


Figure 10-28: State response comparison between the MFIC, the MPC$_{IO}^{mfic}$, and the MPC$_{IIO}^{mfic}$, for the bank-reversal of case five.

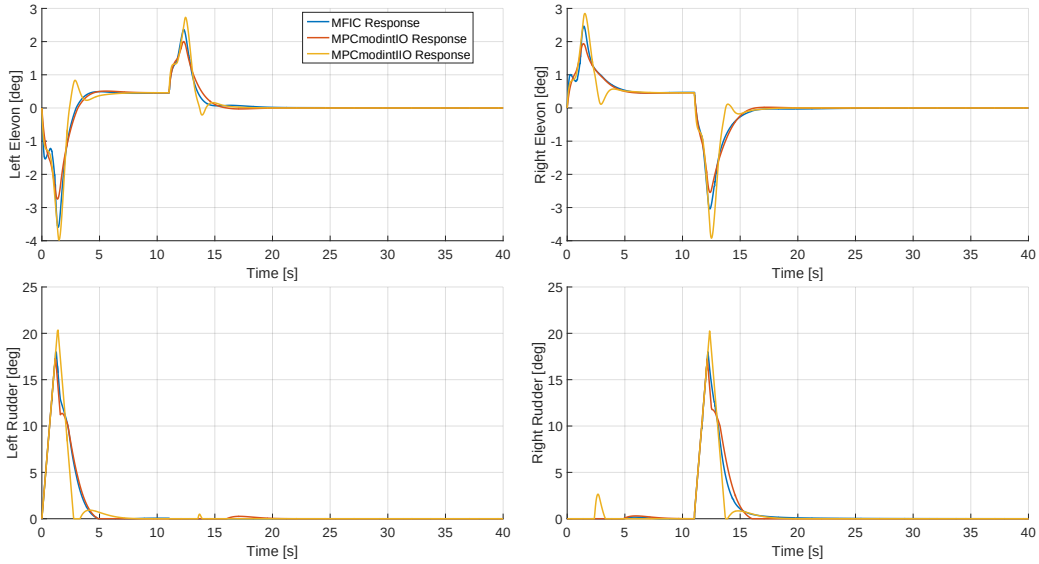


Figure 10-29: Control response comparison between the MFIC, the MPC$_{IO}^{mfic}$, and the MPC$_{IIO}^{mfic}$, for the bank-reversal of case five.

Figure 10-29 shows the nominal control responses corresponding the state responses of Figure 10-28. It can be seen that, for all control responses, the MPC$_{IO}^{mfic}$ is similar to those of the MFIC, albeit with lower amplitude. As mentioned before, the MPC$_{IIO}^{mfic}$ assigns larger control deflections than the other two controllers, which is seen in Figure 10-29 for all control deflections. Furthermore, it can be seen that the control responses of the MPC$_{IIO}^{mfic}$ oscillate more than its IO-model counterpart. Similar to what was found for the oscillating control responses of the FLIC, these oscillations are sensitive to system variations, and it is, therefore, expected that the distribution width of the MPC$_{IIO}^{mfic}$ is larger than that of the other two controllers.

From the comparative analysis of the state variable sensitivity of the controllers, as shown in Figure 10-31, it is found that the distribution width of the integrated errors of the bank angle for the MPC$_{IIO}^{mfic}$ is smaller than those of the other controllers. For the sideslip angle distribution

of the MPC_{IIO}^{mfic} the width is larger than those of both other two controllers, whereas for the angle of attack distribution, its width is comparable to that of the MPC_{IO}^{mfic}. This is in contrast with what was seen for the IIO-model analogue of the MFC, where the use of the IIO-model did significantly decrease the sensitivity compared to that of the IO-model. This comparison shows that it is not necessarily guaranteed that the use of an IIO-model will improve robustness for all state variables. Further comparison with the MFC analogues show that the distribution widths of the MPC_{IO}^{mfic} are, at worst, twice as large as that of the MPC_{IIO}^{mfc}, which is much less than the difference between the two MFC analogues. Furthermore, the pitch rate distributions of the MPCs are similar, and both distribution widths are smaller than that of the MFIC. Furthermore, the distribution widths of the roll and yaw rates of the MPC_{IIO}^{mfc} is four times less than those of its IO-model counterpart, which is due to the fact that the IIO-model does not have an overshoot, and, hence, is not affected by the bank angle overshoot sensitivity to system variations causing increases in the integrated error of the roll and yaw rates. However, particularly for the roll rate, this is offset by the rate overshoot during the end of the acceleration and deceleration phase, which was shown to be sensitive to system variations in the comparative analysis of the MPC analogues of the MFC.

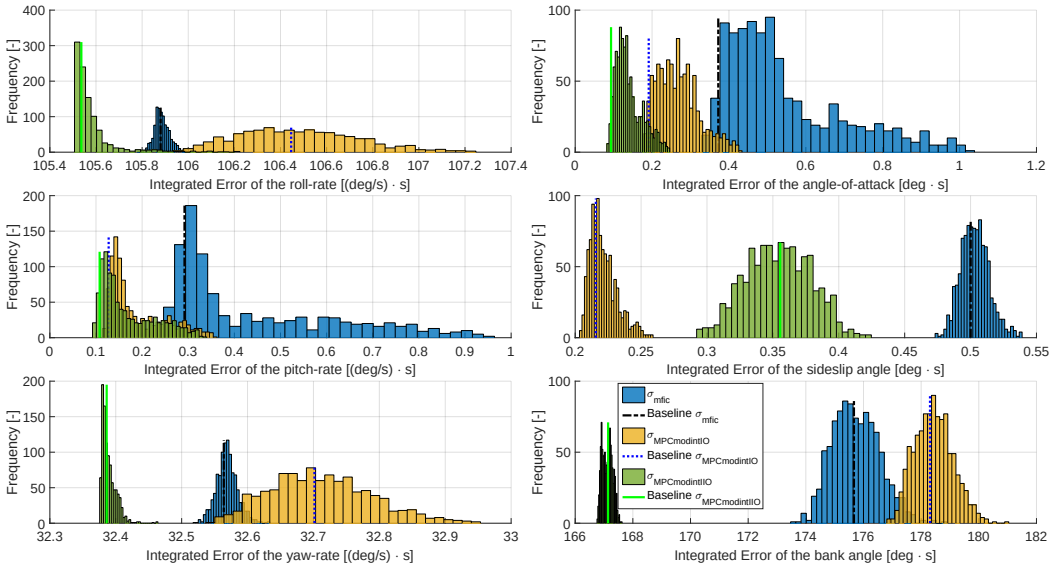


Figure 10-30: Comparative analysis of the sensitivity of the state variables of the MFIC, the MPC_{IO}^{mfic}, and the MPC_{IIO}^{mfic}, for the bank-reversal of case five.

The control effort distributions are shown in Figure 10-31. It can be seen that the elevon distributions are nearly identical for all three controllers. For the rudders, this statement also holds for the MFIC and the MPC_{IO}^{mfic}, this is not unexpected, based upon the fact that the nominal control responses are very similar, as seen in Figure 10-29. The rudder distribution widths of the MPC_{IIO}^{mfic} is larger than those of the other controllers, which is due to sideslip angle oscillation, which is sensitive to system variations, and causes the rudder control effort to increase and oscillate more than the other two controllers, for the same system variations.

Based on the observations of Figure 10-29 it was expected that the MPC_{IIO}^{mfic} will have higher fluctuation values than the other two controllers, as is shown in Figure 10-32. It can be seen that this statement is not true for all system variations. Additionally, the elevon fluctuation distributions of the MPCs are (nearly) identical in distribution widths, although the IIO-model is shifted to larger control effort values. Furthermore, the left elevon fluctuation distributions widths of the MPCs are similar to those of the MFIC, whereas for the right elevon, the distribution width of the MFIC is smaller than those of the MPCs. For the rudder fluctuation values, the MPC_{IIO}^{mfic} has larger distribution widths than the other two controllers, which is due to the sensitivity of the sideslip angle to system variations.

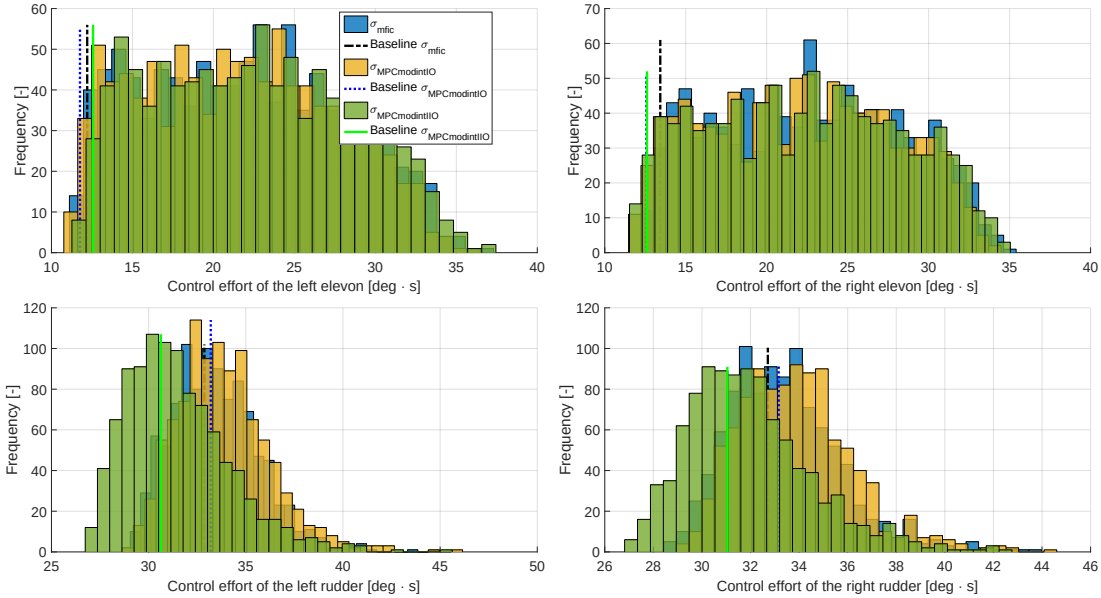


Figure 10-31: Comparative analysis of the sensitivity of the control effort of the MFIC, the MPC_{IO}^{mfic}, and the MPC_{IIO}^{mfic}, for the bank-reversal of case five.

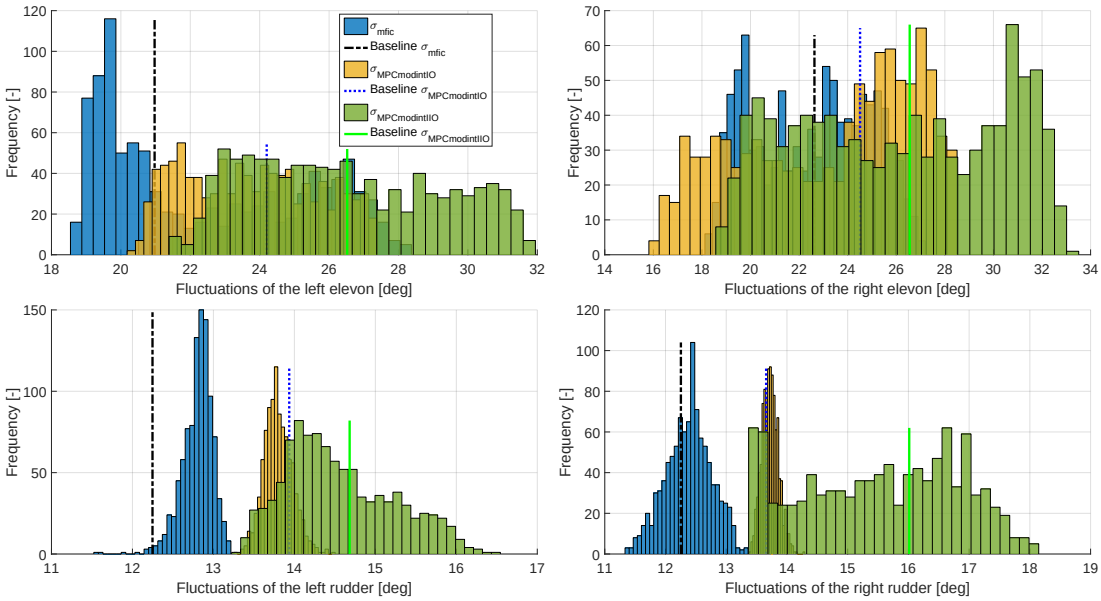


Figure 10-32: Comparative analysis of the sensitivity of the control fluctuations of MFIC, the MPC_{IO}^{mfic}, and the MPC_{IIO}^{mfic}, for the bank-reversal of case five.

Further inspection of all MPCs reveals that the MPC_{IIO}^{mfc} and the MPC_{IO}^{mfic} compete for the best performing MPC. The former has a lower integrated error of the bank angle than the later, but this comes at the cost of a (much) greater integrated error for the sideslip angle. On average, the control effort and fluctuations of both MPCs are similar. Additionally, the selection of the weighting-matrices parameters for the MPC_{IIO}^{mfc} proved more troublesome that those for the MPC_{IO}^{mfic}, which increased the implementation effort of the former. From these arguments, it is included that the MPC_{IO}^{mfic} outperforms the MPC_{IIO}^{mfc}, and, hence, the former is selected as the best performing MPC.

10-5 The Best of Both Worlds

In the previous sections, the benchmark controllers, the NNCs and FLCs, and the MPCs were comparative analysed. Based on these separate analyses it is difficult to determine which control methodology has the highest performance overall. Hence, the best controllers of all three sections will be analysed together in greater detail than the analyses of the previous sections. For the benchmark controllers, it was found that the MFIC performs better than its colleague benchmark controllers. For the NNCs and the FLCs, it was found that the NNIC has the best overall performance. Likewise, the MPC_{IO}^{mfic} was found to be the best overall performing controller among the MPCs. It is noted that the NNIC and the MPC_{IO}^{mfic} have already been comparatively analysed to the MFIC in Sections 10-3 and 10-4, respectively, but not to each other. Hence, this section will focus the differences between the NNIC and the MPC_{IO}^{mfic}.

Instead of focusing on only the bank-reversal of case five, it is decided to also analyse the behaviour in response to a one degree disturbance in the angle of attack and sideslip angle, for the conditions right before the bank-reversal of case five. Additionally, a five degree bank angle disturbance will be analysed at a point along the reference trajectory where all control actuators, including the thrusters, are active simultaneously. In this manner, a detailed analyses can be performed of the performance of the three selected controllers for a variety of settings. It is noted that, for the angle of attack and sideslip disturbances, the same tuning is used for each of the controllers, whereas for the hybrid control point, the controllers are retuned to adapt to the changes in the environmental conditions. For the MFIC, this hybrid case in included in the weighting matrices parameters of Table B-5 and B-6. The NNIC is retrained and the resulting connection weights are shown in Table B-8. For the MPC_{IO}^{mfic}, the hybrid case is added to Table B-15.

Angle of Attack Disturbance

Figure 10-33 shows the longitudinal state variable responses of the three selected controllers for a one degree disturbance in the angle of attack. Only the longitudinal states are shown, because the lateral states are unaffected by the dynamics of the longitudinal states when the roll and yaw rates, and the sideslip angle are zero. Because these lateral states are zero at the start of the simulation, they will remain zero. In practice, the lateral states will not deviate from their initial value more than the floating-point error of the system.

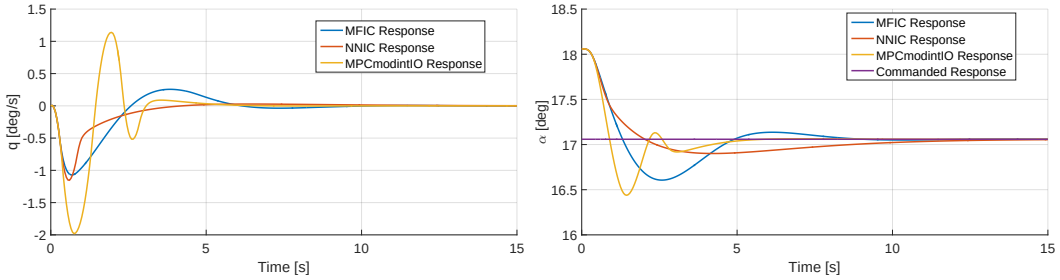


Figure 10-33: State response comparison of the MFIC, the NNIC, and the MPC_{IO}^{mfic}, for a one degree angle of attack disturbance at case five.

From Figure 10-33 it can be seen that the MPC_{IO}^{mfic} eliminates the error faster than the other two controllers, which corresponds with the larger amplitude of its pitch rate response. It can do this because the MPC_{IO}^{mfic} responds more strongly to errors in the angle of attack than the other two controllers, and less to errors in the pitch rate. The combined effect results in a sharper and faster response to eliminating the error. Furthermore, while the NNIC eliminates the error the slowest, it shows no oscillations about the commanded angle of attack and pitch rate. The slowness of the NNIC response is due to the exact same reason as was discussed in Section 10-3. The reason why the NNIC takes the longest to fully eliminate the error is due to the integral term, because it takes time to eliminate the accumulated integral term. This last argument is also true for the MFIC.

It can be seen from Figure 10-34 that the control deflections for both elevons are identical. This is expected, because only longitudinal motion occurs and, therefore, the elevons combine to perform only their elevator function. Furthermore, it can be seen that the MPC_{IO}^{mfic} has the largest control effort values of the three selected controllers. Based on its state response of Figure 10-33, this is not unexpected. Additionally, the oscillations of the elevons correspond directly with the oscillations of the pitch rate. The elevon responses for the NNIC are initially similar to that of the MFIC, but due to the integral term of the NNIC, it settles on a slowly decreasing response. For the MFIC, the elevon response shows an oscillation that is directly related to that of the angle of attack.

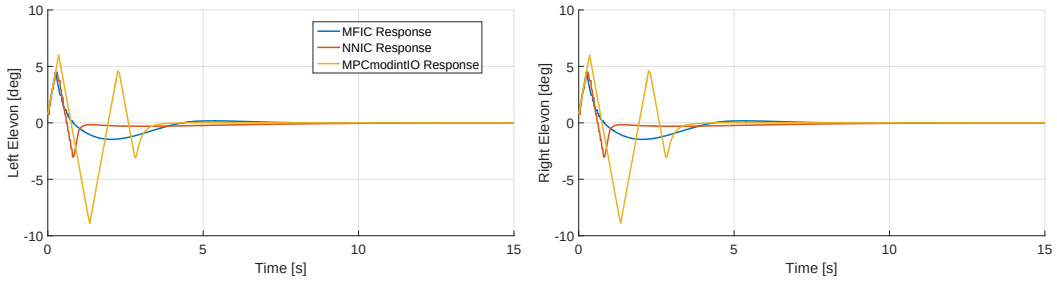


Figure 10-34: Control response comparison of the MFIC, the NNIC, and the MPC_{IO}^{mfic}, for a one degree angle of attack disturbance at case five.

In this scenario, the MPC_{IO}^{mfic} has a better nominal performance than the other two controllers, in terms of eliminating the state errors. This increased performance does come at the price of increased control effort. The NNIC has the lowest control effort, but also eliminates the angle of attack error the slowest of the three controllers.

It is noted that, for this scenario, no sensitivity analysis will be performed. This is because it will only reveal the sensitivity of the longitudinal states on their own, without being simultaneously influenced by the dynamics of the lateral states. While this is useful if one wants to determine the individual contributions to the sensitivity distributions, it is not during bank-reversals when the effects of the coupling between the longitudinal and lateral dynamics is relatively strong.

Sideslip Angle Disturbance

Figure 10-35 shows the state variable responses of the three selected controllers for the one degree disturbance of the sideslip angle. It can be seen that the elimination of the sideslip angle error induces a relatively large bank angle error. The NNIC takes the longest to eliminate the sideslip angle error and the induced bank angle error, which is due to its slow responses and limits the vehicle in its acceleration and deceleration, thereby causing the overshoot. Additionally, the small values for the integral terms of the roll-axis NNIC do not help in eliminating the error quickly. The yaw rate response of the MPC_{IO}^{mfic} is due to it responding to the increasing lateral integral terms, which assigns a larger moment-fraction about the yaw-axis, thereby increasing the amplitude of the yaw rate oscillation. This also induces an error in the bank angle response, which explains its difference compared to the responses of the other two controllers. The large error of the pitch rate for the MFIC is due to the longitudinal controller not being very sensitive to small errors of the longitudinal states, which causes the elevons to assume deflections that induces an even larger pitch rate. The origin of the slow decrease in the pitch rate is due to the small gain of the longitudinal integral term of the MFIC, which takes time to reduce to zero.

The nominal control responses corresponding the state responses of Figure 10-35, are shown in Figure 10-36. It can be seen that each controller employs a different rudder strategy to eliminate the sideslip angle error, and the errors induced by that attempt. For the MPC_{IO}^{mfic}, after one second, it starts using its left rudder to limit the overshoot of the sideslip angle, which in turn induces a larger amplitude of the yaw rate. Simultaneously, the bank angle error grows, which causes the right rudder to deflect. For the NNIC, the right rudder first increases, after which

decreases and increases again, in response to the decreasing sideslip angle error. The left rudder of the NNIC starts to get allocated at the time when the roll rate switches signs, which in turn causes the bank angle error to decrease.

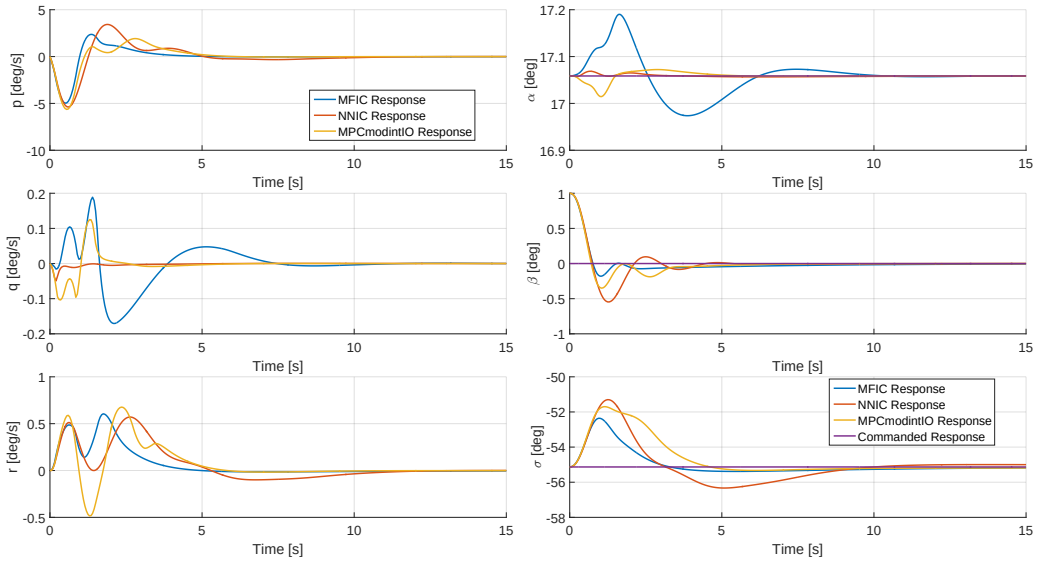


Figure 10-35: State response comparison of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for a one degree sideslip angle disturbance at case five.

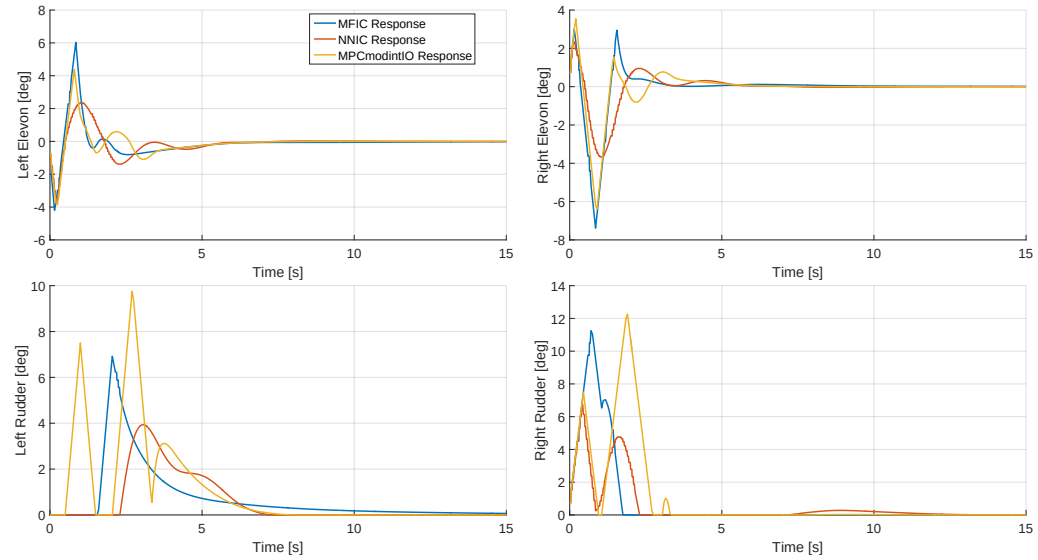


Figure 10-36: Control response comparison of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for a one degree sideslip angle disturbance at case five.

In this scenario, the MFIC has a better nominal performance than the other two controllers, in terms of eliminating the errors and the control effort. This proves that the modified benchmark controllers can still compete with the more advanced control concepts of the NNCs and MPCs.

It is noted that, for this scenario, no sensitivity analysis will be performed. This is because the sensitivity of all state variables, when the response of one affects all the responses of the others, will be shown in the next paragraph, when a comparative analysis will be made during the bank-reversal of case five. Because this scenario is played out at the same location along the reference trajectory, a comparative sensitivity analysis of this scenario would be redundant.

Bank-reversal

Figure 10-37 shows the state variable responses of the three selected controllers. It shows that the bank angle response of the MPC_{IO}^{mfic} is faster than that of the NNIC, and contains no overshoot, unlike the NNIC. From the roll and yaw rate responses, it can clearly be seen that the MPC_{IO}^{mfic} responds more strongly than the NNIC during the acceleration and deceleration phases, and does not suffer from rate overshoot. Furthermore, it can be seen from the angle of attack and sideslip angle response that the MPC_{IO}^{mfic} eliminates the errors quicker than the NNIC.

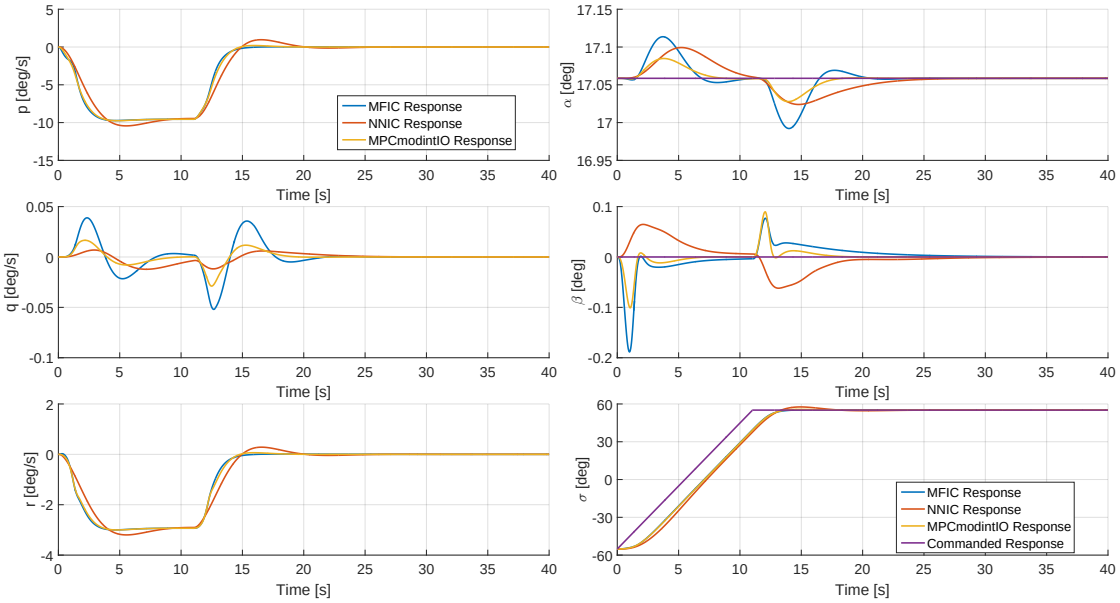


Figure 10-37: State response comparison of the MFIC, the NNIC, and the MPC_{IO}^{mfic}, for the bank-reversal of case five.

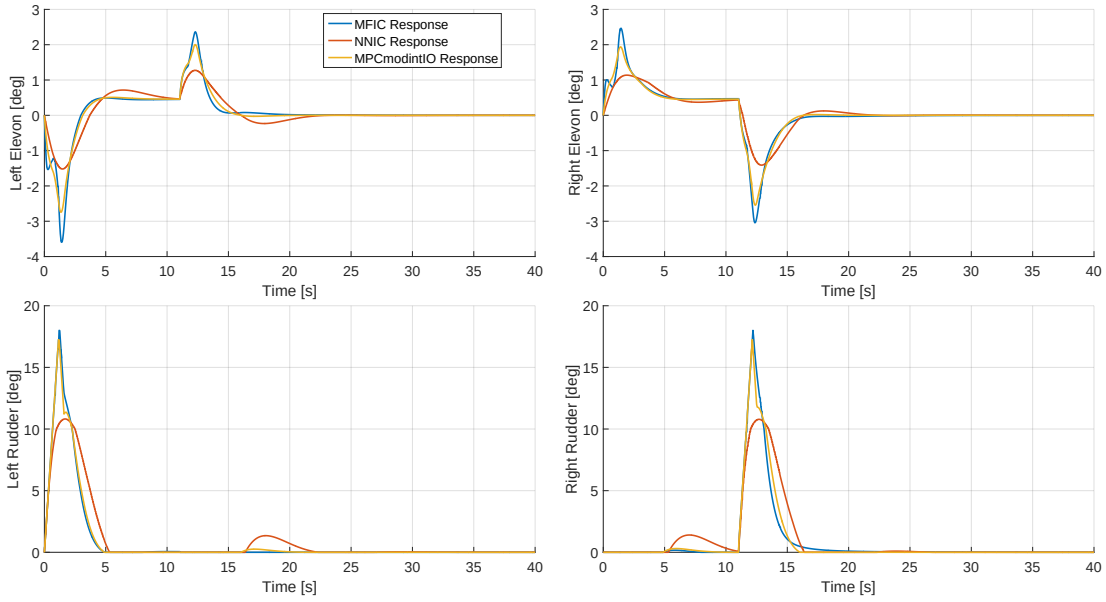


Figure 10-38: Control response comparison of the MFIC, the NNIC, and the MPC_{IO}^{mfic}, for the bank-reversal of case five.

Figure 10-38 shows the nominal control responses corresponding to the state variable responses of Figure 10-37. It can be seen that the control responses of the NNIC are less sharp than those the MPC$_{IO}^{mfic}$, and require less control deflections. This statement is true for all control actuators. It is therefore expected that the NNIC is assigned lower fluctuation values than the MPC$_{IO}^{mfic}$.

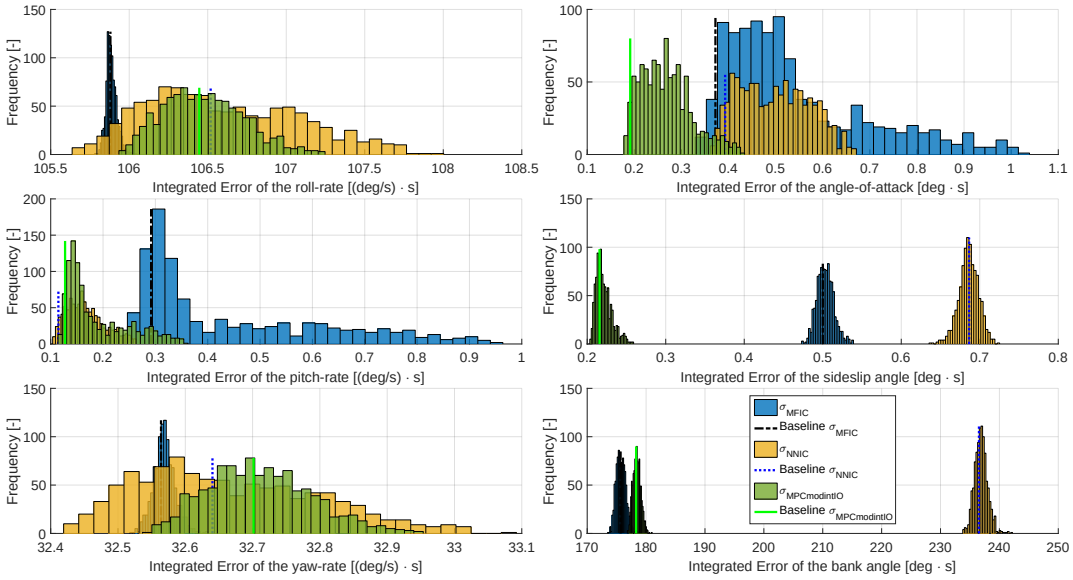


Figure 10-39: Comparative analysis of the sensitivity of the state variables of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for the bank-reversal of case five.

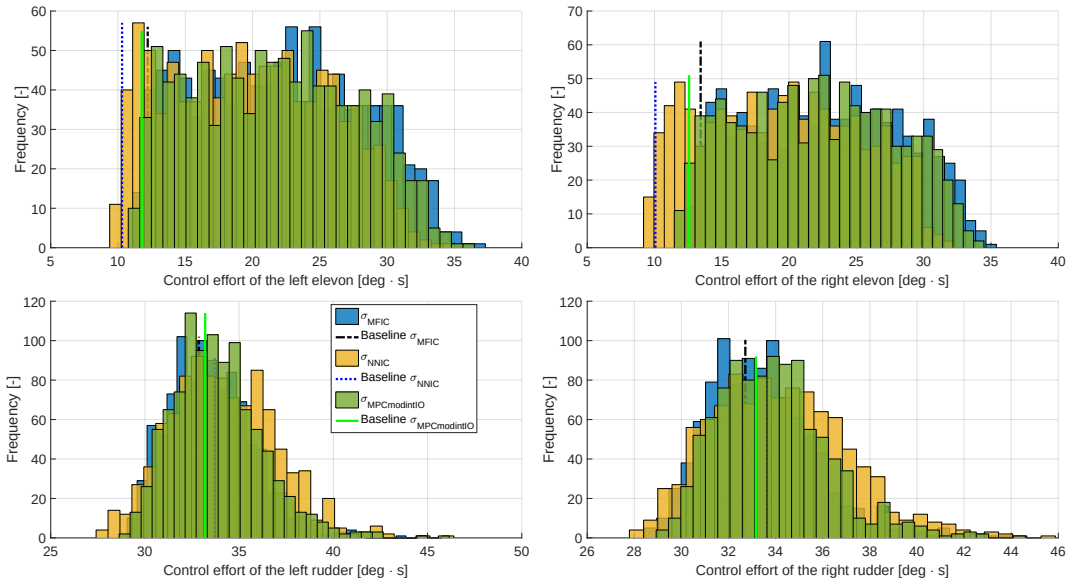


Figure 10-40: Comparative analysis of the sensitivity of the control effort of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for the bank-reversal of case five.

Making a comparative analysis of the sensitivity of the three selected controllers will reveal which controller is the most robust to system variations. Figure 10-39 shows the histograms of the integrated errors of the state variables. It can be seen from the bank angle distributions that the nominal value of the MFIC is lower than those of the FLIC and the MPC$_{IO}^{mfic}$. Combined with the corresponding bank angle responses, it can be concluded that a small change in the response

behaviour can induce a large difference in the values of the integrated errors. In practice, the response differences between two extremes of a distribution are not visible when shown together in a single figure. Furthermore, the distribution widths of the MPC_{IO}^{mfic} are smaller for all lateral variables than those of the NNIC. For the longitudinal variables, the widths of the distributions of the MPC_{IO}^{mfic} are similar to those of the NNIC.

It can be seen from Figure 10-40 that the NNIC and the MPC_{IO}^{mfic} have similar distribution widths and, hence, are similarly sensitive to system variations. In fact, this statement is true when comparing all three controllers. Furthermore, the nominal elevon control effort of the MPC_{IO}^{mfic} are larger than those of the NNIC, but are smaller than those of the MFIC. For the rudder control effort, the roles of the nominal values are reversed; the MPC_{IO}^{mfic} has a larger value than the MFIC, but a smaller value than the NNIC.

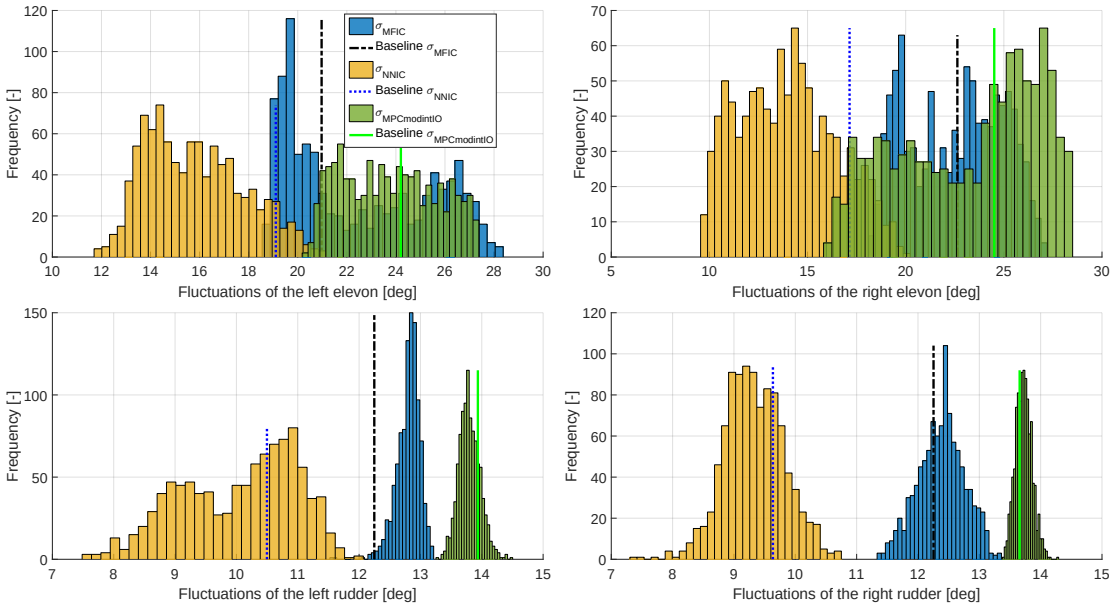


Figure 10-41: Comparative analysis of the sensitivity of the control fluctuations of the MFIC, the NNIC, and the MPC_{IO}^{mfic}, for the bank-reversal of case five.

Besides the control effort of Figure 10-40, the *effective* control effort will still be large when the control surface fluctuates rapidly with small amplitudes. The comparative analysis of the sensitivity of the fluctuation values is shown in Figure 10-41. It can be seen that the MPC_{IO}^{mfic} is assigned larger control fluctuation values for all control surfaces, and the NNIC the lowest. This is not unexpected based upon the control responses shown in Figure 10-38. For all control surface fluctuations, the NNIC is more sensitive to system variations than the MPC_{IO}^{mfic}. Except for the right rudder, the distribution widths of the MPC_{IO}^{mfic} are smaller than those of the MFIC. Hence, on average, the MPC_{IO}^{mfic} shows the least control fluctuations.

Hybrid Control

The performance of the controllers will additionally be evaluated at a point along the reference trajectory where all control actuators, including the thrusters, are active. The initial conditions and parameters of this point are shown in Table 10-3. Evaluating the performance and sensitivity of the controllers at another point along the reference trajectory, with a different set of active actuators, increases the confidence in the conclusions that will be made about the performance and robustness of each of the controllers.

Table 10-3: Overview of the initial conditions and parameters of the hybrid case.

Parameter	Value
Dynamic Pressure [Pa]	250.05
Mach Number [-]	27.00
Cmd. angle of attack [deg]	40.00
Initial Bank angle [deg]	5.00
Final Bank angle [deg]	0.00

Figure 10-42 shows the state variable responses of the three selected controllers to a five degree disturbance of the bank angle at the hybrid case. It can be seen that the MFIC and the NNIC show a similar response, but the former has a larger overshoot than the latter. As a result, the NNIC has a settling time similar to that of the MPC_{IO}^{mfic}, which has no overshoot. The faster elimination of the bank angle error by the NNIC comes at the price of a much larger sideslip angle error. Further inspection reveals that the yaw-axis NNIC optimized in such a manner that it mostly ignored the contribution to its control action due to an bank angle error. The sideslip angle error of the MFIC and the MPC_{IO}^{mfic} has a maximum amplitude of 0.02 degrees and has smoothly decayed to a tenth of the maximum amplitude after approximately ten seconds. Furthermore, it can be seen that the angle of attack response of the NNIC has the largest amplitude, which is induced by the large error in the sideslip angle. Although the NNCs have a tendency to react slower than the other controllers, this is not the case here, because from Figure 10-43 it can be seen that both elevons deflect at their maximum rate of 15 degrees per second. Furthermore, it can be seen that the NNIC has a leftover steady-state error. This is due to the a small gain value for the longitudinal integral term, effectively nullifying its effect.

Additionally, it can be seen that the NNIC has the largest pitch rate amplitude, which is due to the large induced angle of attack error. For the MPC_{IO}^{mfic}, it can be seen that the roll and yaw rate amplitudes are the smallest, which is due to its smooth response in eliminating the error in the bank angle. The roll and yaw rates of the MFIC show a small overshoot corresponding to the bank angle response overshoot. For the roll and yaw rates of the NNIC, oscillations occur, which is due to the error in the sideslip angle causing the error balance to shift.

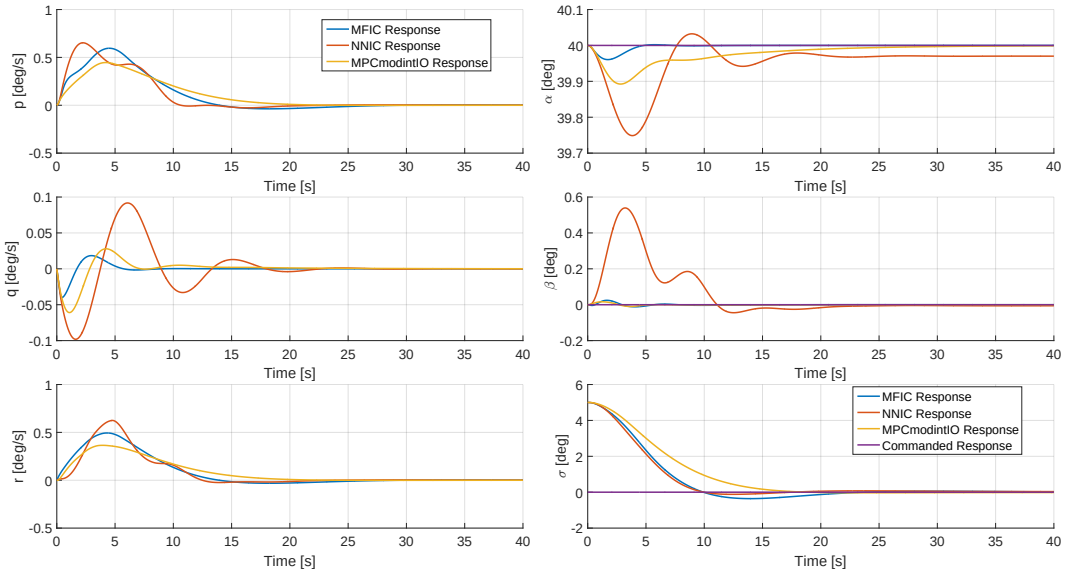


Figure 10-42: State response comparison of the MFIC, the NNIC, and the MPC_{IO}^{mfic}, for a five degree bank angle disturbance for the hybrid case.

From Figure 10-43 it can be seen that, at the start of the simulation, all control surfaces are changing their deflection values at their maximum rates. For the elevon responses of the NNIC, it can clearly be seen that they oscillate, which causes the oscillations in the roll and yaw rates. The elevon responses of the MPC$_{IO}^{mfic}$ shows small oscillations, which are in response to the angle of attack error. For the MFIC, a larger oscillation occurs for the left elevon, which is due to a strong response to the induced error in the angle of attack. This oscillation is also responsible for the small oscillation in its roll rate response. Additionally, it can be seen that the elevons settle to a non-zero value. This is because, at the combination of dynamic pressure and Mach number, the body-flap, which controls the trim stability, is not active and, hence, does not have the correct deflection to ensure trim stability. This will result in an pitch moment coefficient offset, which is corrected by a non-zero deflection of the elevons. It is noted that this is not due to the trim algorithm, but purely due to the controllers. Both elevons have the same non-zero deflection, although it may not appear this way in Figure 10-43 due to the difference in vertical scaling.

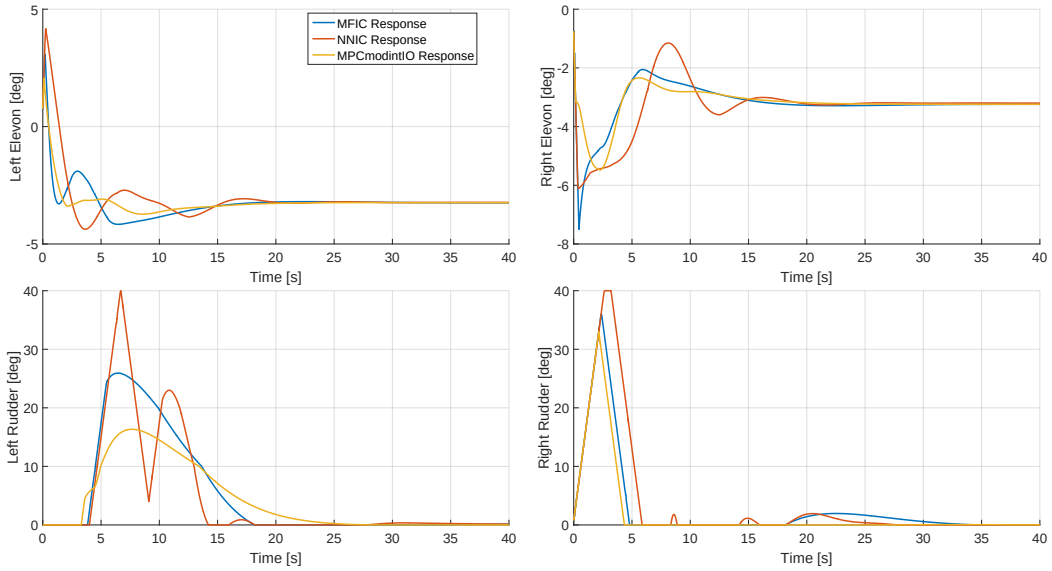


Figure 10-43: Control response comparison of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for a five degree bank angle disturbance for the hybrid case.

It can be seen that the left rudder response of the NNIC grows sharply, then decreases and increases again. This behaviour is purely due to the large error in the sideslip angle. For the MFIC, the left rudder response starts to smoothly decrease when the error in the bank angle has been reduced to less than two degrees. Interestingly, the MPC$_{IO}^{mfic}$ left rudder responds earlier than those of the other two controllers. Further inspection reveals that this is a clear example of its predictive capabilities. The MPC algorithm predicts that it needs to decelerate, even though the error in the bank angle is larger than that of the MFIC when it does so. This predictive capability allows the MPC$_{IO}^{mfic}$ to smoothly eliminate the error in the bank angle without overshoot. From the right rudder response of the NNIC, it can be seen that it saturates. In response, the yaw thrusters assist to comply with the commanded moment coefficients, as is shown in Figure 10-44. Furthermore, due to the rapidly changing commanded moment coefficients, the rudders cannot keep up with their finite deflection rates. In such situations, the yaw thrusters assist to comply to the commanded moment coefficients. The yaw thrusters of the MFIC and the MPC$_{IO}^{mfic}$ are only active before their left rudders are used, suggesting they are used to minimize the induced sideslip angle. Further inspection confirms this to be the case. The yaw thrusters of the NNIC remain active for longer than those of the other two controllers, partly due to the saturated right rudder, and partly due to the commanded moment coefficients changing faster than the rudders can keep up with.

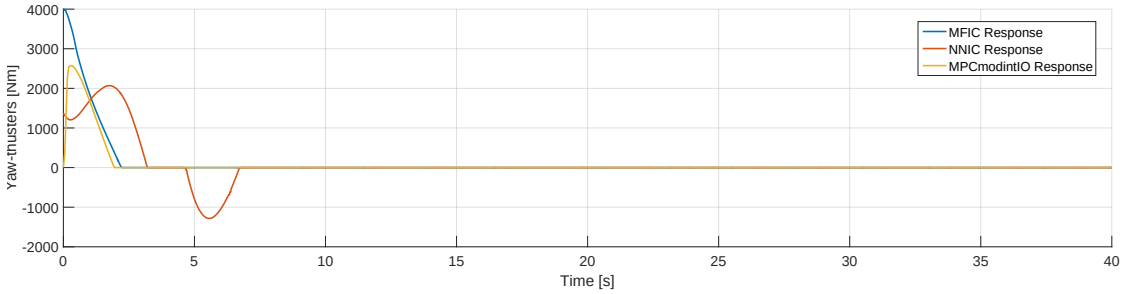


Figure 10-44: Thruster response comparison of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for a five degree bank angle disturbance for the hybrid case.

It is noted that, even though the pitch and roll thrusters are active, they are not used in this scenario. Combined with the rudder responses of the bank-reversal of case five, as shown in Figure 10-38, this clearly demonstrates that sideslip angle is the most challenging to control.

From the comparative analysis of the sensitivity of the state variables, as shown in Figure 10-45, it can be seen that the distribution widths of the NNIC are larger for all state variables than those of the other two controllers, which is due to the sensitivity of the error in the sideslip angle. In particular, the sensitivity of the angle of attack is purely due to changes in the induced error when system variations change the amplitude of the error in the sideslip angle. The larger distribution width of the pitch rate is in response to the sensitivity of the angle of attack. The bank angle distribution widths of the MPC$_{IO}^{mfic}$ and the NNIC are similar in size, but their origins are different. For the NNIC, the distribution width is determined by the variations in the sideslip angle response, which in turn induces changes in the bank angle response. For the MPC$_{IO}^{mfic}$, the distribution width is due to errors in the prediction algorithm of the MPC, which causes the vehicle to respond too slowly or too strongly, which the MPC has trouble accounting for, because it expects the system to behave differently. The error in the predictions accounts for the larger distributions widths of the MPC$_{IO}^{mfic}$ compared to those of the MFIC, for all state variables.

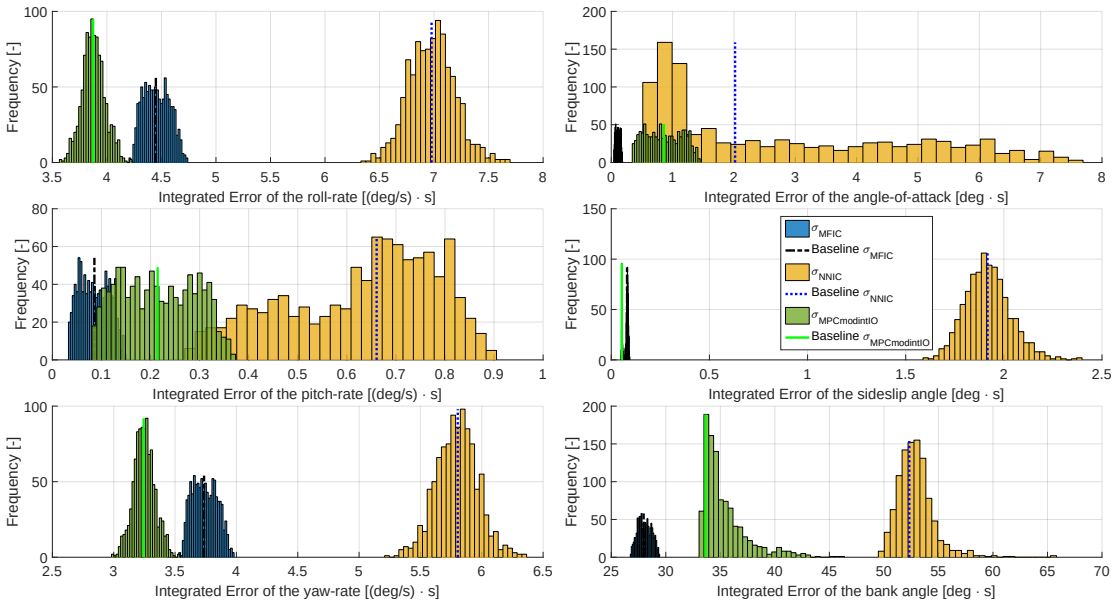


Figure 10-45: Comparative analysis of the sensitivity of the state variables of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for a five degree bank angle disturbance for the hybrid case.

Figure 10-46, it can be seen that the elevon distributions are nearly identical for all three controllers. This is quite surprising, because Figure 10-43 shows that each controller has a different nominal elevon response. The similarity occurs because the non-zero steady-state deflections of the elevon dominate the control effort values. The elevon trim deflection is sensitive to variations in the pitch moment coefficients and, hence, dominates the sensitivity of the elevon control effort. Because each controller is affected by the changes in the pitch moment coefficients equally, the elevon sensitivity is also equal for the three controllers.

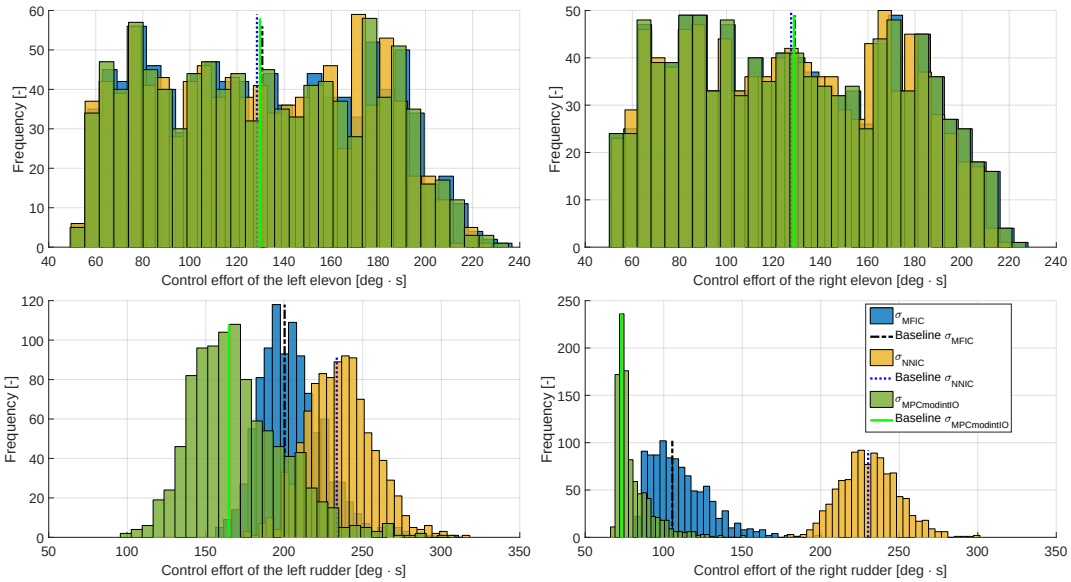


Figure 10-46: Comparative analysis of the sensitivity of the control effort of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for a five degree bank angle disturbance for the hybrid case.

The left rudder of the MPC$_{IO}^{mfic}$ has the largest distribution width, whereas for the right rudder is has the smallest width. This is due the prediction errors that the MPC$_{IO}^{mfic}$ makes when it computes its optimal control action. When the MPC$_{IO}^{mfic}$ has decreased its error in the bank angle, it will start using its left rudder. The prediction errors cause the controller to either respond too strongly or too slowly, which predominantly affects the right rudder because it is responsible for decelerating the vehicle when it approaches the commanded bank angle. Additionally, when the controller responds too slowly, the right rudder is used for longer periods of time, and when it responds too strongly, the bank angle might show an overshoot, which requires the use of the right rudder to correct. In both cases the control effort of the right rudder increases, which explains the skewness of the right rudder distribution of the MPC$_{IO}^{mfic}$. The distribution widths of the MFIC and the NNIC are similar, although the latter has a larger nominal rudder effort than the MFIC. For the right rudder, this statement is true for all system variations. For both the MFIC and the NNIC, the source of the sensitivity lies with the changes in the induced sideslip angle when the system parameters vary. When the controllers respond too strongly, the induced sideslip angle will be larger and, hence, the rudder control effort will increase. Conversely, when the controllers respond too slowly, the induced sideslip angle will be smaller and, hence, the rudder control effort will decrease. This explains the symmetric rudder distributions of the MFIC and the NNIC.

Besides the elevons and the rudders, this scenario also made use of the thrusters, of which only the yaw thrusters were used. Figure 10-47 shows the control effort of the yaw thrusters for each of the controllers. Curiously, the yaw thrusters of the NNIC use the least control effort, which is not completely obvious from Figure 10-44. Further inspection reveals that the lower thruster amplitudes more than compensate for the longer thruster use.

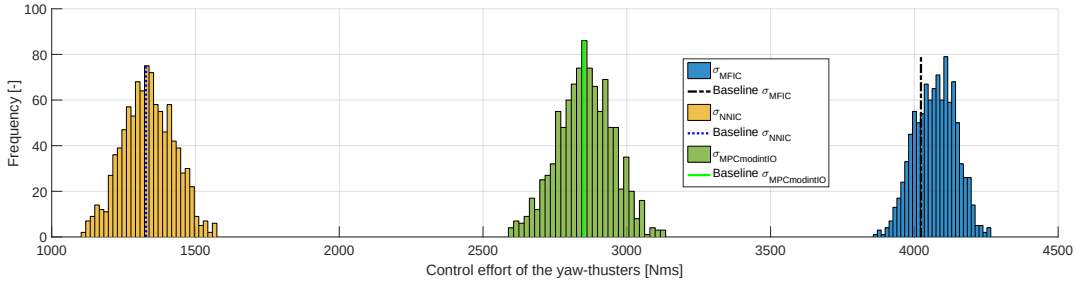


Figure 10-47: Comparative analysis of the sensitivity of the thruster effort of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for a five degree bank angle disturbance for the hybrid case.

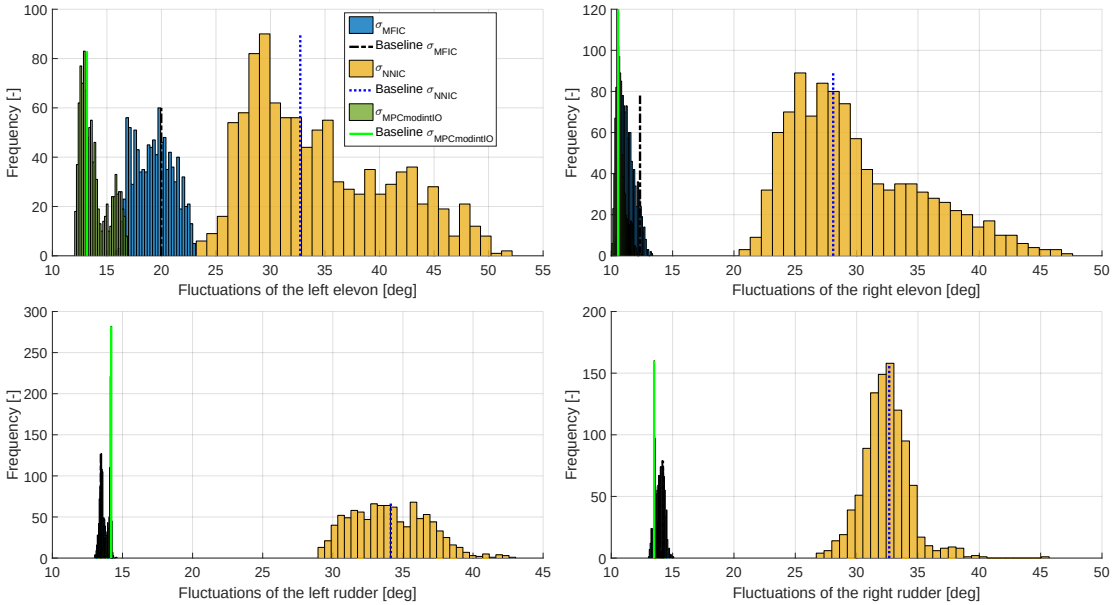


Figure 10-48: Comparative analysis of the sensitivity of the control fluctuations of the MFIC, the NNIC, and the MPC$_{IO}^{mfic}$, for a five degree bank angle disturbance for the hybrid case.

The comparative analysis of the sensitivity of the fluctuation values is shown in Figure 10-48. It clearly shows that the NNIC is assigned the largest fluctuation values, for all system variations, and is the most sensitive to changes in the fluctuation value due to system variations. The fluctuation values and distribution widths are directly related to variations in the sideslip angle response. The MPC$_{IO}^{mfic}$ has the lowest distribution widths for the fluctuations of all the control surfaces, and is, except for the left rudder, assigned the lowest nominal fluctuation values. The decreased distribution widths of the MPC$_{IO}^{mfic}$ are due to the relatively smooth control responses. Even though system variations cause more control effort for the rudders of the MPC$_{IO}^{mfic}$, its deflections increase and decrease at approximately the same rate for all system variations, thereby not influencing its fluctuation values. A similar observation can be made for the MFIC.

The Final Verdict

Based on the comparative analysis of the three controllers of this section, a conclusion can be made about which controller has the overall best performance. For the angle of attack disturbance, it was found that the MPC$_{IO}^{mfic}$ has a better nominal performance, but this came at the price of an corresponding increase in control effort. For the sideslip angle disturbance, it was found that the MFIC has a better overall performance, and although the settling times of the NNIC and the MPC$_{IO}^{mfic}$ were similar to that of the MFIC, they showed larger fluctuations in the state variables.

Using the same tuning for the controllers as was used for the angle of attack and sideslip angle disturbances, a bank-reversal was performed. It was found that the NNIC had the overall worst performance. For the MPC_{IO}^{mfic}, the integrated error of the bank angle has larger values than the MFIC, and it can be seen from Figure 10-37 that it eliminates the error marginally slower. Additionally, it was shown that the nominal elevon and rudder control effort of the MPC_{IO}^{mfic} is marginally smaller and larger than that of the MFIC, respectively. However, the increased response speed came the price of increased control fluctuations, which were larger for the MPC_{IO}^{mfic} than for the other two controllers. Furthermore, it was found that the MPC_{IO}^{mfic} and the MFIC are, on average, equally sensitive to system variations. The control effort of all three selected controllers were shown to be equally sensitive to system variations.

Additionally, a comparative analysis was made for a point along the reference trajectory for which all actuators were active. For this scenario, a five degree error in the bank angle had to be eliminated. It was found that the NNIC had the lowest overall performance, due it having a poorly optimized yaw controller, resulting in large sideslip angle errors. The MFIC eliminated the bank angle error slower than the MPC_{IO}^{mfic}, because of the overshoot of the former. The MPC_{IO}^{mfic}, however, responded slower, but because it has no overshoot, it eliminated the bank angle error the fastest. This is a clear example of, "slow and steady wins the race". The NNIC did eliminate the bank angle error at the same time as the MPC_{IO}^{mfic}, the decrease in overall performance due to the error of the sideslip angle error, makes the cost of the faster bank angle response too high. Additionally, it was shown that the MPC_{IO}^{mfic} had the lowest overall control effort, compared to that of the other two controllers. Furthermore, it was shown that the NNIC was assigned the largest fluctuation values and its control fluctuations were the most sensitive to system variations. The MPC_{IO}^{mfic} was shown to have to lowest control fluctuations, and had the lowest sensitivity.

In the previous sections, statements have been made about the implementation effort for each of the groups of controllers. It was reasoned that the benchmark controllers and the MPCs required much less implementation effort than the NNCs and FLCs, which required simulations that took several hours for each axis of a controller. Additionally, their optimization algorithms contained a large number of parameters that changes the behaviour of the optimization algorithm. The large simulations times imply that it is not practical to do a large number of iterations with different optimization parameters, complicating the design process for the optimal controller.

Based on the above paragraphs, it is concluded that the NNIC can perform similar to the MFIC, but the large simulation times and parameter choices of the optimization algorithm complicates the design process for the optimal controller. Furthermore, it can be concluded that the MPC_{IO}^{mfic} can outperform the MFIC, with a similar design effort. Additionally, the MPC_{IO}^{mfic} has the capability to take constraints of the state and control variables into account, which the MFIC does not. Furthermore, it was shown that the MPC_{IO}^{mfic} is the least sensitive to system variations. It is therefore concluded that the MPC_{IO}^{mfic} is, overall, the best performing controller, outperforming the MFIC.

For completeness, it is mentioned that, in the analysis of the sensitivity of the MPC_{IO}^{mfic}, it was determined that prediction errors are the main source of its sensitivity to system variations. It is possible to extent the definition of the state-space model of Equation 8-4 with an error source, which would be the difference between the predicted and the actual state errors. This will allow the MPCs to take into account the differences between in the internal model and the actual equations of motion and, hence, decrease its sensitivity to system variations.

Chapter 11

Conclusion and Recommendations

After a literature study, a knowledge-gap was identified relating to the comprehensive comparison of conventional and knowledge-based control systems when applied to entry-GNCs. Based on the knowledge-gap, the research question of this thesis was defined as:

Can knowledge-based control systems outperform conventional control systems when applied to entry-GNC systems?

The main research question can be split into three sub-questions which are formulated as follows: **When compared to conventional control systems**,

1. **can knowledge-based control systems improve the performance of entry-GNC systems?**
2. **can knowledge-based control systems improve the robustness of entry-GNC systems?**
3. **can the increase in implementation effort for knowledge-based control systems be justified with a corresponding increase in control performance?**

These research questions, together with the main research question, are answered in Section 11-1. Additionally, the work done in this thesis is only the tip of the iceberg. To this end, Section 11-2 will give an overview of the recommendations for further research.

11-1 Conclusion

In Chapter 10 a comparative analysis was made of the designed controllers to derive insights into the performance and the sensitivity to system variations of the controllers. The obtained insights directly relate to the answers of the first two sub-questions of the research question. The first sub-question relates to the nominal control performance of the KBCSs when compared to those of the (modified) benchmark controllers. Primarily, it was found that all controllers designed in this thesis, including the modified benchmark controllers, perform better than the LQR benchmark controller, as designed by Mooij (1998). It was shown that the MFIC has the highest overall performance compared to the remaining benchmark controllers. Additionally, it was shown

that the NNCs and the FLCs can be optimized using the NEAT and pattern search algorithms, respectively, such that they have similar or larger performance values than the corresponding modified benchmark controllers. However, this is not guaranteed, because the NNIC was shown to perform worse than the MFIC.

Furthermore, comparing the MPC_{IO}^{mfc} and the MPC_{IIO}^{mfc} with the MFC revealed that the MPCs had a higher overall performance. Additionally, the MPC_{IO}^{mfic} and the MPC_{IIO}^{mfic} have a nominal performance similar to that of the MFIC. Additionally, it was shown that the increased performance did not come at the cost of increased control effort and the control surfaces and thrusters did not saturate during the nominal responses for any of the controllers. Hence, it can be concluded that KBCSs *can* improve the nominal performance of entry-GNC systems when compared to conventional control system, but more complex systems of the latter type can equally improve the nominal performance.

The second sub-question relates to the robustness, or sensitivity, of the KBCSs to variations in the moment coefficients of the system, compared to those of the (modified) benchmark controllers. It was found that the sensitivity of a controller to system variations was dependent on the choice and specific tuning of the controller. It was found that, of the (modified) benchmark controllers, the integrated error of the aerodynamic angles of the MFIC was the least sensitive to system variations compared to the other (modified) benchmark controllers. The integrated error of the pitch rate had the largest width compared to that of the other benchmark controllers, but it was found through further inspection that the large widths were due to the too tightly tuned gains. It is interesting to note that the distribution widths of the control effort distributions were similar for the LQIC and the MFIC, and for the MFC and the benchmark controller. Additionally, for the variations in the control fluctuations, the MFC and the MFIC have larger distribution widths than the benchmark controller and the LQIC, for the rudders, and smaller widths for the elevons. However, for all system variations, it was found that the MFC has less fluctuations overall than the MFIC.

Furthermore, for the aerodynamic angles, it was shown that the NNC and the FLC without integral terms were equally sensitive to system variations as the MFC and, by extension, more sensitive than the MFIC. Identical observations were made for the NNCs and the FLCs with integral terms. It was shown that the NNIC was the best performing controller of the KBCSs for the bank-reversal of case five, not including the MPCs. Not only is it the least sensitive to system variation for all state variables, it also has the best aerodynamic angle performance for almost all system variations. Furthermore, it was shown that, nominally, it requires less control effort than the NNC and the FLCs. Additionally, for the NNIC, the sensitivity of the control effort due to system changes is similar to that of the MFIC, although the control fluctuations of the latter are more sensitive to system variations than the former.

Additionally, it was shown that, for the aerodynamic angles, all MPCs are less sensitive to system variations than the MFIC. It was found that the MPC_{IO}^{mfic} has the best overall performance and robustness of the MPCs. Furthermore, it was shown that the MPC_{IIO}^{mfic} is less sensitive to system variations than its IO-model counterpart, as was predicted to be the advantage of using an IIO-model by Ghahramani and Towhidkhah (2009). However, this decreased sensitivity of the aerodynamic angles came at the price of a increased sensitivity to the rudder control effort and the fluctuations of all control surfaces. Moreover, it was shown that the MPCs require similar levels of control effort as the MFIC.

Based on the observations of the previous three paragraph, it was shown that, for the aerodynamic angles, the NNCs and the FLCs are equally sensitive to system variations as the corresponding benchmark controllers. In general, however, the MPCs are more robust than all benchmark controllers. Hence, it is concluded that the KBCSs *can* improve the robustness of entry-GNC systems when compared to conventional control system. Furthermore, the IIO-model MPCs clearly demonstrated increased robustness when compared to the (conventional) benchmark controllers. Whether this increase in robustness occurs for all systems using an IIO-model MPC, is beyond the scope of this thesis.

The third and last sub-question relates to the implementation effort of the KBCSs compared to that of the benchmark controllers. This is an important aspect of controller design, because if an increase in design effort of the KBCSs does not correspond to an advantage (in this report, increased performance and robustness) over the more easily designed benchmark controllers, than the KBCSs will not be selected when a trade-off needs to be made for controller methodology.

Theoretically, NNCs and FLCs have great potential, and are able to approximate any function to any desired degree of accuracy. However, this potential comes at the cost of large degrees of freedom when the controllers needs to be optimized, thereby drastically increasing the implementation effort compared to the benchmark controllers and the MPCs. The optimization of the NNCs and FLCs required simulations that took several hours for each axis of a controller. Additionally, the optimization algorithm contained a large number of parameters that changes the behaviour of the optimization algorithm (default options were chosen in this thesis). The large simulations times imply that it is not practical to do a large number of iterations with different optimization parameters, complicating the design process for the optimal controller. Furthermore, in terms of control behaviour, it was shown that the NNCs and FLCs can have similar *nominal* performance as their corresponding benchmark controllers and can be less sensitive to system variations. For the NNCs and FLCs implemented in this thesis, the increased design effort does not correspond with an increase in nominal performance.

For the MPCs, it was shown that that the MPC_{IIO}^{mfc} had similar nominal performance compared to the MFIC; the best performing benchmark controller. Additionally, the state variable responses of the MPC_{IIO}^{mfc} are less sensitive to system variations than those of the MFIC. Furthermore, the former required less control effort than the latter, and did so with smaller fluctuation values of the control surfaces by a large margin for all system variations. In terms of implementation effort, the MPCs required additional effort to define the matrices relating to the constraints. Additionally, four summation parameters required tuning in addition to the state and control weighting-matrices. However, these matrices have straightforward definitions and the (integer) values of the summation parameters are easily found through iterations. Hence, the implementation effort of an MPC increases only marginally compared to that of the benchmark controllers, yet does reduce the sensitivity to system variations and decreases the effective control effort by decreasing the control fluctuations.

The above three paragraphs reflected on the implementation effort of the KBCSs compared to that of conventional control systems. It was found that, for the NNCs and FLCs implemented in this thesis, the increased design effort did not correspond with an increase in nominal performance, and they did showed, at best, similar levels of robustness compared to the benchmark controllers. Furthermore, for the MPCs, the implementation effort increased only marginally, but this was the price paid for a reduction in the sensitivity to system variations. Hence, it can be concluded that the drastic increase in implementation effort of the NNCs and FLCs was not justified by a corresponding increase in performance and robustness. Additionally, it can be concluded that the marginally increased implementation effort of the MPCs, the MPC_{IO}^{mfic} in particular, is justified by a corresponding increase in robustness of the controller compared to the benchmark controllers.

From the above paragraphs, it can be concluded that the KBCSs can outperform the original benchmark controller of the reference vehicle. However, a more complex LQR, in particular the MFIC, can have a nominal performance better or equal to the other controllers discussed in this thesis. However, it was shown that the NNIC and the MPC_{IO}^{mfic} are the best performing controllers in their respective categories. Furthermore, the MPC_{IO}^{mfic} is, on average, less sensitive to system variations than the MFIC. In the case of the NNIC, the MFIC had a better overall performance and is, on average, less sensitive to system variations than the NNIC. These statements are true for the bank-reversal of case five, as well as the hybrid case. Hence, the main conclusion is that KBCSs *can* outperform conventional control system when applied to entry-GNC systems. In particular, the MPC_{IIO}^{mfc} shows the greatest potential of the KBCSs analysed in this thesis to improve the performance and robustness of conventional control systems.

11-2 Recommendation

Throughout this thesis, short-comings were identified to lead to recommendations regarding research topic. These recommendations of future work are listed and discussed in the remainder of this section in no specific order.

- In this thesis, the performance and robustness of the controllers were evaluated about a single point along a reference trajectory. To derive conclusions about the performance of the control system for varying flight conditions the performance of the control system must be evaluated for a full flight simulation, which requires a guidance system to generate the steering commands to eliminate translation state-errors. Furthermore, a more general conclusion can be derived about the performance and robustness of the control system when (models of) sensors are included in the simulation model.
- For the NNCs, unsupervised learning methods were required due to the fact that there was no data available from which the controller could be optimized. In essence, the neural network did not *know* which functions to approximate. However, a discrete-time linear state-space model can be modelled as a recurrent network with the control commands as the input to the network. Additionally, the output states of this linear state-space network are desired to be zero (in the case of the discrete-time version of the linearised model of Equation 3-22). Therefore, chaining the output of a neural network to the input of the discrete-time linear state-space model allows one to use back-propagation methods commonly used to train neural networks, although changes need to be made to ensure that the "weights" of the state-space network do not change, as they represent the physical characteristics of the vehicle. It is noted that preliminary tests, using a control network without hidden nodes and the discrete-time version of Equations 4-10 and 4-11 as the state-space network, showed control performance similar to that of the longitudinal NNC of Chapter 6.
- In this thesis, the membership functions of the FLCs, representing the gains of the controller, are scheduled as a function of the corresponding state variables. Additionally, similar to the gains of the benchmark controllers, the membership functions can also be scheduled as a function of the dynamic pressure. This would allow the controller to be able to adapt to the varying flight conditions without requiring look-up tables.
- During the optimization process of the FLCs, it was found that the membership functions of several variables clustered together near the allowed ranges of these variables. Clearly, in these cases, to constrain the range of the variables did not allow the FLCs to fully optimize. To optimize the membership functions without the range constraints, the constraints of Equation 7-6 need to be adjusted. This will allow the membership functions to optimize freely, and possibly obtain higher performance values than the FLCs in this thesis.
- Like the NNCs, unsupervised learning methods were required to optimize the FLCs. However, it is possible to use *inverse control* to design a fuzzy logic controller using the discrete-time linear state-space model of the EoM. Furthermore, using the dynamic pressure as an input, it is possible to design the rulebase such that it can adjust to the changing flight conditions by varying the consequent parameters for each rule. Consequently, the output membership functions are replaced with the linearised state-space model, and do not have to be optimized during training. This inverse-model controller will simplify the controller design and makes active use of the available state-space model.
- It is possible to extend the methodology of MPC to include error sources and known disturbances by extending the SPCP model to include (linear) contributions of these sources. In the case of error sources, this allows the controller to adjust the state-prediction values based on the error between the expected and actual states. Additionally, known disturbances can include non-zero reference signals or possible corrections due to damage to the system.

Including error sources and known disturbances allows the controller to make corrections to its internal linear model of the non-linear system, thereby improving the quality of the optimal control action.

- In the current software implementation of the MPC, the parameters and matrices are computed off-line. However, during full flight simulation the matrices will change with the changing flight parameters, and hence need to be updated regularly. This can be accomplished by either scheduling the matrices as functions of the state variables, or by computing them on-line directly.
- It was found that the IIO-model, on its own, showed behaviour reminiscent to having integral terms. However, when integral terms were added to the IIO-model, they negated each others effect. The case of both behaviours is unknown and requires further research to be understood.
- In this thesis, an actuator allocation algorithm was developed. However, unlike the algorithms developed by Wu et al. (2000) and Durham (2001), the algorithm in this thesis did not take the rate limits of the actuators into account. It is possible to implement rate constraints on the actuators by limiting the search-space of the variations of the roll and pitch moment coefficients around the current deflection of the left and right elevons.

Bibliography

Allgöwer, F., Badgwell, T. A., Qin, J. S., Rawlings, J. B., and Wright, S. J., *Advances in Control: Highlights of ECC'99*, chap. Nonlinear Predictive Control and Moving Horizon Estimation — An Introductory Overview, Springer London, London, 1999, pp. 391–449.

Babuska, R., "Knowledge-Based Control Systems," Lecture Notes of SC4081, Delft University of Technology, Jan. 2010.

Bequette, B. W., "Nonlinear control of chemical processes: a review," *Industrial & Engineering Chemistry Research*, Vol. 30, No. 7, 1991, pp. 1391–1413.

van den Boom, T., "Model Predictive Control," Lecture Notes of SC4060, Delft University of Technology, jan 2013.

Bryson, A. E., Jr. and Ho, Y. C., *Applied optimal control*, John Wiley & Sons, 1975.

Cox, C., Neidhoefer, J., Saeks, R., and Lendaris, G., "Neural Adaptive Control of LoFLYTE," https://www.pdx.edu/sites/www.pdx.edu.sysc/files/control-finalabstract.html [Retrieved 27-05-2015], 1998.

Crassidis, J. L. and Markley, F. L., "Attitude Estimation Using Modified Rodriquez Parameters," Paper available from NASA-Goddard Space Flight Center, 1996.

Cybenko, G., "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, Vol. 2, No. 4, 1989, pp. 303–314.

DeCarlo, R., *Linear Systems: A State Variable Approach with Numerical Implementation*, Prentice Hall, NJ, 1989.

Dehkordi, S. K. M., Kashaninia, A., and Nasirian, M., "Comparison of H∞ and GIPC control methods for attitude control of rigid spacecrafts," *Control, Instrumentation, and Automation (ICCIA), 2013 3rd International Conference on*, Dec 2013, pp. 101–106.

Duerksen, N., "Fuzzy Logic Decoupled Lateral Control for General Aviation Airplanes," NASA, NASA Contractor Report 201735, Aug. 1997.

Durham, W. C., "Constrained Control Allocation," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 4, 1993, pp. 717 – 725.

Durham, W. C., "Attainable Moments for the Constrained Control Allocation Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 6, 1994a, pp. 1371 – 1373.

Durham, W. C., "Constrained Control Allocation: Three-Moment Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 2, 1994b, pp. 330 – 336.

Durham, W. C., "Computationally Efficient Control Allocation," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 3, 2001, pp. 519 – 524.

Engelen, C. J. H., "Fuzzy Logic Control of the X38 re-entry vehicle," Delft University of Technology, ESA ESTEC, Tech. rep., Jan. 2000.

Farid, A. M. and Barakti, S. M., "UAV Controller Based on Adaptive Neuro-Fuzzy Inference System and PID," *International Journal of Robotics and Automation*, Vol. 2, No. 2, June 2013, pp. 73–82.

Garcia, C. E., Prett, D. M., and Morari, M., "Model predictive control: Theory and practice - A survey," *Automatica*, Vol. 25, No. 3, May 1989, pp. 335–348.

Gaurav, A. K., "Comparison between Conventional PID and Fuzzy Logic Control for Liquid Flow Control Performance Evaluation of Fuzzy Logic and PID Control by Using MATLAB/Simulink," *International Journal of Innovative Technology and Exploring Engineering*, Vol. 1, No. 1, June 2012, pp. 84–88.

Ghahramani, N. O. and Towhidkhah, F., "Constrained incremental predictive controller design for a flexible joint robot," *ISA Transactions*, Vol. 48, No. 3, 2009, pp. 321 – 326.

Hall, C. E., Gallaher, M. W., and Hendrix, N. D., "X-33 Attitude Conrol System Design for Ascent, Transition, and Entry Flight Regimes," *Guidance, Navigation, and Control and Co-located Conferences*, American Institute of Aeronautics and Astronautics, Aug. 1998, pp. –.

Hughes, J. F., van Dam, A, McGuire, M., Sklar, D. F., Foley, J. D., Feiner, S. K., and Akeley, K., *Computer Graphics: Principles and Practice*, Addison-Wesley Professional, 2013.

Hui, N. B., Mahendar, V., and Pratihar, D. K., "Time-optimal, collision-free navigation of a car-like mobile robot using neuro-fuzzy approaches," *Fuzzy Sets and Systems*, Vol. 157, No. 1, 2006, pp. 2171–2204.

IEEE, *IEEE Standard Glossary of Software Engineering Terminology*, IEEE, Dec. 1990.

Jacchia, L. G., "Thermospheric Temperature, Density, and Composition: New Models," Smithson. Astrophys. Obs. Spec., Rept. No. 375, 1977.

Klancar, C. and Skrjanc, I., "Tracking-error model-based predictive control for mobile robots in real time," *Robotics and Autonomous Systems*, Vol. 55, No. 1, 2007, pp. 460–469.

Kruskal, J. B., "On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem," *Proceedings of the American Mathematical Society*, Vol. 7, No. 1, 1956, pp. 48–50.

Kuipers, J., *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace, and Virtual Reality*, Princeton University Press, Princeton, New Jersey, United States of America, 2002.

Leslie, F. W. and Justus, C. G., "The NASA Marshall Space Flight Center Earth Global Reference Atmospheric Model 2010 Version," NASA, NASA/TM2011216467, 2011.

Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. M., "Constrained model predictive control: Stability and optimality," *Automatica*, Vol. 36, No. 6, June 2000, pp. 789–814.

MBB, "Study on re-entry guidance and control," MBB Space Communication and Propulsion Systems Division, Munich, ESA report reference ESA CR(P) 2652, 1998.

McDowell, D. M. and Irwin, G. W., "Online Neural Control Applied to a Bank-to-Turn Missile Autopilot," *Guidance, Navigation, and Control and Co-located Conferences*, American Institute of Aeronautics and Astronautics, Aug. 1995, pp. 1286–1294.

McFarland, M. B. and Calise, A. J., "Neural Network for Stable Adaptive Control of Air-to-Air Missiles," *Control, Navigation, and Control and Co-located Conferences*, American Institute of Aeronautics and Astronautics, Aug. 1995, pp. 1280–1285.

Mooij, E., "The HORUS-2B Reference Vehicle," Memorandum M-692, May 1995.

Mooij, E., *The Motion of a Vehicle in a Planetary Atmosphere*, Delft University Press, Delft, The Netherlands, 1997.

Mooij, E., *Aerospace-Plane Flight Dynamics: Analysis of guidance and control concepts*, Ph.D. thesis, Delft University of Technology, 1998.

Mooij, E., "Re-entry Systems," Draft Lecture Notes of AE4870B, Delft University of Technology, 2014a.

Mooij, E., "Simulator Development," Lecture slides of AE4870A-B, Delft University of Technology, Oct. 2014b.

Mooij, E. and Ellenbroek, M., "Multi-Functional Guidance, Navigation and Control Simulation Environment - Rapid Prototyping of Space Simulations," *Rapid Prototyping Technology - Principles and Functional Requirements*, edited by Dr. M. Hoque, ISBN: 978-953-307-970-7 15, InTech, 2011, pp. 315–338, Available from: http://www.intechopen.com/books/rapid-prototyping-technology-principles-and-functional-requirements/multifunctional-guidance-navigation-and-control-simulation-environment-rapid\/-prototyping-of-space-s.

Mooij, E. and Wijnands, Q., "Generic Attitude and Orbit Control Simulator development supporting the AOCS software life cycle," *7th International Workshop on Simulation for European Space Programmes*, Nov 2002, pp. –, Noordwijk, The Netherlands.

NASA, NOAA, and USAF, "U.S. Standard Atmosphere 1976," Washington, D.C., Tech. rep., 1976.

Nho, K. and Agarwal, R. K., "Automatic Landing System Design Using Fuzzy Logic," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 2, Mar. 2000, pp. 298–304.

Nour, M. I. H., Ooi, J., and Chan, K. Y., "Fuzzy Logic Control vs. Conventional PID control of an inverted pendulum robot," *International Conference on Intelligent and Advanced Systems, 2007*, Nov. 2007, pp. 209–214.

Ogata, K., *Modern Control Engineering*, Prentice Hall, 5th ed., 2010.

Ricker, N., Subrahmanian, T., and Sim, T., "Case studies of model-predictive control in pulp and paper production," *Proc. 1998 IFAC Workshop on Model Based Process Control*, 1988, pp. 13–22.

de Ridder, S., *Study on Optimal Trajectories and Energy Management Capabilities of a Winged Re-entry Vehicle during the Terminal Area*, Master's thesis, Delft University of Technology, July 2009.

Rutkowski, L., *Flexible Neuro-fuzzy Systems: Structures, Learning and Performance Evaluation*, The Kluwer International Series in Engineering and Computer Science, Kluwer Academic Publishers, 2004.

Saini, G. and Balakrishnan, S. N., "Adaptive critic based neurocontroller for autolanding of aircrafts," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 4, July 1996, pp. 893–897.

Scalera, K., *A Comparison of Control Allocation Methods for the F-15 ACTIVE Research Aircraft Utilizing Real-Time Piloted Simulation*, Master's thesis, Virginia Polytechnic Institute and State University, 1999.

Soeterboek, A. R. M., *Predictive control - A unified approach*, Prentice Hall, Englewood Cliffs, NJ, 1992.

Stanley, K. O. and Miikulainen, R., "Evolving Neural Network through Augmenting Topologies," *Evolutionary Computation*, Vol. 10, No. 2, 2002, pp. 99–127.

Strejc, V., *State Space Theory of Discrete Linear Control*, Academia, Prague, 1981.

Visser, H. G., "Aircraft Performance Optimization," Lecture Notes of AE4447, Delft University of Technology, Jan. 2014.

Wallner, E. M. and Well, K. H., "Non-linear Flight Control Design for the X-38 Using CMAC Neural Networks," *Guidance, Navigation, and Control and Co-located Conferences*, American Institute of Aeronautics and Astronautics, Montreal, Canada, Aug. 2001, pp. –.

Wang, L. X., "Fuzzy systems are universal approximators," *Fuzzy Systems, 1992., IEEE International Conference on*, Mar 1992, pp. 1163–1170.

Weiland, C., *Aerodynamic Data of Space Vehicles*, Springer, Berlin, Germany, 2014.

Wu, S.-F., Engelen, C., Babuska, R., Chu, Q. P., and Mulder, J., *Aerospace Sciences Meetings*, chap. Intelligent flight controller design with fuzzy logic for an atmospheric re-entry vehicle, American Institute of Aeronautics and Astronautics, Jan 2000, pp. 1–12.

Yildirim, S., "Design of a proposed neural network control system for trajectory control of walking robots," *Simulation Modeling Practice and Theory*, Vol. 16, No. 1, 2008, pp. 368–378.

Zaccagnino, E., Malucchi, G., Marco, V., Drocco, A., Dussy, S., and Préaud, J. P., "Intermediate eXperimental Vehicle (IXV), the ESA Re-entry Demonstrator," *Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, Aug. 2011, pp. –.

Zhang, D., Hu, D., Shen, L., and Xie, H., "Design of an artificial bionic neural network to control fish-robot's locomotion," *Neurocomputing*, Vol. 71, No. 1, 2008, pp. 648–654.

Zheng, A. and Morari, M., "Stability of model predictive control with mixed constraints," *IEEE Transactions on Automatic Control*, Vol. 40, No. 10, Oct 1995, pp. 1818–1823.

Appendix A

State-space matrix coefficients

Mooij (2014b) advised to use spherical coordinates for both position and velocity for the use in analytical approximations of the EoM. When the EoM are written using spherical position and velocity in the R-frame, it makes it far easier to understand and interpret the results, and hence this definition will be used a basis for the analytical approximations of the EoM (Mooij, 2014a). The dynamic equations of translational motion can be written in spherical coordinates for position and velocity in the R-frame as:

$$\dot{V} = -\frac{D}{m} - g\sin\gamma + \omega_{cb}^2 R\cos\delta\left(\sin\gamma\cos\delta - \cos\gamma\sin\delta\cos\chi\right) \tag{A-1}$$

$$\begin{aligned} V\dot{\gamma} = \frac{L\cos\sigma - S\sin\sigma}{m} &- g\cos\gamma + 2\omega_{cb}V\cos\delta\sin\chi + \frac{V^2}{R}\cos\gamma \\ &+\omega_{cb}^2 R\cos\delta\left(\cos\delta\cos\gamma + \sin\gamma\sin\delta\cos\chi\right) \end{aligned} \tag{A-2}$$

$$\begin{aligned} V\cos\gamma\dot{\chi} = -\frac{L\sin\sigma + S\cos\sigma}{m} &+ 2\omega_{cb}V\left(\sin\delta\cos\gamma - \cos\delta\sin\gamma\cos\chi\right) + \\ &\frac{V^2}{R}\cos^2\gamma\tan\delta\sin\chi + \omega_{cb}^2 R\cos\delta\sin\delta\sin\chi \end{aligned} \tag{A-3}$$

with the kinematic position equations given as:

$$\dot{R} = V\sin\gamma \tag{A-4}$$

$$\dot{\tau} = \frac{V\sin\chi\cos\gamma}{R\cos\delta} \tag{A-5}$$

$$\dot{\delta} = \frac{V\cos\chi\cos\gamma}{R} \tag{A-6}$$

In the derivation of Equations A-1 through A-6 the assumption has been made that the Earth is spherical (with a spherical gravitational field), and that it rotates with constant angular velocity. This assumption has the effect that one only has a single gravity component, instead of multiple,

and that one does not have to worry about the long period variations in the Earth angular velocity. Note that in Equations A-3 and A-5 there are singularities for $\gamma = \pm 90^\circ$ (vertical flight) and $\delta = \pm 90^\circ$ (north or south pole of the central body). Since these flight conditions do not occur in the missions studied in this report, they are not further considered.

In Section 3-6, the kinematic attitude equations were written as function of the attitude quaternion. The kinematic attitude equations can also be based on the aerodynamic angles α, β, and σ, and are formulated in the B-frame. The following set of equations can be derived (Mooij, 1998):

$$\begin{aligned}\dot{\alpha}\cos\beta = &-p\cos\alpha\sin\beta + q\cos\beta - r\sin\alpha\sin\beta + \sin\sigma\left[\dot{\chi}\cos\gamma - \dot{\delta}\sin\chi\sin\gamma\right.\\ &\left.+ (\dot{\tau}+\omega_{cb})\left(\cos\delta\cos\chi\sin\gamma - \sin\delta\cos\gamma\right)\right] - \cos\sigma\left[\dot{\chi} - \dot{\delta}\cos\chi\right.\\ &\left.- (\dot{\tau}+\omega_{cb})\cos\delta\sin\chi\right]\end{aligned} \tag{A-7}$$

$$\begin{aligned}\dot{\beta} = &\, p\sin\alpha - r\cos\alpha + \sin\sigma\left[\dot{\gamma} - \dot{\delta}\cos\chi - (\dot{\tau}+\omega_{cb})\cos\delta\sin\chi\right] + \cos\sigma[\dot{\chi}\cos\gamma\\ &- \dot{\delta}\sin\chi\sin\gamma + (\dot{\tau}+\omega_{cb})\left(\cos\delta\cos\chi\sin\gamma - \sin\delta\cos\gamma\right)\big]\end{aligned} \tag{A-8}$$

$$\begin{aligned}\dot{\sigma} = &-p\cos\alpha\cos\beta - q\sin\beta - r\sin\alpha\cos\beta + \dot{\alpha}\sin\beta + \dot{\chi}\sin\gamma - \dot{\delta}\sin\chi\cos\gamma\\ &+ (\dot{\tau}+\omega_{cb})\left(\cos\delta\cos\chi\cos\gamma + \sin\delta\sin\gamma\right)\end{aligned} \tag{A-9}$$

Equations A-7 through A-9 will be used for the linearisation of the EoM. Furthermore, $\dot{\gamma}, \dot{\chi}, \dot{\delta}$ and $\dot{\tau}$ represent the effects of the angular velocity of the Earth and the rotation of the horizontal plane. Note that in Equation A-7 there is a singularity for $\beta = \pm 90^\circ$. Since this flight condition does not occur in the missions studied in this report, it is not further considered.

Furthermore, in Section 3-7, it was derived that Equations A-1 to A-9 can be linearised and written as a state-space model as:

$$\Delta\dot{\mathbf{x}} = \mathbf{A}\Delta\mathbf{x} + \mathbf{B}\Delta\mathbf{u} \tag{A-10}$$

In (Mooij, 2014a), the elements of the matrices $\mathbf{A}$ and $\mathbf{B}$ were derived. Here, they are simply restated.

$$\begin{pmatrix}\Delta\dot{V}\\ \Delta\dot{\gamma}\\ \Delta\dot{R}\\ \Delta\dot{p}\\ \Delta\dot{q}\\ \Delta\dot{r}\\ \Delta\dot{\alpha}\\ \Delta\dot{\beta}\\ \Delta\dot{\sigma}\end{pmatrix} = \begin{bmatrix} a_{VV} & a_{v\gamma} & a_{VR} & 0 & 0 & 0 & a_{V\alpha} & 0 & 0\\ a_{\gamma_v} & a_{\gamma\gamma} & a_{\gamma R} & 0 & 0 & 0 & a_{\gamma\alpha} & a_{\gamma\beta} & a_{\gamma\sigma}\\ a_{RV} & a_{R\gamma} & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{p\beta} & 0\\ a_{qV} & 0 & 0 & 0 & 0 & 0 & a_{q\alpha} & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{r\beta} & 0\\ a_{\alpha V} & a_{\alpha\gamma} & a_{\alpha R} & 0 & a_{\alpha q} & 0 & a_{\alpha\alpha} & 0 & a_{\alpha\sigma}\\ a_{\beta V} & a_{\beta\gamma} & a_{\beta R} & a_{\beta p} & 0 & a_{\beta r} & 0 & a_{\beta\beta} & a_{\beta\sigma}\\ a_{\sigma V} & a_{\sigma\gamma} & 0 & a_{\sigma p} & 0 & a_{\sigma r} & a_{\sigma\alpha} & a_{\sigma\beta} & a_{\sigma\sigma}\end{bmatrix}\begin{pmatrix}\Delta V\\ \Delta\gamma\\ \Delta R\\ \Delta p\\ \Delta q\\ \Delta r\\ \Delta\alpha\\ \Delta\beta\\ \Delta\sigma\end{pmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & b_{pa} & 0 & b_{px} & 0 & 0\\ b_{qe} & 0 & 0 & 0 & b_{qy} & 0\\ 0 & b_{ra} & b_{rr} & 0 & 0 & b_{rz}\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}\begin{pmatrix}\Delta\delta_e\\ \Delta\delta_a\\ \Delta\delta_r\\ \Delta M_{T,x}\\ \Delta M_{T,y}\\ \Delta M_{T,z}\end{pmatrix} \tag{A-11}$$

where the non-zero elements for state matrix $\mathbf{A}$ are:

$$a_{VV} = -\frac{1}{mV_0}\left(M_0\frac{\partial C_D}{\partial M}\bar{q}_0 S_{ref} + 2D_0\right) \tag{A-12}$$

$$a_{V\gamma} = -g_0\cos\gamma_0 \tag{A-13}$$

$$a_{VR} = 2\frac{g_0}{R_0}\sin\gamma_0 \tag{A-14}$$

$$a_{V\alpha} = -\frac{1}{m}\frac{\partial C_D}{\partial \alpha}\bar{q}_0 S_{ref} \tag{A-15}$$

$$a_{\gamma V} = \frac{1}{V_0}\left(-\dot{\gamma}_0 + \frac{2V_0}{R_0}\cos\gamma_0\right) + \frac{\cos\sigma_0}{mV_0^2}\left(M_0\frac{\partial C_L}{\partial M}\bar{q}_0 S_{ref} + 2L_0\right) \tag{A-16}$$

$$a_{\gamma\gamma} = -\left(\frac{V_0}{R_0} - \frac{g_0}{V_0}\right)\sin\gamma_0 \tag{A-17}$$

$$a_{\gamma R} = \left(\frac{2g_0}{V_0} - \frac{V_0}{R_0}\right)\frac{\cos\gamma_0}{R_0} \tag{A-18}$$

$$a_{\gamma\alpha} = \frac{\cos\sigma_0}{mV_0}\frac{\partial C_L}{\partial \alpha}\bar{q}_0 S_{ref} \tag{A-19}$$

$$a_{\gamma\beta} = -\frac{\sin\sigma_0}{mV_0}\frac{\partial C_S}{\partial \beta}\bar{q}_0 S_{ref} \tag{A-20}$$

$$a_{\gamma\sigma} = -\frac{L_0}{mV_0}\sin\sigma_0 \tag{A-21}$$

$$a_{RV} = \sin\gamma_0 \tag{A-22}$$

$$a_{R\gamma} = V_0\cos\gamma_0 \tag{A-23}$$

$$a_{p\beta} = \frac{1}{I_{xx}}\frac{\partial C_l}{\partial \beta}\bar{q}_0 S_{ref} b_{ref} \tag{A-24}$$

$$a_{qV} = \frac{M_0}{I_{yy}V_0}\frac{\partial C_m}{\partial M}\bar{q}_0 S_{ref} c_{ref} \tag{A-25}$$

$$a_{q\alpha} = \frac{1}{I_{yy}}\frac{\partial C_m}{\partial \alpha}\bar{q}_0 S_{ref} c_{ref} \tag{A-26}$$

$$a_{r\beta} = \frac{1}{I_{zz}}\frac{\partial C_n}{\partial \beta}\bar{q}_0 S_{ref} b_{ref} \tag{A-27}$$

$$a_{\alpha V} = -\frac{g_0}{V_0^2}\cos\gamma_0\cos\sigma_0 - \frac{1}{mV_0^2}\left(M_0\frac{\partial C_L}{\partial M} + C_L\right)\bar{q}_0 S_{ref} \tag{A-28}$$

$$a_{\alpha\gamma} = -\frac{g_0}{V_0}\sin\gamma_0\cos\sigma_0 \tag{A-29}$$

$$a_{\alpha R} = -\frac{2g_0}{R_0V_0}\cos\gamma_0\cos\sigma_0 \tag{A-30}$$

$$a_{\alpha q} = 1 \tag{A-31}$$

$$a_{\alpha\alpha} = -\frac{1}{mV_0}\frac{\partial C_L}{\partial \alpha}\bar{q}_0 S_{ref} \tag{A-32}$$

$$a_{\alpha\sigma} = -\frac{g_0}{V_0}\cos\gamma_0\sin\sigma_0 \tag{A-33}$$

$$a_{\beta V} = \frac{g_0}{V_0^2}\cos\gamma_0\sin\sigma_0 \tag{A-34}$$

$$a_{\beta\gamma} = \frac{g_0}{V_0}\sin\gamma_0\sin\sigma_0 \tag{A-35}$$

$$a_{\beta R} = \frac{2g_0}{R_0V_0}\cos\gamma_0\sin\sigma_0 \tag{A-36}$$

$$a_{\beta p} = \sin \alpha_0 \tag{A-37}$$
$$a_{\beta r} = -\cos \alpha_0 \tag{A-38}$$
$$a_{\beta\beta} = -\frac{1}{mV_0} \frac{\partial C_S}{\partial \beta} \bar{q}_0 S_{ref} \tag{A-39}$$
$$a_{\beta\sigma} = -\frac{g_0}{V_0} \cos \gamma_0 \cos \sigma_0 \tag{A-40}$$
$$a_{\sigma V} = \frac{\tan(\gamma_0 \sin \sigma_0)}{mV_0^2} \left(M_0 \frac{\partial C_L}{\partial M} + C_L \right) \bar{q}_0 S_{ref} \tag{A-41}$$
$$a_{\sigma\gamma} = \frac{L_0}{mV_0} \sin \sigma_0 \tag{A-42}$$
$$a_{\sigma p} = -\cos \alpha_0 \tag{A-43}$$
$$a_{\sigma r} = -\sin \alpha_0 \tag{A-44}$$
$$a_{\sigma\alpha} = \frac{\tan(\gamma_0) \sin \sigma_0}{mV_0} \frac{\partial C_L}{\partial \alpha} \bar{q}_0 S_{ref} \tag{A-45}$$
$$a_{\sigma\beta} = \frac{\tan(\gamma_0) \cos \sigma_0}{mV_0} \frac{\partial C_S}{\partial \beta} \bar{q}_0 S_{ref} - \frac{L_0}{mV_0} + \frac{g_0}{V_0} \cos \gamma_0 \cos \sigma_0 \tag{A-46}$$
$$a_{\sigma\sigma} = \tan(\gamma_0) \cos \sigma_0 \frac{L_0}{mV_0} \tag{A-47}$$

The non-zero elements of the matrix $\mathbf{B}$ are:

$$b_{pa} = \frac{1}{I_{xx}} \frac{\partial C_l}{\partial \delta_a} \bar{q}_0 S_{ref} b_{ref} \tag{A-48}$$
$$b_{px} = \frac{1}{I_{xx}} \tag{A-49}$$
$$b_{qe} = \frac{1}{I_{yy}} \frac{\partial C_m}{\partial \delta_e} \bar{q}_0 S_{ref} c_{ref} \tag{A-50}$$
$$b_{qy} = \frac{1}{I_{yy}} \tag{A-51}$$
$$b_{ra} = \frac{1}{I_{zz}} \frac{\partial C_n}{\partial \delta_a} \bar{q}_0 S_{ref} b_{ref} \tag{A-52}$$
$$b_{rr} = \frac{1}{I_{zz}} \frac{\partial C_n}{\partial \delta_r} \bar{q}_0 S_{ref} b_{ref} \tag{A-53}$$
$$b_{rz} = \frac{1}{I_{zz}} \tag{A-54}$$

Appendix B

Controller Parameter Tables

B-1 Benchmark Controllers

This section lists the parameter tables of the modified benchmark controllers. First the MFC parameter tables are shown, after which the tables corresponding to the LQIC and the MFIC are shown.

B-1-1 Moment-fraction Controller

Table B-1: Variations of the maximum allowed amplitudes of the state-vector and control variables for the longitudinal MFC depending on which actuators are active.

	Pitch thruster	Pitch thruster and elevator	Elevator
Δq_{max} [deg/s]	1.5	1.5	4
$\Delta \alpha_{max}$ [deg]	1	2.5	2
$\eta_{y,max}$ [-]	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{2}$

Table B-2: Variations of the maximum allowed amplitudes of the state-vector and control variables for the nominal lateral MFC depending on which actuators are active.

	Roll and yaw thruster	Roll and yaw thruster and ailerons	Roll and yaw thruster, aileron and rudder	Yaw thruster, rudder, and aileron
Δp_{max} [deg/s]	1.5	2.5	2.5	10
Δr_{max} [deg/s]	1.5	1.5	10	10
$\Delta \beta_{max}$ [deg]	1	1	2	2
$\Delta \sigma_{max}$ [deg]	4	4	5	10
$\eta_{x_{max}}$ [-]	0.5	0.5	0.5	0.5
$\eta_{z_{max}}$ [-]	1	1	1	1

For the bank-reversal amplitudes, only the maximum allowed amplitude $\Delta \sigma_{max}$ is changed in the phase where the yaw thruster, the rudders, and the ailerons are active. It's value is changed to be 20deg.

B-1-2 Linear Quadratic Integral Controller

Table B-3: Variations of the maximum allowed amplitudes of the state-vector and control variables for the longitudinal LQIC depending on which actuators are active.

	Pitch thruster	Pitch thruster and elevator	Elevator
Δq_{max} [deg/s]	1.5	1.5	4
$\Delta \alpha_{max}$ [deg]	1	1	1
$I_{\alpha,max}$ [deg·s]	2.5	5	10
$\Delta \delta_{e_{max}}$ [deg]	NA	40	40
$\Delta M_{T,y_{max}}$ [Nm]	10,400	10,400	NA

Table B-4: Variations of the maximum allowed amplitudes of the state-vector and control variables for the nominal lateral LQIC depending on which actuators are active.

	Roll and yaw thruster	Roll and yaw thruster and ailerons	Roll and yaw thruster, aileron and rudder	Yaw thruster, rudder, and aileron
Δp_{max} [deg/s]	1.5	2.5	2.5	5
Δr_{max} [deg/s]	1.5	1.5	5	5
$\Delta \beta_{max}$ [deg]	1	1	2	1
$\Delta \sigma_{max}$ [deg]	4	4	4	5
$I_{\beta,max}$ [deg·s]	0.5	0.5	0.5	0.5
$I_{\sigma,max}$ [deg·s]	100	100	100	100
$\Delta \delta_{a_{max}}$ [deg]	NA	40	40	40
$\Delta \delta_{r_{max}}$ [deg]	NA	NA	40	40
$\Delta M_{T,x_{max}}$ [Nm]	1,600	1,600	1,600	NA
$\Delta M_{T,z_{max}}$ [Nm]	7,600	7,600	7,600	7,600

For the bank-reversal amplitudes, the values of $\Delta p_{max}, \Delta r_{max}$, and $\Delta \sigma_{max}$ are changed in the phase were the yaw thruster, the rudder, and the ailerons are active. Their values are changed to 25 deg/s for Δp_{max} and Δr_{max}, and to 30 degrees for $\Delta \sigma_{max}$. Additionally, the value of $I_{\sigma,max}$ is changed to 200 deg·s for all phases.

B-1-3 Moment-fraction Integral Controller

Table B-5: Variations of the maximum allowed amplitudes of the state-vector and control variables for the longitudinal MFIC depending on which actuators are active.

	Pitch thruster	Pitch thruster and elevator	Elevator
Δq_{max} [deg/s]	4	4	5
$\Delta \alpha_{max}$ [deg]	1	1	2.5
$I_{\alpha,max}$ [deg·s]	2.5	2.5	5
$\eta_{y,max}$ [-]	$\sqrt{2}$	$\sqrt{2}$	0.25

Table B-6: Variations of the maximum allowed amplitudes of the state-vector and control variables for the nominal lateral MFIC depending on which actuators are active.

	Roll and yaw thruster	Roll and yaw thruster and ailerons	Roll and yaw thruster, aileron and rudder	Yaw thruster, rudder, and aileron
Δp_{max} [deg/s]	1.5	2.5	5	15
Δr_{max} [deg/s]	1.5	1.5	5	15
$\Delta \beta_{max}$ [deg]	1	1	1	1
$\Delta \sigma_{max}$ [deg]	4	4	6	15
$I_{\beta,max}$ [deg·s]	0.5	1	1	5
$I_{\sigma,max}$ [deg·s]	100	200	200	200
$\eta_{x_{max}}$ [-]	0.5	0.5	0.5	0.5
$\eta_{z_{max}}$ [-]	1	1	1	1

For the bank-reversal amplitudes, the values of $\Delta p_{max}, \Delta r_{max}$, and $\Delta \sigma_{max}$ are changed in the phase were the yaw thruster, the rudder, and the ailerons are active. Their values are changed to 20 deg/s for Δp_{max} and Δr_{max}, and to 30 degrees for $\Delta \sigma_{max}$.

B-2 Neural Network Controllers

The numerical values of the connection weights of the NNCs are shown in Table B-7 for the NNCs with out without integral terms for the bank-reversal of case five. Furthermore, Table B-8 shows the connection weight values for the NNIC during the hybrid case.

Table B-7: Connection weights of the optimal neural networks structure with (NNIC) and without (NNC) integral term for the bank-reversal of case five.

Connection	**NNC**	**NNIC**	**Connection**	**NNC**	**NNIC**
w_{px}	-1.0043	-3.0957	w_{rx}	1.9156	3.6872
$w_{\beta x}$	-0.6377	-0.1499	$w_{\sigma x}$	0.3041	0.1317
$w_{I_\beta x}$	-	-1.2473	$w_{I_\sigma x}$	-	0.0132
w_{qy}	-5.3528	-7.5568	$w_{\alpha y}$	-7.4085	-17.5386
$w_{I_\alpha y}$	-	-0.1437	w_{pz}	0.0595	0.5115
w_{rz}	-1.7990	-3.2665	$w_{\beta z}$	5.5851	5.9046
$w_{\sigma z}$	0.3951	0.2914	$w_{I_\beta z}$	-	4.5107
$w_{I_\sigma z}$	-	0.0124			

Table B-8: Connection weights of the optimal NNIC for the hybrid case.

Connection	**Value**	**Connection**	**Value**
w_{px}	-3.5379	w_{rx}	3.6872
$w_{\beta x}$	-0.0250	$w_{\sigma x}$	0.1756
$w_{I_\beta x}$	-0.3118	$w_{I_\sigma x}$	0.0395
w_{qy}	-7.5568	$w_{\alpha y}$	-17.5386
$w_{I_\alpha y}$	-0.1437	w_{pz}	0.1023
w_{rz}	-5.2263	$w_{\beta z}$	4.9205
$w_{\sigma z}$	0.1942	$w_{I_\beta z}$	0.1504
$w_{I_\sigma z}$	0.0950		

B-3 Fuzzy Logic Controllers

This section shows the membership function parameters describing the location of the peaks of the triangular membership functions. Table B-9 and B-10 show the parameters for the FLIC and the FLC, respectively. It is noted that $(\Delta p_{max})_z$, for example, refers to the membership functions of the roll rate corresponding to the yaw controller.

Table B-9: Membership parameters of the optimal fuzzy logic structure with integral terms.

Parameter	$\mathbf{R}_1$	$\mathbf{R}_2$	$\mathbf{R}_3$	$\mathbf{R}_4$	$\mathbf{R}_5$	$\mathbf{R}_6$	$\mathbf{R}_7$
Δq_{max} [deg/s]	0.125	0.250	0.375	0.500	0.625	0.750	0.875
$\Delta \alpha_{max}$ [deg]	0.125	0.250	0.375	0.500	0.625	0.750	0.875
I_α [deg·s]	0.125	0.250	0.375	0.500	0.625	0.750	0.875
$(\Delta p_{max})_z$ [deg/s]	0.337	0.422	0.565	0.824	0.950	1.064	4.164
$(\Delta r_{max})_z$ [deg/s]	0.900	2.297	2.440	2.574	2.700	3.564	4.415
$(\Delta \beta_{max})_z$ [deg]	0.031	0.046	0.067	0.086	0.647	0.770	0.883
$(\Delta \sigma_{max})_z$ [deg]	4.276	4.549	4.817	5.077	5.329	5.570	5.800
$(I_\beta)_z$ [deg·s]	0.196	0.265	0.395	0.523	0.650	0.767	0.883
$(I_\sigma)_z$[deg·s]	0.056	0.110	0.190	0.324	0.450	0.564	0.665
$(\Delta p_{max})_x$ [deg/s]	0.625	1.250	1.875	2.500	3.125	3.750	4.375
$(\Delta r_{max})_x$ [deg/s]	0.625	1.250	1.875	2.500	3.125	3.750	4.375
$(\Delta \beta_{max})_x$ [deg]	0.125	0.250	0.375	0.500	0.625	0.750	0.875
$(\Delta \sigma_{max})_x$ [deg]	0.7500	1.500	2.250	3.000	3.750	4.500	5.250
$(I_\beta)_x$ [deg·s]	0.125	0.250	0.375	0.500	0.625	0.750	0.875
$(I_\sigma)_x$[deg·s]	0.625	1.250	1.875	2.500	3.125	3.750	4.375
K_{yq} [s/deg]	46.250	47.500	48.750	50.000	51.250	52.500	53.750
K_{ya} [1/deg]	86.250	87.500	88.750	90.000	91.250	92.500	93.750
K_{yia} [1/(deg·s)]	21.875	23.750	25.625	27.50	29.375	31.250	33.125
K_{xp} [s/deg]	6.250	6.500	6.750	7.000	7.250	7.500	7.750
K_{xr} [s/deg]	-17.375	-16.750	-16.125	-15.500	-14.875	-14.25	-13.625
K_{xb} [1/deg]	96.250	97.500	98.750	100.000	101.250	102.500	103.750
K_{xs} [1/deg]	-1.875	-1.750	-1.625	-1.500	-1.375	-1.250	1.125
K_{xib} [1/(deg·s)]	136.250	137.500	138.750	140.000	141.250	142.500	143.750
K_{xis} [1/(deg·s)]	-0.550	-0.500	-0.450	-0.400	-0.350	-0.300	-0.250
K_{zp} [s/deg]	-1.547	-1.494	-1.433	-1.362	-1.282	-1.192	-1.097
K_{zr} [s/deg]	16.077	16.271	17.953	18.877	21.015	21.585	21.825
K_{zb} [1/deg]	-53.307	-53.150	-53.047	-52.802	-52.689	-52.477	-51.875
K_{zs} [1/deg]	-5.560	-5.105	-4.130	-3.885	-3.621	-3.092	-3.051
K_{zib} [1/(deg·s)]	-61.481	-61.302	-61.167	-59.627	-58.257	-57.386	-56.975
K_{zis} [1/(deg·s)]	-0.584	-0.567	-0.548	-0.525	-0.498	-0.467	-0.434

(Continues on next page.)

Table B-10: Membership parameters of the optimal fuzzy logic structure without integral terms.

Parameter	$\mathbf{R}_1$	$\mathbf{R}_2$	$\mathbf{R}_3$	$\mathbf{R}_4$	$\mathbf{R}_5$	$\mathbf{R}_6$	$\mathbf{R}_7$
Δq_{max} [deg/s]	0.125	0.250	0.375	0.500	0.625	0.750	0.875
$\Delta \alpha_{max}$ [deg]	0.125	0.250	0.375	0.500	0.625	0.750	0.875
$(\Delta p_{max})_z$ [deg/s]	5.918	6.102	9.122	9.266	9.411	9.556	9.700
$(\Delta r_{max})_z$ [deg/s]	0.088	0.770	0.998	1.102	1.206	2.372	2.475
$(\Delta \beta_{max})_z$ [deg]	0.091	0.101	0.112	0.170	0.182	0.193	0.220
$(\Delta \sigma_{max})_z$ [deg]	0.243	0.453	0.681	1.033	1.822	2.236	16.589
$(\Delta p_{max})_x$ [deg/s]	3.274	3.374	3.474	3.732	3.895	4.307	9.900
$(\Delta r_{max})_x$ [deg/s]	0.050	3.949	3.999	4.049	4.615	4.665	4.715
$(\Delta \beta_{max})_x$ [deg]	0.930	0.940	0.950	0.960	0.970	0.980	0.990
$(\Delta \sigma_{max})_x$ [deg]	0.200	2.221	2.421	2.621	2.821	3.021	13.990
K_{yq} [s/deg]	36.250	37.500	38.750	40.000	41.250	42.500	43.750
K_{ya} [1/deg]	36.250	37.500	38.750	40.000	41.250	42.500	43.750
K_{xp} [s/deg]	6.020	6.040	6.060	6.080	6.010	6.120	6.140
K_{xr} [s/deg]	-4.640	-4.620	-4.100	-4.080	-4.060	-4.040	-4.020
K_{xb} [1/deg]	18.040	18.080	18.120	18.160	18.200	18.240	19.655
K_{xs} [1/deg]	-4.405	-4.365	-4.325	-4.160	-4.120	-4.080	-4.040
K_{zp} [s/deg]	-0.616	-0.493	-0.464	-0.435	-0.405	-0.376	-0.347
K_{zr} [s/deg]	10.115	10.322	10.530	14.861	15.070	15.276	15.608
K_{zb} [1/deg]	-21.428	-21.118	-20.932	-20.746	-20.560	-20.374	-20.189
K_{zs} [1/deg]	-2.126	-2.076	-2.027	-1.789	-1.739	-1.690	-1.578

B-4 Model Predictive Controllers

This section list the tables containing the parameters of the state and control weighting-matrices of the various MPCs. Sections B-4-1, B-4-2, B-4-3, and B-4-4 list the parameter tables of both process models of the MPC^{bm}, MPC^{mfc}, MPC^{lqic}, and MPC^{mfic}, respectively.

B-4-1 MPC Analogue of the Benchmark Controller

Table B-11: Parameter values of the weighting-matrices for the $\text{MPC}_{\text{IO}}^{\text{bm}}$, scheduled as function of the dynamic pressure. When the actuators are not active, it is noted by its parameter value being Not Active (NA).

	Case 1	Case 2	Case 3	Case 4	Case 5
Δp_{max} [deg/s]	2.5	3	10	13	15
Δq_{max} [deg/s]	4	4	4	4	4
Δr_{max} [deg/s]	2.5	4	10	13	15
$\Delta \alpha_{max}$ [deg]	1	1	1	1	1
$\Delta \beta_{max}$ [deg]	0.1	0.1	0.1	0.1	0.1
$\Delta \sigma_{max}$ [deg]	6	7.5	10	10	10
$\Delta \delta_{a_{max}}$ [deg]	40	120	40	40	40
$\Delta \delta_{e_{max}}$ [deg]	40	40	40	40	40
$\Delta \delta_{r_{max}}$ [deg]	80	50	40	40	40
$\Delta M_{T,x_{max}}$ [Nm]	NA	NA	NA	NA	NA
$\Delta M_{T,y_{max}}$ [Nm]	10,400	NA	NA	NA	NA
$\Delta M_{T,z_{max}}$ [Nm]	7,600	7,600	7,600	7,600	7,600

Table B-12: Parameter values of the weighting-matrices for the MPC^{bm}_{IIO} , scheduled as function of the dynamic pressure. When the actuators are not active, it is noted by its parameter value being NA.

	Case 1	Case 2	Case 3	Case 4	Case 5
Δp_{max} [deg/s]	2.5	4	8	9.5	10
Δq_{max} [deg/s]	4	4	4	4	4
Δr_{max} [deg/s]	3.5	3.5	8	9.5	10
$\Delta \alpha_{max}$ [deg]	1	1	1	1	1
$\Delta \beta_{max}$ [deg]	0.1	0.1	0.1	0.1	0.1
$\Delta \sigma_{max}$ [deg]	5	6	10	10	10
$\delta(\delta_{a_{max}})$ [deg]	6	4	2.1	1.8	1.8
$\delta(\delta_{e_{max}})$ [deg]	2.5	2.5	2.5	2.5	2.5
$\delta(\delta_{r_{max}})$ [deg]	6	4.5	2.1	1.8	1.8
$\delta M_{T,x_{max}}$ [Nm]	NA	NA	NA	NA	NA
$\delta M_{T,y_{max}}$ [Nm]	150	NA	NA	NA	NA
$\delta M_{T,z_{max}}$ [Nm]	200	100	100	100	100

B-4-2 MPC Analogue of the MFC

Table B-13: Parameter values of the weighting-matrices for the MPC^{mfc}_{IO}, scheduled as function of the dynamic pressure.

	Case 1	Case 2	Case 3	Case 4	Case 5
Δp_{max} [deg/s]	5	6	10	13	15
Δq_{max} [deg/s]	4	4	4	4	4
Δr_{max} [deg/s]	5	6	10	13	15
$\Delta \alpha_{max}$ [deg]	1	1	1	1	1
$\Delta \beta_{max}$ [deg]	0.1	0.1	0.1	0.1	0.1
$\Delta \sigma_{max}$ [deg]	8	8	9	10	10
$\eta_{x_{max}}$ [-]	3	3	2.5	2	2
$\eta_{y_{max}}$ [-]	2	2	2	2	2
$\eta_{z_{max}}$ [-]	3	3	2.5	2	2

Table B-14: Parameter values of the weighting-matrices for the MPC^{mfc}_{IIO}, scheduled as function of the dynamic pressure.

	Case 1	Case 2	Case 3	Case 4	Case 5
Δp_{max} [deg/s]	5	5	6.5	9	9
Δq_{max} [deg/s]	4	4	4	4	4
Δr_{max} [deg/s]	5	5	6.5	9	9
$\Delta \alpha_{max}$ [deg]	1	1	1	1	1
$\Delta \beta_{max}$ [deg]	0.1	0.1	0.1	0.1	0.1
$\Delta \sigma_{max}$ [deg]	7	7.5	7.5	9	9
$\delta\eta_{x_{max}}$ [-]	0.15	0.15	0.05	0.05	0.05
$\delta\eta_{y_{max}}$ [-]	0.25	0.15	0.05	0.05	0.05
$\delta\eta_{z_{max}}$ [-]	0.175	0.175	0.06	0.06	0.05

B-4-3 MPC Analogue of the LQIC

Table B-15: Parameter values of the weighting-matrices for the MPC^{lqic}_{IO}, scheduled as function of the dynamic pressure. When the actuators are not active, it is noted by its parameter value being NA.

	Case 1	Case 2	Case 3	Case 4	Case 5
Δp_{max} [deg/s]	2.5	3	10	11	12
Δq_{max} [deg/s]	4	4	4	4	4
Δr_{max} [deg/s]	3	4	10	11	12
$\Delta \alpha_{max}$ [deg]	1	1	1	1	1
$\Delta \beta_{max}$ [deg]	0.1	0.1	0.1	0.1	0.1
$\Delta \sigma_{max}$ [deg]	6	8	10	10	10
$I_{\alpha,max}$ [deg·s]	2.5	2.5	2.5	2.5	2.5
$I_{\beta,max}$ [deg·s]	0.1	0.1	0.1	0.1	0.1
$I_{\sigma,max}$ [deg·s]	80	80	80	80	80
$\Delta \delta_{a_{max}}$ [deg]	40	40	40	40	40
$\Delta \delta_{e_{max}}$ [deg]	40	40	40	40	40
$\Delta \delta_{r_{max}}$ [deg]	80	50	40	40	40
$\Delta M_{T,x_{max}}$ [Nm]	NA	NA	NA	NA	NA
$\Delta M_{T,y_{max}}$ [Nm]	10,400	NA	NA	NA	NA
$\Delta M_{T,z_{max}}$ [Nm]	7,600	7,600	7,600	7,600	7,600

Table B-16: Parameter values of the weighting-matrices for the MPC^{lqic}_{IIO}, scheduled as function of the dynamic pressure. When the actuators are not active, it is noted by its parameter value being NA.

	Case 1	Case 2	Case 3	Case 4	Case 5
Δp_{max} [deg/s]	2.5	4.5	7	9	10
Δq_{max} [deg/s]	1	1	4	4	4
Δr_{max} [deg/s]	3	4	7.5	10	10
$\Delta \alpha_{max}$ [deg]	1	1	1	1	1
$\Delta \beta_{max}$ [deg]	0.1	0.1	0.1	0.1	0.1
$\Delta \sigma_{max}$ [deg]	5	7	10	10	10
$I_{\alpha,max}$ [deg·s]	1	1	2.5	2.5	2.5
$I_{\beta,max}$ [deg·s]	0.1	0.1	0.1	0.1	0.1
$I_{\sigma,max}$ [deg·s]	25	20	50	80	80
$\delta(\delta_{a_{max}})$ [deg]	6	4	2.1	1.8	1.8
$\delta(\delta_{e_{max}})$ [deg]	2.5	2.5	2.5	2.5	2.5
$\delta(\delta_{r_{max}})$ [deg]	6	4.5	2.1	1.8	1.8
$\delta M_{T,x_{max}}$ [Nm]	NA	NA	NA	NA	NA
$\delta M_{T,y_{max}}$ [Nm]	150	NA	NA	NA	NA
$\delta M_{T,z_{max}}$ [Nm]	200	100	100	100	100

(Continues on next page.)

B-4-4 MPC Analogue of the MFIC

Table B-17: Parameter values of the weighting-matrices for the MPC_{IO}^{mfic}, scheduled as function of the dynamic pressure.

	Case 1	Case 2	Case 3	Case 4	Case 5	Hybrid Case
Δp_{max} [deg/s]	2.5	5	9	11	13	2.5
Δq_{max} [deg/s]	4	4	4	4	4	4
Δr_{max} [deg/s]	3	4	9	11	13	3
$\Delta \alpha_{max}$ [deg]	1	1	1	1	1	1
$\Delta \beta_{max}$ [deg]	0.1	0.1	0.1	0.1	0.1	0.1
$\Delta \sigma_{max}$ [deg]	6	8	10	10	10	6
$I_{\alpha,max}$ [deg·s]	2.5	2.5	2.5	2.5	2.5	2.5
$I_{\beta,max}$ [deg·s]	0.1	0.1	0.1	0.1	0.1	0.1
$I_{\sigma,max}$ [deg·s]	80	50	80	80	80	80
$\eta_{x_{max}}$ [-]	2	1	0.5	0.5	0.5	0.5
$\eta_{y_{max}}$ [-]	2	2	2	2	2	1
$\eta_{z_{max}}$ [-]	2	1	0.5	0.5	0.5	2

Table B-18: Parameter values of the weighting-matrices for the MPC_{IIO}^{mfic}, scheduled as function of the dynamic pressure.

	Case 1	Case 2	Case 3	Case 4	Case 5
Δp_{max} [deg/s]	3.5	4.5	7.5	9	10
Δq_{max} [deg/s]	1	1	4	4	4
Δr_{max} [deg/s]	3	4.5	7	9	10
$\Delta \alpha_{max}$ [deg]	1	1	1	1	1
$\Delta \beta_{max}$ [deg]	0.1	0.1	0.1	0.1	0.1
$\Delta \sigma_{max}$ [deg]	7	7	10	10	10
$I_{\alpha,max}$ [deg·s]	1	1	2.5	2.5	2.5
$I_{\beta,max}$ [deg·s]	0.5	0.1	0.1	0.1	0.1
$I_{\sigma,max}$ [deg·s]	25	20	50	50	80
$\delta\eta_{x_{max}}$ [-]	0.25	0.2	0.02	0.02	0.02
$\delta\eta_{y_{max}}$ [-]	0.25	0.15	0.05	0.05	0.02
$\delta\eta_{z_{max}}$ [-]	0.25	0.2	0.02	0.02	0.02