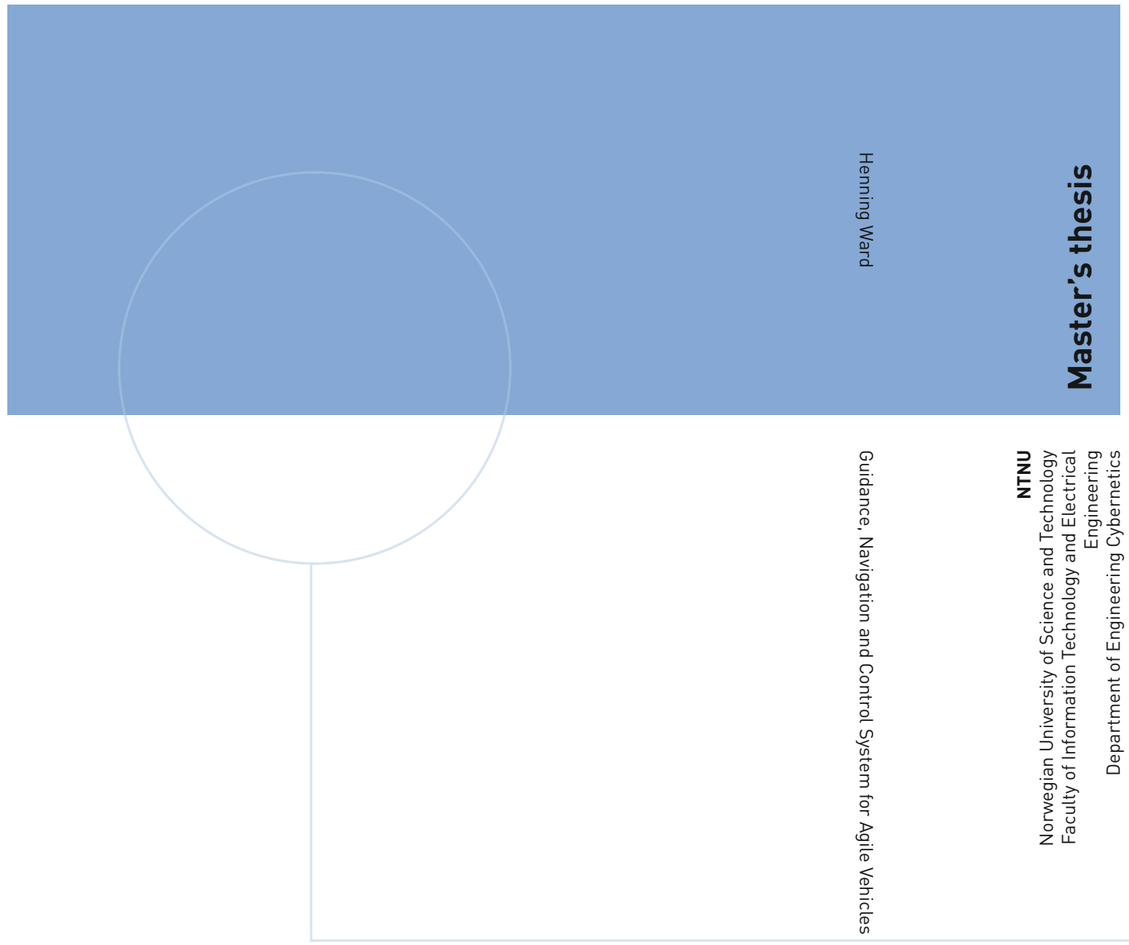

Henning Ward

Guidance, Navigation and Control System for Agile Vehicles

Master's thesis in Engineering Cybernetics Supervisor:
Thor I. Fossen
Co-supervisor: Murat Arcak, UC Berkeley
June 2019

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

Henning Ward

Guidance, Navigation and Control System for Agile Vehicles

Master's thesis in Engineering Cybernetics
Supervisor: Thor I. Fossen
Co-supervisor: Murat Arcak, UC Berkeley
June 2019

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Faculty of Information Technology
and Electrical Engineering
Department of Engineering Cybernetics

MSC THESIS DESCRIPTION SHEET

Name: Henning Ward
Department: Engineering Cybernetics
Thesis title: Guidance, Navigation and Control System for Agile Vehicles

Thesis Description:

The goal of the project is to develop a guidance, navigation and control (GNC) system for a target-tracking scenario. The project includes course autopilot design, target tracking, design of an inertial navigation system using the error-state Kalman filter and development of a guidance law for precision control using output feedback.

The following items should be considered:

1. Literature study and review of GNC systems.
2. Create a simulation environment for the the vehicle and its sensory systems. This is to be used to test the GNC algorithms.
3. Investigate, develop and discuss the need for the error-state Kalman filter for inertial navigation where the goal is to process inertial, magnetometer and GNSS measurements at different measurement rates.
4. Investigate, develop and discuss the need for a guidance and control system for target tracking and precise control using output feedback.
5. Verification and validation by computer simulations in a realistic missile target-tracking scenario.
6. Conclude your findings in a report.

Start date: 2019-01-07
Due date: 2019-07-15

Thesis performed at: Department of Engineering Cybernetics, NTNU
Supervisor: Professor Thor I. Fossen, Dept. of Eng. Cybernetics, NTNU
Co-Supervisor: Professor Murat Arcak, EECS, UC Berkeley, USA

Abstract

A reliable Guidance, Navigation and Control (GNC) system is an important part of a successfull target-tracking scenario. In this thesis, a GNC system is developed and compared to state-of-the-art systems widely used today. The work includes the development of an autopilot, guidance law, target tracking law and a reliable inertial navigation system for precicely controlling an agile vehicle such as aircraft, missiles, etc. using available sensor measurements. In this thesis the GNC system is simulated in a MATLAB/Simulink enviroment using a non-linear generic missile model.

The first part of the GNC system to be investigated is the control system. Two autopilots are considered:

The first design is the widely used three-loop autopilot. Based on desired acceleration commands from the guidance system, the autopilot calculates optimal missile fin deflections to navigate towards a target. The second design consists of two decoupled autopilots for lateral and longitudinal control using course and flight-path-angle as reference commands. By using a Linear-Quadratic Regulator (LQR), based on the linearized generic missile model, fin deflections are produced to achieve the desired missile orientation. Performance and robustness properties are enhanced by using an additional feedback from sideslip and angle-of-attack derivatives.

The second part of the GNC system to be investigated is the navigation system. Without reliable sensors and filters, other subsystems in the control loop will loose track of the Position, Velocity and Attitude (PVA) of the vehicle. A Global Navigation Satellite Systems (GNSS)-aided Multiplicative Extended Kalman Filter (MEKF) using gyro and accceleration biases is derived to ensure convergence of the vehicle states. The MEKF differs from the standard Extended Kalman Filter (EKF) by using quaternion multiplication to update the Inertial Navigation System (INS) estimations for the attitude, resulting in the additional multiplicative property. In a target-tracking scenario, information about the target position, velocity, and, in certain cases, acceleration is important when computing guidance commands. In addition to the estimated missile states provided by the INS, a target-tracking Kalman Filter (KF) is implemented to keep track of the relative states between the target and missile.

Finally, to conclude the GNC design, two guidance laws are compared. The well known Proportional Navigation (PN) law in combination with the three-loop autopilot described above, is compared to a Line-Of-Sight (LOS) design combined with the course and flight-path-angle controlled autopilot. By assuming independent control of the horizontal and vertical plane, the LOS guidance objective is to control the missile towards a vector between its launch platform and the estimated position of interception between the missile and target.

The GNC system is simulated using Simulink and shows promising results in both reference tracking for the autopilot, as well as state-estimations using both KF designs.

Sammendrag

Et pålitelig guiding-, navigasjons- og kontrollsystem (GNC) er en viktig del av et vellykket målsporings-scenario. I denne avhandlingen vil et GNC-system bli utviklet, og sammenlignet med moderne systemer, mye brukt i dag. Dette inkluderer utvikling av en autopilot, sporing, og et pålitelig treghetsnavigasjonssystem for nøyaktig styring av farkosten ved hjelp av tilgjengelige sensormålinger. For denne oppgaven vil GNC-systemet simuleres i et MATLAB / Simulink-miljø, ved hjelp av en ikke-lineær generisk missilmodell.

Den første delen av GNC-systemet som skal undersøkes er kontrollsystemet. To autopiloter sammenlignes:

Den første er den mye brukte three-loop autopiloten. Ved å utnytte akselerasjonskommandoer beregner autopiloten optimale missilfinnebøyninger for å navigere mot målet. Det andre autopilotsystemet består av to uavhengige autopiloter for horisontal og vertikal kontroll ved hjelp av kursreferansekommandoer. Ved å bruke en lineær kvadratisk regulator LQR basert på den lineære generiske missilmodellen, produseres finnebøyninger for å oppnå ønsket missilretning. Egenskaper for ytelse og robusthet undersøkes, og tilpasses ved bruk av en ekstra tilbakekobling fra den deriverte av sideslip og angrepsvinkel.

Den andre delen av GNC-systemet som skal undersøkes er navigasjonssystemet. Uten pålitelige sensorer og filtre vil andre delsystemer i kontrollsløyfen miste kontrollen over posisjon, hastighet og orientering av missilet. Et globalt navigasjonssatellittsystem GNSS brukt for å veilede ett multipliserende utvidet Kalman-filter (MEKF) vil sikre konvergens av missiltilstandene, inkludert gyro- og akselerasjonsforstyrrelser. MEKF er forskjellig fra et standard utvidet Kalman filter da oppdatering av orienteringen ved hjelp av quaternion-multiplikasjon resulterer i den ekstra multiplikative egenskapen. I et målsporingsscenario er informasjon om posisjon, hastighet og noen ganger akselerasjon til målet viktig når refansekommandoer skal regnes ut. I tillegg til å beregne missiltilstandene ved hjelp av treghetsnavigasjonssystemet, vil et ekstra Kalman Filter implementeres for å holde oversikt over de relative tilstandene mellom målet og missilet.

Til slutt vil to guidingslover sammenlignes for å fullføre GNC-designet. Den velkjente ”Propoertional-Navigation”-loven i kombinasjon med ”three-loop” autopiloten beskrevet ovenfor, vil bli sammenlignet med en LOS guidingslov kombinert med autopiloten for horisontal og vertikal kontroll. Ved å anta uavhengig kontroll av horisontalt og vertikalt plan, er målet for LOS guidingsloven å styre missilet mot en vektor mellom en utskytningsplattform og den estimerte posisjonen for optimal kollisjon mellom missilet og målet.

GNC-systemet simuleres ved hjelp av Simulink og viser lovende resultater i både referansesporing for autopiloten, samt tilstandsestimering for begge Kalman filterdesignene.

Preface

This master thesis concludes the mandatory final evaluation of the Master of Engineering program provided by the Department of Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU).

Some of the work done in this master thesis is built upon my previous work last semester in the TTK4550 Specialization project. This includes a Guidance, Navigation and Control System for three degrees of freedom. Details can be found in the attached file "prosjektoppgave_ward.pdf".

Knowledge of basic control theory, Kalman filtering and an understanding of GNC systems will be useful in order to understand the content of this thesis. The reader should also be familiar with classical physics such as kinematics and kinetics expressed in terms of ordinary differntial equations.

I would like to thank my supervisor Professor Thor Inge Fossen and Professor Murat Arcak for supervising and guiding me throughout this thesis. I would also like to thank the "BCCI family" at UC Berkeley, for all the good memories, both inside and outside of the lab. I would especially like to thank Emmanuel Sin for sharing his knowledge and work, including some of the Simulink models and MATLAB code. I also appreciate all the support you have given, and all the social events you have arranged during this semester.

Finally I would like to thank my father for all the support, and for always being there when things have been challenging.

Table of Contents

List of Tables

List of Figures

Abbreviations

CAD Computer Aided Design. 7, 77

CPA Closest Point of Approach. 71, 74, 98

DOF Degrees Of Freedom. 7, 15, 19

ECI Earth Centered Inertial. 9

EKF Extended Kalman Filter. i, 49

FOG Fiber Optic Gyros. 44

GNC Guidance, Navigation and Control. i, iii, xi, 3, 5–7, 15, 20, 75

GNSS Global Navigation Satellite Systems. i, 3, 5, 42, 44, 46, 97

IMU Inertial Measurement Unit. 3, 5, 26, 39, 42, 44, 46, 48, 82, 85

INS Inertial Navigation System. i, ix, xi, 5–7, 25, 41–43, 46, 48, 75, 83, 85, 89, 95, 97

KF Kalman Filter. i, 5–7, 53, 54, 61, 72, 82, 90, 91

LOS Line-Of-Sight. i, vii, ix, xii, 5, 29, 61, 63, 64, 66, 67, 73, 75, 76, 92–96, 98

LQR Linear-Quadratic Regulator. i, vi, 7, 32–34, 38

MEKF Multiplicative Extended Kalman Filter. i, ix, 5, 7, 42, 49, 57, 76, 82, 85, 91, 97

MEMS Micro Electrical Mechanical Systems. 44

NED North East Down. v, 10, 11, 21, 73

ONR Office of Naval Research. 2

PN Proportional Navigation. i, vii, xii, 5, 60, 61, 64–66, 72, 75, 76, 83, 86, 92–98

PVA Position, Velocity and Attitude. i, 3, 5, 41, 42, 44, 46, 49

RMS Root Mean Square. 81, 88

Chapter 1

Introduction

Historically, rockets dates all the way back to 1232, used by the Chinese as unguided missiles against the Mongol besiegers. It was not until the first world war that the idea of using guided missiles erupted (Siouris, 2004). Since then, the efficiency and precision of guided missiles has radically improved.

The main difference between a guided missile and an unguided rocket is the ability to change its course once it is airborne. Just like a cannon ball, an unguided rocket will follow a ballistic trajectory. This makes it considerably inaccurate, making it easily affected by external disturbances such as wind. In contrast to the unguided rocket, the guided missile is able to alter its flight path after being launched. This gives the ability to control the missile, either by a human operator or an autonomous system e.g. a weapon control system. In order to do this, the missile must carry additional components compared to an unguided rocket, including sensors and control systems.

Tactical missiles are able to intercept targets at a range of tens of kilometers, and are mainly used to aim for military targets. Missiles have the possibility to be launched from both air platforms as well as different ground based platforms, including sea-based platforms. These capabilities establishes effective ways to achieve a wide varity of military missions. This includes denying enemy communications or supply routes, establishing air superiority and performing anti-ship cruise missile or ballistic missile defense (Palumbo, 2010). In order to efficiently achieve the goal of interception, a robust guidance navigation and control system has to be implemented.

In this thesis, the focus will be to investiagte GNC-systems applied for defensive purposes.

1.1 Research topic

The work behind this thesis is carried out with the means of a contribution to the Office of Naval Research (ONR) Navy and Marine Corps FY2018 Basic Research Challenge (BRC) Program.

Fom ”ONR Basic Research Challenge Topic 4” (p. 10 - 11) the background and some of the research topics are given as (ONR, 2001):

“ State of the art methods for achieving guaranteed stability for both deterministic and uncertain control systems are based on well-developed mathematical tools such as gain margin analysis, μ analysis, etc. ”

“As multibody control systems play increasingly critical roles in the Navy’s ability to project force, the need for improved analysis and design methods becomes more pressing. Multibody control systems are increasingly prevalent in military and civilian applications ranging from autonomous automobiles to defensive systems.”

“ The proposed research seeks to develop methods with computational cost similar to stability analysis but with the power to provide a mathematically guaranteed assessment of mission performance for multibody control systems.”

“ The success of this proposed research will be measured by how far the gap between current capabilities for a single body system and the ability to mathematically guarantee performance of a multibody system can be bridged. In practical terms, a successful research program will also produce validation data for the mathematical framework. Since this program will be limited to basic research, the validation data will be obtained numerically via simulations. The main validation points are (1) correct performance prediction for a multibody control system, and (2) algorithm efficiency relative to the Monte Carlo method.”

My contribution has been to include uncertainty in the missile states and target tracking information, contributing to a more realistic simulation environment for use in suitable parts of the research.

1.2 Motivation

Many functions must be carried out in order to intercept and negate the target. The purpose of this thesis is to develop a Guidance, Navigation and Control (GNC) system for an tactical missile, using a Global Navigation Satellite Systems (GNSS)/Inertial Measurement Unit (IMU) integrated navigation system, applied for defensive purposes. In this thesis, all the major parts of the GNC system will be investigated, and simulations for different target-tracking scenarios will be carried out.

The main parts of the GNC system is as follows:

1. **Guidance:** The guidance system is responsible of computing essential guidance commands in order to successfully track down and intercept a target. This can be carried out using different algorithms, whereas two different methods will be investigated in this thesis.
2. **Navigation:** Keeping track of the current PVA of the missile is important. The same goes for keeping track of the target state information. Accurate terminal position measurements are inevitable in order to achieve an engagement between the missile and target. This is what the navigation system is responsible for. Without a good inertial reference system for stabilizing target line-of-sight measurements and keeping track of its own state information, computation of accurate missile guidance commands are difficult.
3. **Control:** While the guidance system is responsible of computing guidance commands, the control system is accountable of manipulating the missile actuators such that these guidance commands are successfully achieved. For a missile, this is equivilant of performing fin deflections to manipulate its states using a robust and efficient autopilot.

1.3 Organization of thesis

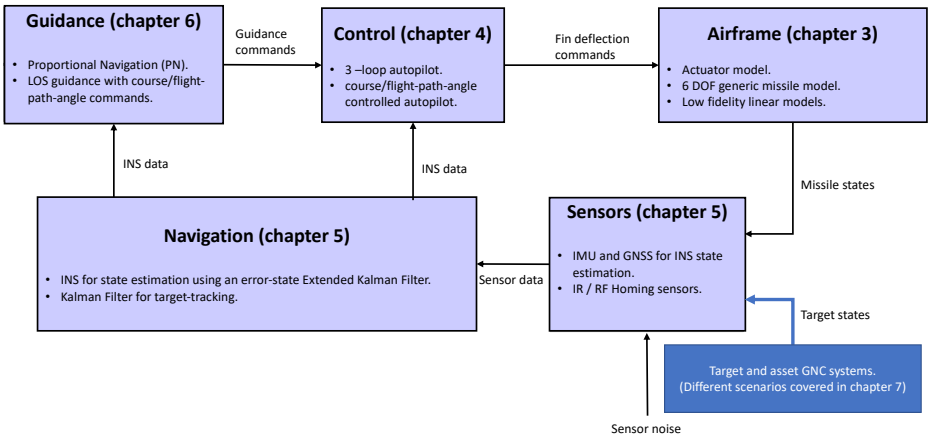

Figure 1.1: Organization of thesis, including the primary content of the main chapters.

The relation between some of the chapters in this thesis, as well as some of the main content is illustrated in Figure 1.1. The organization of the respective chapters are as follows:

Chapter 2: Background

Chapter 2 contains notation and some basic background information.

Chapter 3: Airframe

Chapter 3 will address the generic missile modeling. This includes aerodynamics forces and moments, coefficients and physical characteristics, all needed to construct a realistic missile simulator. Both high-fidelity and low-fidelity models will be outlined. In addition to this, an actuator model will be derived. Finally a reference model to filter out steps in position and attitude references will be discussed.

Figure 1.1 shows that the Airframe is responsible of calculating the missile states based on its previous states and fin deflection commands received from the control system.

Chapter 4: Control System

Today there are several different approaches for designing missile autopilots. This includes different non-linear approach as well as the three-loop-autopilot, which is often the design

topology of choice (Mracek and Ridgely, 2005). This relies on producing fin deflections based on acceleration command inputs for separated pitch- and yaw-channels of the missile dynamics. With appropriate sign changes, the same feedback signals with the same gains can be used for both channels.

One of the main objectives of this report is to investigate the development of a novel autopilot design, using course angle and flight-path angle as reference signals. To ensure desired robustness properties, a feedback from the derivatives of sideslip angle and flight-path angle will be introduced, with the feedback gains as tuneable parameters. This topology will then be compared and combined with the state-of-the-art three-loop autopilot to gain a better insight in the actual performance of the design.

Figure 1.1 shows how the control system receives both data from the Inertial Navigation System (INS), as well as guidance commands from the guidance system in order to produce the desired fin deflection commands. Note that the guidance commands can be both in terms of accelerations and course/flight-path angles. The two different autopilots will be able to transform the two different command types respectively, as outlined above.

Chapter 5: Navigation System

The thesis will also investigate the challenge of estimating the Position, Velocity and Attitude (PVA) using a GNSS/Inertial Measurement Unit (IMU) integrated navigation system. The IMU measurements are only valid for a short time without aiding, as sensors are vulnerable to noise and biases, causing drift in the state estimates (Qi and Moore, 2002). Different approaches may be used to estimate the biases and then update the attitude, position and velocity estimates. In this report, an error-state Multiplicative Extended Kalman Filter (MEKF) with GNSS aiding will be designed. The measurement equations will be calculated in two different ways, one including an additional Kalman Filter (KF) serving as a differentiator for calculating a specific force reference vector.

In addition to this, a 9-state KF will be introduced to estimate the target states based on available target-tracking sensors.

Chapter 6: Guidance System

The state-of-the-art Proportional Navigation (PN) law is widely used in several different applications. This includes surface-to-air, air-to-air and air-to-surface missile engagements, as well as space applications (Palumbo et al., 2010c). The PN law computes accelerations commands which are translated into fin deflection commands in the control system. This well known guidance law will be combined with the three-loop autopilot in contribution of the first GNC system used in later simulations.

The second guidance law derived in chapter 6 is using the Line-Of-Sight (LOS) angle between the predicted target position at the optimal interception time and a launch platform to compute course and flight-path angle commands directly. This differs from the PN law, as it doesn't apply any acceleration commands. The LOS guidance law will be combined

with the course and flight-path angle autopilot derived in chapter 4, which contributes to the second GNC system.

Chapter 7: Implementation

Chapter 7 describes how the GNC system is implemented. Different maneouvers will be used, and a three-body simulation scenario involving an interceptor, threat and a friendly asset will be investigated.

Chapter 8: Simulation Results

Description of cases studies, simulation parameters and simulation results will be carried out. Performance of the navigation system using different sampling rates will be investigated, and a comparison between the guidance laws will be carried out.

Chapter 9: Conclusion and further work

Conclusion of the thesis, as well as suggestions for further work are presented.

1.4 Mission objective

In this thesis, a surface-to-air tactical missile, also named the interceptor, is implemented with the purpose of tracking and intercepting an aerial target, also refered to as the threat. The aerial target is trying to intercept with a friendly asset. The mission objective is to protect the asset, by successfully destruct the threat before it is able to intercept with the friendly asset.

1.5 Requirement spesification

R.1 Implementation of an INS with accurate state estimations.

R.2 Implementation of an INS showing convergence of all state biases.

R.3 Successfull missile/target interception under different conditions, including target doing evasive manouvers.

R.4 Accurate target tracking using KF state estimations.

R.5 Comparing and validating two GNC systems with different guidance and control algorithms.

1.6 Contributions

The main contributions and work done in this thesis are:

Chapter 1: Littearature study and presentation of the missile GNC overview.

Chapter 3: Combination of a 6 Degrees Of Freedom (DOF) missile airframe with an actuator and reference model to create a realistic simulation enviroment. Derivation of low fidelity linearized models based on a high fidelity non-linear model.

Chapter 4: Derivation of a novel autopilot design by using a Linear-Quadratic Regulator (LQR) approach on the low-fidelity model from Chapter 3. The autopilot robustness properties are investigated and improved by using the derivatives of sideslip and angle-of-attack feedback in addition to integral action. This an extended and generalized design compared to the one in my TTK4550 specialization project report. The design is different than the three-loop autopilot in terms of course and flight-path angle references instead of acceleration.

Chapter 5: An INS with an error-state MEKF is derived. Different ways to calculate the measurement equations are explored, including the use of an KF differentiator for specific force reference calculation. This extends and generalizes the work in my specialization project report, where some other measurement equations were used in a simpler simulation environment. The combination of measurement equations and state parametrization in the error-state filter contributes to a design I have not been able to find elsewhere in the litterature.

Chapter 6: A novel guidance law is derived, designed to be used together with the course-controlled autopilot from chapter 4.

Chapter 7: MATLAB/Simulink implementation of the complete GNC system, able to successfully guide a missile towards a target in a realistic target-tracking scenario. This includes the two different autopilots and guidance systems. The INS with the error-state MEKF is implemented. The simulation enviroment also includes the three-body scenario with asset, threat and interceptor.

A way to visualize the rigid body following the given trajectory is derived. By translating Computer Aided Design (CAD) files to a format MATLAB is able to comprehend, and by plotting in a 3D MATLAB plot, the simulations are visualized in an elegant way. This includes better understanding of the rigid body attitude, in addition to better distinction between asset, threat and interceptor.

Chapter 8: Testing and verification of the GNC system through different simulation scenarios. Thoroughly testing the performance by using various sampling rates, initial conditions, maneouvers and different sensor characteristics. The novel methods introduced in this thesis is compared to the state-of-the-art methods presented. The different methods are compared in various ways, including target interception accuracy, state estimation accuracy, total force applied to the rigid body and effectiveness in terms of total flight time until interception.

Chapter 2

Background

2.1 Notation

The identity matrix is denoted $\boldsymbol{I}_{N\times N} \in \mathbb{R}^N$, while $(.)^T$ is the transpose of a matrix.

Coordinate frames are denoted $\{.\}$. Throughout this thesis vectors will be written in bold, while matrices will be written in bold capital letters. Vectors have superscripts to denote what frame the vector is composed in. Superscripts are used to describe relative frames.

Example

$$\boldsymbol{\omega}_{b/n}^{b} \quad \text{angular velocity of } \{\mathrm{b}\} \text{ with respect to } \{\mathrm{n}\} \text{ expressed in } \{\mathrm{b}\}$$

2.2 Coordinate systems

When describing the relationship between different states, calculating navigation equations and deriving the equations of motions for the system, different Cartesian coordinate systems are used:

2.2.1 Inertial Frame

Netwon's laws of motion are formulated in the inertial frame, which is denoted $\{\mathrm{i}\}$. The Earth Centered Inertial (ECI) frame is often used as an approximation of the inertial frame, and is an orthogonal frame that follows the Earth. The ECI frame has its origin at the center of earth, x-axis in the equatorial plane point towards vernal equinox, z-axis pointing along the Earth's rotational axis while the y-axis is completing the right-hand frame.

2.2.2 North-East-Down

The North East Down (NED) frame $\{n\} = (x_n, y_n, z_n)$ has its origin in an arbitrary point on the surface of the earth. The x-axis points towards north, the y-axis points towards east, while the z-axis points towards the center of the earth, completing the right-hand frame. The xy-plane is therefore tangential to the surface of the Earth. In the aircraft and marine craft control system litterature (Fossen, 2011), the Earth's rotation are often neglected, such that a local NED tangential frame can be used. This assumption is used for the rest of this thesis, such that the intertial frame $\{\mathrm{i}\}$ is replaced by the NED frame $\{\mathrm{n}\}$.

2.2.3 BODY

The BODY reference frame $\{b\} = (x_b, y_b, z_b)$ is fixed with respect to the missile. The x-axis points towards the direction of the missile, the y-axis points to the left while the z-axis points down towards the Earth's surface. The origin of the coordinate system coincides with the missile's center of mass. The body-fixed axis are illustrated in Figure 2.1.

2.2.4 Transformation between BODY and NED

To represent the attitude, the unit quaternion is used. The quaternion is defined as

$$\boldsymbol{q} = \begin{bmatrix} \eta \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{bmatrix} \tag{2.1}$$

where η is the real part and $\boldsymbol{\epsilon} = [\epsilon_1 \quad \epsilon_2 \quad \epsilon_3]^T$ is the imaginary part. The unit quaternion is defined as a quaternion satisfying the property $\boldsymbol{q}^T\boldsymbol{q} = 1$. By representing the attitude with unit quaternions, a singularity-free representation is achieved, giving better stability properties than by using e.g. Euler angles. The inverse of a unit quaternion is $\boldsymbol{q}^{-1} = [\boldsymbol{\eta} \quad \boldsymbol{\epsilon}^T]^T$

The unit quaternion is related to the axis and angle of a rotation by Markley (2008)

$$\boldsymbol{q} = \begin{bmatrix} \eta \\ \boldsymbol{\epsilon} \end{bmatrix} = \begin{bmatrix} \cos\frac{\phi}{2} \\ \boldsymbol{e}\sin\frac{\phi}{2} \end{bmatrix} \tag{2.2}$$

where $\boldsymbol{e}$ is a unit vector and ϕ is the angle of rotation.

Denoted as $\otimes$, the quaternion product between two unit quaternions produces a new unit quaternion, satisfying

$$\boldsymbol{q}_1 \otimes \boldsymbol{q}_2 = \begin{bmatrix} \eta_1\eta 2 - \boldsymbol{\epsilon}_1^T\boldsymbol{\epsilon}_2 \\ \eta_1\boldsymbol{\epsilon}_2 + \eta_2\boldsymbol{\epsilon}_1 + \boldsymbol{S}(\boldsymbol{\epsilon}_1)\boldsymbol{\epsilon}_2 \end{bmatrix} \tag{2.3}$$

The rotation matrix between two frames a and b is denoted $\boldsymbol{R}_{\boldsymbol{b}}^{\boldsymbol{a}}$. The relationship between the unit quaternion and the rotational matrix is

$$\boldsymbol{R}_b^n(\boldsymbol{q}) = \boldsymbol{I}_{3x3} + 2\eta\boldsymbol{S}(\boldsymbol{\epsilon}) + 2\boldsymbol{S}^2(\epsilon) \tag{2.4}$$

where $\boldsymbol{R}_b^n$ represents the rotation between the BODY and NED frame, while $\boldsymbol{S}(.)$ is a skew symmetric, satisfying

$$\boldsymbol{S}(\boldsymbol{\epsilon}) = \begin{bmatrix} 0 & -\epsilon_3 & \epsilon_2 \\ \epsilon_3 & 0 & -\epsilon_1 \\ -\epsilon_2 & \epsilon_1 & 0 \end{bmatrix} \tag{2.5}$$

For small values of $\delta\boldsymbol{\epsilon}$, the following approximations holds:

$$\boldsymbol{R}(\delta\epsilon) \approx \boldsymbol{I}_{3x3} + \boldsymbol{S}(\boldsymbol{a}_g) \tag{2.6}$$

$$\boldsymbol{R}(\delta\epsilon)^T \approx \boldsymbol{I}_{3x3} - \boldsymbol{S}(\boldsymbol{a}_g) \tag{2.7}$$

The derivative of the unit quaternion is

$$\dot{\boldsymbol{q}} = \boldsymbol{T}_q(\boldsymbol{q})\boldsymbol{\omega}_{b/n}^b \tag{2.8}$$

where

$$\boldsymbol{T}_q(\boldsymbol{q}) = \frac{1}{2}\begin{bmatrix} -\boldsymbol{\epsilon} \\ \eta\boldsymbol{I}_{3x3} + \boldsymbol{S}(\boldsymbol{\epsilon}) \end{bmatrix} \tag{2.9}$$

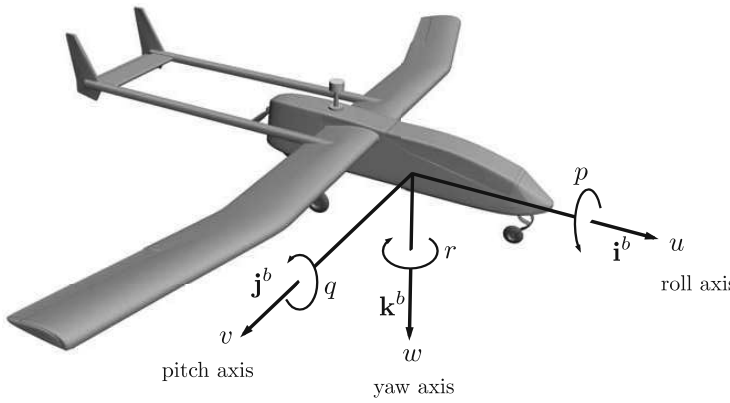

Figure 2.1: Body-fixed axis illustrated on an aircraft.

2.3 Course and flight-path angle

When developing the dynamic equations for the missile, it is important to remember that the aerodynamic forces are not only related to the airspeed, but also to the surrounding air. The relationship between the airspeed $\boldsymbol{V}_a$, ground speed $\boldsymbol{V}_g$ and wind speed $\boldsymbol{V}_w$ yields (Beard and McLain, 2013)

$$\boldsymbol{V}_a = \boldsymbol{V}_g - \boldsymbol{V}_w \tag{2.10}$$

Where $\boldsymbol{V}_g$ can be expressed in the body frame along the body-fixed i^b, j^b, and k^b axis from Figure 2.1 (Weehong Tan et al., 2000) as

$$\boldsymbol{V}_g^b = \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{2.11}$$

Similarily, it can be shown in Beard and McLain (2013) that the expression for the velocity of the rigid body with respect to the airspeed vector can be written as

$$\boldsymbol{V}_a^b = \begin{pmatrix} u_r \\ v_r \\ w_r \end{pmatrix} = \begin{pmatrix} u - u_w \\ v - v_w \\ w - w_w \end{pmatrix} \tag{2.12}$$

By assuming that the wind can be negligible (2.11) and (2.12) are simplified to

$$\boldsymbol{V}_g^b = \boldsymbol{V}_a^b \tag{2.13}$$

By inspecting Figure 2.2 and 2.3 (Beard and McLain, 2013) , the airspeed vector can be related to the sideslip and angle of attack by

$$\begin{pmatrix} u_r \\ v_r \\ w_r \end{pmatrix} = \boldsymbol{V}_a \begin{pmatrix} \cos\alpha\cos\beta \\ \sin\beta \\ \sin\alpha\cos\beta \end{pmatrix} \tag{2.14}$$

Inverting these relationships gives

$$V_a = \sqrt{u_r^2 + v_r^2 + w_r^2} \tag{2.15a}$$

$$\alpha = \tan^{-1}\left(\frac{w_r}{u_r}\right) \tag{2.15b}$$

$$\beta = \sin^{-1}\left(\frac{v_r}{U_r}\right) \tag{2.15c}$$

Which in absence of wind, simply gives

$$V_a = \sqrt{u^2 + v^2 + w^2} \tag{2.15d}$$

$$\alpha = \tan^{-1}\left(\frac{w}{u}\right) \tag{2.15e}$$

$$\beta = \sin^{-1}\left(\frac{v}{U}\right) \tag{2.15f}$$

The course angle χ is the angle between the airspeed vector and the body-fixed x-z plane

$$\chi = \psi + \beta \tag{2.16}$$

and describes the direction of travel for the rigid body. For the lateral autopilot design, it is desirable to control this instead of the heading, as the heading may be an inaccurate approximation of the course if the sideslip angle is sufficiently large. The relationship between these angles are shown in Figure 2.2.

The flight-path angle is the positive angle with respect to the airspeed vector that is used to generate lift. This is defined as

$$\gamma_\alpha = \theta - \alpha \tag{2.17}$$

For the longitudinal autopilot, the flight-path angle is the commanded reference. The longitudinal angles are shown in Figure 2.3. In the absence of wind, $\gamma_\alpha = \gamma$, such that

$$\gamma = \theta - \alpha \tag{2.18}$$

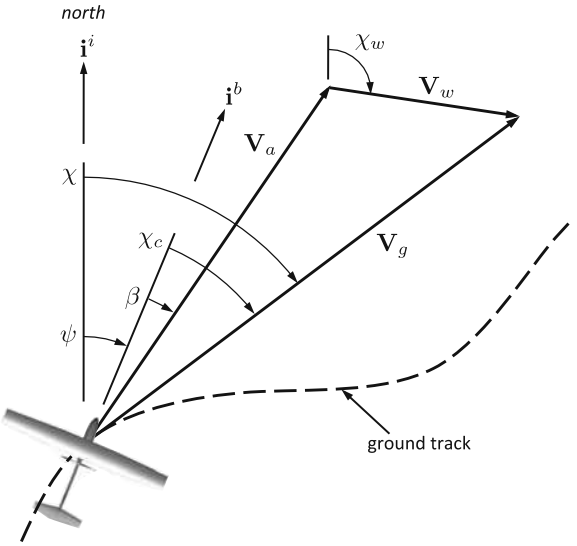

Figure 2.2: Relationship between speed, side-slip and course. The crab angle χ_c equals zero in absence of wind.

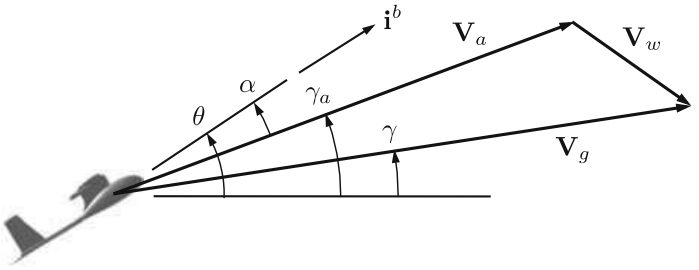

Figure 2.3: Relationship between speed, angle of attack and flight-path angle.

The combination of these control objectives, as well as the assumption of zero roll, are the foundation of the autopilot design.

Chapter 3

Airframe

3.1 Introduction

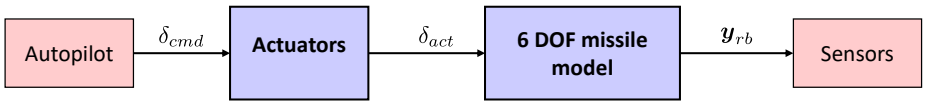

Figure 3.1: Actuator and rigid body model in relation to the rest of the GNC system.

Many flying objects such as aircraft and missiles can be approximated as rigid bodies with gravitational and aerodynamic forces acting on them (Bryson, 2015). By giving the position, velocity orientation and angular velocity of a set of body-fixed axis, it is possible to describe the motions of a rigid body. These parameters are the 6 DOF used in the following model. The missile model used is a generic surface-to-air missile model, and is axis-symmetric with cruciform wings. It is controlled using $skid - to - turn$, by manipulating the fin deflections which are entering the system linearly. When using the course-controlled autopilot, there is also a reference model used to filter out huge steps in the commanded fin deflection δ_{cmd}. Figure 3.1 shows how this chapter relates to the rest of the GNC system.

3.2 Actuator

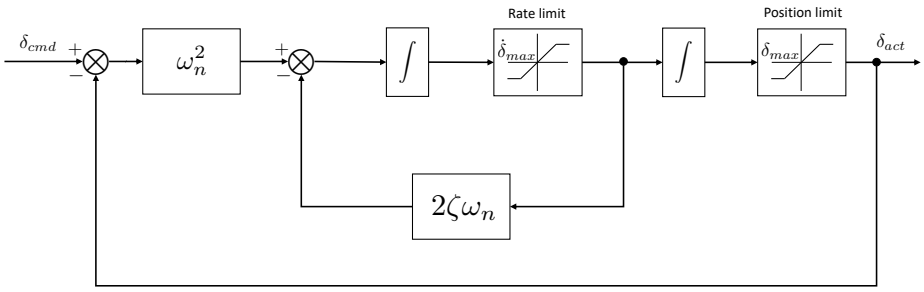

Figure 3.2: Block diagram illustrating the actuator model.

There are three fin actuators, each controlling deflections in pitch, yaw and roll, denoted $[\delta_P, \delta_Y, \delta_R]$. These actuators are generally acting independently, and are each driven by a separate but often identical servomotor (Çimen, 2011). The actuator dynamics are modeled using an actuator saturation that limits the position and rate of the aerodynamic control surface. The saturation function yields

$$\text{sat}(\delta, \delta_{max}) \triangleq \begin{cases} \delta_{max}, & \delta > \delta_{max} \\ \delta, & |\delta| \leq \delta_{max} \\ -\delta_{max}, & \delta < -\delta_{max} \end{cases} \tag{3.1}$$

The actuator dynamics for each seperate actuator channel can be described using a second order nonlinear differential equation (Çimen, 2011)

$$\ddot{\delta}_{act} = (\delta_{cmd} - \text{sat}(\delta_{act}, \delta_{max}))\omega_n^2 - 2\zeta\omega_n^2 \,\text{sat}(\dot{\delta}_{act}, \dot{\delta}_{max}) \tag{3.2}$$

Where ζ is the actuator damping ratio and ω_n is the natural frequency. $\delta_{max} = 25$ degrees and $\dot{\delta}_{max} = 4$ degrees / sec are the saturations for fin deflection position and position rates respectively.

The actuator block diagram, which also represents the Simulink implementation, is shown in figure 3.2.

3.3 Assumptions

The following assumptions are made for the missile model to be valid

- Rigid air frame, i.e. no flexible modes.
- Reference position coincides with the mass center of the missile.
- All sensors are mounted at the missile mass center.
- Good measurements or estimates of angle of attack, sideslip, accelerations and angles are obtained.
- There are no roll, which results in $p = \phi = 0$.
- The missile is operating at a constant speed.

3.4 State vector

The state variables used in the equations of motions for the missile is given in Table 3.1. Even though V_m is treated as a state variable in the missile model, it is assumed to be constant.

Notation	Description
α	Angle of attack
β	Side slip angle
V_m	Total speed of missile
p	Roll angular velocity, rotation around x-axis
q	Pitch angular velocity, rotation around y-axis
r	Yaw angular velocity, rotation around z-axis

Table 3.1: High fidelity missile model states

3.5 Aerodynamic coefficients and physical missile characteristics

3.5.1 Aerodynamic forces and moments

The aerodynamic forces and moments are modeled in terms of the aerodynamic coefficients, and are given by (Weehong Tan et al., 2000)

$$F_x = -\frac{1}{2}\rho V_m^2 A \left[C_{dap} + C_{day} + C_D\right] \tag{3.3a}$$

$$M_x = \frac{1}{2}\rho V_m^2 A \left[C_{l,PY} + C_{l,\alpha_R,\delta}\delta_R + \frac{C_{l,p}d}{2V_m}p\right] \tag{3.3b}$$

$$F_z = \frac{1}{2}\rho V_m^2 A C_{Nz} \tag{3.3c}$$

$$M_y = \frac{1}{2}\rho V_m^2 A \left[C_{My} + \frac{C_{Mtd}d}{2V_m}q\right] \tag{3.3d}$$

$$F_y = \frac{1}{2}\rho V_m^2 A C_{Ny} \tag{3.3e}$$

$$M_y = \frac{1}{2}\rho V_m^2 A \left[C_{Mz} + \frac{C_{Mtd}d}{2V_m}r\right] \tag{3.3f}$$

Where ρ is the air density, A is the reference area and d is the reference distance. Since the roll motion is controlled to zero, δ_R will be assumed equal to zero throughout the rest of the thesis.

3.5.2 Aerodynamic coefficients

By the use of wind-tunnel experiments, good approximations of the aerodynamic coefficients are obtained. By assuming a constant speed of $V_m = 3$ Mach, equal to 1020 m/s, the coefficients are given by Weehong Tan et al. (2000) as

Normal force coefficient:

$$C_{Nz}(\alpha, \delta_P) = -21\alpha - 24.5\alpha^3 - (2 - 0.1\sqrt{|\alpha|})\frac{6}{\pi}\delta_P \tag{3.4a}$$

Pitch moment coefficient:

$$C_{My}(\alpha, \delta_P) = -15.75\alpha - 36.75\alpha^3 - (2 - 0.1\sqrt{|\alpha|})\frac{42}{\pi}\delta_P \tag{3.4b}$$

Side force coefficient:

$$C_{Ny}(\beta, \delta_Y) = -21\beta - 24.5\beta^3 - (2 - 0.1\sqrt{|\beta|})\frac{6}{\pi}\delta_Y \tag{3.4c}$$

Yaw moment coefficient:

$$C_{Mz}(\beta, \delta_Y) = 15.75\beta + 36.75\beta^3 + (2 - 0.1\sqrt{|\beta|})\frac{6}{\pi}\delta_Y \tag{3.4d}$$

Axial force coefficient due to pitch motions:

$$C_{dap}(\alpha, \delta_P) = 0.5(\alpha + \delta_P)^2 \tag{3.4e}$$

Axial force coefficient due to yaw motions:

$$C_{day}(\beta, \delta_Y) = 0.5(\beta + \delta_Y)^2 \tag{3.4f}$$

Roll moment coefficient:

$$C_{l,PY}(\alpha, \beta) = -2.63\alpha_R^2 \sin 4\phi_A \tag{3.4g}$$

Roll fin effectiveness:

$$C_{l,\alpha_R,\delta}(\alpha, \beta) = 2.5 - 1.25\cos(\pi\alpha_R) \tag{3.4h}$$

Axial drag coefficient:

$$C_D = 0.2 \tag{3.4i}$$

Damping coefficients:

$$C_{Mtd}(M = 3) = -720 \quad C_{l,p}(M = 3) = -10 \tag{3.4j}$$

3.5.3 Physical characteristics

Since the dynamic pressure $\frac{1}{2}\rho V_m^2$ is related to the speed and altitude of the missile, it is necessary to approximate both the altitude and velocity. The altitude h of the missile is assumed to be about 7500 meter, while the speed as stated above is 3 Mach. The mass of the missile is 17 kg, while the diameter is 0.15 meter.

3.6 High-fidelity model

For the high fidelity model, the 6 DOF non-linear model described in Weehong Tan et al. (2000) is used. The equations of motions (EoM) yields

$$\dot{\alpha} = \cos^2\alpha \left[\frac{F_z Q_{\alpha\beta}}{mV_m} - p\tan\beta\right] + q - \sin\alpha\cos\alpha \left[\frac{F_x Q_{\alpha\beta}}{mV_m} - r\tan\beta\right] \tag{3.5}$$

$$\dot{q} = \frac{1}{I_{yy}}\left[M_y + (I_{zz} - I_{xx})rp\right] \tag{3.6}$$

$$\dot{\beta} = \cos^2\beta \left[\frac{F_y Q_{\alpha\beta}}{mV_m} + p\tan\alpha\right] - r - \sin\beta\cos\beta \left[\frac{F_x Q_{\alpha\beta}}{mV_m} - q\tan\alpha\right] \tag{3.7}$$

$$\dot{r} = \frac{1}{I_{zz}} \left[M_z + (I_{xx} - I_{yy}) pq \right] \tag{3.8}$$

$$\dot{V_m} = \frac{1}{mQ_{\alpha\beta}} \left[F_x + F_y \tan\beta + F_z \tan\alpha \right] \tag{3.9}$$

$$\dot{p} = \frac{1}{I_{xx}} \left[M_x + (I_{yy} - I_{zz}) qr \right] \tag{3.10}$$

Where $Q_{\alpha\beta}$ is defined as

$$Q_{\alpha\beta} = \sqrt{1 + \tan^2\alpha + \tan^2\beta}$$

While

$$I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix}$$

is the moment of inertia matrix, with zero elements except on the diagonal.

3.7 Airframe variables

The state variables from section 3.4 will be used to calculate the necessary variables needed for the rest of the GNC system . All the variables are contained in the vector $\boldsymbol{y}_{rb}$:

$$\boldsymbol{y}_{rb} = \begin{pmatrix} V_m \\ \alpha \\ \beta \\ \boldsymbol{\omega}^b_{b/n} \\ \boldsymbol{f}^b \\ \boldsymbol{v}^b_{b/n} \\ \boldsymbol{\Theta}_{nb} \\ \boldsymbol{p}^n_{b/n} \\ \boldsymbol{v}^n_{b/n} \end{pmatrix} \tag{3.11}$$

$\boldsymbol{\omega}^b_{b/n} = [p \quad q \quad r]^T$ is the angular velocity and $\boldsymbol{f}^b = [N_x \quad N_y \quad N_z]^T$ is the specific force. $\boldsymbol{v}^b_{b/n} = [u \quad v \quad w]^T$ is the velocity given in body frame, and is calculated from (2.14).

$\boldsymbol{\Theta}_{nb} = [\phi \quad \theta \quad \psi]^T$ is the attitude. The relationship between $\dot{\boldsymbol{\Theta}}_{nb}$ and the angular velocity $\boldsymbol{\omega}^b_{b/n}$ is (Fossen, 2011)

$$\dot{\boldsymbol{\Theta}}_{nb} = \boldsymbol{T}_{\boldsymbol{\Theta}}(\boldsymbol{\Theta}_{nb})\boldsymbol{\omega}^b_{b/n} \tag{3.12}$$

where

$$\boldsymbol{T}_{\boldsymbol{\Theta}}(\boldsymbol{\Theta}_{nb}) = \begin{bmatrix} 1 & \sin(\phi)\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

The attitude can then be calculated by integrated $\boldsymbol{\omega}^b_{b/n}$ after the rotation in (3.12), given that an initial attitude is provided.

The velocity in the NED frame $\boldsymbol{v}^n_{b/n}$ is calculated by a simple rotation $\boldsymbol{v}^n_{b/n} = \boldsymbol{R}^n_b \boldsymbol{v}^b_{b/n}$ while the position in NED frame is obtained from integrating $\dot{\boldsymbol{p}}^n_{b/n} = \boldsymbol{v}^n_{b/n}$ given an initial position.

3.8 Low-fidelity model

To calculate the feedback terms of the autopilot, a simplified model is used. Separation between the roll, pitch and yaw channel is a common assumption when designing autopilots for tactical missiles (Mracek and Ridgely, 2005). The low-fidelity model is therefore a decoupled and linearized version of the original high fidelity model about a given operating point.

In order to understand the dynamics of the system and to design a autopilot working in both lateral and longitudinal direction, it is often useful to linearize the system. This can, for instance, be done using gain scheduling. Gain scheduling is a common technique for controlling nonlinear systems. The dynamics of the system will often change from one operating condition to another. If the dynamics change in such a way that a single set of controller gains obtained around one set of operating points are insufficient, gain scheduling may be a considerable approach. The technique involves linearizing the system about different operating points, such that different sets controller gains are obtained. This will provide the desired performance and stability throughout the entire range of operating conditions for the system.
When deriving the course and flight-path-angle autopilot in this thesis, the model is only linearized about the equilibrium point for both the longitudinal and lateral dynamics. This will affect the system and will aggravate the performance when operating far from the linearized operating point. The three-loop autopilot on the other hand, is using Gain Scheduling. This is done by students under supervision of Professor Murat Arcak during the spring semester 2019.

3.8.1 Linearized lateral model

When deriving the equations for the lateral dynamics, (3.7) and (3.8) from the equations of motion is considered. By assuming decoupled roll, pitch and yaw dynamics, the simplification yields

$$\dot{\beta} = \cos^2\beta \frac{F_y Q_{\alpha\beta}}{mV_m} - r - \sin\beta\cos\beta \frac{F_x Q_{\alpha\beta}}{mV_m} \tag{3.13}$$

$$\dot{r} = \frac{M_z}{I_{zz}} \tag{3.14}$$

This is linearized about $\beta^* = 0$, $r^* = 0$, ${\delta_Y}^* = 0$ and can then be expressed as

$$\begin{bmatrix} \dot{\beta} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} -2.11 & -1 \\ 423.97 & -1.44 \end{bmatrix} \begin{bmatrix} \beta \\ r \end{bmatrix} + \begin{bmatrix} 0.39 \\ 719.75 \end{bmatrix} \delta_Y \tag{3.15}$$

3.8.2 Linearized longitudinal model

When deriving the autopilot for the longitudinal dynamics, it is only necessary to consider (3.5) and (3.6) from the equations of motion. By asuming decoupled roll, pitch and yaw dynamics, (3.5) and (3.6) can be further simplified as

$$\dot{\alpha} = \cos^2\alpha \frac{F_z Q_{\alpha\beta}}{mV_m} + q - \sin\alpha\cos\alpha \frac{F_x Q_{\alpha\beta}}{mV_m} \tag{3.16}$$

$$\dot{q} = \frac{M_y}{I_{yy}} \tag{3.17}$$

By linearizing this about $\alpha^* = 0$, $q^* = 0$ and ${\delta_P}^* = 0$ the linearized equations of motion for the longitudinal motion can be expressed as

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} -0.07 & 1 \\ -423.79 & -1.44 \end{bmatrix} \begin{bmatrix} \alpha \\ q \end{bmatrix} + \begin{bmatrix} 0 \\ -719.75 \end{bmatrix} \delta_P \tag{3.18}$$

3.9 Reference model

For trajectory tracking, a third order reference model is typically chosen to filter steps in the position and attitude reference (Fossen, 2011). For course control, consider the transfer function

$$\frac{\boldsymbol{\eta}_d}{\boldsymbol{r}^n}(s) = \frac{\omega_n^3}{s^3 + (2\zeta+1)\omega_n s^2 + (2\zeta+1)\omega_n^2 s + \omega_n^3} \tag{3.19}$$

where $\boldsymbol{r}^n$ denotes the reference input, while $\boldsymbol{\eta}_d = [\chi_d \quad \dot{\chi}_d \quad \ddot{\chi}_d]^T$ is the desired course vector.

The state-space representation yields

$$\dot{\boldsymbol{\eta}}_d = \boldsymbol{A}_r \boldsymbol{\eta}_d + \boldsymbol{B}_r \boldsymbol{r}^n \tag{3.20}$$

where

$$\boldsymbol{A}_r = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -\omega_n^3 & -(2\zeta+1)\omega_n^2 & -(2\zeta+1)\omega_n \end{bmatrix} \tag{3.21}$$

and

$$\boldsymbol{B}_r = \begin{bmatrix} 0 \\ 0 \\ \omega_n^3 \end{bmatrix} \tag{3.22}$$

satisfying

$$\lim_{t\to\infty} \boldsymbol{\eta}_d(t) = \boldsymbol{r}^n \tag{3.23}$$

Note that by using the third-order reference model, information about $\dot{\chi}_d$ and $\ddot{\chi}_d$ is obtained, giving more flexibility when it comes to choosing reference trajectories for the control system design.

Chapter 4

Control System

4.1 Introduction

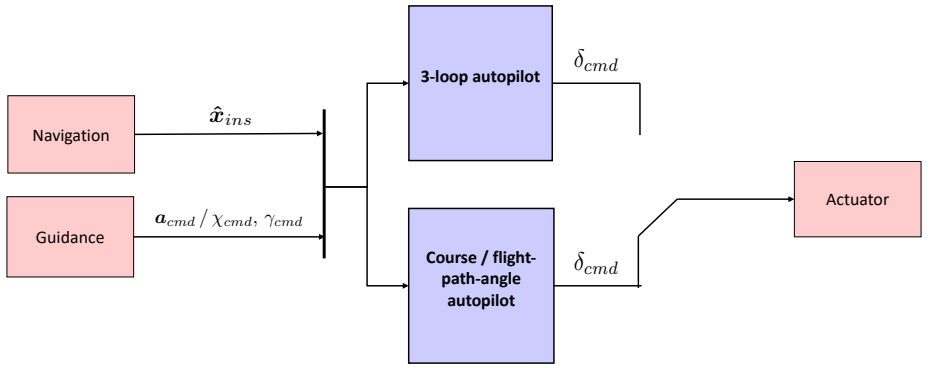

Figure 4.1: The autopilots uses information from the INS and guidance commands to calculate desired fin deflection commands. In this thesis, two different designs will be compared.

The design goal of missile autopilot is to produce a stable response to a given set of command inputs. The implementation of such autopilots has successfully been employed over the last 50 years, and the classical three-loop autopilot has been the desirable design topology (Mracek and Ridgely, 2005). Due to the acceleration command input, many of the design challenges presented by the homing missile relates to the need of integrating the autopilot into the guidance loop without heading errors in the terminal phase as well as avoiding stability problems (Horton, 1995). The controllers are usually designed to use

gain scheduling in order to operate in large flight envelopes (Mracek and Ridgely, 2005). Two different autopilots will be discussed in this chapter - the first one is the widely used three-loop autopilot, while the other one will be designed to rely on course and flight-path-angle command inputs.

4.1.1 Three-loop autopilot

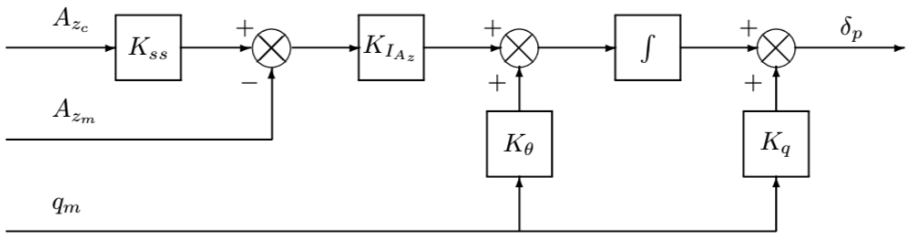

Figure 4.2: Classical three loop topology for pitch control. The same topology can be used for yaw control by applying the correct sign changes (Mracek and Ridgely, 2005).

The classic three loop autopilot uses acceleration and angular rate feedback as the sensed quantities to produce the desired fin deflections. There are several different typology's that uses the given feedback quantities to produce the desired performance. The robustness of several different topology's was studied by Mracek and Ridgely (2005), which concluded that the classical three-loop-autopilot gave the overall best robustness properties.

The basic longitudinal dynamics can be written, according to Mracek and Ridgely (2005), as

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{4.1a}$$

$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}u \tag{4.1b}$$

where

$$\boldsymbol{x} = \begin{bmatrix} \alpha \\ q \end{bmatrix} \quad u = \delta p \quad \boldsymbol{y} = \begin{bmatrix} A_{z_m} \\ q_m \end{bmatrix} \tag{4.1c}$$

The classic three loop topology are shown in figure 4.2. A_{z_c} represents the commanded acceleration, A_{z_m} is the measured linear acceleration, while q_m is the measured angular velocity about the perpendicular axis. These measured quantities are obtained by using an IMU. The same topology is valid for both pitch and yaw channel autopilots, by appropriate sign changes in the feedback signals (Mracek and Ridgely, 2005).

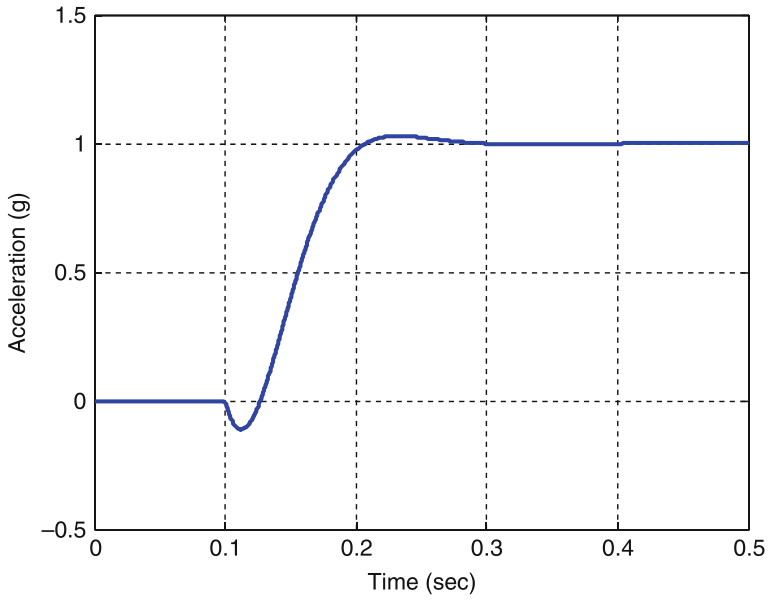

Figure 4.3: Acceleration response when a step input is applied to the classical three loop autopilot design (Mracek and Ridgely, 2005).

It is worth noticing that there is a integrator in the topology shown in figure 4.2. This prevents an infinite command rate as the system sees a step command on the input. However, this causes the system to be non-minimum phase, resulting in the missile moving in the wrong direction before moving in the commanded direction, as seen from the step response in figure 4.3. This problem may be handled using different approaches than the classical designs, e.q. an Model Predictive Control approach (Sefastsson, 2016). This problem will not be present when implementing the course-controlled autopilot presented in this thesis.

By using different feedback typologies, the open-loop properties will be different. This means that the different three-loop designs will have varied robustness properties, despite identical closed loop responses.

4.2 Course-commanded lateral autopilot

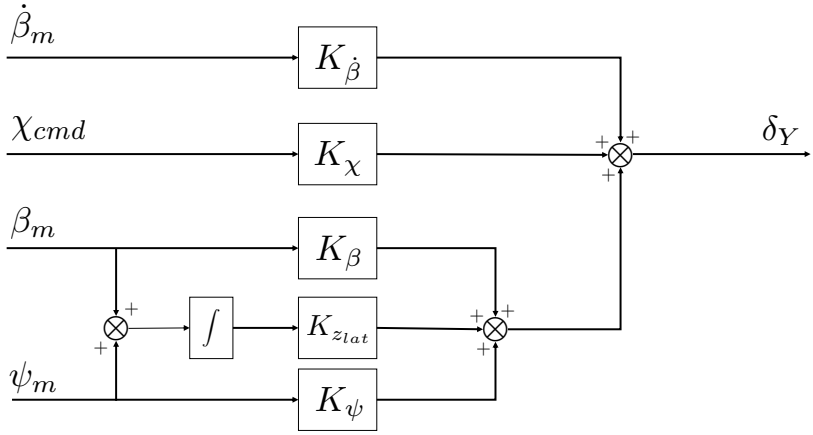

Figure 4.4: Course commanded autopilot for lateral control.

A reliable autopilot design with appropriate command structures are crucial to achieve the overall desired performance of the missile. This means that different autopilot commands may be necessary during different phases, e.g. during launch, mid-course and when the missile is closing up upon the target (Çimen, 2011). For agile turns and during vertical launches, angle of attack and sideslip angle commands may be preferred, while flight-path-angle and course is sometimes used during the launch phase.

For the autopilot design investigated in this report, course and flight-path-angle command inputs are applied. In contradiction to the three-loop autopilot, the course and flight-path-angle can now be controlled directly, without any additional outer-loop controller. This is illustrated in Figure 4.4, where χ_{cmd} denotes the commanded course, while β_m and ψ_m denotes measured quantities. Note that the optimal feedback control does not involve feedback from the measured angular velocity r_m, even though it is designed as a full-state-feedback system.

Furthermore, an additional feedback from $\dot{\beta}_m$ and, for the pitch channel $\dot{\alpha}_m$, grants an additional way to alter the robustness properties of the system. By introducing the additional feedback from $\dot{\beta}_m$, the course controlled design is able to adjust these properties through a tuneable parameter. This gives a robustness-tuning flexibility not present in the three-loop autopilot.

As there is no pure integrator in this autopilot design, the minimum-phase problem will not be an issue. When it comes to handling steps on the reference inputs, an additional reference model will be used to introduce saturation on the command rate.

The LOS steering law makes it easy to produce the desired course and flight-path angle given a LOS vector with the desired look-ahead distance (Fossen, 2011). Since the autopilot controls the course directly, the parameters needed, and consequently the complexity of the autopilot, may be reduced compared to the traditional three-loop autopilot.

When deriving the autopilot for the longitudinal dynamics, the linearized expression for the yaw motion (3.15) is considered. The purpose of the autopilot is to control the course χ, and it is therefore necessary to augment the model with another state. By introducing ψ as a third state, the model becomes

$$\begin{bmatrix} \dot{\beta} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -2.11 & -1 & 0 \\ 423.97 & -1.44 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \beta \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} 0.39 \\ 719.75 \\ 0 \end{bmatrix} \delta_Y \tag{4.2}$$

Since the relationship between the state vector and the course χ is

$$\chi = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta \\ r \\ \psi \end{bmatrix} \tag{4.3a}$$

The C matrix related to the state-space model is simply written as

$$\boldsymbol{C} = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \tag{4.3b}$$

When designing autopilots, the robustness properties of the system will be related to the performance of the system. The systems robustness properties will generally be improved by the expense of the performance of the system. It is desirable to be able to adjust these properties by design, and therefore a new feedback term from $\dot{\beta}$ to χ_d with a gain $K_{\dot{\beta}}$ is introduced. This will represent a tuneable parameter in the state space representation of the linearized system.

The augmented input to the system is formulated as

$$\delta_Y = \delta'_Y + K_{\dot{\beta}} \dot{\beta} \tag{4.4}$$

Such that (4.2) becomes

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\beta} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -2.11 & -1 & 0 \\ 423.97 & -1.44 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \beta \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} 0.39 \\ 719.75 \\ 0 \end{bmatrix} \delta'_Y + \begin{bmatrix} 0.39\, K_{\dot{\beta}} \\ 719.75\, K_{\dot{\beta}} \\ 0 \end{bmatrix} \dot{\beta} \tag{4.5a}$$

The new feedback term is subtracted from the identity matrix on the left side

$$\begin{bmatrix} 1 - 0.39\, K_{\dot{\beta}} & 0 & 0 \\ -719.75\, K_{\dot{\beta}} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\beta} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -2.11 & -1 & 0 \\ 423.97 & -1.44 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \beta \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} 0.39 \\ 719.75 \\ 0 \end{bmatrix} \delta'_Y \tag{4.5b}$$

The left hand matrix is inversed

$$\begin{bmatrix}\dot{\beta}\\ \dot{r}\\ \dot{\psi}\end{bmatrix} = \underbrace{\begin{bmatrix}1-0.39\,K_{\dot{\beta}} & 0 & 0\\ -719.75\,K_{\dot{\beta}} & 1 & 0\\ 0 & 0 & 1\end{bmatrix}^{-1}\begin{bmatrix}-2.11 & -1 & 0\\ 423.97 & -1.44 & 0\\ 0 & 1 & 0\end{bmatrix}}_{\mathbf{A}}\begin{bmatrix}\beta\\ r\\ \psi\end{bmatrix} \tag{4.5c}$$

$$+\underbrace{\begin{bmatrix}1-0.39\,K_{\dot{\beta}} & 0 & 0\\ -719.75\,K_{\dot{\beta}} & 1 & 0\\ 0 & 0 & 1\end{bmatrix}^{-1}\begin{bmatrix}0.39\\ 719.75\\ 0\end{bmatrix}}_{\mathbf{B}}\delta'_Y \tag{4.5d}$$

This can be summarized by writing the system in state-space form, such that it becomes

$$\dot{x} = \boldsymbol{A}x + \boldsymbol{B}u \tag{4.6a}$$

$$y = \boldsymbol{C}x + Du \tag{4.6b}$$

where

$$x = \begin{bmatrix}\beta\\ r\\ \psi\end{bmatrix} \quad u = \delta_Y \quad y = \chi \tag{4.6c}$$

$$\boldsymbol{C} = \begin{bmatrix}1 & 0 & 1\end{bmatrix} \quad D = 0 \tag{4.6d}$$

While $\boldsymbol{A}$ and $\boldsymbol{B}$ are given in (4.5c).

By investigating the bode plot in figure 4.5, it is clear that the parameter $K_{\dot{\beta}}$ is reducing the bandwidth of the system, making it slower. In many cases, is it desirable to achieve the highest bandwidth as possible, as this makes the reaction time and performance of the system better (Balchen et al., 2004). For a missile, the response time and performance is of course important, but without a robust system, the missile will, at worst, not be working at all. As seen in Figure 4.6, feedback from $\dot{\beta}$ gives damping to the system, removing oscillations from the step reponse.

From figure 4.5, it is worth noticing that the feedback term $K_{\dot{\beta}}$ prevents the phase from dropping far below 180 degrees. This is an important robustness property, as gain crossover frequency close to the phase crossover frequency can make the system closed-loop unstable.

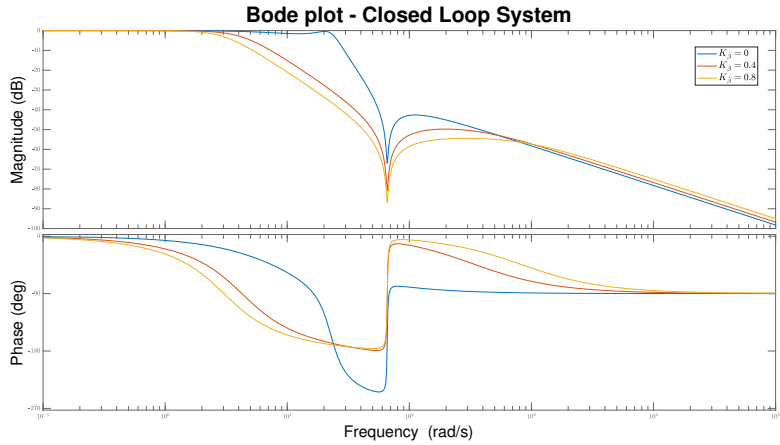

Figure 4.5: Bode plot of the closed-loop lateral autopilot for different values of $K_{\dot{\beta}}$

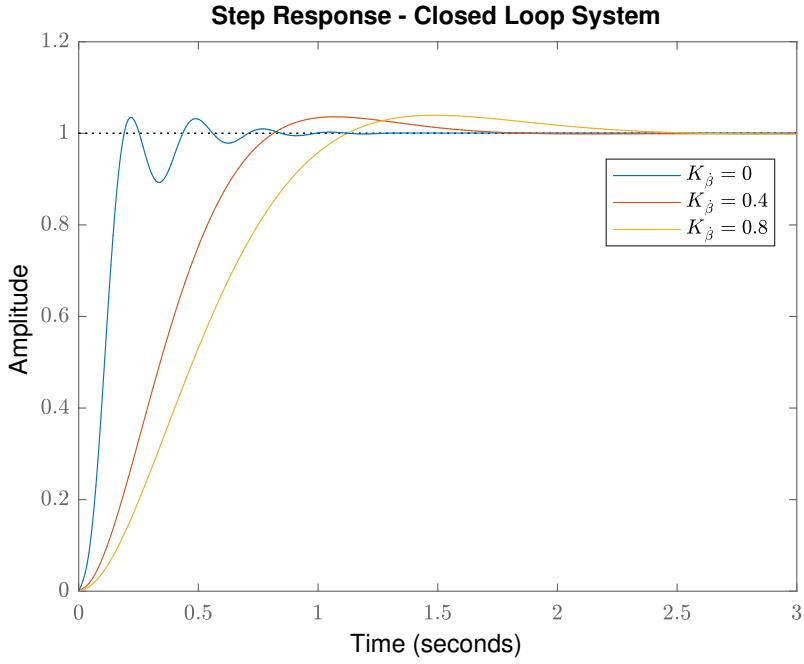

Figure 4.6: Step Response of the closed-loop lateral autopilot for different values of $K_{\dot{\beta}}$

In Table 4.1, different characteristics of the system is given for different values of $K_{\dot{\beta}}$

Table 4.1: Stability characteristics for different values of $K_{\dot{\beta}}$

$K_{\dot{\beta}}$	Phase margin	Gain margin	Bandwidth	Closed loop stable?
-0.2	NaN	NaN	78.1	No
0.0	-0.3	0	24.37	No / Marginal
0.2	45.2	33.7	5.68	yes
0.4	33.5	41.3	4.15	yes
0.6	27.8	46.9	3.42	yes
0.8	24.3	52.5	2.98	yes
1.0	21.9	NaN	2.68	yes

As Table 4.1 shows, a robust system is achieved at the expense of the system performance. $0 < K_{\dot{\beta}} \leq 0.4$ is used in further simulations.

4.2.1 Integral action

As the system will experience disturbances such as gravity and wind, the convergence of the LQR controller might leave the system with a small offset compared to the commanded value. The missile is dependent on hitting a target with high precission, such an offset can't be tolerated. Adding an integral term to the controller will remove this undesired offset.

Consider a state-space model on the form

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{4.7}$$

$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} \tag{4.8}$$

The system is now augmented with the new intergral state $\boldsymbol{z}$, where

$$\dot{\boldsymbol{z}} = \boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} \tag{4.9}$$

which is used to extract the integral states from the state vector. The system can now be rewritten as a standard LQR problem on the form

$$\dot{\boldsymbol{x}}_a = \boldsymbol{A_a}\boldsymbol{x_a} + \boldsymbol{B_a}\boldsymbol{u} \tag{4.10}$$

where $\boldsymbol{x}_a = [\boldsymbol{z}^T \quad \boldsymbol{x}^T]^T$ and

$$\boldsymbol{A}_a = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{C} \\ \boldsymbol{0} & \boldsymbol{A} \end{bmatrix}, \quad \boldsymbol{B}_a = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{B} \end{bmatrix} \tag{4.11}$$

The importance of including integral action in the LQR controller is shown in Figure 4.7. The figure shows how the missile react to commanded flight path angles in the vertical plane. As the gravity is pulling the missile downwards with a constant acceleration, the flight path angle will converge to a value slightly lower than the commanded flight path angle. If this offset is too large, this may result in the missile to intercept a point slightly below the target. This problem is solved when integral action is included, as the flight path angle now converges towards the commanded flight path angle without any offset.

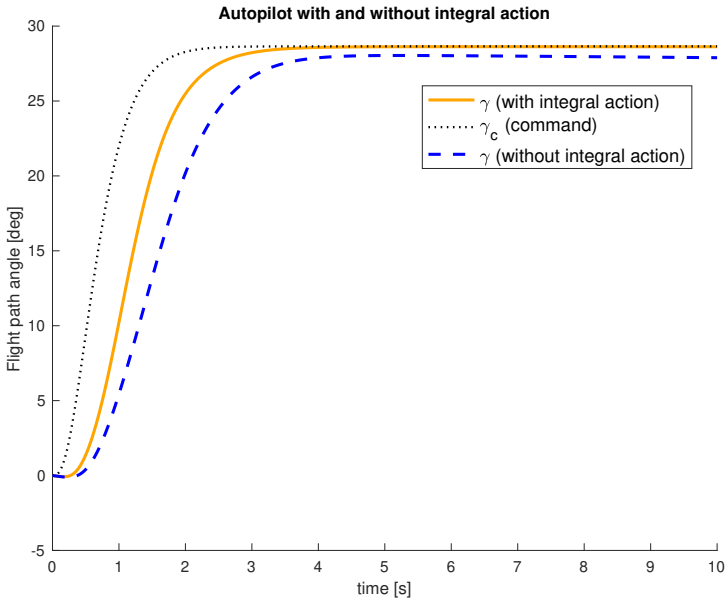

Figure 4.7: Autopilot performance with and without intergral action.

4.2.2 LQR design for trajectory tracking

In order to design the linear optimal control law, the system must be shown to be controllable. The state and input matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ from (4.6) must satisfy the controllability condition to ensure that a control $\boldsymbol{u}(t)$ can drive any arbitrary state $\boldsymbol{x}(t_0)$ to another arbitrary state $\boldsymbol{x}(t_1)$ for $t_1 > t_0$. The condition requires the controllability matrix $\mathcal{C}$ to have full rank (Fossen, 2011):

$$\mathcal{C} = [\boldsymbol{B} \mid \boldsymbol{AB} \mid \ldots \mid (\boldsymbol{A})^{n-1}\boldsymbol{B}] \tag{4.12}$$

For $K_{\dot{\beta}} = 0.4$, the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ from (4.6) becomes

$$\boldsymbol{A} = \begin{bmatrix} -2.50 & -1.18 & 0 \\ 295.13 & -342.16 & 0 \\ 0 & 1.0 & 0 \end{bmatrix} \quad \boldsymbol{B} = \begin{bmatrix} 0 \\ 719.75 \\ 0 \end{bmatrix} \tag{4.13}$$

Where $\boldsymbol{x} = [\beta \quad r \quad \psi]^T$

By adding integral action, the state-space model can be written on the form as in (4.10), where

$$\boldsymbol{A}_{lat} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & -2.50 & -1.18 & 0 \\ 0 & 295.13 & -342.16 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \boldsymbol{B}_{lat} = \begin{bmatrix} 0 \\ 0 \\ 719.75 \\ 0 \end{bmatrix} \tag{4.14}$$

where the augmented state vector is

$$\boldsymbol{x}_{lat} = [z_{lat} \quad \beta \quad r \quad \psi]^T \tag{4.15}$$

where

$$\dot{z}_{lat} = \boldsymbol{C}\,\boldsymbol{x} = \chi \tag{4.16}$$

$\mathcal{C} = \left[\boldsymbol{B}_{lat} \mid \boldsymbol{A}_{lat}\boldsymbol{B}_{lat} \mid \boldsymbol{A}_{lat}^2\boldsymbol{B}_{lat} \mid \boldsymbol{A}_{lat}^3\boldsymbol{B}_{lat}\right]$ can easily be shown to have full rank using the MATLAB command rank(ctrb(A, B)), thus proving the system to be controllable.

Optimal control theory involves finding the optimal control law under certain predefined criteria. For a LQR with tracking reference $\boldsymbol{x}_d = 0$, the optimal controller is found by minimizing the quadratic cost function (Fossen, 2011)

$$\begin{aligned} J &= \min_u \left\{ \frac{1}{2} \int_0^T \left(\boldsymbol{y}^\top \boldsymbol{Q} \boldsymbol{y} + \boldsymbol{u}^\top \boldsymbol{R} u \right) \mathrm{d}t \right. \\ &\left. = \frac{1}{2} \int_0^T \left(\boldsymbol{x}^\top \boldsymbol{C}^\top \boldsymbol{Q} \boldsymbol{C} \boldsymbol{x} + \boldsymbol{u}^\top \boldsymbol{R} u \right) \mathrm{d}t \right\} \end{aligned} \tag{4.17}$$

where $\boldsymbol{R} = \boldsymbol{R}^T > 0$ and $\boldsymbol{Q} = \boldsymbol{Q}^T > 0$ are weighting matrices. The steady-state solution to this problem is

$$u = \underbrace{-\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P}_\infty}_{\boldsymbol{G}} x \tag{4.18}$$

Where $\boldsymbol{P_\infty}$ is found by solving the algebraic Ricatti equation

$$\boldsymbol{P}_\infty \boldsymbol{A} + \boldsymbol{A}^\top \boldsymbol{P}_\infty - \boldsymbol{P}_\infty \boldsymbol{B} \boldsymbol{R}^{-1} \boldsymbol{B}^\top \boldsymbol{P}_\infty + \boldsymbol{C}^T \boldsymbol{Q} \boldsymbol{C} = 0 \tag{4.19}$$

For a time-varying reference signal, the LQ trajectory-tracking problem can be transformed into an LQR problem by formulating the states as error states:

$$\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{x}_d \tag{4.20}$$

For a linear time-invariant system, the approximation of the steady-state solution is found by solving

$$J = \min_{u} \left\{ \frac{1}{2} \int_0^T \left(\boldsymbol{e}^\top \boldsymbol{Q} \boldsymbol{e} + \boldsymbol{u}^\top \boldsymbol{R} u \right) \mathrm{d}t \right\} \tag{4.21}$$

Where $T \to \infty$. By solving (4.19), it is shown by Fossen (2011) that the optimal steady-state control law can be written as

$$\boldsymbol{u} = \boldsymbol{G}_1 \boldsymbol{x} + \boldsymbol{G}_2 \boldsymbol{y}_d + \boldsymbol{G}_3 \boldsymbol{w} \tag{4.22}$$

where $\boldsymbol{y}_d$ is the reference feedforward and $\boldsymbol{w}$ is a constant disturbance that are assumed measured, and

$$\boldsymbol{G}_1 = -\boldsymbol{R}^{-1} \boldsymbol{B}^T \boldsymbol{P}_\infty \tag{4.23}$$

$$\boldsymbol{G}_2 = -\boldsymbol{R}^{-1} \boldsymbol{B}^T (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{G}_1)^{-T} \boldsymbol{C}^T \boldsymbol{Q} \tag{4.24}$$

$$\boldsymbol{G}_3 = \boldsymbol{R}^{-1} \boldsymbol{B}^T (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{G}_1)^{-T} \boldsymbol{P}_\infty \boldsymbol{E} \tag{4.25}$$

The matrices $\boldsymbol{G}_{1,lat}$, $\boldsymbol{G}_{2,lat}$ and $\boldsymbol{G}_{3,lat}$ are calculated using the lqtracker.m function in the MSS toolbox (Fossen, 2011):

$$\boldsymbol{G}_{1,lat} = \begin{pmatrix} K_{z_{lat}} \\ K_\beta \\ K_r \\ K_\psi \end{pmatrix} = \begin{pmatrix} -0.71 \\ -0.79 \\ 0 \\ -1.16 \end{pmatrix} \tag{4.26}$$

$$\boldsymbol{G}_{2,lat} = \begin{pmatrix} K_\chi \\ 0 \end{pmatrix} = \begin{pmatrix} 0.71 \\ 0 \end{pmatrix} \tag{4.27}$$

$$\boldsymbol{G}_{3,lat} = 0 \tag{4.28}$$

4.3 Flight-path angle commanded longitudinal autopilot

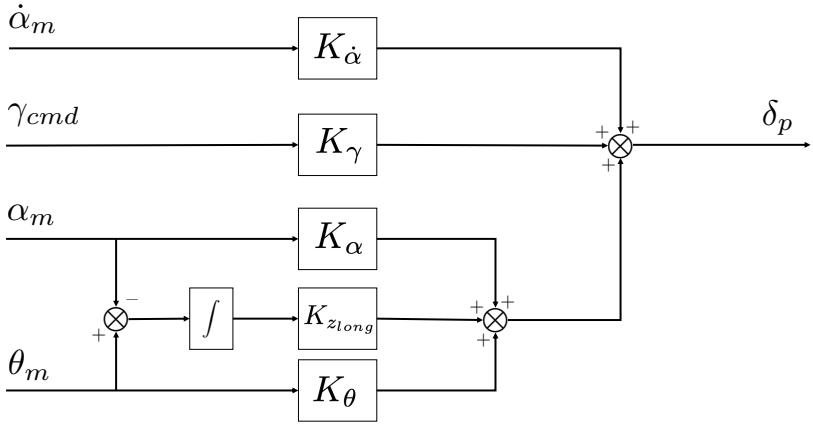

Figure 4.8: Flight-path angle commanded autopilot for longitudinal control.

In this section the autopilot for longitudinal control is derived. The topology is much alike the one for the lateral control, as can be seen in figure 4.8.

When deriving the autopilot for the longitudinal dynamics, the linearized expression for the yaw motion (3.18) is considered. The purpose of the autopilot is to control the flight path angle γ, and it is therefore necessary to augment the model with another state. By introducing θ as a third state, the model becomes

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.07 & 1 & 0 \\ -423.79 & -1.44 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} 0.39 \\ -719.75 \\ 0 \end{bmatrix} \delta_P \tag{4.29}$$

Since the relationship between the state vector and the flight path angle γ is

$$\gamma = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix} \tag{4.30}$$

The C matrix related to the longitudinal state space model is written as

$$\boldsymbol{C} = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \tag{4.31}$$

Like for the lateral autopilot, it is desirable to introduce a feedback term to improve the performance of the system.

The augmented input to the system is formulated as

$$\delta_P = \delta'_P + K_{\dot{\alpha}}\dot{\alpha} \tag{4.32}$$

Such that (4.29) becomes

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.07 & 1 & 0 \\ -423.79 & -1.44 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix} \tag{4.33a}$$

$$+ \begin{bmatrix} -0.39 \\ -719.75 \\ 0 \end{bmatrix} \delta'_P + \begin{bmatrix} -0.39\,K_{\dot{\alpha}} \\ -719.75\,K_{\dot{\alpha}} \\ 0 \end{bmatrix} \dot{\alpha} \tag{4.33b}$$

The new feedback term is subtracted from the identity matrix on the left side

$$\begin{bmatrix} 1 + 0.39\,K_{\dot{\alpha}} & 0 & 0 \\ 719.75\,K_{\dot{\alpha}} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.07 & 1 & 0 \\ -423.79 & -1.44 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} -0.39 \\ -719.75 \\ 0 \end{bmatrix} \delta'_P \tag{4.33c}$$

The left hand matrix is inversed in the same way as for the lateral system:

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 + 0.39\,K_{\dot{\alpha}} & 0 & 0 \\ 719.75\,K_{\dot{\alpha}} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} -0.07 & 1 & 0 \\ -423.79 & -1.44 & 0 \\ 0 & 1 & 0 \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix} \tag{4.33d}$$

$$+ \underbrace{\begin{bmatrix} 1 + 0.39\,K_{\dot{\alpha}} & 0 & 0 \\ 719.75\,K_{\dot{\alpha}} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} -0.39 \\ -719.75 \\ 0 \end{bmatrix}}_{\mathbf{B}} \delta'_P \tag{4.33e}$$

This can be summarized by writing the system in the same form as (4.6), such that

$$\boldsymbol{x} = \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix} \quad u = \delta_P \quad y = \gamma \tag{4.34a}$$

$$\boldsymbol{C} = [-1 \quad 0 \quad 1] \quad D = 0 \tag{4.34b}$$

While $\boldsymbol{A}$ and $\boldsymbol{B}$ are given in (4.33).

4.3.1 Integral action

By adding integral action in the same manner as for the lateral autopilot, the augmented system becomes

$$\boldsymbol{A}_{long} = \begin{bmatrix} 0 & -1 & 0 & 1 \\ 0 & -1.85 & 0.87 & 0 \\ 0 & 107.15 & -250.69 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \boldsymbol{B}_{long} = \begin{bmatrix} 0 \\ 0 \\ -622.54 \\ 0 \end{bmatrix} \tag{4.35}$$

where the state vector is

$$\boldsymbol{x}_{long} = \begin{bmatrix} z_{long} & \alpha & q & \theta \end{bmatrix}^T \tag{4.36}$$

where

$$\dot{z}_{long} = \boldsymbol{C}\,\boldsymbol{x} = \gamma \tag{4.37}$$

4.3.2 LQR control

As the system can be shown to have full rank, an LQR control law can be implemented in the same way as for the lateral dynamics. Again, by using the lqtracker.m function in the MSS toolbox (Fossen, 2011) to calculate the matrices $\boldsymbol{G}_{1,long}$, $\boldsymbol{G}_{2,long}$ and $\boldsymbol{G}_{3,long}$:

$$\boldsymbol{G}_{1,long} = \begin{pmatrix} K_{z_{long}} \\ K_\alpha \\ K_q \\ K_\theta \end{pmatrix} = \begin{pmatrix} 0.71 \\ -0.40 \\ 0 \\ -1.39 \end{pmatrix} \tag{4.38}$$

$$\boldsymbol{G}_{2,long} = \begin{pmatrix} K_\gamma \\ 0 \end{pmatrix} = \begin{pmatrix} -0.71 \\ 0 \end{pmatrix} \tag{4.39}$$

$$\boldsymbol{G}_{3,long} = 0 \tag{4.40}$$

4.4 Calculation of $\dot{\alpha}$ and $\dot{\beta}$

As shown in the previous sections, both $\dot{\alpha}$ and $\dot{\beta}$ are used as feedback terms in the autopilot designs. As these parameters are not obtained directly as measurements, these must be calculated based on already available information. From (2.15), we have that

$$\alpha = \tan^{-1}\left(\frac{w}{u}\right) \tag{4.41}$$

such that the derivative can be expressed as

$$\dot{\alpha} = \frac{1}{\sqrt{1 + \left(\frac{w}{u}\right)^2}} \frac{\dot{w}u - \dot{u}w}{\sqrt{u^2 + w^2}} \tag{4.42}$$

From (2.15), we have that

$$\beta = \sin^{-1}\left(\frac{v}{U}\right) \tag{4.43}$$

such that the derivative can be expressed as

$$\dot{\beta} = \frac{1}{\sqrt{1+\left(\frac{v}{U}\right)^2}} \frac{\dot{v}U - \dot{U}v}{\sqrt{v^2+U^2}} \tag{4.44}$$

$$\tag{4.45}$$

From (5.12) $\dot{u}$, $\dot{v}$ and $\dot{w}$ can be expressed in terms of accelerometer and gyro measurements from the IMU. By assuming that the biases are estimated and compensated for, we have that

$$\dot{u} = a_x - qw + rv - g\sin\theta \tag{4.46}$$
$$\dot{v} = a_y - ru + pw + g\cos\theta\sin\phi \tag{4.47}$$
$$\dot{w} = a_z - pv + qu + g\cos\theta\cos\phi \tag{4.48}$$

In case U is not constant, the expression for $\dot{U}$ is

$$\dot{U} = \frac{1}{U}\left(2v\dot{v} + 2u\dot{u} + 2w\dot{w}\right) \tag{4.49}$$

The results in expression for $\dot{\alpha}$ and $\dot{\beta}$ where all the involved variables are either estimated or measured.

Chapter 5

Navigation System

5.1 Introduction

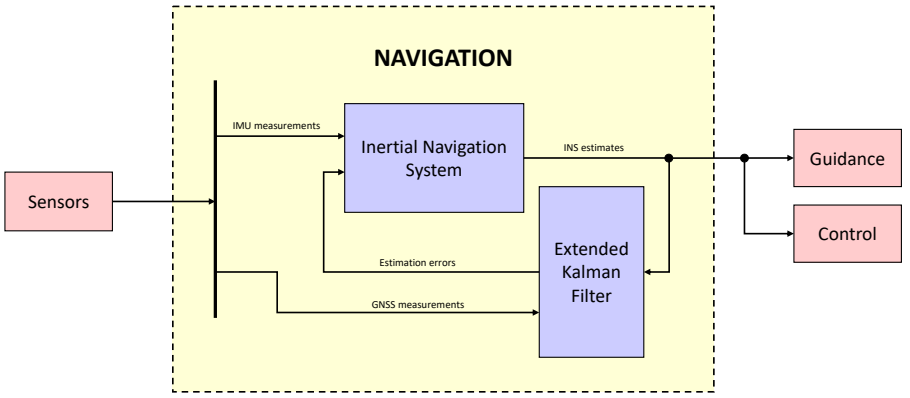

Figure 5.1: Navigation system

The navigation system is reponsible of obtaining accurate state estimations based on sensor measurements. In this chapter, an error-state Kalman Filter will be used to calculate error corrections in the navigation equations, such that the Guidance and Control systems receives more precise state estimates. Since the INS calculates PVA based on integration of sensor measurements (see section 5.5), errors will occur as the measurements are only valid for a short time. This is caused by drift from noisy measurements and biases. Figure 5.1 shows how the navigation system for the interceptor state estimations are designed.

Note that the frequency of IMU and GNSS measurement updates typically are not the same. As an IMU operates at a much higher frequency than what can be expected from a GNSS receiver, the INS system is able to calculate estimates at a much higher frequency than what is possible with a direct Kalman Filter design, where the sampling rate are saturated based on the GNSS update frequency.

Two different methods of computing the measurement equations for the MEKF will be discussed. The first one involves using an additional Kalman Filter for calculating the specific-force reference, while the other one relies on estimation using sensor measurements directly.

There will also be derived a Kalman Filter for keeping track of the relative PVA between the interceptor and target. This is explained in section 5.8.

5.2 The Indirect Extended Kalman Filter process

There are two classical ways to implement a Kalman filter. The two different approaches are known as the $direct$ and $indirect$ Kalman filter. Both designs are built upon a discrete-time state-space model (Bryne and Fossen, 2016):

$$\boldsymbol{x}[k+1] = \boldsymbol{A}_d[k]\boldsymbol{x}[k] + \boldsymbol{B}_d\boldsymbol{u}[k] + \boldsymbol{E}_d\boldsymbol{w}[k] \tag{5.1a}$$

$$\boldsymbol{y}[k] = \boldsymbol{C}[k]\boldsymbol{x}[k] + \boldsymbol{D}_d[k]\boldsymbol{u}[k] + \boldsymbol{\epsilon}[k] \tag{5.1b}$$

where $\boldsymbol{A}_d$, $\boldsymbol{B}_d$, $\boldsymbol{E}_d$ describes the the process model, $\boldsymbol{C}_d$, $\boldsymbol{D}_d$ describes the measurement model, $\boldsymbol{E}_d$ and $\boldsymbol{\epsilon}$ are the process and measurement noise vectors.

For the indirect Kalman filter, the states are formulated as error-states. The filter can then be used to calculate the errors in terms of state errors and bias errors. This stands out from the direct approach of the Kalman filter, as the states of the filter is the error dynamics instead of only the regular states. The error state-space model is formulated as:

$$\delta\dot{\boldsymbol{x}}[k+1] = \boldsymbol{A}_d[k]\delta\boldsymbol{x}[k] + \boldsymbol{E}_d\delta\boldsymbol{w}[k] \tag{5.2a}$$

$$\delta\boldsymbol{y}[k] = \boldsymbol{H}_d[k]\delta\boldsymbol{x}[k] \tag{5.2b}$$

where $\delta(.)$ denotes the error state.

When dealing with an indirect/error-state Kalman filter, we differentiate between the true, nominal and error-state of the system, where the true state is the composition between the nominal state and the error-state (Solà, 2017). The IMU measurement is considered as a large signal, while the error state is a small signal. Nominal state values are obtained by integrating the high frequency IMU data. These dead-reckoning positioning states do not take noise and biases into account, and as a consequence of this, they will drift. In parallel to the integration of the nominal state, the indirect Kalman filter is used to provide corrections for the nominal state. When new aiding measurements arrive, the error-state's mean are injected into the nominal state, and afterwards reset to zero. The correction steps

are shown in (5.3). The $\oplus$ symbols represents the appropriate compositions, either sums or quaternion products. The co-variance matrix is then updated according to this reset. Figure 5.2 (Bryne and Fossen, 2016) shows how the indirect Kalman filter behaves:

$$\hat{\boldsymbol{x}}_{ins}[k] \leftarrow \hat{\boldsymbol{x}}_{ins}[k] \oplus \delta\hat{\boldsymbol{x}}^{+}[k] \tag{5.3a}$$

$$\delta\hat{\boldsymbol{x}}^{+}[k] \leftarrow \boldsymbol{0} \tag{5.3b}$$

$$k \leftarrow k+1 \tag{5.3c}$$

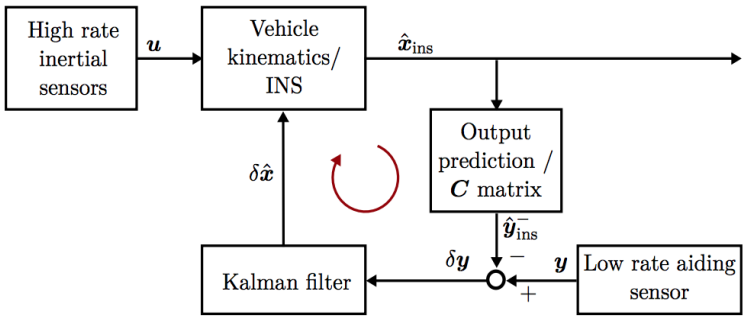

Figure 5.2: Indirect (feedback) Kalman filter for INS.

The estimation of $\delta\hat{\boldsymbol{x}}^{+}$ is done every time step according to

$$\delta\hat{\boldsymbol{x}}^{+}[k] = \delta\hat{\boldsymbol{x}}^{-}[k] + \boldsymbol{K}[k](\delta y[k] - \boldsymbol{h}(\delta\hat{\boldsymbol{x}}^{-}[k])) \tag{5.4}$$

while the Kalman gain and co-variance updates are calculated as

$$\boldsymbol{K}[k] = \hat{\boldsymbol{P}}^{-}[k]\boldsymbol{H}_d^{T}[k](\boldsymbol{H}_d[k]\hat{\boldsymbol{P}}^{-}[k]\boldsymbol{H}_d^{T}[k] + \boldsymbol{R}_d[k])^{-1} \tag{5.5}$$

$$\hat{\boldsymbol{P}}^{+}[k] = (\boldsymbol{I} - \boldsymbol{K}[k]\boldsymbol{H}_d[k])\hat{\boldsymbol{P}}^{-}[k](\boldsymbol{I} - \boldsymbol{K}[k]\boldsymbol{H}_d[k])^{T} + \boldsymbol{K}[k]\boldsymbol{R}[k]\boldsymbol{K}^{T}[k] \tag{5.6}$$

$$\hat{\boldsymbol{P}}^{-}[k+1] = \boldsymbol{H}_d[k]\hat{\boldsymbol{P}}^{+}[k]\boldsymbol{H}_d^{+}[k] + \boldsymbol{E}_d[k]\boldsymbol{Q}_d[k]\boldsymbol{E}_d[k]^{T} \tag{5.7}$$

where

$\boldsymbol{K}$	is the Kalman gain
$\delta\hat{\boldsymbol{x}}^{-}, \delta\hat{\boldsymbol{x}}^{+}$	are the priori and apostriori error measurements
$\boldsymbol{Q}_d, \boldsymbol{R}_d$	is the co-variance matrices for process and measurement noise, and
$\hat{\boldsymbol{P}}^{-}, \hat{\boldsymbol{P}}^{+}$	are the apriori and apostriori co-variance matrix estimates.

Since the error $\delta\hat{\boldsymbol{x}}^{+}[k]$ is reset before the arrival of a new measurement, the last term of (5.4) becomes redundant, and can be simplified to

$$\delta\hat{\boldsymbol{x}}^{+}[k] = \delta\hat{\boldsymbol{x}}^{-}[k] + \boldsymbol{K}[k]\delta y[k] \tag{5.8}$$

5.3 Sensors

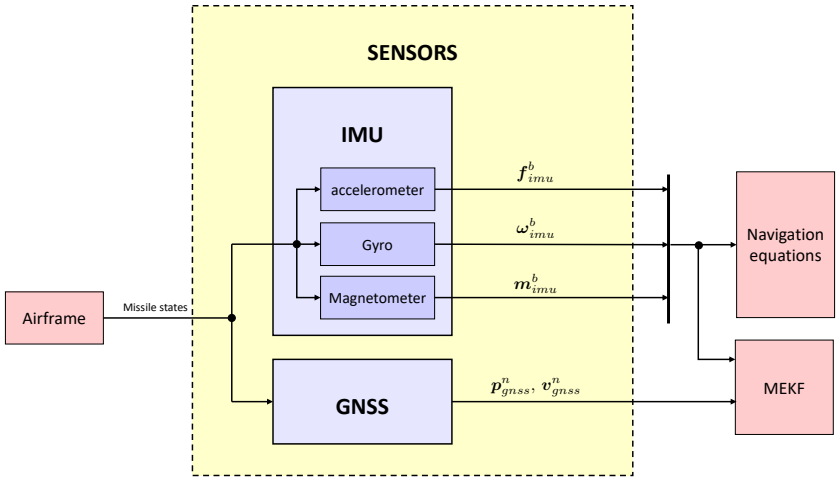

Figure 5.3: Sensors in relation to the rest of the control system.

The states of a rigid body can be estimated using different kind of sensors, often by using accelerometer and gyroscope readings. The IMU contains a cluster of three accelerometers, three gyroscopes and three magnetometers. The IMU readings will provide estimations for the acceleration and angular rates respectively, which by integration are used to obtain the PVA of the rigid body. This leads to dead-reckoning position and attitude estimates, and may be inaccurate due to noise and bias contamination of the sensor readings. To be able to estimate the biases, aiding techniques based on GNSS measurements of position and velocity, magnetometer readings and relative force estimation is used.

5.3.1 Rate gyro measurement

Ring laser gyros and Fiber Optic Gyros (FOG) has been used for some time, and are expected to be the standard for high accuracy strap-down inertial systems (Fossen, 2011). Traditionally, Micro Electrical Mechanical Systems (MEMS) has been expected to be used for low and medium cost applications. However, this assumption has been challenged by new MEMS systems as the UTC Aerospace Systems TITAN® MEMS IMU, which offers performance to rival that of a FOG (UTC Aerospace Systems, 2017). The TITAN® MEMS IMU provides the characteristics for the sensor parameters used in simulations.

By assuming that the sensors are mounted in the body-frame origin with small misalignment error, the gyro output can be expressed according to (Fossen, 2011) as

$$\boldsymbol{\omega}^b_{imu} \approx \boldsymbol{\omega}^b_{b/n} + \boldsymbol{b}^b_{ars} + \boldsymbol{\omega}^b_{ars} \tag{5.9}$$

where $\boldsymbol{b}^b_{ars} = [b^b_{ars,\phi} \quad b^b_{ars,\theta} \quad b^b_{ars,\psi}]^T$ is the unknown bias modeled as a Wiener process

$$\dot{\boldsymbol{b}}^b_{ars} = \boldsymbol{\omega}_{b,ars}$$

and $\boldsymbol{\omega}^b_{ars}$ is Gaussian white noise.

5.3.2 Accelerometer measurement

By employing a proof mass held in place by a suspension, the relative displacement of the mass can be used to measure the acceleration of the sensor. The displacement of the acceleration transducers can easily be converted to acceleration by using a simple force balance analysis:

$$m\ddot{x} + k\dot{x} = ky(t) \tag{5.10}$$

where x is the inertial position of the proof mass, while $y(t)$ is the inertial position of the sensor housing.

Accelerometers measure the specific force in the body frame of the vehicle. The measured acceleration is therefore the total acceleration of the casing, minus the gravity pulling the casing towards the center of the earth. The mathematical expression of accelerometer measurements can be modeled as (Beard and McLain, 2013) (Fossen, 2011):

$$\boldsymbol{f}^b_{imu} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} = \frac{d\boldsymbol{v}}{d_{tb}} + \omega^b_{b/i} \times \boldsymbol{v} - \mathbf{R}^b_n \begin{pmatrix} o \\ o \\ g \end{pmatrix} + \boldsymbol{b}^b_{acc} + \boldsymbol{\omega}^b_{acc} \tag{5.11}$$

Where $\boldsymbol{\omega}_{b/i}$ is is the angular rotation in body-frame with respect to the inertial frame.

$\boldsymbol{b}^b_{acc} = [b^b_{acc,x} \quad b^b_{acc,y} \quad b^b_{acc,z}]^T$ is the unknown bias modeled as a Wiener process

$$\dot{\boldsymbol{b}}^b_{acc} = \boldsymbol{\omega}_{b,acc}$$

and $\boldsymbol{\omega}^b_{acc}$ is Gaussian white noise.

For local navigation, the NED frame can be assumed inertial, which gives

$$\boldsymbol{\omega}^b_{b/i} \approx \boldsymbol{\omega}^b_{b/n}$$

$$\begin{aligned} a_x &= \dot{u} + qw - rv + g\sin\theta + b_{acc,x} + \omega_{acc,x} \\ a_y &= \dot{v} + ru - pw - g\cos\theta\sin\phi + b_{acc,y} + \omega_{acc,y} \\ a_z &= \dot{w} + pv - qu - g\cos\theta\cos\phi + b_{acc,z} + \omega_{acc,z} \end{aligned} \tag{5.12}$$

From (5.12) it is seen that the accelerometer measures linear acceleration, Coriolis acceleration and gravitational acceleration.

5.3.3 Magnetometer measurement

The last part of the IMU measurements are obtained from a cluster of three magnetometers. The magnetic field of the earth can be compared to a simple bar magnet. Originating at a point near the South Pole and stretching to a point near the North Pole, the magnetic field is varying in both strength and direction about the face of the Earth (Fossen, 2011). The magnetic field is different all over the globe. The magnetic field in the horizontal plane is known, and can easily be found by using an online calculator. In Berkeley, CA the magnetic field is approximately[1] $\boldsymbol{m}^n = [22494.35 \quad 5372.67 \quad 42301.72]^T$. By mounting the magnetometers orthogonal and aligned with the body axes, the magnetometer readings can be transformed to the horizontal plane accoriding to (Fossen, 2011)

$$\boldsymbol{m}^{\,b}_{imu} = \boldsymbol{R}^b_n \boldsymbol{m}^n + \boldsymbol{b}^b_{mag} + \boldsymbol{w}^b_{mag} \tag{5.13}$$

where $\boldsymbol{m}^n = [m_N \quad m_E \quad mD]$ represents the magnetometer measurements.

$\boldsymbol{b}^b_{mag} = [b^b_{mag,x} \quad b^b_{mag,y} \quad b^b_{mag,z}]^T$ is the unknown bias modeled as a Wiener process

$$\dot{\boldsymbol{b}}^{\,b}_{mag} = \boldsymbol{\omega}_{b,mag}$$

and $\boldsymbol{\omega}^b_{mag}$ is Gaussian white noise.

5.3.4 Global Navigation Satellite System

The GNSS, uses space satellites to achieve position and navigation measurements, and is widely used in both civil and military applications (Zhang et al., 2017). While the INS provides fast high-precision PVA estimates for a short time, they will, after some time, start drifting because of the sensor bias and noise. The integration of GNSS measurements will provide highly accurate position aiding, preventing the estimations from drifting over time. By using the carrier phase Doppler measurements, the velocity of the receiver may also be calculated with a standard deviation ranging from 0.01 to 0.05 m/s (Beard and McLain, 2013).

The majority of GPS receivers nowadays are updated with a frequency of 1 Hz. For some low speed application, the position and velocity updates can be received with a frequency as low as 0.1 Hz, while for high speed navigation, sampling rates as high as 10 Hz is often necessary (Salih et al., 2013). The Trimble® Serial Embdedded GPS Receiver (SEGR) is a family of Embedded GPS Receivers (EGR) that supports airborn and other high accuracy applications. According to their datasheet (Trimble, 2012), an aiding rate of 1-50 Hz is obtainable.

[1] http://geomag.nrcan.gc.ca/calc/mfcal-en.php

5.4 Attitude model

As the attitude is assumed unknown, it has to be estimated using an attitude estimator. There are different options for how to represent the attitude, and therefore a comparison between Euler angles and the Hamilton quaternion follows. When using Euler angles, the attitude is represented by the three parameters

$$\boldsymbol{\Theta} = [\phi \quad \theta \quad \psi]^T \tag{5.14}$$

The matrix representation of the attitude yields (Fossen, 2011)

$$\boldsymbol{R}_b^n(\boldsymbol{\Theta}_{nb}) =$$

$$\begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\phi s\theta s\psi & -c\psi s + \phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{5.15}$$

It follows that

$$\dot{\psi} = q\frac{\sin\phi}{\cos\theta} + r\frac{\cos\phi}{\cos\theta} \tag{5.16}$$

And it is easy to see that the pitch angle $\theta = 90$ degrees represents a singularity, resulting in only local stability for an observer using Euler angles.

The four-parameter quaternion attitude representation $\boldsymbol{q} = [\eta \quad \epsilon_1 \quad \epsilon_2 \quad \epsilon_3]^T$ does not have any singularities, and can achieve almost-global or semi-global stability (M.Innocenti and D.Fragopoulos, 2004). Quaternions are therefore chosen to represent the attitude of the system.

In extended Kalman filtering problems the error terms may be treated additive, i.e. $\boldsymbol{q} = \hat{\boldsymbol{q}} + \delta\boldsymbol{q}$ (Crassidis et al., 2007). This common approach represents a non singular parameterization of the attitude in the filter state vector. However, adding two unit quaternion together will not produce a new unit quaternion, a problem that often are solved with frequent renormalizations (Maley, 2013). An elegant alternative is to use the quaternion product between the estimated quaternion $\hat{\boldsymbol{q}}$ and the error quaternion $\delta\boldsymbol{q}$ to produce the injection term for the approximation of the true state:

$$\boldsymbol{q} = \hat{\boldsymbol{q}} \otimes \delta\boldsymbol{q} \quad \Leftrightarrow \quad \delta\boldsymbol{q} = \hat{\boldsymbol{q}}^{-1} \otimes \boldsymbol{q} \tag{5.17}$$

$$\boldsymbol{R}_b^n(\boldsymbol{q}) = \boldsymbol{R}_b^n(\hat{\boldsymbol{q}} \otimes \boldsymbol{\delta q}) = \boldsymbol{R}_{\hat{b}}^n(\hat{\boldsymbol{q}})\boldsymbol{R}_{\hat{b}}^{\hat{b}}(\boldsymbol{\delta q}) \tag{5.18}$$

Where $\{\hat{b}\}$ is the estimated body-frame. Another benefit of using the error quaternion multiplication is that the terms needed for parametrization are reduced from four to three, since η easily can be produced by taking $\eta = \sqrt{1 - \epsilon^T\epsilon}$ when $q = [\eta \quad \epsilon_1 \quad \epsilon_2 \quad \epsilon_3]^T$ is a unit quaternion.

The time derivative of the error $\delta \boldsymbol{q}$ is found by differentiating (5.17)

$$\delta \dot{\boldsymbol{q}} = \begin{bmatrix} \delta \dot{\eta} \\ \delta \dot{\boldsymbol{\epsilon}} \end{bmatrix} = \hat{\boldsymbol{q}}^{-1} \otimes \dot{\boldsymbol{q}} \tag{5.19}$$

$$= \frac{1}{2} \hat{\boldsymbol{q}}^{-1} \otimes \boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{w}^b_{b/n} \end{bmatrix} \tag{5.20}$$

$$= \frac{1}{2} \boldsymbol{\delta q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{w}^b_{b/n} \end{bmatrix} \tag{5.21}$$

The vector part $\delta \dot{\boldsymbol{\epsilon}}$ is then written as

$$\delta \dot{\boldsymbol{\epsilon}} = \frac{1}{2} [\boldsymbol{I}_{3x3} \sqrt{1 - \delta \boldsymbol{\epsilon}^T \delta \boldsymbol{\epsilon}} + \boldsymbol{S}(\delta \epsilon)] \boldsymbol{\omega}^b_{b/n} \tag{5.22}$$

5.5 Inertial Nagivation System equations

The vehicle is assumed to be equipped with an IMU consisting of three accelerometers, three angular rate sensors and three magnetometers.

The sensors models are augmented with Gaussian white noise and and time-varying biases such that the IMU measurements can be described as in (5.9) and (5.11):

$$\boldsymbol{f}^b_{imu} = (\boldsymbol{R}^n_b)^T \boldsymbol{f}^b_n + \boldsymbol{b}^b_{acc} + \boldsymbol{w}^b_{acc} \tag{5.23}$$

$$\boldsymbol{\omega}^b_{imu} = \boldsymbol{\omega}^b_{b/n} + \boldsymbol{b}^b_{ars} + \boldsymbol{\omega}^b_{ars} \tag{5.24}$$

Where $\boldsymbol{f}^b_n$ is the true specific force and $\boldsymbol{\omega}^b_{b/n}$ is the true angular rate.

$\boldsymbol{b}^b_*$ is the unknown bias modeled as a Wiener process

$$\dot{\boldsymbol{b}}^b_* = \boldsymbol{\omega}_{b,*}$$

and $\boldsymbol{\omega}_*$ is Gaussian white noise.

Solving the sensor measurements for the true specific force and angular rate gives

$$\boldsymbol{f}^n_b = \boldsymbol{R}^n_b (\boldsymbol{f}^b_{imu} - \boldsymbol{b}^b_{acc} - \boldsymbol{w}^b_{acc}) \tag{5.25a}$$

$$\boldsymbol{\omega}^b_{b/n} = \boldsymbol{\omega}^b_{imu} - \boldsymbol{b}^b_{ars} - \boldsymbol{\omega}^b_{ars} \tag{5.25b}$$

The INS sensor estimates are defined as

$$\boldsymbol{f}^n_{ins} = \boldsymbol{R}^n_{\hat{b}} (\hat{\boldsymbol{q}}) (\boldsymbol{f}^b_{imu} - \boldsymbol{b}^b_{ins,acc}) \tag{5.26a}$$

$$\boldsymbol{\omega}^b_{ins} = \boldsymbol{\omega}^b_{imu} - \boldsymbol{b}^b_{ins,ars} \tag{5.26b}$$

For the inertial frame, the strap-down navigation equations with quaternion representation for the attitude are given by Jay (2008):

$$\dot{\boldsymbol{p}}^{\boldsymbol{n}}{}_{b/n} = \boldsymbol{v}^{\boldsymbol{n}}{}_{b/n} \tag{5.27a}$$

$$\dot{\boldsymbol{v}}^{\boldsymbol{n}}{}_{b/n} = \boldsymbol{f}_b^n + \boldsymbol{g}^n \tag{5.27b}$$

$$\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega}_{b/n}^b \end{bmatrix} \tag{5.27c}$$

By inserting (5.26) into (5.27) we obtain PVA estimates as well as bias estimates. The inertial navigation equations yields

$$\dot{\boldsymbol{p}}^{\boldsymbol{n}}{}_{ins} = \boldsymbol{v}^{\boldsymbol{n}}{}_{ins} \tag{5.28}$$

$$\dot{\boldsymbol{v}}^{\boldsymbol{n}}{}_{ins} = \boldsymbol{R}_b^n(\hat{\boldsymbol{q}})(\boldsymbol{f}_{imu}^b - \boldsymbol{b}_{ins,acc}^b) + \boldsymbol{g}^n \tag{5.29}$$

$$\dot{\boldsymbol{q}}_{ins} = \frac{1}{2}\boldsymbol{q}_{ins} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega}_{imu}^b - \boldsymbol{b}_{ins,ars}^b \end{bmatrix} \tag{5.30}$$

$$\dot{\boldsymbol{b}}_{ins,ars}^b = 0 \tag{5.31}$$

$$\dot{\boldsymbol{b}}_{ins,acc}^b = 0 \tag{5.32}$$

$$\tag{5.33}$$

5.6 Error-state equations

Due to modeling errors, sensor drift and noise, there will be an error propagation between the INS estimates and the true states. The error between these two states has to be estimated and compensated to assure that the system behaves in a satisfactory manner. This is where the EKF explained in section 5.2 will be used.

The error-state equations between the true states and the INS measurements are introduced as

$$\delta \boldsymbol{p}_{b/n}^n = \boldsymbol{p}_{b/n}^n - \boldsymbol{p}_{ins} \tag{5.34}$$

$$\delta \boldsymbol{v}_{b/n}^n = \boldsymbol{v}_{b/n}^n - \boldsymbol{v}_{ins} \tag{5.35}$$

$$\boldsymbol{q} = \boldsymbol{q}_{ins} \otimes \delta \boldsymbol{q} \tag{5.36}$$

$$\delta \boldsymbol{b}_{ars} = \boldsymbol{b}_{ars} - \boldsymbol{b}_{ins,ars} \tag{5.37}$$

$$\delta \boldsymbol{b}_{acc} = \boldsymbol{b}_{acc} - \boldsymbol{b}_{ins,acc} \tag{5.38}$$

As the error state equations includes the quaternion multiplicative term $\otimes$ for quaternion estimation, the name $multiplicative$ extended Kalman filter MEKF is used to describe the filter.

5.6.1 Gibbs vector

There are several ways to parametrize the quaternion, including the use of Euler angles and Hamilton quaternion as explained in section 5.4. For the quaternion error in the MEKF, the Gibbs vector will be used as parametrization. The Gibbs vector is given by Markley (2008) as

$$\boldsymbol{g}_{\text{gibbs}} = \frac{\delta\boldsymbol{\epsilon}}{\delta\eta} \tag{5.39}$$

By scaling the Gibbs vector by the factor 2, the variance will be given in radians squared, which is equivalent to angle errors using a first-order approximation. By defining

$$\frac{\boldsymbol{a}_g}{2} = \frac{\delta\boldsymbol{\epsilon}}{\delta\eta} \tag{5.40}$$

where $\boldsymbol{a}_g$ denotes a rotation such that

$$\frac{\boldsymbol{a}_g}{2} = \boldsymbol{e}\tan\frac{\phi}{2} \tag{5.41}$$

where $\boldsymbol{e}$ is a unit vector and ϕ is an angle of rotation as defined in (2.2). It can be seen in (5.41), that this parametrization ensures that the magnitude of $\boldsymbol{a}_g$ approximates ϕ for small rotations (Markley, 2008).

The imaginary part $\boldsymbol{\epsilon}$ of the quaternion error can be calculated from $\boldsymbol{a}_g$ as follows:

$$\boldsymbol{a}_g = 2\frac{\delta\boldsymbol{\epsilon}}{\delta\eta} \tag{5.42a}$$

$$\boldsymbol{a}_g^2 = 4\frac{\delta\boldsymbol{\epsilon}^2}{\delta\eta^2} = 4\frac{\delta\boldsymbol{\epsilon}^2}{1-\delta\boldsymbol{\epsilon}^2} \tag{5.42b}$$

$$\frac{\boldsymbol{a}_g^2}{4}(1-\delta\boldsymbol{\epsilon}^2) = \delta\boldsymbol{\epsilon}^2 \tag{5.42c}$$

$$\boldsymbol{a}_g^2 - \boldsymbol{a}_g^2\,\delta\boldsymbol{\epsilon}^2 - 4\delta\boldsymbol{\epsilon}^2 = 0 \tag{5.42d}$$

$$\delta\boldsymbol{\epsilon}^2\,[4+\boldsymbol{a}_g^2] = \boldsymbol{a}_g^2 \tag{5.42e}$$

which finally gives

$$\delta\boldsymbol{\epsilon} = \sqrt{\frac{\boldsymbol{a}_g^2}{4+\boldsymbol{a}_g^2}} \tag{5.42f}$$

$$= \frac{\boldsymbol{a}_g}{\sqrt{4+\boldsymbol{a}_g^2}} \tag{5.42g}$$

Now the quaternion error can be expressed in terms of $\boldsymbol{a}_g$ as:

$$\delta\boldsymbol{q}(\boldsymbol{a}_g) = \begin{bmatrix}\delta\eta\\ \delta\boldsymbol{\epsilon}\end{bmatrix} = \frac{1}{\sqrt{4+\boldsymbol{a}_g^2}}\begin{bmatrix}2\\ \boldsymbol{a}_g\end{bmatrix} \tag{5.43}$$

The Kalman filter equations are based on the discrete system explained in (5.2a). Consider the following model:

$$\delta \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{E}\,\boldsymbol{w} \tag{5.44}$$
$$\delta \boldsymbol{y} = h(\delta \boldsymbol{x}) + \boldsymbol{v} \tag{5.45}$$

By defining

$$\delta \boldsymbol{x} = [(\delta \boldsymbol{p}^n)^T \quad (\delta \boldsymbol{v}^n)^T \quad \boldsymbol{a}_g^T \quad (\delta \boldsymbol{b_{ars}})^T \quad (\delta \boldsymbol{b_{acc}})^T]^T \tag{5.46}$$

The non-linear equations describing the system will be derived.

The position error is simply given as

$$\delta \dot{\boldsymbol{p}}^n_{b/n} = \delta \boldsymbol{v}^n_{b/n} \tag{5.47}$$

for the velocity error, the equations given in section 5.5, as well as the relationship given in (5.4) is considered

$$\delta \dot{\boldsymbol{v}}^n_{b/n} = \dot{\boldsymbol{v}}^n_{b/n} - \dot{\boldsymbol{v}}^n_{ins} \tag{5.48}$$
$$= \boldsymbol{R}(\hat{q})\boldsymbol{R}(\boldsymbol{a_g})(\boldsymbol{f}^b_{imu} - \boldsymbol{b}_{ins,acc} - \delta \boldsymbol{b}_{ins,acc} - \boldsymbol{w}_{acc}) + \boldsymbol{g}^n \tag{5.49}$$
$$- \boldsymbol{R}(\hat{q})(\boldsymbol{f}^b_{imu} - \boldsymbol{b}_{ins,acc}) - \boldsymbol{g}^n \tag{5.50}$$

By using (2.6), this can be approximated as

$$\approx \boldsymbol{R}(\hat{q})(1 + S(\boldsymbol{a_g}))(\boldsymbol{f}^b_{imu} - \boldsymbol{b}_{ins,acc} - \delta \boldsymbol{b}_{ins,acc} - \boldsymbol{w}_{acc}) \tag{5.51}$$
$$- \boldsymbol{R}(\hat{q})(\boldsymbol{f}^b_{imu} - \boldsymbol{b}_{ins,acc}) \tag{5.52}$$
$$= -\boldsymbol{R}(\hat{q})S(\boldsymbol{f}^b_{imu} - \boldsymbol{b}_{ins,acc} - \delta \boldsymbol{b}_{ins,acc} - \boldsymbol{w}_{acc})\boldsymbol{a_g} \tag{5.53}$$
$$- \boldsymbol{R}(\hat{q})(\delta \boldsymbol{b}_{ins,acc} - \boldsymbol{w}_{acc}) \tag{5.54}$$

By substituting $\boldsymbol{a}_g$ in (5.22), and by using (5.26), the following expression for $\dot{\boldsymbol{a}}_g$ is obtained:

$$\dot{\boldsymbol{a}}_g = \left(\boldsymbol{I}_{3x3} + \frac{1}{4}\boldsymbol{a}_g^T\boldsymbol{a}_g\right)\left(\boldsymbol{\omega}^b_{b/n} - \boldsymbol{\omega}^b_{ins}\right) - \frac{1}{2}\boldsymbol{S}(\boldsymbol{\omega}^b_{b/n} + \boldsymbol{\omega}^b_{ins})\boldsymbol{a}_g \tag{5.55}$$

This expression is identical to the one in Markley (2008). Ignoring higher order terms, gives

$$\dot{\boldsymbol{a}}_g \approx \left(\boldsymbol{\omega}^b_{b/n} - \boldsymbol{\omega}^b_{ins}\right) - \frac{1}{2}\boldsymbol{S}(\boldsymbol{\omega}^b_{b/n} + \boldsymbol{\omega}^b_{ins})\boldsymbol{a}_g \tag{5.56}$$

From (5.25) and (5.26):

$$\boldsymbol{\omega}^b_{b/n} = \boldsymbol{\omega}^b_{imu} - \boldsymbol{b}^b_{ars} - \boldsymbol{w}^b_{ars} \tag{5.57}$$
$$\boldsymbol{\omega}^b_{ins} = \boldsymbol{\omega}^b_{imu} - \boldsymbol{b}^b_{ins,ars} \tag{5.58}$$

Which implies that

$$\boldsymbol{\omega}_{b/n}^{b} + \boldsymbol{\omega}_{ins}^{b} = 2\boldsymbol{\omega}_{imu}^{b} - \boldsymbol{b}_{ars} - \boldsymbol{b}_{ins,ars} - \boldsymbol{w}_{ars}^{b} \tag{5.59}$$

$$\boldsymbol{\omega}_{b/n}^{b} - \boldsymbol{\omega}_{ins}^{b} = -\delta\boldsymbol{b}_{ars} - \boldsymbol{w}_{ars}^{b} \tag{5.60}$$

By substituting (5.59) and (5.60) into (5.56), the expression for $\boldsymbol{a}_g$ can be written as

$$\dot{\boldsymbol{a}}_g \approx -\delta\boldsymbol{b}_{ars} - \boldsymbol{w}_{ars}^{b} - -\frac{1}{2}\boldsymbol{S}(2\boldsymbol{\omega}_{imu}^{b} - \boldsymbol{b}_{ars} - \boldsymbol{b}_{ins,ars} - \boldsymbol{w}_{ars}^{b})\boldsymbol{a}_g \tag{5.61}$$

$$\dot{\boldsymbol{a}}_g \approx -\boldsymbol{S}(\boldsymbol{\omega}_{imu}^{b} - \boldsymbol{b}_{ins,ars})\boldsymbol{a}_g - \delta\boldsymbol{b}_{ars} - \boldsymbol{w}_{ars} + \frac{1}{2}\boldsymbol{S}(\delta\boldsymbol{b}_{ars} + \boldsymbol{w}_{ars})\boldsymbol{a}_g \tag{5.62}$$

The bias errors are modeled as first order Gauss-Markov processes, which are simply given by

$$\delta\dot{\boldsymbol{b}}_{ars} = -\frac{1}{T_{ars}}\delta\boldsymbol{b}_{ars} + \boldsymbol{w}_{b,ars} \tag{5.63}$$

$$\delta\dot{\boldsymbol{b}}_{acc} = -\frac{1}{T_{acc}}\delta\boldsymbol{b}_{acc} + \boldsymbol{w}_{b,acc} \tag{5.64}$$

5.7 Measurement Equations

The measurement vector is defined as

$$\boldsymbol{y} = [(\boldsymbol{p}_{b/n}^{n})^T \quad (\boldsymbol{v}_{b/n}^{n})^T \quad (\boldsymbol{f}_{imu}^{b})^T \quad (\boldsymbol{m}_{imu}^{b})^T]^T \tag{5.65}$$

While the corresponding INS estimates yields

$$\boldsymbol{y}_{ins} = [(\boldsymbol{p}_{ins}^{n})^T \quad (\boldsymbol{v}_{ins}^{n})^T \quad (\boldsymbol{f}_{ins}^{b})^T \quad (\boldsymbol{m}_{ins}^{b})^T]^T \tag{5.66}$$

The measurement equations are expressed as the error between the measurements and the predicted measurements from the INS. It can be written in terms of the error state $\delta\boldsymbol{x}$ as

$$\delta\boldsymbol{y} = \boldsymbol{y} - \boldsymbol{y}_{ins} = h(\delta\boldsymbol{x}) + \boldsymbol{v} \tag{5.67}$$

The position and velocity errors are simply given as

$$\boldsymbol{p}_{b/n}^{n} - \boldsymbol{p}_{ins} = \delta\boldsymbol{p}_{b/n}^{n} \tag{5.68}$$

$$\boldsymbol{v}_{b/n}^{n} - \boldsymbol{v}_{ins} = \delta\boldsymbol{v}_{b/n}^{n} \tag{5.69}$$

$$\tag{5.70}$$

While the magnetometer measurement equation can be written as

$$\boldsymbol{m}_{imu}^{b} - \boldsymbol{m}_{ins}^{b} = [\boldsymbol{R}(\boldsymbol{a}_g)^T \boldsymbol{R}(\hat{\boldsymbol{q}})^T - \boldsymbol{R}(\hat{\boldsymbol{q}})^T]\boldsymbol{m}^n + \boldsymbol{w}_{mag} + \boldsymbol{b}_{mag}^{b} \tag{5.71}$$

$$= [\boldsymbol{R}(\boldsymbol{a}_g)^T - \boldsymbol{I}_{3x3}]\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{m}^n + \boldsymbol{w}_{mag} + \boldsymbol{b}_{mag}^{b} \tag{5.72}$$

$$\approx -\boldsymbol{S}(\boldsymbol{a}_g)\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{m}^n + \boldsymbol{w}_{mag} + \boldsymbol{b}_{mag}^{b} \tag{5.73}$$

These measurement equations can be summarized and expressed in the form given in (5.67) as

$$\delta\boldsymbol{y} = \begin{bmatrix} \boldsymbol{p}_{b/n}^{n} - \boldsymbol{p}_{ins} \\ \boldsymbol{v}_{b/n}^{n} - \boldsymbol{v}_{ins} \\ \boldsymbol{f}_{imu}^{b} - \boldsymbol{f}_{ins}^{b} \\ \boldsymbol{m}_{imu}^{b} - \boldsymbol{f}_{ins}^{b} \end{bmatrix} = \underbrace{\begin{bmatrix} \delta\boldsymbol{p}_{b/n}^{n} \\ \delta\boldsymbol{v}_{b/n}^{n} \\ -\boldsymbol{S}(\boldsymbol{a}_g)\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{f}_{ins}^{n} \\ -\boldsymbol{S}(\boldsymbol{a}_g)\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{m}^n + \boldsymbol{w}_{mag} \end{bmatrix}}_{h(\delta\boldsymbol{x})} + \underbrace{\begin{bmatrix} \boldsymbol{w}_{pos} \\ \boldsymbol{w}_{vel} \\ \boldsymbol{w}_{acc} + \boldsymbol{b}_{acc}^{b} \\ \boldsymbol{w}_{mag} + \boldsymbol{b}_{mag}^{b} \end{bmatrix}}_{\boldsymbol{v}} \tag{5.74}$$

$$\tag{5.75}$$

While $\boldsymbol{p}_{ins}^{n}$, $\boldsymbol{v}_{ins}^{n}$ and $\boldsymbol{m}_{ins}^{b}$ are all easily obtained from the Navigation equations. The estimate of $\boldsymbol{f}_{ins}^{b}$ in the measurement equations will be derived using the two diferent methods shown next.

5.7.1 Estimation of $\boldsymbol{f}_{ins}^{b}$

Measurements of $\boldsymbol{f}_{ins}^{n}$ is not obtained directly from the navigation equations, so two different methods of estimation will be presented. The first one involves the design of a KF serving as a fast differentiator, estimating $\boldsymbol{f}_{ins}^{n}$ by integration of $\boldsymbol{p}_{ins}^{n}$ and $\boldsymbol{v}_{ins}^{n}$ from the INS output. The second method relies on using a pseudo-measurement, involving the use of angular velocity for calculation. The two different methods are illustrated in figure 5.4.

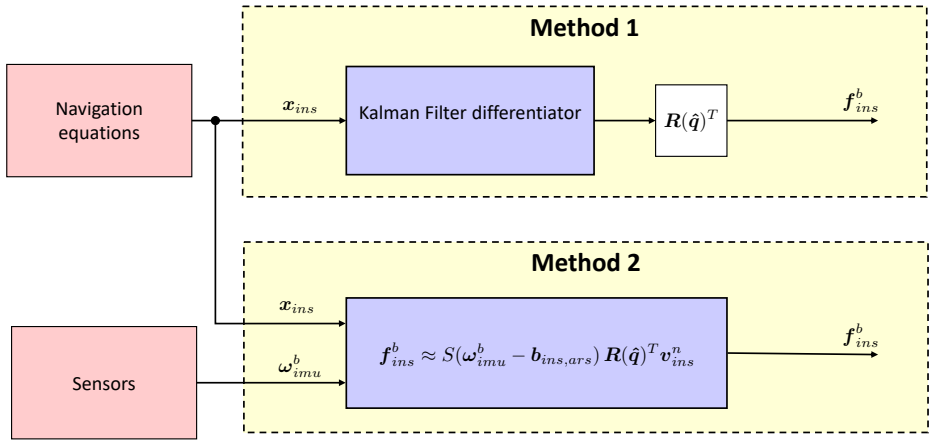

Figure 5.4: Two different methods to estimate $\boldsymbol{f}^b_{ins}$. Method 1 uses a fast differentiator for position and velocity integration, while the second method uses pseudo measurements.

Method 1: Kalman Filter differentiator

9-state KF will serve as a differentiator for estimation of $\boldsymbol{f}^n_{ins}$. The KF uses the position and velocity estimates $\boldsymbol{p}^n_{ins}$ and $\boldsymbol{v}^n_{ins}$ from the INS to calculate an estimate of $\boldsymbol{f}^n_{ins}$. Unlike the KF previously discussed, a $direct$ linear KF is sufficient.

The KF equations for continous time yields

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{p}}^n_{ins} \\ \dot{\boldsymbol{v}}^n_{ins} \\ \dot{\boldsymbol{f}}^n_{ins} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & \boldsymbol{I}_{3x3} & 0 \\ 0 & 0 & \boldsymbol{I}_{3x3} \\ 0 & 0 & 0 \end{bmatrix}}_{\boldsymbol{A}} \begin{bmatrix} \boldsymbol{p}^n_{ins} \\ \boldsymbol{v}^n_{ins} \\ \boldsymbol{f}^n_{ins} \end{bmatrix} + \underbrace{\begin{bmatrix} 0_{3x1} \\ I_{3x1} \\ 0_{3x1} \end{bmatrix}}_{\boldsymbol{B}} \boldsymbol{g}^n + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \boldsymbol{I}_{3x3} \end{bmatrix}}_{\boldsymbol{E}} \boldsymbol{w}_{acc} \tag{5.76}$$

with measurements

$$\boldsymbol{y} = \underbrace{\begin{bmatrix} \boldsymbol{I}_{3x3} & \boldsymbol{0}_{3x3} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & \boldsymbol{I}_{3x3} & \boldsymbol{0}_{3x3} \end{bmatrix}}_{\boldsymbol{C}} \boldsymbol{x} + \begin{bmatrix} \boldsymbol{v}_{pos} \\ \boldsymbol{v}_{vel} \end{bmatrix} \tag{5.77}$$

Where v_* and w_* are white noise, $\boldsymbol{g}^n = \boldsymbol{u} = [0 \quad 0 \quad 9.81\, m/s^2]^T$ is the standard acceleration of gravity on Earth.

The Kalman gain and co-variance updates are obtained by

$$\boldsymbol{K}[k] = \hat{\boldsymbol{P}}^-[k]\boldsymbol{C}_d^T[k](\boldsymbol{C}_d[k]\hat{\boldsymbol{P}}^-[k]\boldsymbol{C}_d^T[k] + \boldsymbol{R}_d[k])^{-1} \tag{5.78}$$

$$\hat{\boldsymbol{P}}^+[k] = (\boldsymbol{I} - \boldsymbol{K}[k]\boldsymbol{C}_d[k])\hat{\boldsymbol{P}}^-[k](\boldsymbol{I} - \boldsymbol{K}[k]\boldsymbol{C}_d[k])^T + \boldsymbol{K}[k]\boldsymbol{R}[k]\boldsymbol{K}^T[k] \tag{5.79}$$

$$\hat{\boldsymbol{P}}^-[k+1] = \boldsymbol{C}_d[k]\hat{\boldsymbol{P}}^+[k]\boldsymbol{C}_d^+[k] + \boldsymbol{E}_d[k]\boldsymbol{Q}_d[k]\boldsymbol{E}_d[k]^T \tag{5.80}$$

The state corrections yields

$$\hat{\boldsymbol{x}}^{+}[k] = \hat{\boldsymbol{x}}^{-}[k] + \boldsymbol{K}[k](y[k] - \boldsymbol{C}_d\hat{\boldsymbol{x}}^{-}[k]) \tag{5.81}$$

and finally the new states are predicted as

$$\hat{\boldsymbol{x}}^{-}[k+1] = \boldsymbol{A}_d\hat{\boldsymbol{x}}^{+}[k] + \boldsymbol{B}_d[k]\boldsymbol{u}[k] \tag{5.82}$$

The performance of the filter is shown in Figure 5.5, which verifies that the filter tracks the true value of $\boldsymbol{f}^n_{ins}$ in a satisfactory manner.

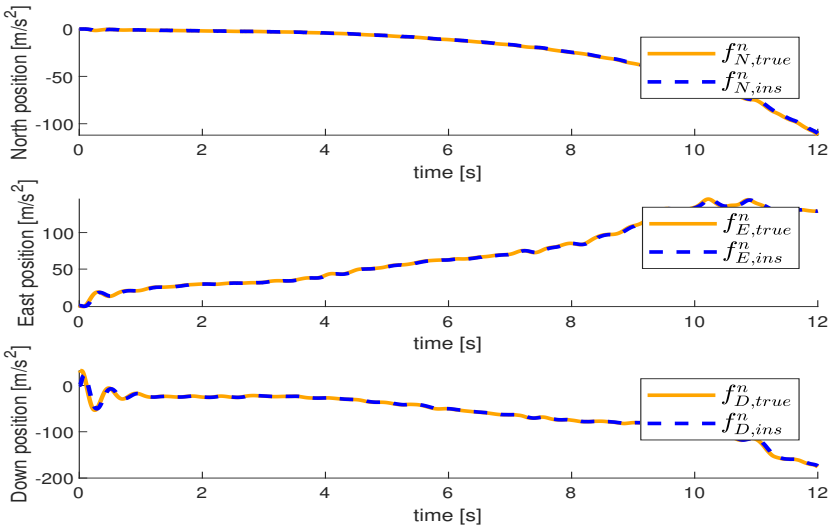

Figure 5.5: $\boldsymbol{f}^n_{ins}$ estimation using method 1.

The accelerometer measurement equation for method 1 can be written as

$$\begin{aligned}
\boldsymbol{f}^b_{imu} - \boldsymbol{f}^b_{ins} &= [\boldsymbol{R}(\boldsymbol{a}_g)^T\boldsymbol{R}(\hat{\boldsymbol{q}})^T - \boldsymbol{R}(\hat{\boldsymbol{q}})^T]\boldsymbol{f}^n_{ins} + \boldsymbol{w}_{acc} + \boldsymbol{b}^b_{acc} - \boldsymbol{b}^b_{acc,ins} && (5.83)\\
&= [\boldsymbol{R}(\boldsymbol{a}_g)^T - \boldsymbol{I}_{3x3}]\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{f}^n_{ins} + \boldsymbol{w}_{acc} + \delta\boldsymbol{b}_{acc} && (5.84)\\
&\approx -\boldsymbol{S}(\boldsymbol{a}_g)\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{f}^n_{ins} + \boldsymbol{w}_{acc} + \delta\boldsymbol{b}_{acc} && (5.85)\\
&= \boldsymbol{S}(\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{f}^n_{ins})\boldsymbol{a}_g + \boldsymbol{w}_{acc} + \delta\boldsymbol{b}_{acc} && (5.86)
\end{aligned}$$

Method 2: Pseudo measurement

Instead of using the differentiator proposed in the previous section, estimation of $\boldsymbol{f}^b_{ins}$ can be obtained by the following expression.

$$\boldsymbol{v}_{ins}^{n} = \boldsymbol{R}(\hat{\boldsymbol{q}})\boldsymbol{v}_{ins}^{b} \tag{5.87}$$

$$\boldsymbol{f}_{ins}^{b} = \dot{\boldsymbol{v}}_{ins}^{n} = \dot{\boldsymbol{R}}(\hat{\boldsymbol{q}})\boldsymbol{v}_{ins}^{b} + \boldsymbol{R}(\hat{\boldsymbol{q}})\dot{\boldsymbol{v}}_{ins}^{b} \tag{5.88}$$

$$= \boldsymbol{R}(\hat{\boldsymbol{q}})\left[\dot{\boldsymbol{v}}_{ins}^{b} + S(\boldsymbol{\omega}_{imu}^{b} - \boldsymbol{b}_{ins,ars})\boldsymbol{v}_{ins}^{b}\right] \tag{5.89}$$

By assuming $\dot{\boldsymbol{v}}_{ins}^{b} << \boldsymbol{v}_{ins}^{b}$, the expression can be further simplified to

$$\dot{\boldsymbol{v}}_{ins}^{n} \approx \boldsymbol{R}(\hat{\boldsymbol{q}})S(\boldsymbol{\omega}_{imu}^{b} - \boldsymbol{b}_{ins,ars})\boldsymbol{v}_{ins}^{b} \tag{5.90}$$

This assumption may be inaccurate during phases where the interceptor is doing maneouvers which involves rapid movements. When $\dot{\boldsymbol{v}}_{ins}^{n}$ is considerably large, this measurement equations will cause oscillations compared to the actual value of $\boldsymbol{f}_{ins}^{b}$.

For the second method, the measurement equation will be slightly different. The accelerometer measurement equation for method 2 yields

$$\boldsymbol{f}_{imu}^{b} - \boldsymbol{f}_{ins}^{b} = [\boldsymbol{R}(\boldsymbol{a}_g)^T\boldsymbol{R}(\hat{\boldsymbol{q}})^T - \boldsymbol{R}(\hat{\boldsymbol{q}})^T]\boldsymbol{f}_{ins}^{n} + \boldsymbol{w}_{acc} + \boldsymbol{b}_{acc}^{b} - \boldsymbol{b}_{acc,ins}^{b} \tag{5.91}$$

$$\approx [\boldsymbol{R}(\boldsymbol{a}_g)^T - \boldsymbol{I}_{3x3}]\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{R}(\hat{\boldsymbol{q}})S(\boldsymbol{\omega}_{imu}^{b} - \boldsymbol{b}_{ins,ars})\boldsymbol{v}_{ins}^{b} + \boldsymbol{w}_{acc} + \delta\boldsymbol{b}_{acc} \tag{5.92}$$

$$\approx S(\boldsymbol{\omega}_{imu}^{b} - \boldsymbol{b}_{ins,ars})\boldsymbol{v}_{ins}^{b}\boldsymbol{a}_g + \boldsymbol{w}_{acc} + \delta\boldsymbol{b}_{acc} \tag{5.93}$$

5.7.2 Method 1 and 2 comparison

Table 5.7.2 shows state estimation errors for the two different methods described in subsection 5.7.1. The table shows that there is a decent amount of equivalence between the methods.

The second method provides a significant lower final error of the attitude estimation than what the first method is able to. This might be a result of the second method including the angular velocity directly in the measurement equation.

In comparison the specific force will not contribute with any information about the yaw angle. Both the roll and pitch angle will directly affect how the gravity contributes to the specific force vector. The yaw angle will on the other hand stay the same. This will serve as a disadvantage for the first method.

Also note that the RMS values are slightly higher for the second method, probably caused by ocillations in the $\boldsymbol{f}_{ins}^{b}$ estimations due to large values of $\dot{\boldsymbol{v}}_{ins}^{n}$.

FIgure 5.6 shows the error in bias calculations using the two different methods. The errors are computed according to

$$b_{*,err} = \sum_{j=1}^{M}\sum_{i=1}^{N}|\boldsymbol{b}_{*}(j,i) - \boldsymbol{b}_{*,ins}(j,i)| \tag{5.94}$$

	Attitude (deg)			Position (m)			Velocity (m/s)		
	roll	pitch	yaw	north	east	down	north	east	down
method 1									
max error	-0.93	-2.0	-2.3	-0.33	0.57	-0.88	-0.29	-0.38	-0.45
final error	0.17	-0.28	0.12	-0.33	0.57	-0.88	0.087	0.14	0.43
RMS	0.44	1.1	1.1	0.12	0.23	0.48	0.11	0.1	0.11
method 2									
max error	-0.87	-2.5	2.4	-0.34	0.59	-0.83	-0.29	-0.38	-0.45
final error	0.13	-0.2	$9.3e{-3}$	-0.34	0.59	-0.83	0.1	0.16	0.43
RMS	0.35	1.5	1.4	0.12	0.24	0.46	0.12	0.099	0.12

Table 5.1: Two different methods of estimating $\boldsymbol{f}^b_{ins}$ for use in MEKF measurement equation, as described in subsection 5.7.1 and 5.7.1 . The table shows estimation errors for the different states.

where N is the number of bias samples, and M is the bias vector dimension. M equals to three for both the acceleration bias (north, east, down) and gyro bias (roll, pitch, yaw).

Example: $\boldsymbol{b}_{ars,ins}(2, 35)$ corresponds to the the 35th bias sample for INS estimate of the pitch bias.

Notice how the gyro bias converges faster for method 1 than method 2. This might be related to the assumption in (5.87). When $\dot{\boldsymbol{v}}^n_{ins}$, the measurement equation will be inaccurate, which may cause slower convergence of the bias estimate.

It is reason to be believe that this might affect the accuracy of state estimations as well. Table 5.7.2 shows that the RMS values for the attitude error for method 2 is higher for pitch and yaw angle, while the roll angle is slightly lower. The fact that metod 2 gives generally higher RMS values substantiates to the statement above, as the second method is more vulnerable as a result of large values of $\dot{\boldsymbol{v}}^n_{ins}$.

For the rest of the simulations carried out in this thesis, the first method will be used, as it shows better bias estimation performance.

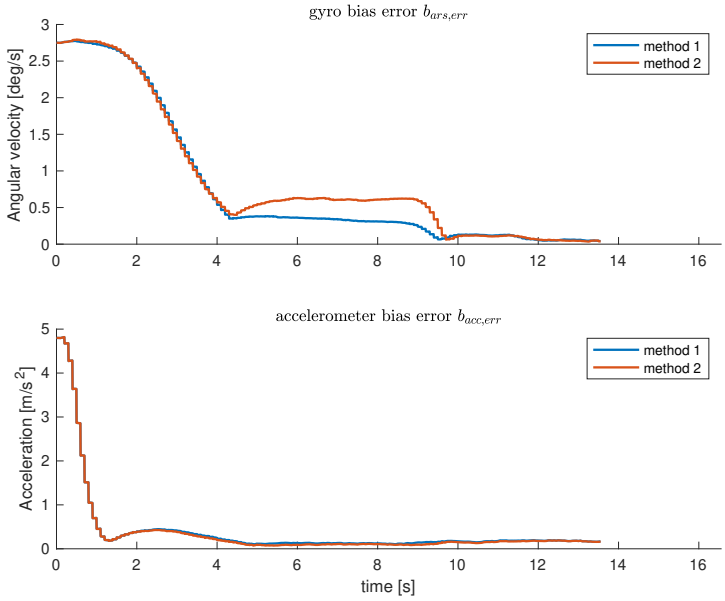

Figure 5.6: Comparison between error in bias estimation using the two different methods described in subsection 5.7.1 and 5.7.1.

5.7.3 Discrete-time matrices

To get the system on the form as in (5.2a), the error state and measurement equations need to be discretized. Discretizing the system with the Euler method gives

$$\boldsymbol{A}_d = \boldsymbol{I}_{15} + h\frac{\partial \boldsymbol{f}_k}{\partial x_k}\bigg|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_k} =$$

$$\boldsymbol{I}_{15} + h\begin{bmatrix} 0 & \boldsymbol{I}_{3x3} & 0 & 0 & 0 \\ 0 & 0 & -\boldsymbol{R}(\hat{q})S(\boldsymbol{f}^b_{imu} - \boldsymbol{b}_{ins,acc}) & 0 & -\boldsymbol{R}(\hat{q}) \\ 0 & 0 & -S(\boldsymbol{\omega}^b_{imu} - \boldsymbol{b}_{ins,ars}) & -\boldsymbol{I}_{3x3} & 0 \\ 0 & 0 & 0 & -\frac{\boldsymbol{I}_{3x3}}{T_{ars}} & 0 \\ 0 & 0 & 0 & 0 & -\frac{\boldsymbol{I}_{3x3}}{T_{acc}} \end{bmatrix} \tag{5.95}$$

$$\boldsymbol{E}_d = h\begin{bmatrix} 0 & 0 & 0 & 0 \\ -\boldsymbol{R}(\hat{q}) & 0 & 0 & 0 \\ 0 & -\boldsymbol{I}_{3x3} & 0 & 0 \\ 0 & 0 & \boldsymbol{I}_{3x3} & 0 \\ 0 & 0 & 0 & \boldsymbol{I}_{3x3} \end{bmatrix} \tag{5.96}$$

Finally, the discrete-time measurement matrix $\boldsymbol{H}_d$, applying method 1 for $\boldsymbol{f}^n_{ins}$ estimation, yields

$$\boldsymbol{H}_d = \frac{\partial \boldsymbol{h}_k}{\partial \delta x_k}\bigg|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_k} = \begin{bmatrix} \boldsymbol{I}_{3x3} & 0 & 0 & 0 \\ 0 & \boldsymbol{I}_{3x3} & 0 & 0 \\ 0 & 0 & \boldsymbol{S}(\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{f}^n_{ins}) & 0 \\ 0 & 0 & \boldsymbol{S}(\boldsymbol{R}(\hat{\boldsymbol{q}})^T\boldsymbol{m}^n) & 0 \end{bmatrix} \tag{5.97}$$

where

$$\boldsymbol{Q}_d \approx \sigma^2_{w_k}\boldsymbol{I}_{15x15} \tag{5.98}$$

$$\boldsymbol{R}_d \approx \sigma^2_{v_k}\boldsymbol{I}_{15x15} \tag{5.99}$$

and the property $\delta\hat{\boldsymbol{\epsilon}} = 0$ has been used.

5.8 Target tracking

Inertial guidance systems may be sufficient to guide ballistic missiles to a target with fixed coordinates, for example, a place on earth known in advance. The problem is that these methods are not well suited for guiding towards moving targets with unpredictable coordinates like enemy cruise-missiles or other threats. When the target coordinates are not known in advance of the missile launch, real-time target sensing and corresponding manouvering changes are required for an interception to occur.

The missile flight can be divided into three phases: The boost, midcourse and terminal phase. In the boost phase, onboard inertial guidance systems are usually used to calculate

an arrival point at the end of the boost-phase. In the midcourse, off-board target tracking methods are often used to produce the desired course of the missile, to bring it close to the target. When the missile is close to the target, onboard sensors are usually taking over as the missile enters terminal phase. The terminal phase can begin anywhere from tens of seconds down to only a few seconds before intercept, depending on the missile capability and the mission objective. As the boost and midcourse phases may accumulate residual errors, the terminal phase serves to reduce the final distance between the interceptor and target below a specified level Palumbo (2010).

5.9 Homing systems

Homing systems can be classified in three general groups:

5.9.1 Passive homing systems

A passive system is designed to detect by measuring natural emanations or radiation such as heat, light and sound waves Siouris (2004). The passive system is therefore based on using the characteristics of radiation from the target itself to measure the angular direction of the target relative to the missile. Passive systems do not provide target range or closing velocity information, which may be an disadvantage as some guidance techniques, including PN, requires this information. Common examples of passive systems are infrared and radio-frequency seekers Palumbo (2010).

5.9.2 Semiactive homing systems

While the passive system only uses emitted signals from the target, semiactive systems uses a reflected wave emitted by a beam of light, laser, IR or RF from an external source, i.e. a radar. In addition to angular direction, semiactive systems are able to provide missile-target closing velocity and angular direction to the target, which can help the overall guidance accuracy in some instances. Due to the fact that an external source is used to produce the emitted signal, the semiactive system has the advantage that no additional size or weight to the missile is necessary Palumbo (2010).The illumination must be present at all times during the flight of the missile Siouris (2004).

5.9.3 Active homing systems

In an active system, the target is illuminated and tracked by an on-board sensor on the missile itself. An advantage is that the active system can provide relative range, range rate, and angular direction measurements. The additional information can improve the guidance accuracy even more. As the missile carries the tracking equipment, the active system comes with a high cost of additional power drain and weight. This usually restricts

active systems to be used before the terminal phase of the flight, after some other form of guidance has brought the missile to within a short distance of the target Palumbo et al. (2010c).

5.10 Target-tracking filter

By assuming a semiactive or active homing systems, the measured quantities can be used to produce measurements of relative position between the missile and the target.

As shown in Palumbo et al. (2010b), relative position "measurements" can be obtained as pseudo-measurements composed of noisy LOS angle and relative range measurements between the interceptor and target Palumbo et al. (2010c). Hence, these measurements must be filtered, and estimates of the relative velocity must be obtained from these noise pseudo-measurements of the relative position, as these measurements are required by the PN-algorithm. Some guidance laws also requires measurements of the target acceleration perpendicular to the missile-target LOS, so this information will also be estimated based on the relative position measurements.

To achieve this, a 9-state linear KF, similar to the one introduced in section 5.7.1 will be used. The stochastic continous-time model yields

$$\begin{bmatrix} \dot{\boldsymbol{p}}_{\boldsymbol{r}}(t) \\ \dot{\boldsymbol{v}}_{\boldsymbol{r}}(t) \\ \dot{\boldsymbol{a}}_{\boldsymbol{T}}(t) \end{bmatrix} = \begin{bmatrix} 0 & \boldsymbol{I}_{3x3} & 0 \\ 0 & 0 & \boldsymbol{I}_{3x3} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{\boldsymbol{r}}(t) \\ \boldsymbol{v}_{\boldsymbol{r}}(t) \\ \boldsymbol{a}_{\boldsymbol{T}}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ -\boldsymbol{1}_{3x1} \\ 0 \end{bmatrix} \boldsymbol{a}_I + \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{I}_{3x3} \end{bmatrix} \boldsymbol{w}_{T,acc} \quad (5.100)$$

with measurements

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{I}_{3x3} & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{\boldsymbol{r}}(t) \\ \boldsymbol{v}_{\boldsymbol{r}}(t) \\ \boldsymbol{a}_{\boldsymbol{T}}(t) \end{bmatrix} + \boldsymbol{v}_{r,pos} \quad (5.101)$$

Where $\boldsymbol{p}_r$ and $\boldsymbol{v}_r$ are relative position and velocity, while $\boldsymbol{a}_I$ is the interceptor's acceleration. $\boldsymbol{v}_{r,pos}$ is white noise modeled as a wiener process.

By discretizing, the following model is obtained:

$$\begin{bmatrix} \dot{\boldsymbol{p}}_{\boldsymbol{r}}[k+1] \\ \dot{\boldsymbol{v}}_{\boldsymbol{r}}[k+1] \\ \dot{\boldsymbol{a}}_{\boldsymbol{T}}[k+1] \end{bmatrix} = \boldsymbol{I}_{9x9} + h \begin{bmatrix} 0 & \boldsymbol{I}_{3x3} & 0 \\ 0 & 0 & \boldsymbol{I}_{3x3} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{\boldsymbol{r}}[k] \\ \boldsymbol{v}_{\boldsymbol{r}}[k] \\ \boldsymbol{a}_{\boldsymbol{T}}[k] \end{bmatrix} + h \begin{bmatrix} 0 \\ -\boldsymbol{1}_{3x1} \\ 0 \end{bmatrix} \boldsymbol{a}_I + h \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{I}_{3x3} \end{bmatrix} \boldsymbol{w}_{T,acc} \quad (5.102)$$

The Kalman gain, covariance and new state updates are computed as explained in (5.78) - (5.82).

For initialization of the filter, four position samples $\{\boldsymbol{p}_m(1), \boldsymbol{p}_m(2), \boldsymbol{p}_m(3), \boldsymbol{p}_m(4)\}$ are obtained, such that the initial values can be estimated as (Palumbo et al., 2010c)

$$\hat{\boldsymbol{p}}_r[0] = \sum_{i=1}^{4} \frac{\boldsymbol{p}_m(i)}{4} \tag{5.103}$$

$$\hat{\boldsymbol{v}}_r[0] = \frac{\boldsymbol{p}_m(4) - \boldsymbol{p}_m(3)}{\Delta t} \tag{5.104}$$

$$\hat{\boldsymbol{a}}_T[0] = \frac{\hat{\boldsymbol{v}}_r[0]}{2\Delta t} - \frac{\boldsymbol{p}_m(2) - \boldsymbol{p}_m(1)}{2\Delta t^2} \tag{5.105}$$

where Δt is the time between each position sample measurement.

Chapter 6

Guidance System

6.1 Introduction.

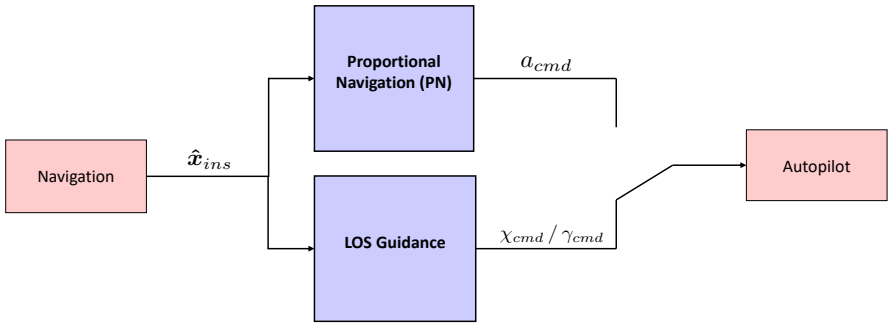

Figure 6.1: Two different guidance laws are implemented. The first one is the state-of-the-art Proportional Navigation law, while the second one is a LOS guidance law.

It is the guidance law that mainly distinguish an unguided projectile from a guided missile. The primary function of the guidance law is to generate steering guidance commands given some information about the missile and target as inputs. Usually, the guidance laws are in form of the magnitude and direction of normal acceleration that the missile needs to apply (NPTEL, 2012).

6.1.1 Line of Sight (LOS)

Some of the most classical guidance laws are based on the LOS vector (NPTEL, 2012). The idea is to guide the missile on a LOS course in an attempt to keep it on a line between the reference point and the target. The line of sight vector is defined as the vector between the waypoint/target and a reference point. The reference point might be a control station on-ground, but if the target tracker is on-board of the missile, the LOS vector will be the straight line between the missile and target.

6.2 Proportional navigation (PN)

Proportional navigation is perhaps the most commonly used guidance laws in modern missile guidance. The guidance law has nothing to do with navigation. The reason behind the somewhat misleading name comes from the limitation of the vocabulary of guidance litterature back in the early days of development (NPTEL, 2012).

When the interceptor and target is on a collision course, there is no relative velocity between the two bodys perpendicular to the LOS vector between them. This means that the LOS rate is equal to zero, while the closing velocity is positive. This is the idea behind the PN law - if the LOS rate at anytime is non-zero, then the guidance law should command the autopilot to do a fin deflection to cancel the LOS rate (NPTEL, 2012). If we assume a planar engagement, the expression for the commanded missile acceleration a_{M_c} is defined as (Palumbo et al., 2010a)

$$a_{M_c} = NV_c\dot{\lambda} \tag{6.1}$$

Where N is called the navigation constant, V_c is the closing velocity and $\dot{\lambda}$ is the LOS rate in an inertial reference frame. For a three dimensional case, the LOS rate must simply be measured by two seperate instruments mutually perpendicular to the sensor boresight. Information about the LOS rate $\dot{\lambda}$ and closing velocity V_c are derived based on target sensor measurement that are available. To obtain good estimates, a semi-active or active system with on-board sensors are necessary. From (3), (4) and (6) in Palumbo et al. (2010a):

$$\bar{\boldsymbol{v}} \triangleq \frac{\partial}{\partial t}\bar{\boldsymbol{r}} = \dot{R}\bar{\mathbf{1}}_{\boldsymbol{r}} + R\frac{\partial}{\partial t}\bar{\mathbf{1}}_{\boldsymbol{r}} \tag{6.2}$$

$$\bar{\boldsymbol{n}} \triangleq \frac{\partial}{\partial t}\bar{\mathbf{1}}_r \tag{6.3}$$

$$\bar{\mathbf{1}}_\omega \triangleq \bar{\mathbf{1}}_r \times \bar{\mathbf{1}}_n \tag{6.4}$$

where $\bar{\boldsymbol{v}}$ is the relative velocity, R is the distance, $\bar{\boldsymbol{r}}$ is the LOS vector between the missile and target. $\bar{\boldsymbol{n}}$ is the LOS rate vector. $\bar{\mathbf{1}}_*$ is a unit vector. These vectors are all illustrated in figure 6.2 (Palumbo et al., 2010a).

(6.5)

By combining (6.2) - (6.4):

$$\bar{\boldsymbol{v}} = \dot{R}\,\bar{\mathbf{1}}_r + R|\bar{\boldsymbol{n}}|\,\bar{\mathbf{1}}_n \tag{6.6}$$

$$\bar{\mathbf{1}}_r \times \bar{\boldsymbol{v}} = \dot{R}\,(\bar{\mathbf{1}}_r \times \bar{\mathbf{1}}_r) + R|\bar{\boldsymbol{n}}|\,(\bar{\mathbf{1}}_r \times \bar{\mathbf{1}}_n) \tag{6.7}$$

$$\bar{\mathbf{1}}_r \times \bar{\boldsymbol{v}} = R|\bar{\boldsymbol{n}}|\,(\bar{\mathbf{1}}_r \times \bar{\mathbf{1}}_n) \tag{6.8}$$

$$\bar{\mathbf{1}}_r \times \bar{\boldsymbol{v}} = R|\bar{\boldsymbol{n}}|\,\bar{\mathbf{1}}_w \tag{6.9}$$

(6.4) can be rewritten as

$$\bar{\mathbf{1}}_w \times \bar{\mathbf{1}}_r = \bar{\mathbf{1}}_n \tag{6.10}$$

the LOS rate vector $\bar{\boldsymbol{n}}$ yields

$$\bar{\boldsymbol{n}} = \dot{\lambda} = (\bar{\mathbf{1}}_r \times \bar{\boldsymbol{v}}) \times \frac{\bar{\mathbf{1}}_r}{R} \tag{6.11}$$

Finally, it follows from 6.5 that the range rate can be expressed as

$$\dot{R} = -V_c = \bar{\boldsymbol{v}} \cdot \bar{\mathbf{1}}_r \tag{6.12}$$

This shows how the required parameters for the PN law can be derived by the use of relative position and relative velocity measurements obtained from a semi-active or active seeker.

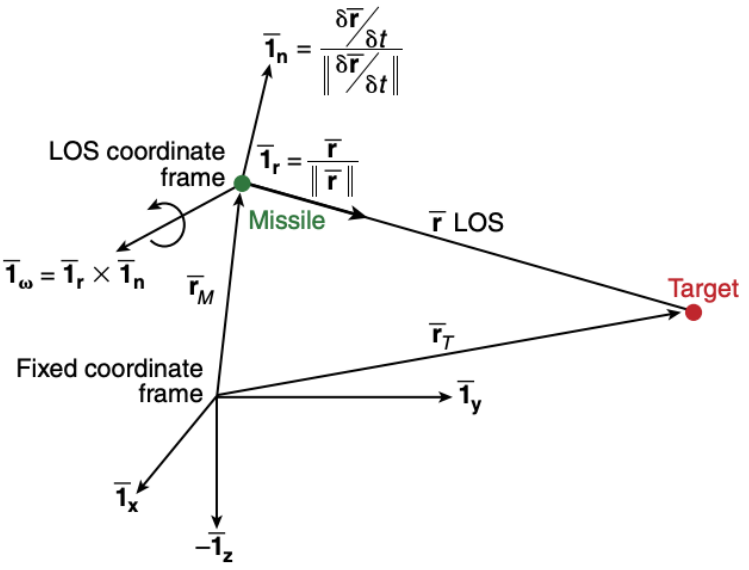

Figure 6.2: LOS coordinate frame used for PN law derivation.

6.3 LOS guidance with course and flight-path-angle commands

While the PN algorithm needs information about relative position and relative velocity, the course and flight-path-angle autopilot design only relies on information about relative position. This makes it possible to use low-cost sensors on the ground, in addition to tracking threats further away from the asset/interceptor than what is possible with sensors available on-board of the interceptor.

In a practical system, it is highly advantageous to keep the software as simple as possible. By utilizing the autopilot with course and flight-path-angle commands explained in Chapter 4, an intuitive design based on trigonometric relationship is derived. While most classical LOS guidance laws produces acceleration or angular velocity commands (NPTEL, 2012), this guidance law produces the desired course and flight-path-angle as commands directly. The design has a lot of similarities to the enclosure based steering for waypoint tracking, explaned in section 6.3.1.

In section 6.2 it was explained how the PN law needs information about the relative velocity, and produces acceleration commands instead of course and flight-path-angle commands for calculating fin deflections. One advantage with this approach, is the ability to easier control the turning rate of the missile. This makes the PN law more robust against threats doing evasive and unpredictable manouvers, compared to a design that does not use velocity information directly in computation of fin deflection commands. By changing from the LOS to the PN guidance law when the interceptor is closing up on the target, i.e. entering the terminal phase of the flight, might therefore improve the chance of interception. As the accuracy in tracking of the target degrades with distance from the control station, the chances are that the tracking is not precise enough to cause an interception (Palumbo et al., 2010a). By switching to on-board sensors during the terminal-phase, this problem could be avoided.

6.3.1 Enclosure based steering for waypoint tracking

Several path and waypoint tracking methods are based on LOS steering laws. By considering the vertical plane, the idea is to properly assign a value to $\chi(t)$ to obtain satisfactory steering control. One of these methods are called enclosure based steering. By considering a circle with sufficient large radius $R > 0$, enclosing $\boldsymbol{p^n} = [x, y]^T$, two intersections on the straight line between the last and next waypoint are obtained. The method computes desired course angle χ_d as

$$\chi_d(t) = \text{atan2}(y_{\text{los}} - y(t), x_{\text{los}} - x(t)) \tag{6.13}$$

where

$$[[x_{\text{los}} - x(t)]^2 + [[y_{\text{los}} - y(t)]^2 = R^2 \tag{6.14}$$

$$\tan(\alpha_k) = \frac{y_{\text{los}} - y_k}{x_{\text{los}} - x_k} = \text{constant} \tag{6.15}$$

must be solved in order to obtain $\boldsymbol{p}^n_{\text{los}}$ Fossen (2011). A submarine moving towards a waypoint using enclosure based steering is shown in figure 6.3. This well-established method for waypoint tracking represents the basic ideas for designing the target tracking guidance law presented below.

6.3.2 LOS guidance for target tracking

For missile guidance the LOS vector is often defined as the straight line between a ground station and the target Fossen (2011). This differs from the illustration in Figure 6.3 (Fossen, 2011), where the LOS vector is defined as the straight line between the marine vessel and the next waypoint. Also, the target position is no longer constant. The basic principle is to guide the missile on a course in an attempt to keep it on the straight line joining the target/threat and the control point/launch station.

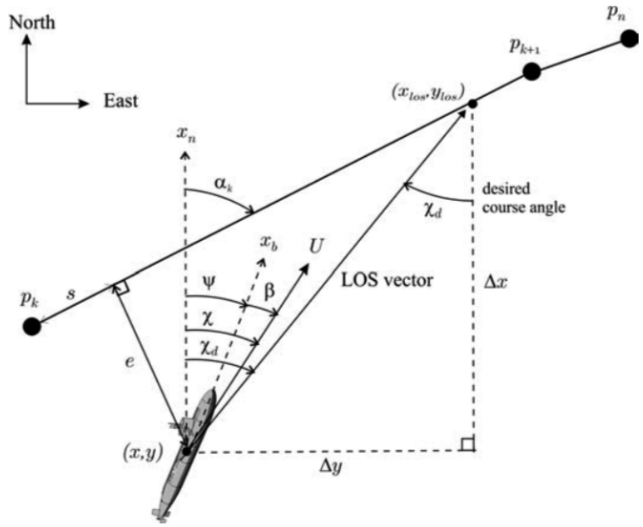

Figure 6.3: Encolsure based steering.

Similar to the course and flight path angle controlled autopilots, the LOS guidance tracking equations are assuming decoupling between the north/down and north/east planes. Figure 6.4 shows the trigonometric relationship between the launch platform, target and interceptor. The interceptor's position is denoted $p_I = (x_I, y_I, z_I)$, the threat position is $p_T = (x_T, y_T, z_T)$ and the launch platform is $p_L = (x_L, y_L, z_L)$. The interceptor's position decomposed in the north/east plane is given as $P_{I_{xy}}$.

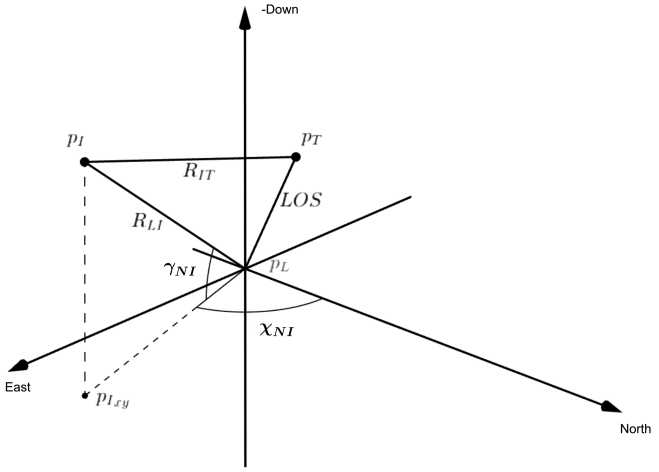

Figure 6.4: Illustration of the guidance system in three dimensions.

6.3.3 Vertical guidance system

The angles between north, threat and interceptor in the $vertical$ plane are calculated as

$$\gamma_{NT} = \sin^{-1}\left(\frac{z_T - z_L}{|p_T - p_L|}\right) \tag{6.16}$$

$$\gamma_{NI} = \sin^{-1}\left(\frac{z_I - z_L}{|p_I - p_L|}\right) \tag{6.17}$$

$$\theta_v = \gamma_{NI} - \gamma_{NT} \tag{6.18}$$

where $p_c = (x_c, y_c, z_c)$ corresponds to the reference position in the vertical plane used to calculate the commanded flight path angle. The distance between the launch platform and the interceptor in the vertical plane is

$$R_{LIv} = \cos(\chi_{NI})\, R_{LI} \tag{6.19}$$

while the distance between the interceptor and threat is

$$R_{ITv} = \sqrt{(x_T - x_I)^2 + (z_T - z_I)^2} \tag{6.20}$$

Furthermore, e_v, r_v and finally the desired commanding flight path angle γ_c is calculated as

$$e_v = \sin(\theta_v) R_{LIv} \tag{6.21}$$

$$r_v = \sqrt{R_{LIv}^2 - e_v^2} \tag{6.22}$$

$$|p_c - p_L| = r_v + k_v \sqrt{R_{ITv}^2 - e_v^2} \tag{6.23}$$

$$z_c = \sin(\gamma_{NT}) |p_c - p_L| \tag{6.24}$$

$$\gamma_c = -\text{atan2}(z_c - z_I, x_c - x_I) \tag{6.25}$$

where k_v is used to adjust what point on the LOS vector between the launch platform and threat that the interceptor will aim at.

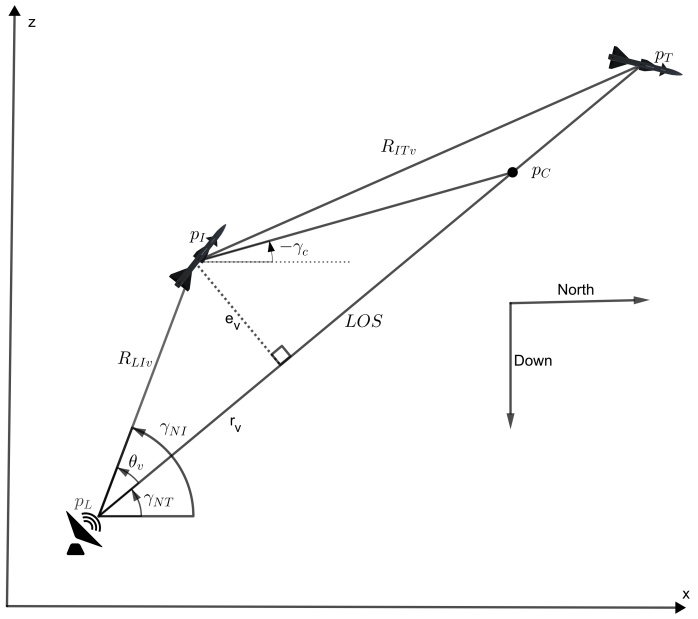

Figure 6.5: The guidance system decomposed in the vertical plane.

6.3.4 Horizontal guidance system

The angles between north, threat and interceptor in the $horizontal$ plane are calculated as

$$\chi_{NT} = \tan^{-1}\left(\frac{y_T - y_L}{x_T - x_L}\right) \tag{6.26}$$

$$\chi_{NI} = \tan^{-1}\left(\frac{y_I - y_L}{x_I - x_L}\right) \tag{6.27}$$

$$\psi_h = \chi_{NT} - \chi_{NI} \tag{6.28}$$

where $p_c = (x_c, y_c, z_c)$ corresponds to the reference position in the vertical plane used to calculate the commanded flight path angle. The distance between the launch platform and the interceptor in the horizontal plane is

$$R_{LIh} = \cos(\gamma_{NI})\, R_{LI} \tag{6.29}$$

while the distance between the interceptor and threat is

$$R_{ITh} = \cos(\gamma_{NI}) \sqrt{(x_T - x_I)^2 + (y_T - y_I)^2} \tag{6.30}$$

Furthermore, e_h, r_h and finally the desired commanding flight path angle χ_c is calculated as

$$e_h = \sin(\psi_h)\, R_{LIh} \tag{6.31}$$

$$r_h = \sqrt{R_{LIh}^2 - e_h^2} \tag{6.32}$$

$$|p_c - p_L| = r_h + k_h \sqrt{R_{ITh}^2 - e_h^2} \tag{6.33}$$

$$z_c = \sin(\chi_{NT})\, |p_c - p_L| \tag{6.34}$$

$$\chi_c = \text{atan2}(y_c - y_I, x_c - x_I) \tag{6.35}$$

where k_h is used to adjust what point on the LOS vector between the launch platform and threat that the interceptor will aim at.

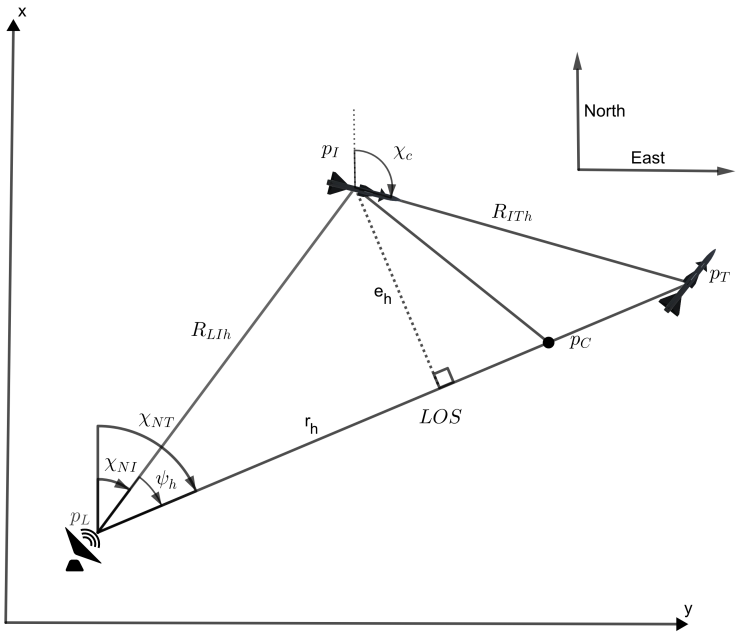

Figure 6.6: The guidance system decomposed in the horizontal plane

6.3.5 Future target position estimation

The guidance system described above involves controlling the missile towards the straight line between the launch platform and the target. As long as the target has a non-zero velocity, the result will be that the missile is following a trajectory that is clearly not optimal in terms of total distance traveled.

If estimating the targets position in a future time, the guidance law can be modifed to track a point that is ahead of the target. By defining $\Delta t := t^* - t$ where t is the current time and t^* is a future time, the targets position at time t^* can be estimated as

$$p_T(t^*) = p_T(t) + v_T(t)\Delta t \tag{6.36}$$

Δt will be chosen as the time which it will take the missile to intercept the target or to arrive at the Closest Point of Approach (CPA). In the litterature, this is known as time-to-go t_{go} (Palumbo et al., 2010b). Figure 6.7 shows the engagement geometry between the missile and target.

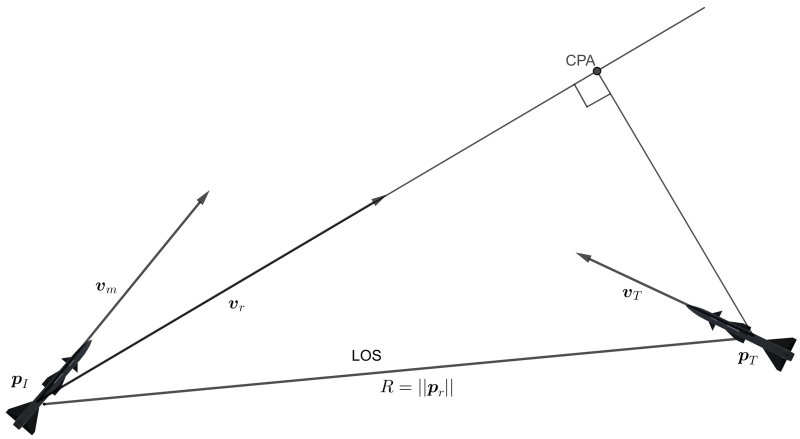

Figure 6.7: Missile-target engagement geometry.

By defining the relative position as $\boldsymbol{p}_r := p_T - p_I$ and the relative velocity $\boldsymbol{v}_r := v_T - v_I$, the future target-missile relative position at time t^* can be stated as (Palumbo et al., 2010b):

$$\bar{\boldsymbol{r}}(t^*) = \bar{\boldsymbol{r}}(t) + \bar{\boldsymbol{v}}(t)\Delta t \tag{6.37}$$

By inspecting figure 6.7, illustrated by the parpendicual line between the target and CPA, it is easy to see that the following condition holds:

$$\bar{\boldsymbol{r}}(t^*) \cdot \bar{\boldsymbol{v}}(t^*) = 0 \tag{6.38}$$

By combining (6.37) and (6.38), and by assuming constant velocity, the expression for $t_{go} := \Delta t$ yields

$$t_{go} = -\frac{\boldsymbol{p}_r(t) \cdot \boldsymbol{v}_r(t)}{\boldsymbol{v}_r(t) \cdot \boldsymbol{v}_r(t)} \tag{6.39}$$

Now that estimation of t_{go} is obtained, the targets predicted position at can be estimated from (6.36) as

$$\tilde{p}_T = p_T(t) + v_T(t)t_{go} \tag{6.40}$$

where $\tilde{p}_T$ is the predicted position of the target at the closest point of approach, given that both the missile and the target is moving with a constant velocity.

Even though the prediction assumes information about relative velocity, the calculation of t_{go} is only used as an approximate reference for calculation of the future estimation of the targets position. This relaxes the need of precise measurements. It is therefore assumed that the KF derived in section 5.10 from on-ground measurements will give a good enough approximation of the relative velocity. On the other hand, the lower precission might

cause the inteceptor to miss the target during the terminal phase. The PN law has higher demands regarding preciseness of the estimates as it is directly involved in calculations of the commanded acceleration, making the need of on-board sensors desired.

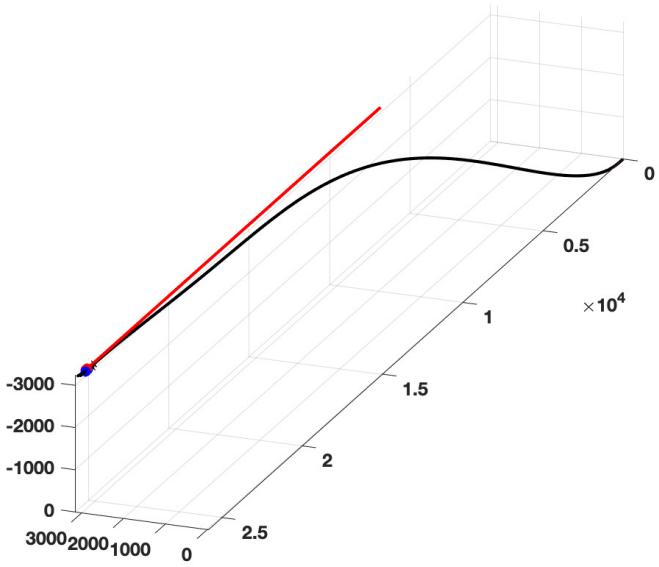

Figure 6.8: Missile tracking the current target position, p_T. Interception time: 8.725 sec

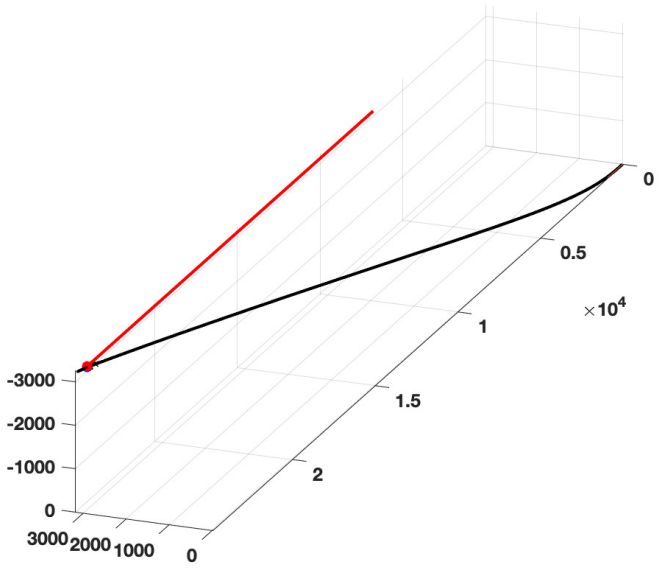

Figure 6.9: Missile tracking an estimated future target position, $\tilde{p}_T$. Interception time: 8.250 sec

Figure 6.8 and 6.9 shows how the interceptor is tracking and intercepting a target. The initial position of the threat in NED is $\boldsymbol{p}_T(0) = [7000 \quad 3100 \quad -3200]^T$ while it is moving at a constant velocity of $\boldsymbol{v}_T = [2100 \quad 0 \quad 0]^T$. The target is modeled as a point mass. The interceptor has the initial position $\boldsymbol{p}_I(0) = [0 \quad 0 \quad 0]^T$, initial attitude $\boldsymbol{\Theta}(0) = [0 \quad 0 \quad 0]^T$ and is traveling with a constant velocity magnitude of $V_m = 1000$ m/s. In figure 6.8, the guidance algorithm is following the LOS vector between the launch platform at $\boldsymbol{p}_L = [0 \quad 0 \quad 0]^T$ and the targets position p_T. In figure 6.9, the LOS vector is pointing from the launch platform towards the predicted location of the target $\tilde{p}_T$, as given in (6.40). While the missile is able to successfully intercept in both simulations, the result in figure 6.9 gives a reduction of 0.475 seconds (or 5.5%) in the total time from launch to interception.

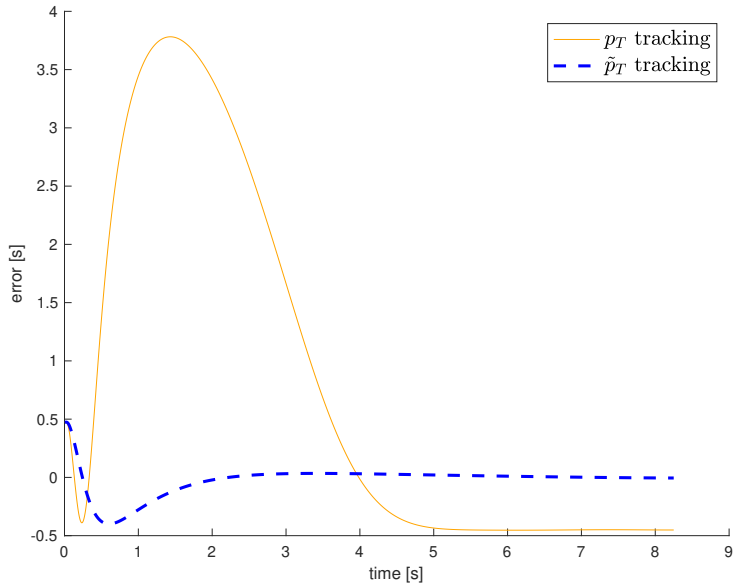

Figure 6.10: Time-to-go estimation error.

Figure 6.10 shows the error between the calculated t_{go} from (6.39) and the actual time remaining until interception. The time error is clearly closer to zero throughout the whole simulation when $\tilde{p}_T$ is used.

As t_{go} uses CPA for calculation, the deviation from zero can be interpreted as a measurement of how optimal the trajectory is. The peak with amplitude 3.8 for p_T after 1.4 seconds shows that p_T is not the optimal choice. This can be intuetively confirmed by insepecting the beginning of the trajectory shown in figure 6.8.

Chapter 7

Implementation

7.1 Introduction

The complete GNC system, including all subsystems described in this thesis have been implemented in a MATLAB/Simulink enviroment. This chapter will address some important features and aspects regarding the implementation.

In order to meet the mission objective described in section 1.4, an asset and threat needs to be present in the simulation enviroment in addition to the interceptor.

Figure 7.1 shows the interaction between asset, threat and interceptor. The dark square in 7.1 can be replaced by the light squares in figure 1.1, and vice versa.

The bullet points in 7.1 describes the different maneouvers, control and navigation systems that are available.

Switch blocks are introduced to easily switch between using the true states and the estimated states from INS and tracking KF. The three-loop autopilot and PN law is placed in a single subsystem, while the course/flight-path angle autopilot and the LOS guidance is placed in another. This makes it easy to switch between the different GNC systems, by introducing another switch block.

Some of the coding, including the three-loop autopilot, a video recording of missile trajectories, PN law and an M-S function for the rigid bodies are implemented in a Simulink environment by students under the supervision of Professor Murat Arcak during the spring semester 2019. The three-loop autopilot has been implemented using Gain Scheduling.

7.2 Simulation environment

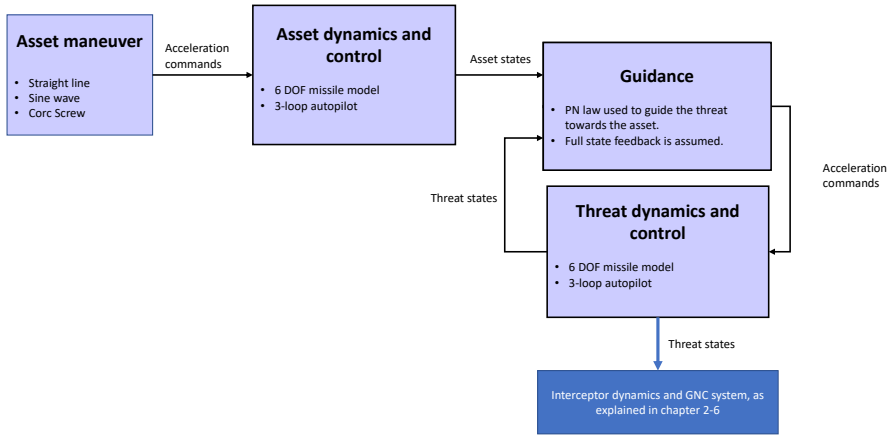

Figure 7.1: Relationship between the three rigid bodies and their control systems.

There are three parties involved in the simulations, all of them modeled as a rigid body using the equations of motions from chapter 3.

7.2.1 Interceptor

The interceptors mission is to intercept a threat before the threat is able to hit the asset. The three-loop autopilot with the PN law, the course/flight-path-angle autopilot with LOS guidance is both implemented and compared for different scenarios described in the next chapter. The interceptor will also be using sensor models, the MEKF and target-tracking filters will be implemented and tested under different circumstances. The interceptor will track the threat independently of the asset maneuver, and will not have any knowledge in advance about the threat's guidance system or trajectory.

7.2.2 Threat

The threats objective is to hit the asset before the interceptor is able to destroy it. It is assumed full-state feedback for the threat, that means filters for state estimation and asset tracking is unnecessary. For tracking and trajectory calculation, the threat is applying the three-loop autopilot combinated with the PN law. The threat has no knowledge about the interceptor, so the guidance law will not optimize its trajectory in terms of avoiding being destroyed by the interceptor.

7.2.3 Asset

The asset is moving independently of both the threat and interceptor. Full-state feedback is assumed, and the trajectory is calculated by using the simple maneuvers described in section 7.4. No information about the threat or interceptor is obtained, so the asset will not choose its maneuvers based on the trajectory of the incoming threat or interceptor.

7.3 Missile animation

For better visualization of the attitude and trajectory of the missile, an animation of the missile's body is introduced. While a simple marker in a 3D plot will visualize the trajectory in a satisfactory manner, a body animation of the missile will further improve the ability to investigate the missile's attitude. It is reasonable to assume that the course and flight-path-angle may be different than the missile's attitude, as sideslip and angle-of-attack can not be expected to be equal zero. By modifing the code from Riley (2003), verticies and faces of a CAD file are extracted as a .mat file (see cad2mat.m). Further, by modifying the code from Scordamaglia (2016) a 3D body of the .mat file is displayed in the same orientation as the missile's attitude. Finally, this is displayed as a 3D animation, showing the trajectory of both the interceptor and threat. Figure 7.2 shows an example of how an interceptor tracking a point mass can be visualized. The black line shows the missile's trajectory, while the red line corresponds to the point mass. This simulation was done using the guidance law derived in section 6.3.

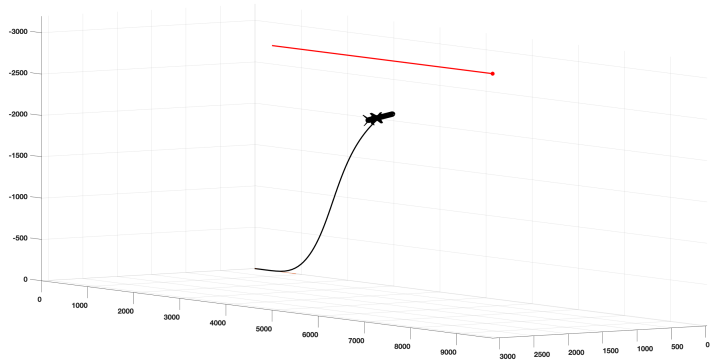

Figure 7.2: Missile chasing a moving point mass.

7.4 Maneuvers

Instead of using one of the proposed guidance laws to track a target, some simple maneuvers can be implemented in order to create a desired trajectory. These maneuvers are used for asset simulation, and in some cases, for the threat as well. The different maneuvers described in this section calculates acceleration commands which are fed into the three-loop autopilot described in chapter 4. It is assumed that the rigid body is initialized with a predefined initial position $\boldsymbol{p}_0$, velocity $\boldsymbol{v}_0$ and attitude $\boldsymbol{\Theta}_0$.

7.4.1 Straight line

The straight line maneouver is the simplest one. The straight line trajectory is given by simply choosing the acceleration command to zero

$$a_c = [0 \quad 0 \quad 0]$$

The three-loop autopilot will then try to keep the rigid body moving on a straight line.

7.4.2 Sine wave in yaw

The acceleration command used to create a sine wave yields

$$a_c = [0 \quad 0 \quad a_\psi]$$

where

$$a_\psi = 1.5 \sin(2\pi t)$$

7.4.3 Corc Screw

The acceleration command used to create a corcscrew maneouver is calculated as

$$a_c = [0 \quad a_\theta \quad a_\psi]$$

where

$$a_\theta = 1.5 \sin(2\pi t + \pi/2)$$

and

$$a_\psi = 1.5 \sin(2\pi t)$$

Figure 7.3 shows the corc screw maneouver performed by the asset.

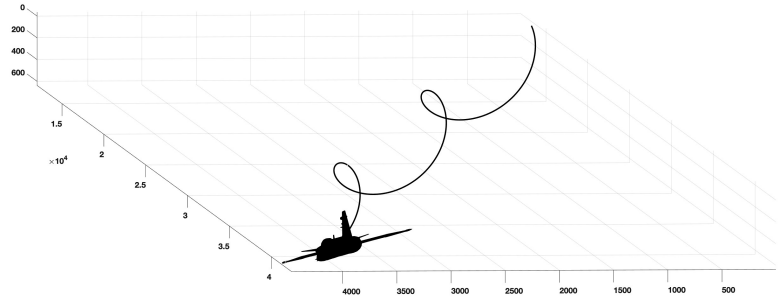

Figure 7.3: Asset performing the corc screw maneouver.

7.5 Quaternion normalization

Numerical round-off errors may occur, and will cause a violation of the unit constraint on the quaternion. To prevent this, the following normalization is used

$$\boldsymbol{q}[k+1] = \frac{\boldsymbol{q}[k+1]}{|\boldsymbol{q}[k+1]|} \tag{7.1}$$

Chapter 8

Simulation results

8.1 Description of case studies

As a measurement of the performance, the result is given as Root Mean Square (RMS) errors, calculated by the following formula

$$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{x}(i) - x_{true}(i)\right)^2}$$

8.1.1 Stop condition

The simulation will automatically finish if the missile either miss or intercepts the target. To simulate an interception, the missile has to pass the target within a predefined distance. The range is chosen to be 6 m, as it is assumed that an explosion within this distance is sufficient in order to successfully destroy the target. If a miss between the interceptor and threat occurs, the distance between them will start increasing instead of decreasing. In other words, the range rate $\dot{R}$ will become negative.

The following stop conditions are then obtained:

if $R < 6$ **then**
  print 'interception sucessfull';
  return true;
else if $R > 6$ **and** $\dot{R} < 0$ **then**
  print 'interception unsucessfull';
  return true;
else return false;

Naturally, these conditions apply both between the interceptor and threat, as well as between the threat and asset. If one of the interceptions suceeds, the other interception is automatically considered unsucsessfull.

8.2 Parameters and initial conditions

The system is sampled with a constant sample step size of $h = 0.005$ seconds, identical to the IMU sampling rate of 200 Hz.

8.2.1 Kalman filter parameters

The following discrete KF tuning matrices have been used for the MEKF

$$Q_{k,mekf} = \text{diag}\,(1\text{e}{-10}_{1\times 3} \quad 1\text{e}{-14}_{1\times 2} \quad 1\text{e}{-13} \quad 1\text{e}{-8}_{1\times 2} \quad 1\text{e}{-7} \quad 1\text{e}{-12}_{1\times 3})$$
$$R_{k,mekf} = \frac{1}{t_{mekf}}\text{diag}\,(10_{1\times 3} \quad 1\text{e}{-3}_{1\times 3} \quad 5_{1\times 3} \quad 1\text{e}{-2}_{1\times 3})$$
$$\hat{P}_{0,mekf} = \text{diag}\,(1\text{e}{-1}_{1\times 3} \quad 1\text{e}{-2}_{1\times 3} \quad 1\text{e}{-10}_{1\times 3} \quad 2\text{e}{-6}_{1\times 3} \quad 5\text{e}{-2}_{1\times 3})$$

For target tracking, the Kalman filter tuning matrices yields

$$Q_{k,tkf} = \text{diag}\,(1\text{e}{-6}_{1\times 3} \quad 1\text{e}{-5}_{1\times 3} \quad 1\text{e}{-8}_{1\times 3})$$
$$R_{k,tkf} = \frac{1}{t_{tkf}}\text{diag}\,(1_{1\times 3})$$
$$\hat{P}_{0,tkf} = \text{diag}\,(1\text{e}{-2}_{1\times 3} \quad 2.25_{1\times 3} \quad 20_{1\times 3})$$

The Kalman filter used for $\boldsymbol{f}_{ins}^{n}$ estimation, is initialized as

$$Q_{k,fkf} = \text{diag}\,(1\text{e}9_{1\times 3} \quad 1\text{e}8_{1\times 3} \quad 1\text{e}11)$$
$$R_{k,fkf} = \frac{1}{t_{fkf}}\text{diag}\,(10_{1\times 3} \quad 1\text{e}{-3}_{1\times 3})$$
$$\hat{P}_{0,fkf} = \text{diag}\,(1\text{e}{-5}_{1\times 9})$$

8.2.2 Autopilot and reference model parameters

The autopilot feedback gains are chosen as

$$K_{\dot{\alpha}} = 0.1$$
$$K_{\dot{\beta}} = 0.1$$

The tracking and input weight matrices for the flight-path-angle controller are chosen as

$$Q_\gamma = \text{diag}\,(1 \quad 1.5) \qquad R_\gamma = 2$$

while the tracking and input weight matrices for the course controller are

$$Q_\chi = \text{diag}\,(1 \quad 1.5) \qquad R_\chi = 2$$

For the reference model, $\omega_n = 100$ and $\gamma = 1$ are used for both the lateral and longitudinal autopilot.

8.2.3 Guidance law parameters

For the LOS guidance, $k_h = 0.9$ and $k_v = 0.9$ are used. The guidance controller is also programmed to intercept at the predicted target position $\tilde{p}$.

The proportional navigation constant for the PN law is set to be $N = 5$.

8.2.4 Interceptor initial conditions

The interceptor is modeled as the rigid body with the characteristics given in chapter 3.

Initial position:

$$p_I^n(0) = (-2000m \quad -100m \quad 0)^T \tag{8.1}$$

Initial velocity:

$$v_I^n(0) = (3000m/s \quad 0 \quad 0)^T \tag{8.2}$$

Initial attitude:

$$\Theta_I^n(0) = (0 \quad 0 \quad 0)^T \tag{8.3}$$

(8.4)

The interceptor is intercepting the threat using the proportional navigation algorithm. The Interceptor uses filters from chapter 5 for INS estimation and for threat tracking.

8.2.5 Threat initial conditions

The threat is modeled as the rigid body with the characteristics given in chapter 3.

Initial position:

$$p_T^n(0) = (6000m \quad 6000m \quad -6000m)^T \tag{8.5}$$

Initial velocity:

$$v_T^n(0) = (2600m/s \quad 0 \quad 0)^T \tag{8.6}$$

Initial attitude:

$$\Theta_T^n(0) = (0 \quad 0 \quad 0)^T \tag{8.7}$$

$$\tag{8.8}$$

The threat is intercepting the asset using the proportional navigation algorithm. It is assumed that the threat has full knowledge about its own and the assets states, that is, full state feedback.

8.2.6 Asset initial conditions

The asset is modeled as the rigid body with the characteristics given in chapter 3.

Initial position:

$$p_A^n(0) = (12000m \quad 0 \quad 0)^T \tag{8.9}$$

Initial velocity:

$$v_A^n(0) = (2000m/s \quad 0 \quad 0)^T \tag{8.10}$$

Initial attitude:

$$\Theta_A^n(0) = (0 \quad 0 \quad 0)^T \tag{8.11}$$

$$\tag{8.12}$$

The asset is using the Cork-screw maneuver to evade from the threat.

Simulations are carried out using a gyro measurement bias defined as

$$\boldsymbol{b}_{acc}^b := [b_{acc,x}^b \quad b_{acc,y}^b \quad b_{acc,z}^b]^T$$

modeled as a Wiener process. The bias is initialized as

$$\boldsymbol{b}_{ars,0}^b = (1.3\text{e}{-2}\ \text{rad/s} \quad 1.6\text{e}{-2}\ \text{rad/s} \quad 1.9\text{e}{-2}\ \text{rad/s})^T$$

where

$$\dot{\boldsymbol{b}}_{ars}^{b} = \boldsymbol{\omega}_{b,ars}$$

and $\boldsymbol{\omega}_{b,ars}$ is white noise. The bias is saturated such that

$$-1.4\,\boldsymbol{b}_{ars,0}^{b} \leq |\boldsymbol{b}_{ars}^{b}| \leq 1.4\,\boldsymbol{b}_{ars,0}^{b}$$

to avoid the bias to grow far from the initial value. The estimated gyro bias is defined as

$$\hat{\boldsymbol{b}}_{ars}^{b} := [\hat{b}_{ars,\phi}^{b} \quad \hat{b}_{ars,\theta}^{b} \quad \hat{b}_{ars,\psi}^{b}]^{T}$$

Simulations are carried out using a acceleration measurement bias defined as

$$\boldsymbol{b}_{acc}^{b} := [b_{acc,x}^{b} \quad b_{acc,y}^{b} \quad b_{acc,z}^{b}]^{T}$$

modeled as a Wiener process. The bias is initialized as

$$\boldsymbol{b}_{acc,0}^{b} = \left(1.3\ \text{m/s}^2 \quad 1.6\ \text{m/s}^2 \quad 1.9\ \text{m/s}^2\right)^{T}$$

where

$$\dot{\boldsymbol{b}}_{acc}^{b} = \boldsymbol{\omega}_{b,acc}$$

and $\boldsymbol{\omega}_{b,acc}$ is white noise. The bias is saturated such that

$$-1.4\,\boldsymbol{b}_{acc,0}^{b} \leq |\boldsymbol{b}_{acc}^{b}| \leq 1.4\,\boldsymbol{b}_{acc,0}^{b}$$

The estimated acceleration bias is defined as

$$\hat{\boldsymbol{b}}_{acc}^{b} := [\hat{b}_{acc,x}^{b} \quad \hat{b}_{acc,y}^{b} \quad \hat{b}_{acc,z}^{b}]^{T}$$

8.3 State estimation

The objective of the first case study is to investigate the error-state MEKF performance.

The first case study is carried out using three different sets of INS sampling rates. The time step for the Simulink system is $h = 0.005$ seconds for all cases.

For the first simulation, both the IMU and GNSS measurements are obtained at a sampling rate of $t_{gnss} = t_{imu} = 200$ Hz. This is an unrealistic simulation scenario, as GNSS receivers is not able to maintain such a high aiding rate, see subsection 5.3.4. The reason behind this case study is to investigate the MEKF's performance to give precise state estimations for more realistic sampling rates, by comparing them to the ideal but unrealistic scenario presented in this case.

For the second case GNSS data is obtained at a reduced frequency of $t_{gnss} = \frac{t_{imu}}{20} = 10$ Hz. This means that the INS equations will still be updated at the same rate as the high-frequency IMU measurements, but the corrections will be computed at a lower frequency.

For the third case, the aiding updates will be further decreased to $t_{gnss} = \frac{t_{imu}}{100} = 2$ Hz.

The target tracking measurements are assumed to be obtained at a rate of $t_{tkf} = 200$ Hz for all the three simulation.

Case 1:	**Case 2:**	**Case 3:**
$t_{imu} = 200$ Hz	$t_{imu} = 200$ Hz	$t_{imu} = 200$ Hz
$t_{tkf} = 200$ Hz	$t_{tkf} = 200$ Hz	$t_{tkf} = 200$ Hz
$t_{gnss} = 200$ Hz	$t_{gnss} = 10$ Hz	$t_{gnss} = 2$ Hz
$t_{mekf} = 200$ Hz	$t_{mekf} = 10$ Hz	$t_{mekf} = 2$ Hz

8.3.1 Simulation results

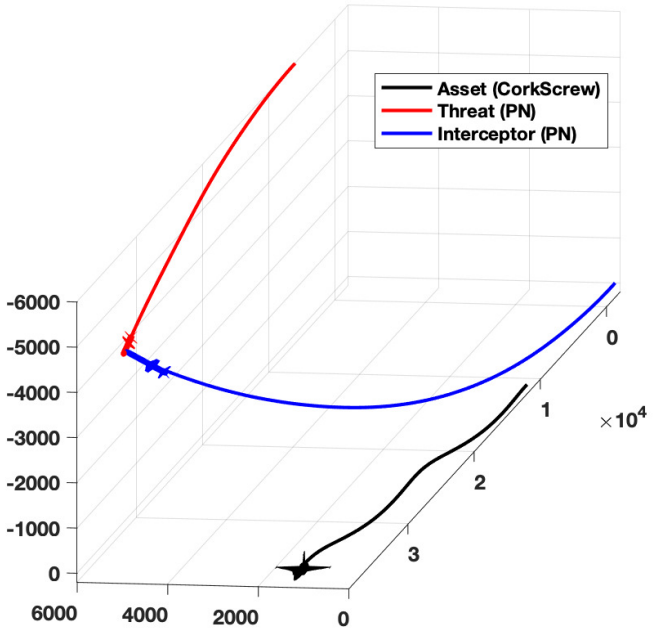

Figure 8.1: Case 2: Trajectory

Figure 8.1 shows the trajectories for the asset, threat and interceptor. Both the interceptor and threat are using PN, while the asset is using a corc screw manouver.

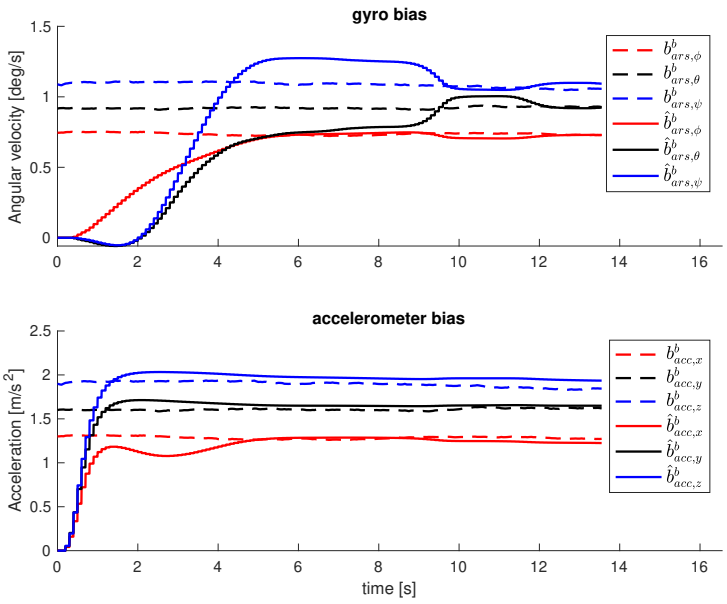

Figure 8.2: Case 2: Bias estimation.

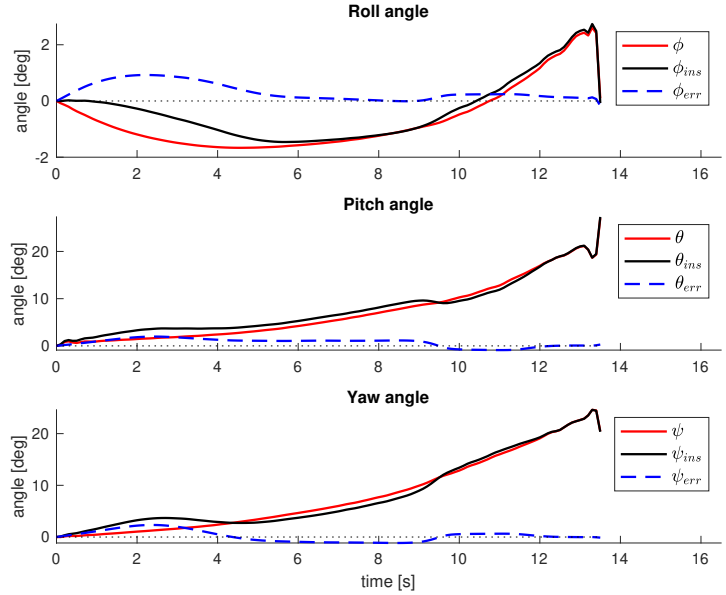

Figure 8.3: Case 2: Atittude estimation.

Figure 8.2 shows that the system is able to estimate all the gyro and accelerometer biases. The figure illustrates bias estimation using the sampling rate in *Simulation 2*. This shows acceptable estimations for cases where the IMU measurements and the GNSS measurements are obtained at different sampling rates. Figure 8.3 shows estimations and estimation errors for the interceptors attitude. Notice that the attitude error is mainly present in the first six seconds of the simulation. This is an expected result, as it corresponds to the time of convergence for bias estimation shows in Figure 8.2.

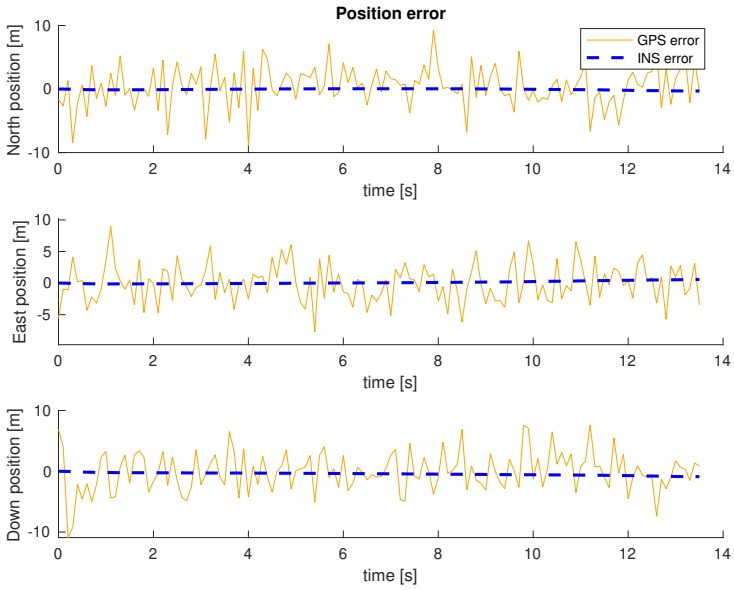

Figure 8.4: Case 2: Position error estimation.

Figure 8.4 shows how the GNSS noise is seemingly completely filtered out from the position measurements.

Table 8.3.1 shows state estimation errors for the three different simulations. *Max error* is the maximum difference between the true state and the estimated state throughout the simulation. *Final error* is the difference between the true state and the estimated state at the end of the simulation. *RMS* is the RMS value as explained in section 8.1.

Notice that *case 1* and *case 2* gives almost the same RMS values. This verifies that the a sampling rate of $t_{gnss} = 10$ Hz and $t_{imu} = 200$ Hz is sufficient to obtain promising state estimations. When the sampling conditions given in *case 3* are applied, the state estimation accuracy is slightly decreased .

Table 8.1: INS estimation errors for different sampling rates

	Attitude (deg)			Position (m)			Velocity (m/s)		
	roll	pitch	yaw	north	east	down	north	east	down
Case 1									
Max error	-0.86	-2.0	-2.4	-0.39	0.75	-0.72	0.29	-0.32	0.86
Final error	0.081	-0.31	0.26	-0.39	0.73	-0.72	0.29	0.29	0.86
RMS	0.32	1.0	0.99	0.13	0.33	0.31	0.092	0.089	0.1
Case 2									
Max error	-0.93	-2.0	-2.3	-0.33	0.57	-0.88	-0.29	-0.38	-0.45
Final error	0.17	-0.28	0.12	-0.33	0.57	-0.88	0.087	0.14	0.43
RMS	0.44	1.1	1.1	0.12	0.23	0.48	0.11	0.1	0.11
Case 3									
Max error	-2.1	-4.9	5.3	-1.9	-2.8	-3.7	1.1	-1.3	-1.4
Final error	0.048	0.65	-0.77	-1.9	-1.5	-3.1	0.33	0.86	0.99
RMS	1.0	2.3	2.7	1.0	1.8	2.6	0.6	0.6	0.57

8.3.2 Target tracking

Figure 8.5 shows the error between true and estimated relative position, relative velocity and threat's acceleration. Figure 8.6 shows how the linear KF performance for the north, east and down directions seperately. Here, relative position measurements are obtained with a standard deviation of $\sigma_r = 2$ m.

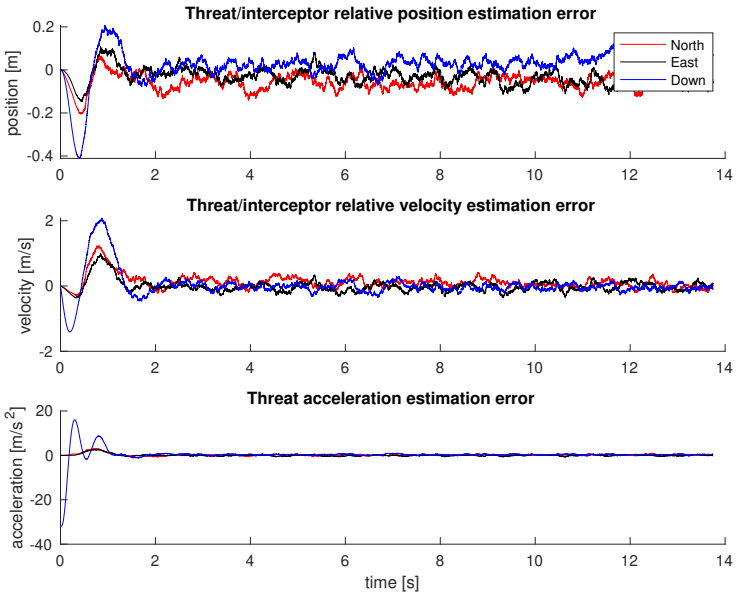

Figure 8.5: Target tracking estimation errors.

Figure 8.6 shows estimated position errors compared to the measurement errors, which gives satisfactory results.

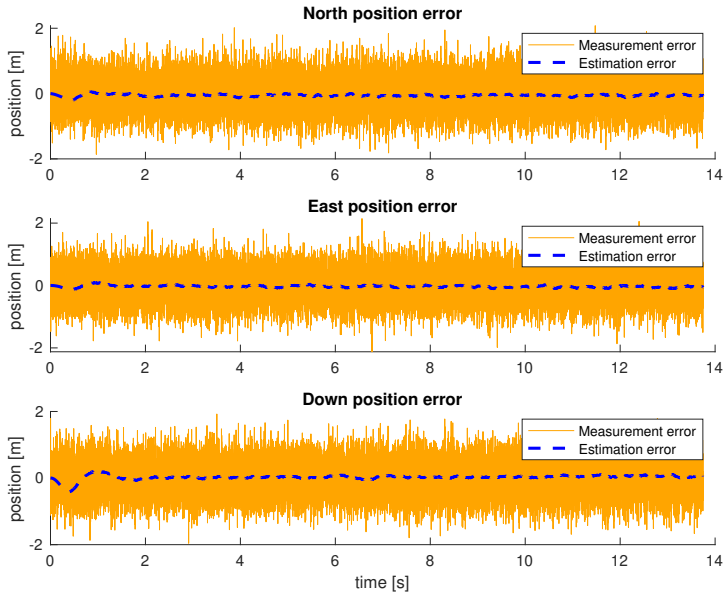

Figure 8.6: Target tracking estimated vs measured relative position error.

The filter uses its state estimates to linearize the state equations on the fly. It may quickly diverge if the estimation error becomes too great or if the process is modeled incorrectly. That is, the performance of the target-tracking KF is directly related to the estimation accuracy of the MEKF, as information about the interceptor's acceleration in inertial frame needs to be accurate.

8.4 Guidance law comparison

This section will aim to investigate the main differences, disadvantages and advantages between the two different guidance laws.

8.4.1 Force acting on interceptor

By using the same conditions presented in the previous section, the specific force L2-norm acting on the interceptor's body is illustrated in figure 8.7

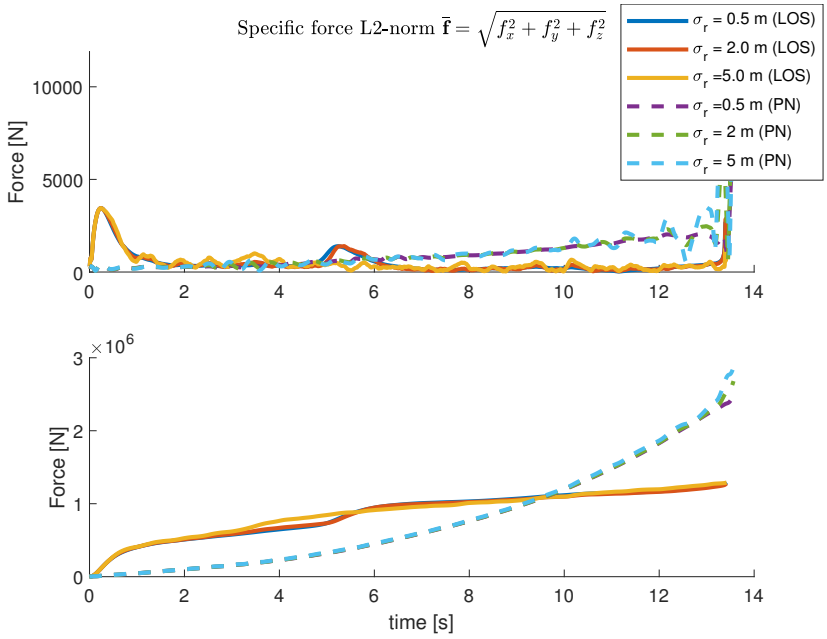

Figure 8.7: Specific force acting on interceptor following a threat. σ_r is the standard deviation of the relative position measurement noise. The bottom subplot shows the accumulated force over time.

Figure 8.7 shows that after a while, the PN law will start to acheive a larger accumulated force acting on the missile than the LOS law. As the LOS law is predicting the future position of target at interception time, one can assume that the force acting on the interceptor will be larger at the initial part of the launch. The PN law on the other hand, will apply the most agressive fin deflections when closing up upon the target. This can be verified by looking at the figure. How much impact this has in a real scenario is hard to say, but it is closely related to fuel usage. If fuel consumption is a issue, there might be scenarios where the LOS law will make the missile able to cover a larger distance before the fuel runs out.

This can also be interpreted as a way to show noise sensitivity of the two guidance laws. As σ_r affects both the relative position and relative velocity estimates which again is used in the PN law, it is reasonable to say that the PN law may be more vulnerable to inaccurate measurements than the LOS law.

Furthermore, three different case studies has been carried out to measure the difference in simulation time, specific force L2-norm, final distance and interception verification. Videos of the simulations are saved in the folder ”animation” in the .zip file.

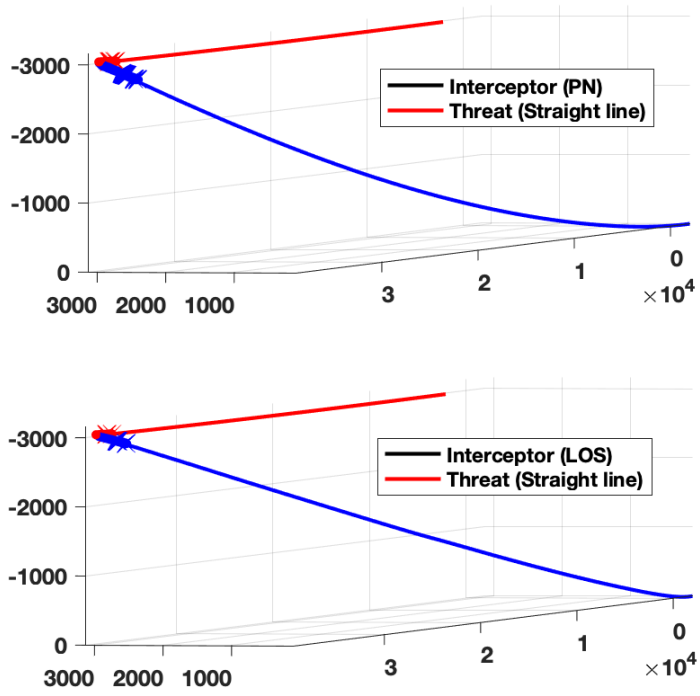

Figure 8.8: Interceptor following a threat moving on a straight line. The LOS law is calculating the optimal trajectory towards the point of interception, that is on a striaght line from the launch platform. See "StraightLine_LOS.avi" and "StraightLine_PN.avi" in .zip file for video of simulation.

8.4.2 Case 1: Threat moving on a straight line

In the first case, the threat is simulated to move on a straight line, such that the acceleration input to the threat becomes $\boldsymbol{a_c} = [0 \quad 0 \quad 0]$. This is a predictale trajectory, and the LOS guidance should therefore be able to predict the point of interception well.

8.4.3 Case 2: Threat doing sine wave

In the second case, the threat is experiencing a sine wave input in the yaw channel $\boldsymbol{a_c} = [0 \quad 0 \quad a_\psi]$ where $a_\psi = 1.5\sin(2\pi t)$. This is a more unpredictable trajectory, as (6.40) gives inaccurate estimations when the velocity is rapidly changing. This will cause the interceptor to constantly alter its trajectory. Figure 8.9 shows how the interceptor behaves using two different guidance laws. Note that the PN law is able to guide the interceptor on a straighter path towards the threat than the LOS law, which compliance the reasoning above.

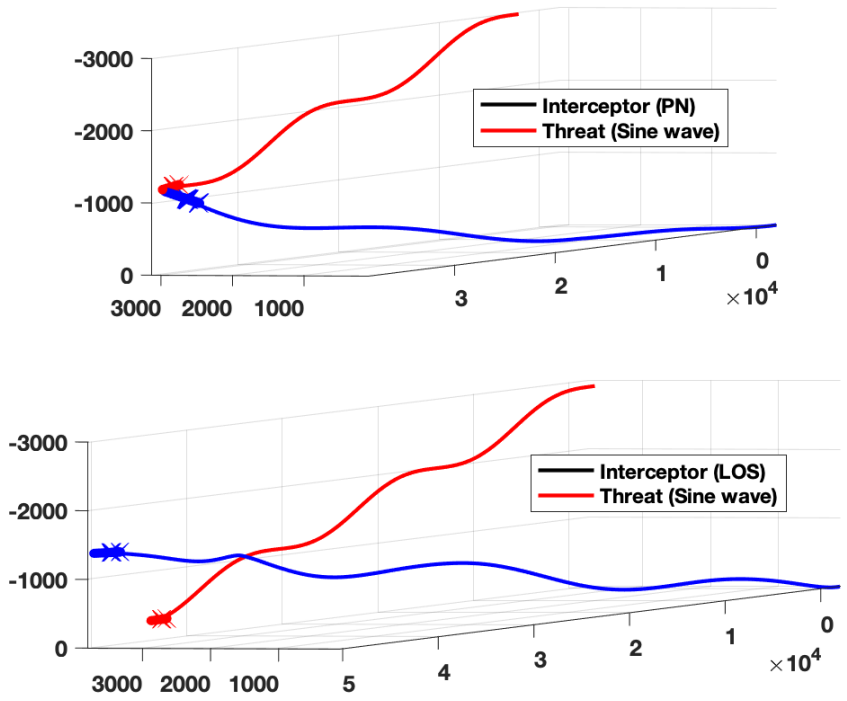

Figure 8.9: Interceptor following a threat doing the sine wave maneuver. The LOS law is not doing well when the threat is using an unpredictable maneuver. See ”Sine_LOS.avi” and ”Sine_PN.avi” in .zip file for video of simulation.

8.4.4 Case 3: Threat intercepting an asset

This case study is similar to the one in section 8.3, as the asset applies the PN law to intercept an asset, using the corc screw maneuver.

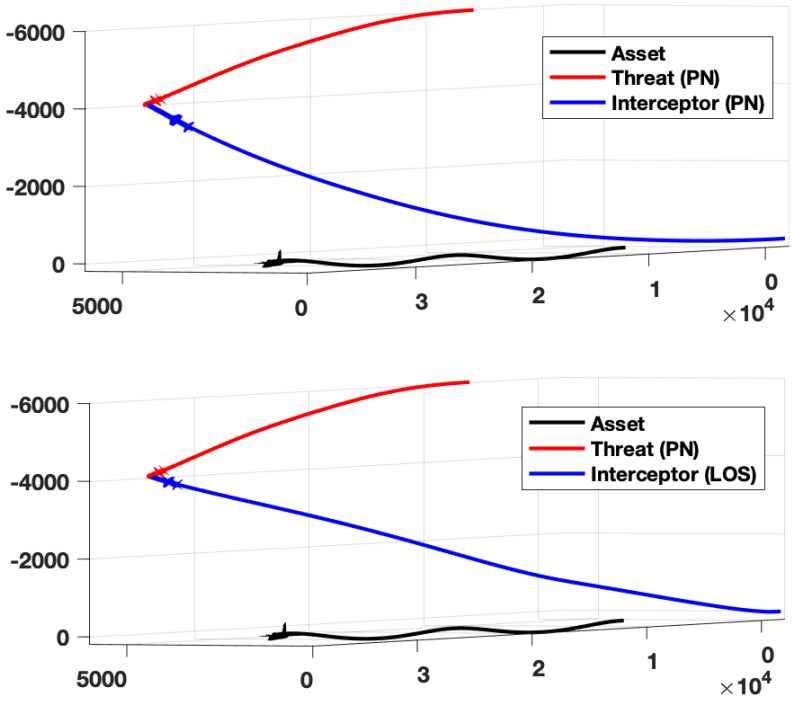

Figure 8.10: Interceptor following a threat applying the PN law to intercept an asset. Similar to the sine wave maneuver, this makes the target maneuver unpredictable. See "Intercept_LOS.avi" and "Intercept_PN.avi" in .zip file for video of simulation.

8.4.5 Simulation results

Table 8.4.5 shows simulation results for all three cases. $Time$ is the total simulation time from missile being launched till either successful or unsuccessful interception. $Force$ is the L2-norm of the summation of the specific force acting on the missile throughout simulation. $Distance$ is the final distance between the interceptor and threat. $Interception$ is a Boolean variable determining if the interception is successful or not. FSF means full state feedback such that all states are known in advance. This is synonymous to removing the INS system, and just use the missile states from the airframe as input to the guidance and control blocks shown in Figure 1.1.

Case 1, where the threat moves on a straight line, results in successful interceptions with both the LOS and PN for full state feedback, and $\sigma_r = 0.5$. This is expected, as the threat's trajectory is predictable, and easy to follow for both laws. Notice that the LOS law is able to intercept the target even when $\sigma_r = 5$. This is reasonable, as the LOS law is less sensitive to measurement noise than the PN law.

	LOS			PN		
	FSF	$\sigma_r = 0.5$	$\sigma_r = 5$	FSF	$\sigma_r = 0.5$	$\sigma_r = 5$
Case 1						
Time [s]	14.0	14.0	14.1	13.7	13.8	13.9
Force [N]	5.7e5	7.3e5	1.1e6	8.9e5	1.3e6	2.2e6
Distance [m]	5.3	5.0	4.4	5.3	4.2	8.7
Interception	✓	✓	✓	✓	✓	✗
Case 2						
Time [s]	17.8	17.8	17.8	13.6	13.6	13.6
Force [N]	3.6e6	3.4e6	3.7e6	1.4e6	1.7e6	2.3e6
Distance [m]	10.0	9.8.0	11.7	5.4	4.6	5.8
Interception	✗	✗	✗	✓	✓	✓
Case 3						
Time [s]	13.4	13.4	13.4	13.5	13.5	13.6
Force [N]	1.1e6	1.3e6	1.4e6	2.1e6	2.4e6	2.9e6
Distance [m]	11.5	12.1	11.4	4.8	2.6	8.3
Interception	✗	✗	✗	✓	✓	✗

Table 8.2: Interception time, accumulated specific force and interception information for different simulation cases using PN and LOS guidance laws.

Case 2, where the threat moves in a sine wave, shows bad performance for the LOS law. The LOS law computes its trajectory by predicting the threat's position at the closest point of approach. Since the threat has a highly time-varying attitude caused by the sine wave, this position will vary a lot. This can be seen in Figure 8.9, as the interceptor's trajectory for LOS results in a wave with significantly higher amplitude than for the PN law. From Table 8.4.5, both the simulation time, force and final distance is notably higher for LOS than PN.

Case 3, where the threat tries to intercept an asset, shows similar simulation time results for both methods, but the final distance is generally larger for LOS than PN. Note that for both this case and for the straight line tracking presented in *Case 1*, the specific force norm is much higher for PN than LOS.

Chapter 9

Conclusion and further work

9.1 Conclusion

Different GNC designs for a missile-target engagement have been implemented and tested:

- Three-loop autopilot with PN guidance law.
- Course / flight-path angle autopilot with LOS guidance law.
- Quaternion based MEKF with GNSS aiding.
- KF differentiator used to estimate MEKF measurement reference, compared to another approach using pseudo-measurements from the INS.
- Kalman Filter for target tracking.
- Three-body simulation enviroment including 6-DOF models of asset, threat and interceptor.

Simulations has been carried out using different scenarios, to observe the effect on state estimations, as well as the overall performance of the two GNC systems. The three-body simulation enviroment using high-fidelity models is used to produce more realistic results, where both the threat and interceptor have specific objectives.

The simulation results shows promising results for missile state-estimations, even with sample frequencies as low as 2 Hz. The KF differentiator for $\boldsymbol{f}_{ins}$ reference calculation results in fast bias estimations, while the pseudo-measurement method, which uses yaw angle estimation, shows better accuracy with an final error as low as $9.3\text{e}{-3}$ for the simulation in section 5.7. Both methods result in simulations where the interceptor is able to complete the mission objective successfully.

The two different guidance and control systems have been tested and compared. The calculated trajectories are slightly different for the two methods, and their strengths and weaknesses are shown to be dependent of the threat's trajectory.

The LOS guidance law shows good performance when the threat moves in a predictable way, such as on a straight line. As the LOS guidance uses the predicted threat's position at CPA, straight line maneouvers result in the missile following a seemingly optimal trajectory. Since the LOS guidance law uses course and flight-path angle commands, it is shown to have problems intercepting a threat conducting evasive or unpredictable movements right before interception, even though it is able to guide the missile to a point close to the threat. The LOS law is also shown to have a lower magnitude of specific force acting on the missile. This is due to the fact that the missile estimates the threat's position, and that the guidance commands do not involve rapid changes in attitude.

The PN law shows good overall performance when tracking down and intercepting the target, both for straight-line tracking, sine wave tracking and for the three-body scenario. However, the PN law has reduced performance when the relative position estimate for the target-tracking filter has a high magnitude. Simulations shows that a relative position standard deviation of $\sigma_r = 5$ m results in unsuccessfull interception for the straight-line case and the three-body case, when the requirement for interception is $R \leq 6$ m. By increasing the standard deviation of the relative position noise, the PN law also shows exponential growth in total specific force acting on the rigid body. This problem does not seem to be present for the LOS guidance law.

9.2 Further Work

In this thesis, the course and flight-path-angle controllers are calculated by using a linearized low-fidelity model about a fixed operating point. The controller performance may therefore be improved by using Gain scheduling, such that the controller gains are calculated for different operating points in the flight envelope, saved in i.e. a look-up table. This approach is used for the three-loop autopilot, implemented by the students under supervision of Professor Murat Arcak.

Section 8.4 shows that even though the PN law is better at achieving successful interceptions, both laws are able to compute a trajectory that brings the interceptor close to the target. Therefore, it may be interesting to combine the two guidance laws by utilizing a switch at a predefined distance or time-to-go estimate from the target. This may result in a more optimal trajectory than when using each guidance law separately.

Bibliography

Balchen, J. G., Andresen, T., Foss, B. A., 2004. Reguleringsteknikk. NTNU, Institutt for teknisk kybernetikk.

Beard, R. W., McLain, T. W., 2013. Small Unmanned Aircraft: Theory and Practice Randal W. Beard and Timothy W. McLain , Princeton Univ. Press , Princeton, NJ. No. 1. PRINCETON UNIVERSITY PRESS.

Bryne, T. H., Fossen, T. I., aug 2016. Introductory Lecture Notes on Aided Inertial Navigation Systems. Tech. rep., NTNU, Trondheim.

Bryson, A. E. J., 2015. Control of Spacecraft and Aircraft. Vol. 3. PRINCETON UNIVERSITY PRESS.

Çimen, T., 2011. A generic approach to missile autopilot design using state-dependent nonlinear control. IFAC Proceedings Volumes (IFAC-PapersOnline) 18 (PART 1), 9587–9600.

Crassidis, J. L., Markley, F. L., Cheng, Y., 2007. Survey of Nonlinear Attitude Estimation Methods. Journal of Guidance, Control, and Dynamics 30 (1), 12–28.
URL http://arc.aiaa.org/doi/10.2514/1.22452

Fossen, T. I., 2011. Handbook of Marine Craft Hydrodynamics and Motion Control. John Wiley & Sons Ltd.

Horton, M. P., 1995. Autopilots for tactical missiles: an overview. Proceedings of the Institution of Mechanical Engineers. Part I, Journal of systems and control engineering 209 (2), 127–139.

Jay, F., 2008. Aided Navigation: GPS with High Rate Sensors. The McGraw-Hill Companies.
URL http://www.amazon.com/Aided-Navigation-High-Rate-Sensors/dp/0071493298

Maley, J. M., 2013. quaternion multiplicative extended kalman filter for nonspinning guided projectiles. Army Research Labratory, ARL 2016-Augus (July), 8043–8048.

Markley, F. L., 2008. Attitude Error Representations for Kalman Filtering. Journal of Guidance, Control, and Dynamics 26 (2), 311–317.

M.Innocenti, D.Fragopoulos, 2004. Stability considerations in quaternion attitude control using discontinuous Lyapunov functions. Control Theory & Applications Iee Proceedings.

Mracek, C., Ridgely, D., 2005. Missile Longitudinal Autopilots: Comparison of Multiple Three Loop Topologies. AIAA Guidance, Navigation, and Control Conference and Exhibit.

NPTEL, 2012. Missile Guidance Laws, 100–109.

ONR, 2001. Broad Agency Announcement (BAA) for the Office of Naval Research (ONR) Navy and Marine Corps FY2018 Basic Research Challenge (BRC) Program.

Palumbo, N. F., 2010. Guest Editor's Introduction: Homing Missile Guidance and Control. Johns Hopkins APL Technical Digest (Applied Physics Laboratory) 29 (1), 2–8.

Palumbo, N. F., Blauwkamp, R. A., Lloyd, J. M., 2010a. Basic Principles of Homing Guidance, 25–41.

Palumbo, N. F., Blauwkamp, R. A., Lloyd, J. M., 2010b. Modern homing missile guidance theory and techniques. Johns Hopkins APL Technical Digest (Applied Physics Laboratory) 29 (1), 42–59.

Palumbo, N. F., Harrison, G. A., Blauwkamp, R. A., Marquart, J. K., 2010c. Guidance filter fundamentals. Johns Hopkins APL Technical Digest (Applied Physics Laboratory) 29 (1), 60–70.

Qi, H., Moore, J. B., 2002. Direct Kalman filtering approach for GPS/INS integration. IEEE Transactions on Aerospace and Electronic Systems 38 (2), 687–693.

Riley, D., 2003. CAD2MATDEMO.M.

Salih, A. A. A.-a., Liyana, N., Che, A., Zaini, A., Zhahir, A., 2013. The Suitability of GPS Receivers Update Rates for Navigation Applications 7 (6), 1012–1019.

Scordamaglia, V., 2016. Trajectory and Attitude Plot Version 3.

Sefastsson, U., 2016. Evaluation of Missile Guidance and Autopilot through a 6 DOF Simulation Mode. Ph.D. thesis, KTH ROYAL INSTITUTE OF TECHNOLOGY.

Siouris, G. M., 2004. Missile Guidance and Control Systems. Applied Mechanics Reviews 57 (6), B32.
URL http://appliedmechanicsreviews.asmedigitalcollection.asme.org/article.aspx?articleid=1398173

Solà, J., 2017. Quaternion kinematics for the error-state Kalman filter. CoRR, vol. abs/1711.02508, 2017. [Online].
URL http://arxiv.org/abs/1711.02508

Trimble, 2012. Serial Embedded GPS Receiver (SEGR) Solution for RF and DAE Embedded Applications.

UTC Aerospace Systems, 2017. TITAN ® Tactical Grade Inertial Measurement Unit (IMU) Product Benefi ts.
URL www.utcaerospacesystems.com/gnc

Weehong Tan, Packard, A., Balas, G., 2000. Quasi-LPV modeling and LPV control of a generic missile. Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334) (June), 3692–3696 vol.5.
URL http://ieeexplore.ieee.org/document/879259/

Zhang, C., Li, X., Gao, S., Lin, T., Wang, L., 2017. Performance analysis of global navigation satellite system signal acquisition aided by different grade inertial navigation system under highly dynamic conditions. Sensors (Switzerland) 17 (5).

Appendix A

Simulink Diagrams

This appendix includes some of the most important parts of the Simulink environment. For the full system, see the attached .zip file.

The following colors are used to distinguish between different block types:

- Subsystems are colored light blue.
- Embedded Matlab functions are colored orange.
- Stop condition subsystems are gray.
- ”In” and ”From workspace” are colored Cyan.
- ”Out” is colored Magenta.
- ”To workspace” is colored yellow.
- The remaning blocks are white.

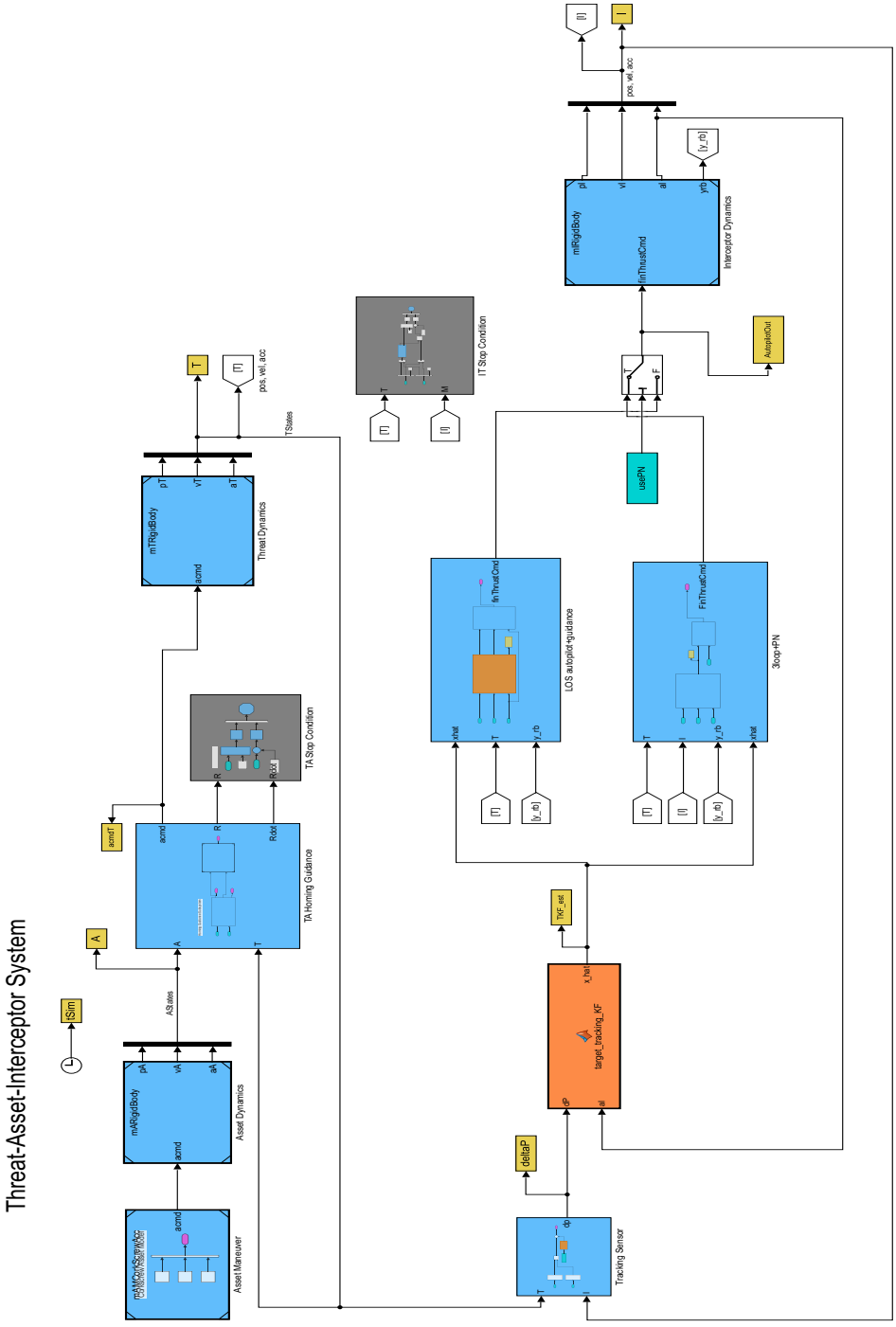

Figure 9.1: Threat-Asset-Interceptor Simulink System.

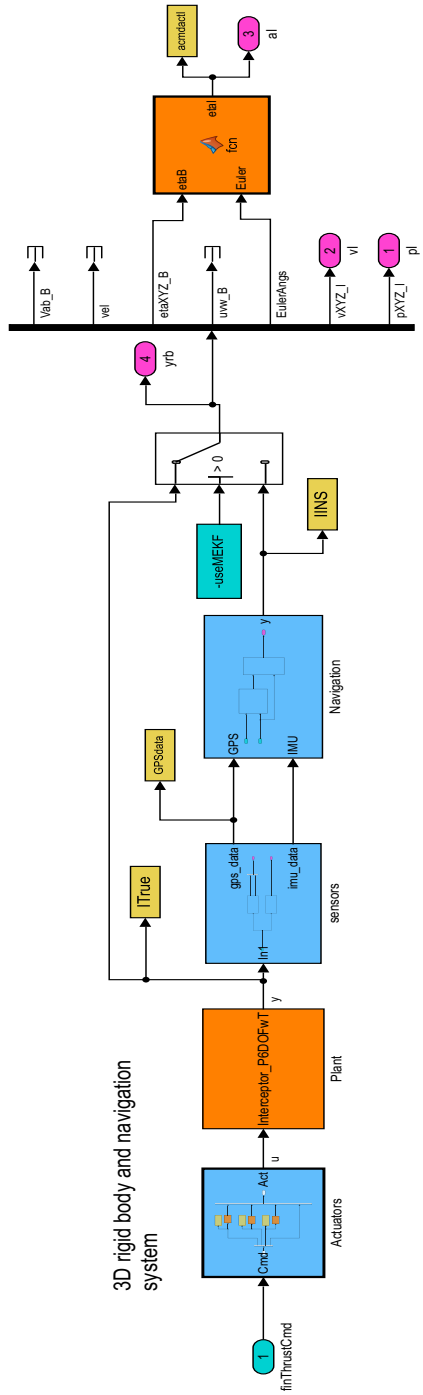

Figure 9.2: Rigid body, sensor and Navigation system.

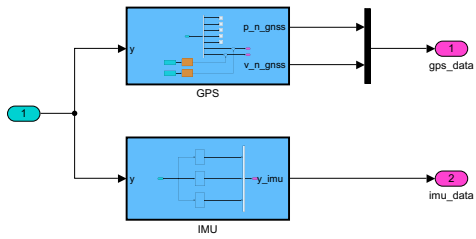

Figure 9.3: Sensor model

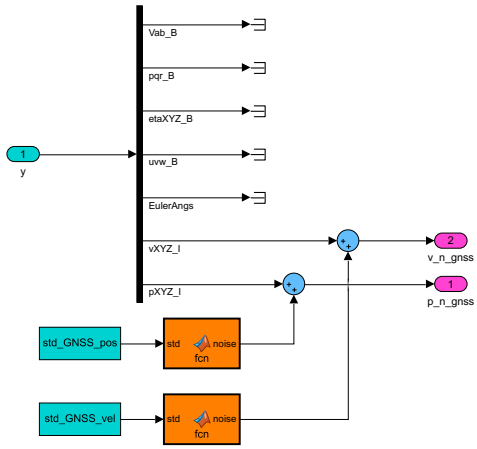

Figure 9.4: GPS model

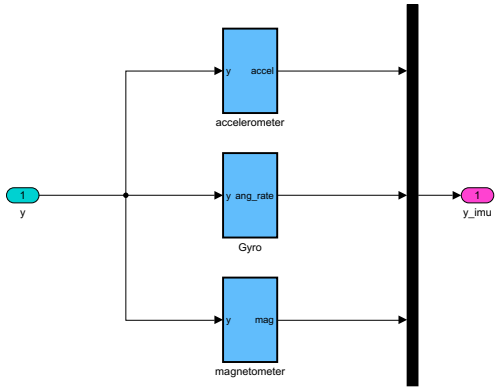

Figure 9.5: IMU model

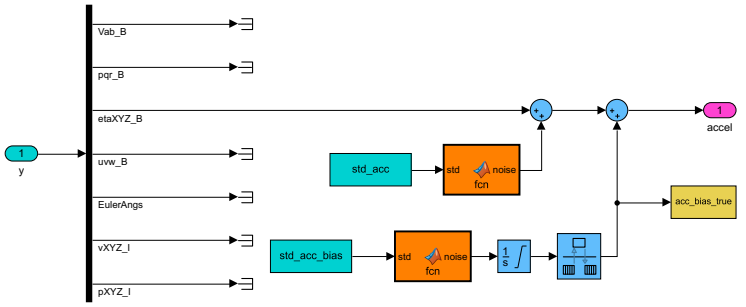

Figure 9.6: Accelerometer model

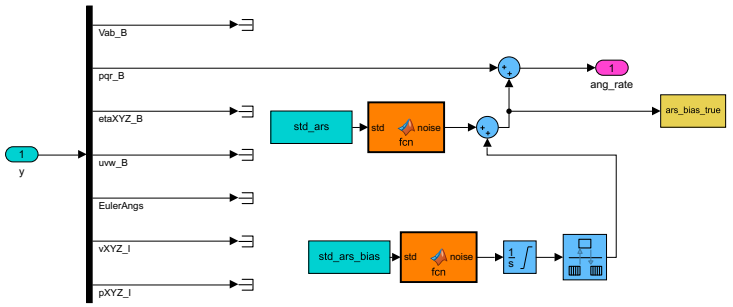

Figure 9.7: Gyro model

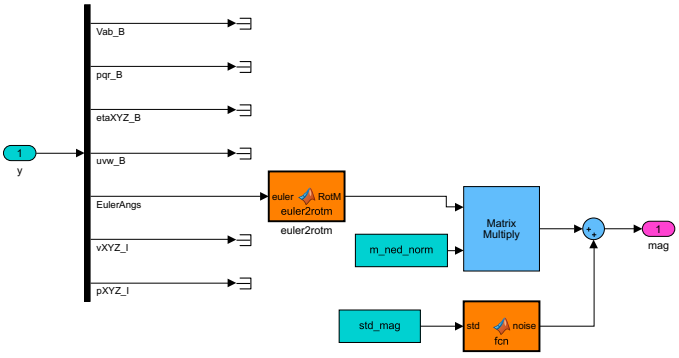

Figure 9.8: Magnetometer model

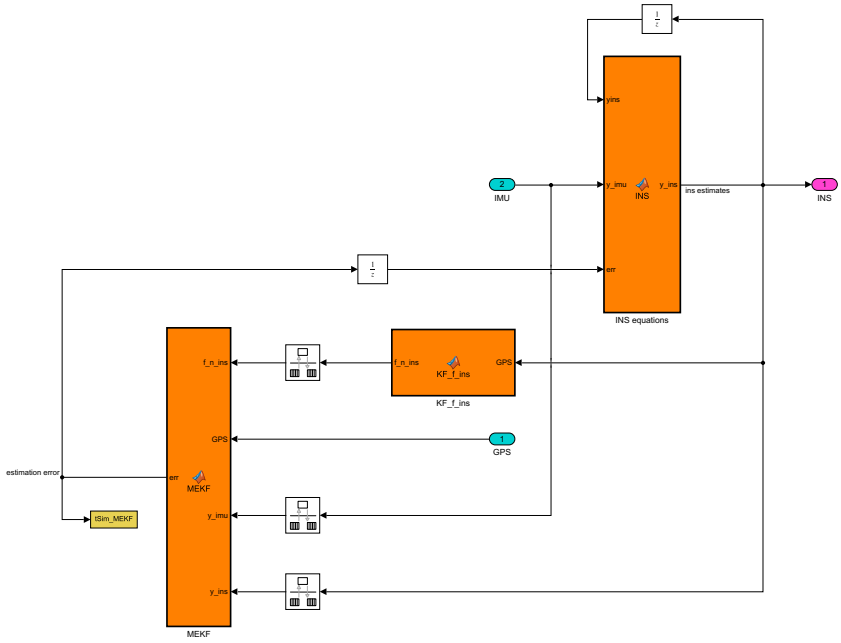

Figure 9.9: INS system

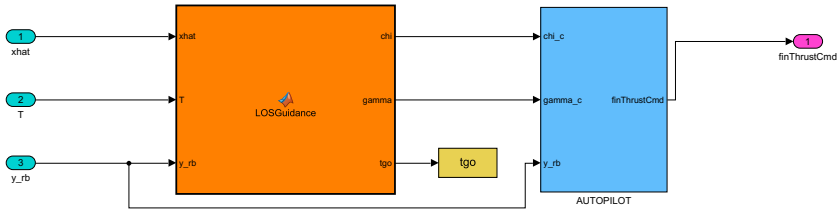

Figure 9.10: LOS guidance and course / flight-path-angle autopilot.

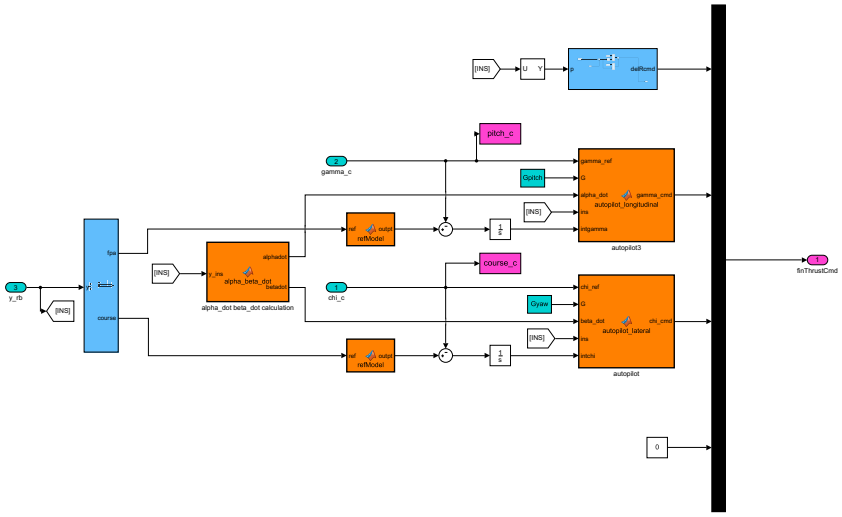

Figure 9.11: LOS autopilot.

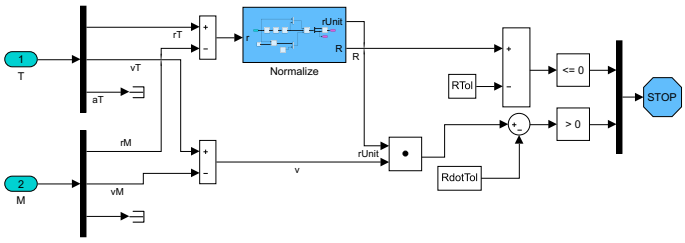

Figure 9.12: Stop condition (Implementation by Professor Arcaks students, UCB).

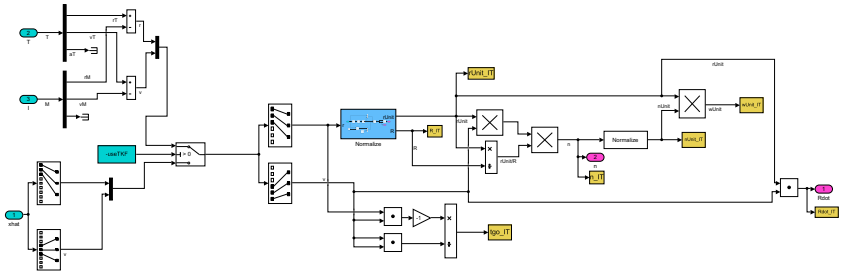

Figure 9.13: Missile relative kinematics (Implementation by Professor Arcaks students, UCB).

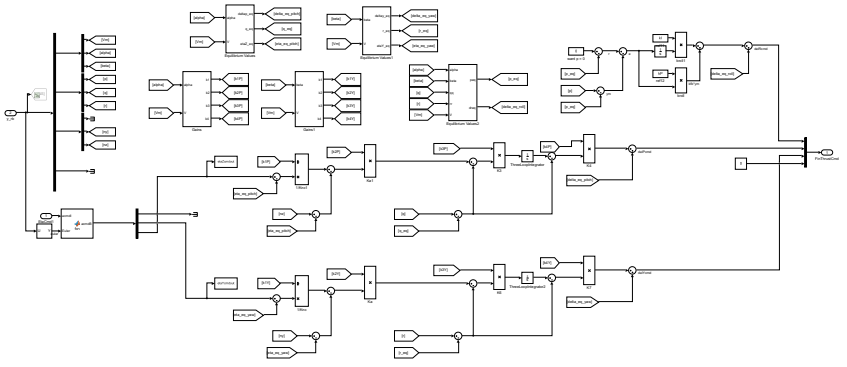

Figure 9.14: Three-loop autopilot (Implementation by Professor Arcaks students, UCB).

Appendix B

MATLAB code

This appendix contains some of the MATLAB code used in this thesis. Only some of the main embedded MATLAB functions used in the Simulink environment are attached.

An overview of all the Matlab simulation files and their descriptions are found in Table 9.1. "UCB" in the Author column means that the code is made by the Professor Arcaks students at UC Berkeley.

The Asterisk "*", is used when there are several files with similar names and functionality. **Example:** mTM*.slx refers to several different threat maneuvers, for example the corc screw maneuver (mTMCorkScrewAcc.slx) and the straight line maneuver (mTMConstantVel.slx).

Table 9.1: MATLAB / Simulink files and descriptions.

Filename	Description	Author
DRIVER.m	Main file to run simulations.	Ward/UCB
mTop.slx	Main file for Simulink environment.	Ward/UCB
mTopParams.m	Parameter file.	Ward/UCB
MEKF.m	Multiplicative extended Kalman filter.	Ward
KF_f_ins.m	Kalman filter for $\boldsymbol{f}_{ins}$ estimation.	Ward
INS_equations.m	Inertial Navigation System equations.	Ward
autopilot_longitudinal.m	Longitudinal autopilot (pitch).	Ward
autopilot_lateral.m	Lateral autopilot (course).	Ward
LOSGuidance.m	LOS guidance law algorithm.	Ward
eps2euler.m	Returns the Euler angles from $\boldsymbol{\epsilon}$.	Ward
linearizations.m	Linearization of the missile model.	Ward
Yaw_analysis.m	Stability analysis of linearized yaw dynamics.	Ward
euler2rotm.m	Returns a rotation matrix with Euler angles as input.	Ward
randNoise.m	Returns gaussian noise with std as input.	Ward
alpha_beta_dot.m	Calculates $\dot{\alpha}/\dot{\beta}$ from INS data.	Ward
LQR_control.m	Computes the optimal feedback gains based on a linear state-space model.	Ward
y_rb_calc.m	Calculates the airframe variables vector $\boldsymbol{y}_{rb}$.	Ward
LinStateSpace_yaw.m	Returns the linear state space representation in yaw, for a given $K_{\dot{\beta}}$.	Ward
LinStateSpace_pitch.m	Returns the linear state space representation in pitch, for a given $K_{\dot{\alpha}}$.	Ward
refModel.m	Third order reference model.	Ward

Continued on next page

Table 9.1 – continued from previous page

Filename	Description	Author
eps2q.m	Build quaternion from ϵ.	Ward
*GPNLaw.slx	PN guidance law for interceptor, threat andasset.	Ward/UCB
ExtractResults.m	Extract data from "To Workspace" vectors.	Ward/UCB
ProcessResults.m	Store data in structures.	Ward/UCB
PlotResults.m	Returns all the plots and saves simulation video.	Ward/UCB
m*RigidBody.slx	Simulink files for rigid bodies and navigation system.	Ward/UCB
Interceptor_P6DOFwT.m	6DOF rigid body model for Interceptor.	Ward/UCB
load_mdlrefs.m	Loads all referenced models.	UCB
set_param_mdlrefs.m	Does set_param to all the referenced model.	UCB
ModelOptions.m	Option file for maneuver, dynamics and guidance laws.	UCB
mTM*.slx	Simulink files for different threat maneuvers.	UCB
mAM*.slx	Simulink files for different asset maneuvers.	UCB
m*PointMass.slx	Double integrator point mass model.	UCB
Threat_P6DOFwT.m	6DOF rigid body model for threat.	UCB
Asset_P6DOFwT.m	6DOF rigid body model for Asset.	UCB

MEKF

```
function err = MEKF(f_n_ins, GPS, y_imu, y_ins, MEKFinit)
%
% Multiplicate extended Kalman filter
%
% Input:
% f_n_ins      specific force reference from KF differentiator
% GPS          GPS measurements
% y_imu        IMU sensor measurements (accelerometer, gyro, magnetometer)
% y_ins        measurements from INS
% q_ins        quaternion measurement
% b_ins_ars    bias estimation for gyro
% w_b_nb       angular velocity
% h            sampling rate
%
% Output:
% err          error state from MEKF, injection term for INS
%
% Notes:
% comment/uncomment f_b_ins2 and the 4th line in H matrix to switch
    between
% method 1 and 2 for f_b_ins reference computation
%

% Author:   Henning Ward
% Date:     June 2019
%

%%

Z3 = zeros(3);
I3 = eye(3);
err = zeros(16, 1);
h = MEKFinit.h;

m_ned       =  [22494.35 5372.67 42301.72]'; %magnetic field (UC Berkeley
    , USA) [nT]
    persistent R Q P_hat


    if isempty(R)
        R = MEKFinit.R;
        Q = MEKFinit.Q;
        P_hat = MEKFinit.P_hat;

    end

    Tars = MEKFinit.Tars;
    Tacc = MEKFinit.Tacc;

    eps_ins     = y_ins(7:9);
    q_ins       = eps2q(eps_ins);
    %q_ins normalization to prevent numerical errors
    q_ins       = q_ins/norm(q_ins);
    R_ins       = Rquat(q_ins);
```

```
53
54
55      p_n_nb       = GPS(1:3);
56      v_n_nb       = GPS(4:6);
57      f_b_imu      = y_imu(1:3);
58      m_b_imu      = y_imu(7:9)/norm(y_imu(7:9));
59      y            = [p_n_nb; v_n_nb; f_b_imu; m_b_imu];
60
61      p_ins        = y_ins(1:3);
62      v_ins        = y_ins(4:6);
63
64      b_ins_ars = y_ins(10:12);
65      b_ins_acc = y_ins(13:15);
66
67      w_b_nb       = y_imu(4:6);
68
69      f_b_ins      = R_ins' * f_n_ins;
70
71      %uncomment for method 2
72      %f_b_ins      = Smtrx(w_b_nb   b_ins_ars) * R_ins' * v_ins;
73
74      m_b_ins      = R_ins' * (m_ned/norm(m_ned));
75
76
77      y_ins        = [p_ins; v_ins; f_b_ins; m_b_ins];
78
79      ag_param = 2; %gibbs parametrization
80      a_g = ag_param * q_ins(2:4) / q_ins(1);
81
82      A =      [Z3 I3 Z3 Z3 Z3
83                Z3 Z3  R_ins * Smtrx(f_b_imu   b_ins_acc) Z3  R_ins
84                Z3 Z3  Smtrx(w_b_nb   b_ins_ars)  I3 Z3
85                Z3 Z3 Z3  I3/Tars Z3
86                Z3 Z3 Z3 Z3  I3/Tacc];
87
88      E =     [ Z3 Z3 Z3 Z3
89                R_ins Z3 Z3 Z3 %w_acc
90                Z3  I3 Z3 Z3   %w_ars
91                Z3 Z3 I3 Z3    %ars_bias (noise)
92                Z3 Z3 Z3 I3];  %acc_bias (noise
93
94
95      H = [I3 Z3 Z3 Z3 Z3
96           Z3 I3 Z3 Z3 Z3
97           Z3 Z3 Smtrx(f_b_ins) Z3 Z3
98           Z3 Z3 Smtrx(m_b_ins) Z3 Z3];
99
100
101     %uncomment for method 2
102     %     H = [I3 Z3 Z3 Z3 Z3
103     %     Z3 I3 Z3 Z3 Z3
104     %     Z3 Z3 Smtrx(Smtrx(w_b_nb   b_ins_ars)* R_ins' * v_n_nb) Z3 Z3
105     %     Z3 Z3 Smtrx(m_b_ins) Z3 Z3];
106
107     % Discrete time model
108     Ad = eye(15) + h * A;
109     Ed = h * E;
```

```
110
111     dy = y    y_ins;
112
113
114     %% KF update
115     % KF gain
116     K = P_hat * H' / (H * P_hat * H' + R);
117
118     % new x_hat
119     x_est = K * dy;
120     %
121     p_est = x_est(1:3);
122     v_est = x_est(4:6);
123     ag_est = x_est(7:9);
124     q_est = 1/sqrt(ag_param^2 + ag_est'*ag_est)*[ag_param ag_est'];
125
126     b_ins_ars_est = x_est(10:12);
127     b_ins_acc_est = x_est(13:15);
128     %
129
130     % Covariance update
131     P_hat = (eye(15) K*H) * P_hat * (eye(15)    K*H)' + K*R*K';
132     P_hat = (P_hat + P_hat')/2;
133
134
135     % Covariance predictor (k+1)
136     P_hat = Ad * P_hat * Ad' + Ed * Q * Ed';
137
138
139 err = [p_est; v_est; q_est; b_ins_ars_est; b_ins_acc_est];
```

Listing 9.1: MEKF.m

INS equations

```
function y_ins = INS_equations(yins,y_imu, err, INSinit)
%
% Inertial Navigation System (INS) equations
%
% Input:
% yins          INS estimates
% y_imu         IMU measurements
% err           error from MEKF
% y_ins         measurements from INS
% INSinit       INS initialization
%
% Output:
% y_ins         INS estimates
%
% Author:    Henning Ward
% Date:      May 2019
%

%%

v_b = zeros(3, 1);
y_init = zeros(15, 1);
yins = yins + y_init;
err_init = zeros(16, 1);
err = err + err_init;
h = INSinit.h;

persistent ins_init prevErr

%initialization
if isempty(ins_init)

    newMeasurement = false;
    ins_init = yins;
    v_b = ins_init(4:6);
    err = zeros(16, 1);
    err(7) = 1;
    yins = INSinit.x_ins;
    yins(10:15) = 0.000001;
    prevErr = err;

else
    newMeasurement = false;
  if err(7) < 0.001
        err(7) = 1; %quaternion initialization q_0 = [1 0 0 0]
    end

    if (prevErr ~= err)
        newMeasurement = true;
    end
    prevErr = err;

    f_b_imu = y_imu(1:3);
    w_b_nb = y_imu(4:6);
```

```
55
56      xdot_ins = zeros(15, 1);
57
58      p_ins        = yins(1:3);
59      v_ins        = yins(4:6);
60      eps_ins      = yins(7:9);
61      q_ins        = eps2q(eps_ins);
62      %q_ins normalization to prevent numerical errors
63      q_ins        = q_ins/norm(q_ins);
64      b_ins_ars    = yins(10:12);
65      b_ins_acc    = yins(13:15);
66      R_ins        = Rquat(q_ins);
67
68
69      f_n_imu = R_ins * f_b_imu;
70
71
72      if (newMeasurement)
73      %% Move error and reset
74      p_ins = p_ins + err(1:3);
75      v_ins = v_ins + err(4:6);
76      q_ins = quatmultiply(q_ins', err(7:10)')';
77      q_ins = q_ins/norm(q_ins);
78      b_ins_ars = b_ins_ars + err(11:13);
79      b_ins_acc = b_ins_acc + err(14:16);
80      yins = [p_ins; v_ins; q_ins(2:4); b_ins_ars; b_ins_acc];
81      end
82
83      %% Strapdown INS equations
84      xdot_ins(1:3) = yins(4:6);
85      xdot_ins(4:6) = f_n_imu    R_ins * b_ins_acc;
86      qins_dot = 0.5 * quatmultiply(q_ins', [0; w_b_nb    b_ins_ars]');
87      xdot_ins(7:9) = qins_dot(2:4);
88      xdot_ins(10:12) = 0;
89      xdot_ins(13:15) = 0;
90
91      yins = yins + h * xdot_ins;
92  end
93
94  y_ins = yins;
```

Listing 9.2: INS_equations.m

f_{ins} estimation

```
function f_n_ins  = KF_f_ins(GPS, KFfinit)
%
% KF differentiator for f_ins reference calculation
%
% Input:
% GPS          GPS measurements
% KFinit       Init parameters
%
% Output:
% f_n_ins      f_ins estimation
%
%
% Author:    Henning Ward
% Date:      May 2019
%

%%

m2feet  = 1/0.3048;
persistent R Q P_hat I3 Z3 xhat g

f_n_ins = zeros(3, 1); %memory allocation

    p_meas        = GPS(1:3);
    v_meas        = GPS(4:6);
    h = KFfinit.h;

    %initialization
    if isempty(R)
        Z3 = zeros(3);
        I3 = eye(3);
        g =  9.81*m2feet;
        xhat = KFfinit.xhat;

        P_hat        = KFfinit.P_hat;
        Q            = KFfinit.Q;
        R =      KFfinit.R;

    else



        % Error model
        A =     [Z3 I3 Z3
                 Z3 Z3 I3
                 Z3 Z3 Z3];

        B = [zeros(1, 5) ones(1, 1) zeros(1, 3)]';

        E =  [   I3 Z3 Z3
                 Z3 I3 Z3
                 Z3 Z3 I3];

        C = [I3 Z3 Z3
```

```
55              Z3  I3  Z3];
56  %%
57
58          % Discrete time model
59          Ad = eye(9) + h * A;
60          Ed = h * E;
61
62          % Measurements
63          y = [p_meas; v_meas];
64
65          % KF gain
66          K = P_hat * C' / (C * P_hat * C' + R);
67
68          % corrector
69          xhat = xhat + K * (y   C * xhat);
70          P_hat = (eye(9) K*C) * P_hat * (eye(9)   K*C)' + K*R*K';
71          P_hat = (P_hat + P_hat')/2;
72
73          xhat = Ad * xhat + h * B * g;
74
75          P_hat = Ad * P_hat * Ad' + Ed * Q * Ed';
76
77      end
78      f_n_ins = xhat(7:9);
```

Listing 9.3: KF_f_ins.m

Target tracking KF

```
function x_hat = target_tracking_KF(dP, aI, TKF)
%
% KF for target tracking
%
% Input:
% dp            relative position measurement
% aI            Interceptor acceleration measurement from INS
% TKF           TKF init
% T             Threat / Target states
%
% Output:
% x_hat         Threat / Target estimated states
%
% Notes:
% Because of high values for acceleration, the position estimates tends to
% drift if sample time is too low.
%
% Author:    Henning Ward
% Date:      May 2019
%

%%

h = TKF.h;
Z3 = zeros(3);
I3 = eye(3);

persistent R Q P_hat xhat

    % initialization
    if isempty(R)
        xhat = TKF.xhat;
        P_hat = TKF.P_hat;
        Q = TKF.Q;
        R = TKF.R;

    else

        % Error model
        A =     [Z3 I3 Z3
                 Z3 Z3 I3
                 Z3 Z3 Z3];

        B = [zeros(3, 3)  eye(3) zeros(3, 3)]';

        E =  [ Z3 Z3 I3]';

        C = [I3 Z3 Z3];

        % Discrete time model
        Ad = eye(9) + h * A;
        Ad(1, 3) = 0.5*h^2;
        Ed = h * E;

```

```
55          % Measurements
56          y = dP;
57
58          % KF gain
59          K = P_hat * C' / (C * P_hat * C' + R);
60
61          % corrector
62          xhat = xhat + K * (y    C * xhat);
63          P_hat = (eye(9) K*C) * P_hat * (eye(9)    K*C)' + K*R*K';
64          P_hat = (P_hat + P_hat')/2;
65
66          xhat = Ad * xhat + h * B * aI;
67
68          P_hat = Ad * P_hat * Ad' + Ed * Q * Ed';
69
70      end
71
72  x_hat = xhat;
```

Listing 9.4: target_tracking_KF.m

LOS guidance law

```
function [chi, gamma, tgo] = LOSGuidance(xhat, T, y_rb)
%
% LOS guidance law
%
% Input:
% xhat          Interceptor / Threat relative state estimates
% T             Threat position, velocity and acceleration
% y_rb          Rigid body states
%
% Output:
% chi           Course angle
% gamma         flight path angle
% tgo           time to go
%
% Author:    Henning Ward
% Date:      May 2019
%

%%

persistent initfunc

%initialization
if isempty(initfunc)
    initfunc = true;
    chi = 0;
    gamma = 0;
    tgo = 0;
else

    %position threat
    pT = T(1:3);
    vT = T(4:6);

    %position interceptor
    pI = y_rb(19:21);
    vI = y_rb(16:18);

    %position Launcher
    pL = [0 0 0]';

    pR = xhat(1:3);
    vR = xhat(4:6);
    tgo =  (pR' * vR) / (vR' * vR);

    vT = vI + vR;
    pT = pT + vT * tgo;

    %distance between launch platform and interceptor
    RLI = norm(pI   pL);

    %angle between horizontal plane and interceptor
    gammaNI = asin(pI(3)/RLI);

```

```
    %angle between NORTH and interceptor
    chiNI = atan2(pI(2)   pL(2), pI(1)   pL(1));
    %angEM = atan2(pM(2)   pL(2), pM(1)   pL(1));

    %distance between launch platform and threat
    RLT = norm(pT   pL);

    %angle between horizontal plane and threat
    gammaNT = asin((pT(3)  pL(3)) /RLT);

    %angle between NORTH and threat
    chiNT = atan2(pT(2)   pL(2), pT(1)   pL(1));
    %theta = atan2(pT(2)   pL(2), pT(1)   pL(1));

    %% COURSE  (horizontal)
    %angle between interceptor and threat horizontal plane
    thetah    = chiNT   chiNI;
    %distance between launch platform and interceptor in horizontal plane
    RLIh    = cos(gammaNI) * RLI;
    eh      = sin(thetah) * RLIh;
    rh       = sqrt(RLIh^2  eh^2);
    Rith    = cos(gammaNI) * abs(norm(pT(1:2) pI(1:2)));
    kh = 0.9;

    ahdh = sqrt(abs(Rith^2  eh^2));
    disth = rh + kh * ahdh;
    xc = cos(chiNT) * disth;
    yc = sin(chiNT) * disth;
    chi = atan2(yc   pI(2), xc   pI(1));

    %% AoA (vertical, North/ Down)
    aITv    = gammaNI   gammaNT;
    %distance between launch platform and interceptor in verical plane
    RLIv = cos(chiNI) * RLI;
    ev      = sin(aITv) * RLIv;
    rv       = sqrt(RLIv^2  ev^2);
    Ritv    = abs(norm(pT(1:2:3) pI(1:2:3)));
    kv = 0.9;

    ahdv = sqrt(abs(Ritv^2  ev^2));
    distv = rv + kv * ahdv;
    zc = sin(gammaNT) * distv;
    gamma =  atan2(zc   pI(3), xc   pI(1));
end
```

Listing 9.5: LOSGuidance.m