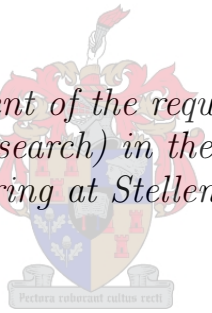
Optimal estimation and sensor selection for autonomous landing of a helicopter on a ship deck

by

Shaun George Irwin



Thesis presented in fulfilment of the requirements for the degree of Master of Engineering (Research) in the Faculty of Electrical and Electronic Engineering at Stellenbosch University

Supervisor: Prof. Thomas Jones

December 2014

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

September 2014

Abstract

This thesis presents a complete state estimation framework for landing an unmanned helicopter on a ship deck. In order to design and simulate an optimal state estimator, realistic sensor models are required. Selected inertial, absolute and relative sensors are modeled based on extensive data analysis. The short-listed relative sensors include monocular vision, stereo vision and laser-based sensors.

A state estimation framework is developed to fuse available helicopter estimates, ship estimates and relative measurements. The estimation structure is shown to be both optimal, as it minimises variance on the estimates, and flexible, as it allows for varying degrees of ship deck instrumentation. Deck instrumentation permitted ranges from a fully instrumented deck, equipped with an inertial measurement unit and differential GPS, to a completely uninstrumented ship deck. Optimal estimates of all helicopter, relative and ship states necessary for the autonomous landing on the ship deck are provided by the estimator. Active gyro bias estimation is incorporated into the helicopter's attitude estimator. In addition, the process and measurement noise covariance matrices are derived from sensor noise analysis, rather than conventional tuning methods.

A full performance analysis of the estimator is then conducted. The optimal relative sensor combination is determined through Monte Carlo simulation. Results show that the choice of sensors is primarily dependent on the desired hover height during the ship motion prediction stage. For a low hover height, monocular vision is sufficient. For greater altitudes, a combination of monocular vision and a scanning laser beam greatly improves relative and ship state estimation. A communication link between helicopter and ship is not required for landing, but is advised for added accuracy.

The estimator is implemented on a microprocessor running real-time Linux. The successful performance of the system is demonstrated through hardware-in-the-loop and actual flight testing.

Opsomming

Hierdie tesis bied 'n volledige sensorfusie- en posisieskattingstruktuur om 'n onbemande helikopter op 'n skeepsdek te laat land. Die ontwerp van 'n optimale posisieskatter vereis die ontwikkeling van realistiese sensormodelle ten einde die skatter akkuraat te simuleer. Die gekose inersie-, absolute en relatiewe sensors in hierdie tesis is op grond van uitvoerige dataontleding getipeer, wat eenoogvisie-, stereovisie- en lasergegronde sensors ingesluit het.

'n Innoverende raamwerk vir die skatting van relatiewe en skeepsposisie is ontwikkel om die beskikbare helikopterskattings, skeepskattings en relatiewe metings te kombineer. Die skattingstruktuur blyk optimaal te wees in die beperking van skattingsvariansie, en is terselfdertyd buigsaam aangesien dit vir wisselende mates van skeepsdekinstrumentasie voorsiening maak. Die toegelate vlakke van dekinstrumentasie wissel van 'n volledig geïnstrumenteerde dek wat met 'n inersiemetingseenheid en 'n differensiële globale posisioneringstelsel (GPS) toegerus is, tot 'n algeheel ongeïnstrumenteerde dek. Die skatter voorsien optimale skattings van alle vereiste helikopter-, relatiewe en skeepsposisies vir die doeleinde van outonome landing op die skeepsdek. Aktiewe giro-sydige skatting is by die posisieskatter van die helikopter ingesluit. Die proses- en metingsmatrikse vir geruiskovariansie in die helikopterskatter is met behulp van 'n ontleding van sensorgeruis, eerder as gebruiklike instemmingsmetodes, afgelei.

'n Volledige werkingsontleding is daarna op die skatter uitgevoer. Die optimale relatiewe sensorkombinasie vir landing op 'n skeepsdek is met Monte Carlo-simulasie bepaal. Die resultate toon dat die keuse van sensors hoofsaaklik van die gewenste sweefhanghoogte gedurende die voorspellingstadium van skeepsbeweging afhang. Vir 'n lae sweefhanghoogte is eenoogvisie-sensors voldoende. Vir hoër hoogtes het 'n kombinasie van eenoogvisie-sensors en 'n aftaslaserbundel 'n groot verbetering in relatiewe en skeepsposisieskatting teweeggebring. 'n Kommunikasieskakel tussen helikopter en skip is nie 'n vereiste vir landing nie, maar word wel aanbeveel vir ekstra akkuraatheid.

Die skatter is op 'n mikroverwerker met intydse Linux in werking gestel. Die suk-

sesvolle werking van die stelsel is deur middel van hardeware-geïntegreerde simulasie en werklike vlugtoetse aangetoon.

Acknowledgements

I would like to express my sincere gratitude to the following:

- Armscor, the Defence Research and Development Board and the National Research Foundation for their financial assistance of this research.
- Prof Jones for his unbounded enthusiasm and invaluable insight throughout this project and for managing to survive an onslaught of relentless and grueling questions. All of the nuggets of wisdom are much appreciated!
- Dr Corné van Daalen for his advice regarding the estimation structure.
- Past and present lab engineers, in particular Wiaan Beeton and Nico Alberts, as well as Phil Bellstedt, for their assistance in flight testing and help in troubleshooting many hardware-related issues.
- Michael Basson, the safety pilot, for his expert handling of Bessie in rather gusty conditions.
- Craig Robinson from Google Street View for providing advice on automatic differentiation and observability testing.
- Anouk Albien for proof reading this rather lengthy document.
- All of my friends inside and outside the lab for all the healthy (and continual) distractions and for providing many laughs and good memories.
- My family for all the motivation and support they have provided over the course of my Master's and the many years beforehand.

Contents

List of Figures

List of Tables

Nomenclature

Notational Convention

$\boldsymbol{x}$	True value of $\boldsymbol{x}$
$\hat{\boldsymbol{x}}$	Estimated value of $\boldsymbol{x}$
$\tilde{\boldsymbol{x}}$	Measured value of $\boldsymbol{x}$
$\bar{\boldsymbol{x}}$	Weighted mean of measurements of $\boldsymbol{x}$
$\hat{\mathbf{x}}_{k\|k-1}$	A priori state estimate
$\hat{\mathbf{x}}_{k\|k}$	A posteriori state estimate
$\mathbf{I}_{nxn}$	n by n identity matrix
$\mathbf{0}_{nxn}$	n by n matrix of zeros
$\mathbf{DCM}_{\boldsymbol{\theta}_{deck}}$	Direction cosine matrix corresponding to Euler angles $\boldsymbol{\theta}_{deck}$. Transforms from inertial to body reference frame.
$\mathbf{DCM}^{T}_{\boldsymbol{\theta}_{deck}}$	Inverse DCM corresponding to Euler angles $\boldsymbol{\theta}_{deck}$. Transforms from body to inertial reference frame.
$(\mathbf{DCM}_{\boldsymbol{\theta}_{deck}})_{i,j}$	Element of the DCM matrix at the i^{th} row and j^{th} column

Subscripts

heli	Helicopter reference frame
ship	Ship reference frame
deck	Ship deck reference frame
mono	Monocular vision sensor reference frame
stereo	Stereo vision sensor reference frame

rel	Relative reference frame
$sensor$	Sensor reference frame
cam	Camera reference frame
$laser$	Laser sensor reference frame
$offset$	Offset of a relative sensor from the relative reference frame

Symbols

$\boldsymbol{\theta}$	Attitude expressed using Euler 3-2-1 convention
ϕ	Roll angle
θ	Pitch angle
ψ	Yaw angle
$\sigma^2(\tau)$	Allan variance
τ	Window period of a signal
τ_{gm}	Correlation time of a Gauss-Markov process
ω_{gm}	Gaussian distributed white noise
$\mathbf{\Phi}$	Discrete state transition matrix
$\mathbf{\Gamma}$	Discrete input matrix
$\mathbf{C}$	Output matrix
$\mathbf{g}$	Gravity vector
$\mathbf{J}$	Jacobian
$\mathbf{K}$	Kalman gains
$\mathbf{N}$	A diagonal covariance matrix corresponding to noise sources
$\mathbf{O}$	Observability matrix
$\boldsymbol{p}$	NED position of vehicle
$\mathbf{P}$	State error covariance matrix
$\mathbf{Q}$	Process noise covariance matrix
$\mathbf{R}$	Measurement noise covariance matrix
$S_\Omega(f)$	Power spectral density of signal Ω
$\mathbf{u}$	Driving input of Kalman filter
$\mathbf{v}$	Measurement noise
$\mathbf{w}$	Process noise

Accronyms

AD	Automatic differentiation
ARW	Angular random walk
ATOL	Automated take-off and landing
ATV	All-terrain vehicle
CAN	Controller Area Network
CEP	Circular error probability
CG	Center-of-gravity
COM	Computer-on-module
CTP	Conventional terrestrial pole
DCM	Direction cosine matrix
DGPS	Differential GPS
ECEF	Earth centered, earth fixed
EKF	Extended Kalman filter
ESL	Electronic Systems Laboratory
KF	Kalman filter
FIR	Finite impulse response filter
FFT	Fast Fourier Transform
FPGA	Field-programmable gate array
GPS	Global Positioning System
Gyro	Gyroscope
HIL	Hardware-in-the-loop
HRF	Helderberg Radio Fliers Club
IDE	Integrated development environment
IIR	Infinite impulse response filter
IMU	Inertial measurement unit
LPF	Low-pass filter
MEMS	Microelectromechanical system
MRW	Measurement random walk
MSE	Mean-square error
NED	North-East-Down
OBC	Onboard computer
PC	Personal computer
PCB	Printed circuit board

PSD	Power spectral density
RMSE	Root-mean-square error
RTK	Real-time kinematic
RRR	Rate random run
RRW	Rate random walk
SEP	Spherical error probability
SLAM	Simultaneous Localisation and Mapping
SVD	Singular value decomposition
UAV	unmanned aerial vehicle
UDP	User Datagram Protocol
UKF	Unscented Kalman filter
VRW	Velocity random walk

Chapter 1

Introduction

1.1 Automated Take-off and Landing

In recent years unmanned aerial vehicles (UAVs) have found widespread military use. Although commonplace in land-based military operations, UAVs have begun to emerge in naval scenarios too. In 2006 the US Navy's Fire Scout became the first UAV to land autonomously on a navy ship (refer to Figure 1.1a). More recently, in 2013, Northrop Grumman's X-47B became the first autonomous aircraft to perform an arrested landing upon an aircraft carrier (refer to Figure 1.1b).

(a) MQ-8C Fire Scout UAV landing upon a United States frigate. Photo obtained from [1].

(b) X-47B landing upon an aircraft carrier. Photo obtained from [2].

Figure 1.1: Historic landings of UAVs at sea.

Rough seas, turbulent weather and rapidly changing airflow at the rear of the ship contribute to the difficulty of the take-off and landing of an aircraft on a ship deck [3]. The landing deck, in most ships, is located at the stern of the ship. The ship rotates around a pivot point located towards the center of the vessel. In heavy swell this can result in the landing deck undergoing enormous and rapid heaves [3]. Data

captured from a South African Navy Corvette reveals that the deck can heave up to 9 meters at a maximum rate of 3.5 meters per second in moderately rough seas [4]. In addition, the ship was observed to roll up to 7 degrees at a maximum rate of 4.5 degrees per second [4].

The estimation of both the aircraft and relative states is required in order to land a UAV successfully on a ship deck. Motion prediction of the ship deck is required to enable the safe landing of the aircraft. Therefore, estimation of the ship states is also required. These sensors can either be onboard the aircraft, on the ship deck or distributed between the two.

This project investigates the sensory requirements needed to land an unmanned autonomous helicopter on a ship's deck. An optimal state estimator, capable of estimating all the states necessary for landing, is created. The estimator is implemented, simulated and flight tested on a radio controlled helicopter.

1.2 Related Research in the Stellenbosch University Electronics Systems Lab

Stellenbosch University's Electronic Systems Laboratory (ESL) conducts research on the autonomous control and navigation of vehicles. This includes fixed and rotary-wing aircraft, quadbikes, submarines and satellites. Several core research groups exist in the ESL that focus on specific long term objectives. The Autonomous Take-Off and Landing (ATOL) research group seeks to develop control systems for the landing of unmanned aeroplanes, helicopters and quadcopters on a ship deck. A summary of the ATOL research that has been conducted so far, depicted in Figure 1.2, follows.

In 2003 Carstens [5] initiated unmanned helicopter research in the ESL. This resulted in the development of a vehicle-independent kinematic state estimator and control system capable of flying an electrically-powered helicopter autonomously. The complementary filter [6] and Kahn-Hudson filter[1] were investigated as alternative state estimator forms before opting for the simpler kinematic-based estimator. Satisfactory results were found in using this estimator for slow, stable, near-hover flight. Two rate gyroscopes and GPS velocity measurements were used to estimate roll, pitch and heading. The estimator, however, was constrained to operate in near-hover attitudes and required a velocity sufficiently large to enable the GPS to estimate the heading. Due to limited processing power, estimation and control were performed offboard and actuator commands were transmitted via a radio link to the vehicle.

[1] Unpublished, but fully described by Carstens [5].

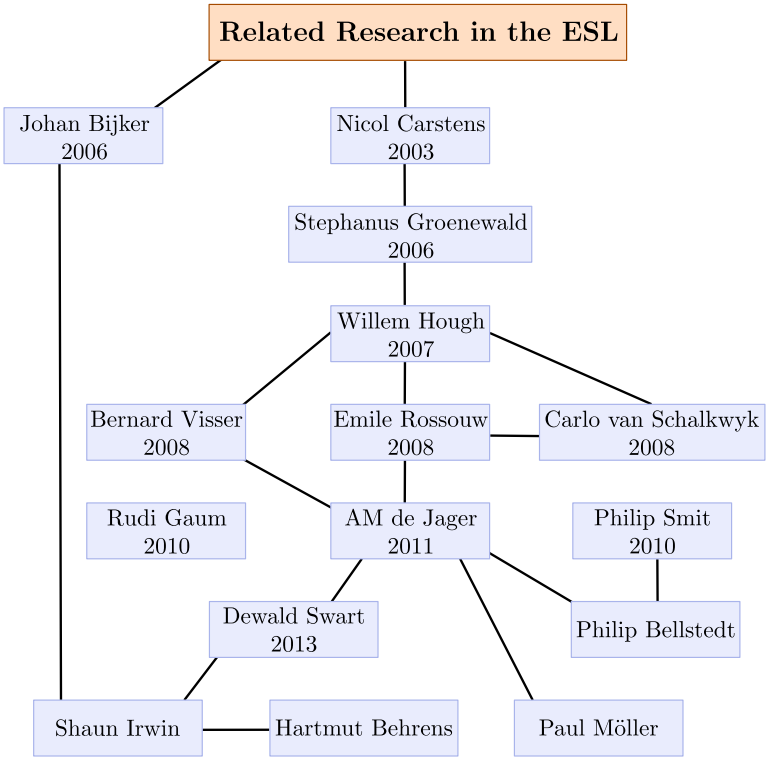


Figure 1.2: Past related research in the ESL.

Research was continued on rotary-wing UAVs by Groenewald [7]. The X-Cell methanol-powered radio controlled helicopter was selected to replace the electric helicopter used by Carstens [5]. This was in order to ensure a significantly larger payload capacity. A Pentium III PC/104-based computer running Linux was used to provide onboard computing. A Controller Area Network (CAN) bus was introduced to enable communication between sensors, servo-motors, the onboard computer and other modules.

Although not part of the ATOL research within the ESL, Bijker [8] developed a quaternion-based estimator to estimate airship states. Allan variance, a technique discussed in Chapter 3, was used to characterise the stochastic bahaviour of the inertial sensors and magnetometer.

In 2007, Hough [9] developed a quaternion-based kinematic estimator by using the TRIAD algorithm to measure the attitude of the helicopter. This permitted full state estimation regardless of the vehicle's attitude and velocity. Three-axis accelerometers, gyroscopes and magnetometers, as well as single-point GPS were used in a strap-down configuration. Translational and rotational vehicle states were estimated in separate linear and extended Kalman filters, respectively. This is the form of the estimator currently used within the ESL. Minor changes have since been imple-

mented. The rate gyroscopes exhibited significant drift, although implementation of bias estimation was left for future research.

Visser [10] pioneered vision-based estimation for precision landing of an unmanned fixed-wing aircraft. Infra-red markers located on the runway were used to provide accurate measurement of the aircraft's position during its final approach until touchdown.

The monocular vision-based approach used by Visser [10] was incorporated by de Jager [11] to enable the landing of a helicopter autonomously on a stationary platform. Five infra-red markers were placed on the landing platform in a specifically chosen asymmetric configuration to allow calculation of the relative pose of the camera. The system was successfully demonstrated in hardware-in-the-loop simulation. The quaternion-based estimator developed by [9] was modified. This allowed for attitude states to be estimated in Euler angles for simplicity. It was deemed that the flight envelope of the helicopter would avoid singularities. Position and attitude measurements obtained from the monocular vision system were used to update the estimator states.

Swart [3] expanded on the monocular vision-based approach used by [11] to enable the helicopter to land on a rolling, pitching and heaving platform. Unlike [11], the vision measurements were used to measure the landing platform states relative to the helicopter, rather than improving the helicopter's state estimates. This was necessary as the assumption of a stationary landing surface was no longer valid. [3] placed a concentric square pattern on the landing platform and determined the relative pose of the camera using singular value decomposition (refer to Section 4.2). This allowed the helicopter to obtain accurate pose estimates several meters above the landing platform, at which point the entire pattern is in view. In addition, the helicopter could also obtain accurate pose estimates just above the platform, when only the innermost square is visible to the camera. Motion prediction of a ship deck's heave state was also demonstrated in simulation. The first completely autonomous landing of the ESL helicopter on a stationary surface was successfully demonstrated in a flight test. A photograph of the helicopter hovering above the landing surface is shown in Figure 1.3.

In 2013, Möller demonstrated the first landing of a UAV on a translating platform. The SLaDe II quadcopter, developed in the ESL, successfully achieved a fully autonomous landing on a trailer moving horizontally at 30 meters per second. The Novatel GPS receiver's ALIGN mode was used to provide highly accurate relative position and velocity measurements of the quadcopter and the landing platform.

Figure 1.3: Autonomous landing of the X-Cell helicopter using monocular vision. Photo obtained from [3].

A new avionics system, a replacement for the aging PIC-based onboard computer used in all the ESL vehicles, is being developed by Hartmut Behrens. Sensors, servo-boards, onboard computer and other modules communicate with one another via Ethernet. Onboard processing is performed using a real-time Linux operating system running on a Gumstix microcontroller. The system is designed to be modular and backwards compatible with the existing avionics system using a CAN to Ethernet converter.

In addition to these studies, a significant contribution has been made by laboratory engineers. In particular, Gaum drove much of the helicopter work during 2009 to 2010.

1.3 Problem Statement

The landing of an unmanned helicopter on a ship deck requires knowledge of the helicopter, relative and ship states.

A successful autonomous landing on a stationary surface using a monocular vision sensor was already demonstrated by Swart [3]. The relative roll, pitch and height measurements from a monocular vision sensor significantly degrade in accuracy the further away the helicopter hovers above the deck. The measurements of these states are crucial to the safe landing of the helicopter. This thesis investigates the possibility of using alternative relative sensors and combinations thereof, to ensure sufficiently accurate estimates of these states. This is in order to determine the minimal set of sensors required to autonomously land an unmanned helicopter on a ship deck.

The estimating of the states for a UAV to land on a ship is currently performed by sensors placed on the ship and transmitting commands to the UAV [12; 13]. This approach simplifies the sensory requirements of the UAV. The UAV, however, can only land on ships instrumented with the required sensors. Moreover, in a military environment the UAV is vulnerable to communications failures that could potentially prevent the UAV from being able to land. Complex electronic warfare countermeasures would be required to minimise the susceptibility to communication failure and jamming.

The primary objectives of this thesis are to develop a state estimation framework and to determine the minimal degree of instrumentation required for the ship deck. The possibility of an uninstrumented deck will be examined. Practical implementation of the chosen system, however, is outside of the scope of this thesis.

The X-Cell helicopter used by the ESL is already equipped with the necessary sensors and estimator to estimate its own states. The gains of this estimator, however, have been determined through trial-and-error based on simulation results using basic sensor models. Accurate models of the sensor noise are created in order to build a more realistic simulation testbed of the estimator.

The process noise of the kinematic estimator used in the ESL essentially contains measurement noise from the inertial sensors. The process noise covariance matrices will be directly determined through analysis of the sensor noise. This means that no tuning of the Kalman noise covariance matrices is required, leading to the possibility of the Kalman gains being determined more optimally for the given sensors.

Bias estimation is also introduced in the helicopter estimator to reduce the impact of slowly changing biases that are synonymous with relatively low-cost MEMS gyroscopes.

1.4 Contributions

The following contributions were made by this thesis to ATOL research at Stellenbosch University:

- Realistic simulation models of the sensors currently in use in the ESL were developed. Models of potentially useful relative sensors, including stereo vision, single and multiple laser rangefinders, were also created.
- A flexible framework for estimation of all the states necessary for landing autonomously on a ship deck is presented and implemented. This estimator

enables various sensor configurations to be used. Varying degrees of ship instrumentation are supported by this unified estimation framework.

- The estimation framework was analysed for the cases of fully instrumented, partially instrumented and uninstrumented decks. For each case, several sets of sensor configurations were compared.
- A modified version of the previous ESL estimator was implemented. It features active gyroscope bias estimation and the Kalman noise matrices are determined directly from sensor noise analysis.
- The helicopter estimator was implemented on a Gumstix microcomputer running a real-time version of Linux. Hardware-in-the-loop and actual flight testing was conducted to verify the performance of the estimator.

1.5 Thesis Outline

Chapter 2 examines optimal estimation, focusing primarily on the Kalman filter. An explanation of the reference frames relevant to the task of landing an unmanned helicopter on a ship deck is given.

Chapter 3 reviews analyses of the noise characteristics of the inertial and absolute sensors currently used by the ESL helicopter. Simulation models of the sensors are developed.

Chapter 4 outlines the relative sensors available. The most useful subset of sensors are short-listed. The short-listed relative sensors are then modelled and stochastic noise models developed.

The state estimator is discussed in Chapter 5. Factors important to the design of the estimator are outlined. Thereafter, the final design of the estimator is explained, demonstrating how the proposed estimator design meets the requirements. A detailed discussion is presented on the estimation of the relative and ship states.

Chapter 6 summarises the existing helicopter state estimator. Modifications are proposed. These include the removal of biases from the gyroscope measurements and the calculation of the Kalman filter noise matrices directly from sensor models developed in Chapter 3.

Chapter 7 analyses the estimator. Performance of the gyroscope bias estimation is determined. A sensitivity analysis of biases in the helicopter attitude determination algorithms is presented. Simulations of various combinations of relative sensors are conducted. Decisions regarding the optimal set of relative sensors are presented.

The practical implementation of the proposed estimator is discussed in Chapter 8. Hardware-in-the-loop simulations are performed and flight test results are discussed.

Finally, conclusions are drawn and recommendations made in Chapter 9.

Chapter 2

Background

This chapter gives an overview of the concept of optimal state estimation, which is the core topic of this thesis. Particular emphasis will be placed on a specific form of optimal estimator, known as the Kalman filter.

To estimate the various helicopter and ship states several reference frames need to be defined. These reference frames and their mathematical interrelationships are subsequently explained.

2.1 State Estimation

Consider a physical system that can be modelled as a set of state variables $\mathbf{x}$ related by first-order differential equations with inputs $\mathbf{u}$ and outputs $\mathbf{y}$. The states of the system cannot, in general, be measured directly from the outputs. Furthermore, the system dynamics are subject to disturbances $\mathbf{w}$, which prevent the possibility of developing a prefect mathematical model of the system that tracks the internal states. A state estimator is therefore required.

The estimator contains a mathematical model of the physical system and receives the same inputs as the physical system. The outputs of the physical system are measured to provide state feedback. This prevents unmodelled disturbances from causing the estimated states $\hat{\mathbf{x}}$ to diverge from the true states. Figure 2.1 shows a block diagram of the system and the state estimator.

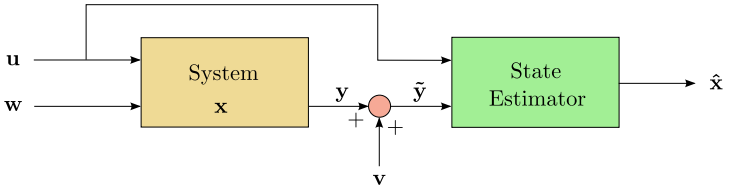


Figure 2.1: State estimator.

2.1.1 Kalman Filter

The Kalman filter is an optimal recursive estimator that estimates the states of time-varying linear systems [14; 15; 16]. The filter enables estimates of a system's states to be constructed from a series of incomplete, noisy measurements, provided a model of the system is known. As such, it is widely used in diverse applications ranging from aircraft and spacecraft navigation [17; 18; 19] to econometrics [20; 21; 22] and medicine [23; 24].

The Kalman filter will yield the exact conditional probability density of the state error, provided that the system is linear and that the process noise $\mathbf{w}_k$ and measurement noise $\mathbf{v}_k$ are Gaussian-distributed white noise. If the Gaussian assumption is removed the state error covariance will still be minimised. However, it will no longer reflect the true state error covariance [25]. A large class of practical applications meet the Gaussian assumption. Process noise is often a lumped parameter, which results in an approximately Gaussian distribution[1]. Many sensors also exhibit Gaussian noise on their outputs [21].

A discrete-time linear system can be modelled as follows:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{\Phi}_k \mathbf{x}_{k-1} + \mathbf{\Gamma}_k \mathbf{u}_{k-1} + \mathbf{w}_k \\ \mathbf{y}_k &= \mathbf{C}_k \mathbf{x}_k + \mathbf{v}_k, \end{aligned} \tag{2.1.1}$$

where $\mathbf{\Phi}_k$ is the state transition matrix, $\mathbf{\Gamma}_k$ is the input matrix and $\mathbf{C}_k$ is the output matrix at timestep k. The Kalman filter equations for such a system are separated into a control update step [28]:

$$\begin{aligned} \mathbf{P}_{k|k-1} &= \mathbf{Q}_k + \mathbf{\Phi}_k \mathbf{P}_{k-1|k-1} \mathbf{\Phi}_k^T \\ \hat{\mathbf{x}}_{k|k-1} &= \mathbf{\Phi}_k \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{\Gamma}_k \mathbf{u}_{k-1} \end{aligned} \tag{2.1.2}$$

and a measurement update step:

$$\begin{aligned} \mathbf{P}_{k|k} &= \mathbf{P}_{k|k-1} \mathbf{C}_k^T (\mathbf{R}_k + \mathbf{C}_k \mathbf{P}_{k|k-1} \mathbf{C}_k^T)^{-1} \\ \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k (\mathbf{y}_k - \mathbf{C}_k \hat{\mathbf{x}}_{k|k-1}). \end{aligned} \tag{2.1.3}$$

$\mathbf{Q}_k$ and $\mathbf{R}_k$ are the covariance matrices corresponding to the process noise and measurement noise, respectively. The control update predicts the next state of the system one timestep into the future, which results in an increase in uncertainty of the state error covariance matrix $\mathbf{P}_k$. The measurement update reduces the uncertainty.

The Kalman gains $\mathbf{K}_k$ are calculated as follows:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{C}_k^T (\mathbf{R}_k + \mathbf{C}_k \mathbf{P}_{k|k-1} \mathbf{C}_k^T)^{-1}. \tag{2.1.4}$$

[1] The central limit theorem states that the sum of several random variables featuring well-defined means and variances will be approximately Gaussian distributed [26; 27].

This choice of gains results in the minimisation of the mean-square error (MSE) of the estimates.

Nonlinear Systems

In many systems the dynamics or the measurements of the states are nonlinear [28]. In these cases the extended Kalman filter (EKF) is frequently used. First-order Taylor series expansion is used to create linear approximations of nonlinear state transition equations and measurement updates [28]. The resultant estimator resembles a standard Kalman filter, although it will be suboptimal due to the linear approximations used. The less well-behaved nonlinear systems require other forms of estimators to be used, such as the unscented Kalman filter [29; 30] and the particle filter [31; 32].

Simon [15], Maybeck [25] and Brown and Hwang [33] give a more thorough explanation of the Kalman filter and its nonlinear counterparts. These should be consulted by the interested reader.

2.2 Reference Frames

The Kalman filter described above forms part of the overall estimation process to land a helicopter on a ship deck. The complete estimator requires several reference frames to estimate the various helicopter, relative and ship states. The five primary reference frames used in the estimation process are:

1. North-East-Down (NED) reference frame
2. Relative reference frame
3. Ship (body) reference frame
4. Ship deck reference frame
5. Sensor reference frames

Figure 2.2 illustrates the relationships of these reference frames and indicates their respective axes. As can be seen in the figure, $\mathbf{p}_{heli}$, $\mathbf{p}_{ship}$ and $\mathbf{p}_{deck}$ are the location of the helicopter, ship and ship deck within the NED reference frame, respectively. The relative position of the ship deck within the relative reference frame is denoted by $\mathbf{p}_{rel}$.

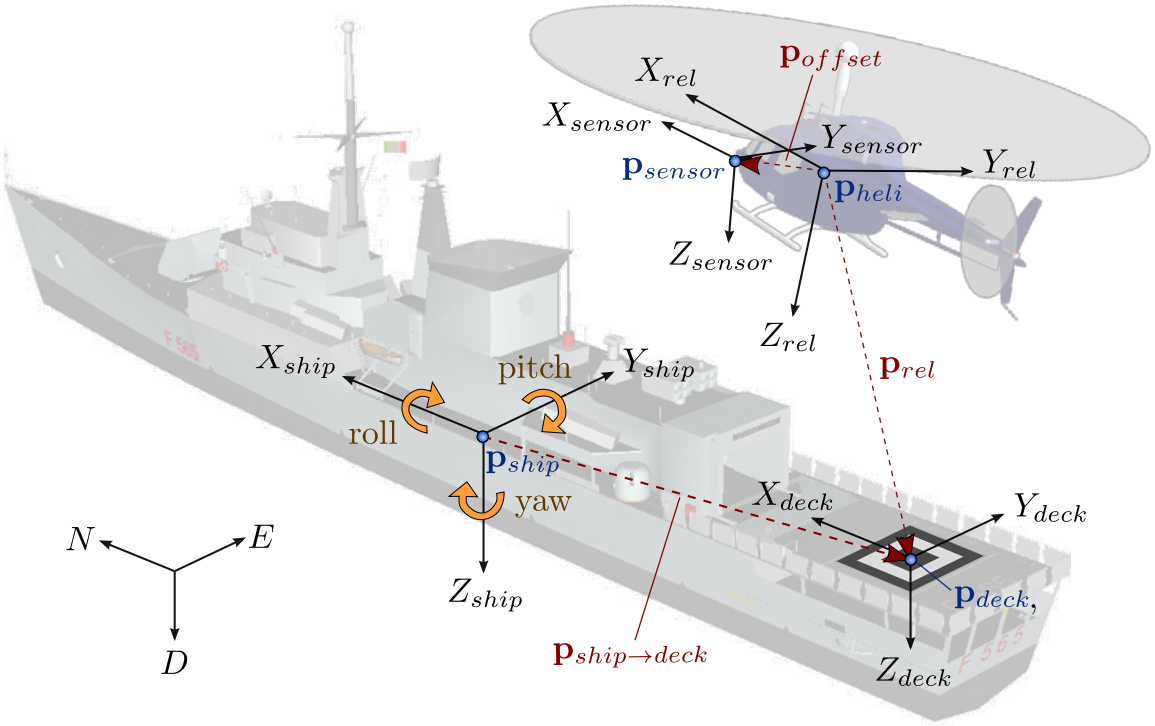


Figure 2.2: Reference frames.

2.2.1 Earth Reference Frames

The position of a helicopter or ship relative to the earth can be described using various coordinate systems. The most useful ones in terms of the research focus are outlined here. The earth centered, earth fixed (ECEF) axis system will be discussed to describe positions using latitude and longitude. The NED reference frame will be used to determine a position on the surface relative to a nearby starting location. The relationship between ECEF and NED axis systems will then be defined. It will be demonstrated that, for the purposes of this research, the NED axis system is indistinguishable from an inertial reference frame and thus can be treated as one.

ECEF Axis Systems

The ECEF rectangular axis system is a right-handed axis system whose origin is defined as the center of mass of the earth. The X-axis passes through the equator at the prime meridian and the Z-axis passes through the conventional terrestrial pole (CTP) [8].

The ECEF geocentric axis system is a more convenient way of expressing positions within the ECEF reference frame. Spherical coordinates, rather than rectangular coordinates, are used. The earth's surface is divided into lines of latitude λ and lon-

gitude ϕ. Lines of longitude pass through the North and South Poles, with positions East of 0° longitude, the prime meridian, defined as positive. 0° latitude is defined as the equator. Positions North of the equator are defined as being positive [8]. The altitude of a point $\mathbf{p}$ above the surface of the earth is given by h. The earth is approximated as a sphere with radius R. Figure 2.3 shows the relationship between the ECEF rectangular and geocentric axis systems.

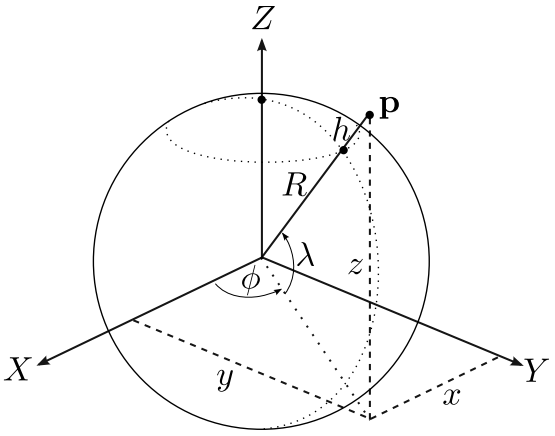


Figure 2.3: ECEF rectangular and geocentric axis systems.

ECEF geocentric coordinates can be transformed into ECEF rectangular coordinates as follows [8]:

$$\begin{aligned} x &= (R+h)\cos(\lambda)\cos(\phi) \\ y &= (R+h)\cos(\lambda)\sin(\phi) \\ z &= (R+h)\sin(\lambda). \end{aligned} \tag{2.2.1}$$

NED Reference Frame

The ECEF reference frame is useful in describing a point relative to Earth. However, to simplify the task of local navigation on the earth's surface the NED reference frame can be used. In this reference frame, the North and East axes are tangent to the surface of the geocentric axis system, with the North axis pointing to true North. The geocentric axis system approximates the earth as being round, which means that the Down axis points to the center of the earth. Figure 2.4 indicates the relationship between the ECEF and NED axis systems.

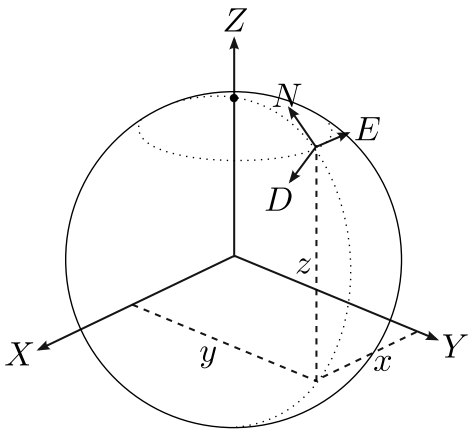


Figure 2.4: ECEF and NED reference frames.

NED coordinates can be transformed into ECEF rectangular coordinates as follows:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -\sin\lambda\cos\phi & -\sin\phi & -\cos\lambda\cos\phi \\ -\sin\lambda\sin\phi & \cos\phi & -\cos\lambda\sin\phi \\ \cos\lambda & 0 & -\sin\lambda \end{bmatrix} \begin{bmatrix} N \\ E \\ D \end{bmatrix}. \tag{2.2.2}$$

Although the NED axis system approximates the earth's surface as being flat, for the purposes of this thesis the flat earth approximation is sufficient. This may only be assumed provided that the helicopter and ship do not move far from the NED origin and that their velocities are low.

Inertial Reference Frame

The NED reference frame, described above, is not an inertial reference frame as it rotates with the earth. A highly accurate gyroscope placed at a fixed coordinate within the NED axis system would, therefore, detect the Coriolis effect. The MEMS gyroscopes used in this project are not sufficiently sensitive to measure the fictitious forces introduced by the rotating reference frame, since the Coriolis effect is far smaller than the noise levels of the sensors. The NED reference frame can, therefore, not be distinguished from an inertial reference frame using the sensors in this project. This means that Newton's laws, which are applicable only to inertial reference frames [9; 34], can be safely applied and the Coriolis effect can be ignored.

2.2.2 Relative Reference Frame

The relative reference frame is the reference frame within which measurements and estimates relative to the helicopter are taken. It's origin is located at the center

of mass of the helicopter. Its axes are fixed to the translation and rotation of the helicopter's airframe. As such, it is synonymous with the helicopter body reference frame.

The position and attitude of the ship deck relative to the helicopter are calculated as follows:

$$\begin{aligned} \mathbf{p}_{rel} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{heli}}(\mathbf{p}_{deck} - \mathbf{p}_{heli}) \\ \mathbf{DCM}_{\boldsymbol{\theta}_{rel}} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{heli}}\mathbf{DCM}_{\boldsymbol{\theta}_{deck}}. \end{aligned} \tag{2.2.3}$$

$\mathbf{DCM}_{\boldsymbol{\theta}}$ is used to denote a direction cosine matrix (DCM) corresponding to the vector of Euler angles $\boldsymbol{\theta}$.

2.2.3 Ship Reference Frame

The ship reference frame is defined similarly to the helicopter body reference frame. The axes are fixed to the translation and rotation of the ship. The importance of the location of the origin of the ship body reference frame is explained in Section 5.2.3.

The positive roll (ϕ), pitch (θ) and yaw (ψ) angle directions of the ship are shown in Figure 2.2. The roll, pitch and yaw directions are defined in the same manner for the helicopter.

2.2.4 Ship Deck Reference Frame

The ship deck reference frame has a constant offset in translation, $\mathbf{p}_{ship \to deck}$, from the ship reference frame, described above. The value of this offset is defined in Chapter 5.2.3.

The ship deck position and attitude are related to the ship position and attitude within the NED reference frame as follows:

$$\begin{aligned} \mathbf{p}_{deck} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{ship}}\mathbf{p}_{ship \to deck} + \mathbf{p}_{ship} \\ \boldsymbol{\theta}_{deck} &= \boldsymbol{\theta}_{ship}. \end{aligned} \tag{2.2.4}$$

2.2.5 Sensor Reference Frames

The ship deck's states are measured by a relative sensor within the sensor's local reference frame. The deck position and attitude within a relative sensor's reference frame is

$$\begin{aligned} \mathbf{p}_{sensor} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{offset}}\mathbf{p}_{rel} - \mathbf{p}_{offset} \\ \mathbf{DCM}_{\boldsymbol{\theta}_{sensor}} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{offset}}\mathbf{DCM}_{\boldsymbol{\theta}_{rel}}. \end{aligned} \tag{2.2.5}$$

$\mathbf{p}_{offset}$ and $\boldsymbol{\theta}_{offset}$ are the translational and rotational offsets of the sensor relative to the helicopter body reference frame, respectively.

Equations (2.2.3) and (2.2.5) can be rearranged and combined into a single set of position and attitude equations to form the equations used by Swart [3][2]:

$$\begin{aligned} \mathbf{p}_{deck} &= \mathbf{DCM}_{\boldsymbol{\theta}_{heli}}(\mathbf{p}_{sensor} + \mathbf{p}_{offset}) + \mathbf{p}_{heli} \\ \mathbf{DCM}_{\boldsymbol{\theta}_{deck}} &= \mathbf{DCM}_{\boldsymbol{\theta}_{heli}}\mathbf{DCM}_{\boldsymbol{\theta}_{offset}}\mathbf{DCM}_{\boldsymbol{\theta}_{sensor}}. \end{aligned} \tag{2.2.6}$$

These equations enable the ship deck states to be calculated from relative sensor readings. However, using Equations (2.2.3) and (2.2.5) separately, rather than in combination, is advantageous when fusing the measurements from multiple sensors, as discussed in Chapter 5.

2.3 Chapter Summary

This chapter presented an overview of the concept of optimal estimation. The Kalman filter was introduced as a focal point for the present research. It was explained that the Kalman filter provides optimal estimates when the system is linear and its process and measurement noise is composed of Gaussian distributed white noise. The extended Kalman filter, the unscented Kalman filter or the particle filter may be used instead if the system is nonlinear.

Reference frames important to the estimation of the helicopter, relative and ship states were then introduced. These consist of the NED, relative, ship, ship deck and sensor reference frames. The NED reference frame was shown to resemble an inertial reference frame in this project. This was as a result of the accelerometers and gyroscopes used being unable to distinguish the fictitious forces arising from the rotational reference frame. The equations to transform coordinates between each reference frame were subsequently given.

Thorough analyses of the sensors on board the helicopter are conducted in the following chapter. Noise models will be derived for each sensor that will be used to determine the process and measurement noise covariance matrices that were introduced in this chapter.

[2] [3] uses a different notation although the results are exactly equivalent.

Chapter 3

Inertial and Absolute Sensors

The landing of a helicopter on a ship deck requires the measurement and estimation of the helicopter's states, the states of the ship deck relative to the helicopter, and the states of the ship. In order to achieve this, a suite of sensors is needed. Previous ESL projects [5; 7] have investigated a set of sensors for estimation of the inertial states of the helicopter. This set of sensors includes three-axis gyroscopes, accelerometers, magnetometers and a GPS. As this set of sensors has been tested and proven in flight, no changes need to be made to this set.

A perfect measurement, however, does not exist in reality. All sensors exhibit some form of corruption of the true signal being measured. This corruption manifests in two distinct forms of error: systematic errors and stochastic errors. Systematic errors take the form of deterministic biases that can be removed from the measurements through calibration. In contrast, stochastic errors are random disturbances that can, at best, be modelled by their statistical distributions.

Due to the presence of these errors, sensors need to be calibrated and accurate sensor models developed in order to develop an accurate aircraft navigation system. This section provides an overview of a widely used technique for statistical modelling of sensors, known as the Allan variance. The inertial sensors (rate gyroscope and accelerometer) and the absolute sensors (magnetometer and GPS) will be analysed and modelled using this technique.

The sensors currently used on board the ESL helicopter are listed below in Table 3.1. The ADIS16405 is a complete six degree-of-freedom inertial measurement unit (IMU) containing a three-axis gyroscope, accelerometer and magnetometer. However, the Honeywell HMC2003 magnetometer, is used instead due to its superior performance. Most vehicles in the ESL presently use the Novatel OEMV-1G. This is a high-end GPS receiver that supports single-point, differential and real-time kinematic (RTK) modes of operation, as discussed in Section 3.5.

Table 3.1: Inertial and absolute sensors on board the helicopter.

Sensor type	**Model**
Gyroscope	ADIS16405
Accelerometer	
Magnetometer	Honeywell HMC2003
GPS	Novatel OEMV-1G

3.1 Allan Variance

3.1.1 Overview

Allan variance is a technique for detecting the presence of and determining the power spectral densities of common types of measurement noise. Allan variance was originally developed as a method for determining the frequency stability of precision clocks [35; 36]. This technique was later used to characterise gyroscope and accelerometer measurement noise [37], and subsequently became the standard noise modelling technique [38].

Consider a signal Ω_k consisting of N samples that has been sampled uniformly every t_s seconds. Now subdivide Ω_k into adjacent n-sample clusters. Each cluster will have a period of $\tau = n \cdot t_s$ seconds. The standard Allan variance of Ω_k is defined as follows [8]:

$$\begin{aligned}\sigma^2(\tau) &= \frac{1}{2}\langle(\bar{\Omega}_{k+n}(\tau) - \bar{\Omega}_k(\tau))^2\rangle_\tau \\ &= \frac{1}{2(N-1)}\sum_{k=1}^{N-1}(\bar{\Omega}_{k+n}(\tau) - \bar{\Omega}_k(\tau))^2,\end{aligned} \tag{3.1.1}$$

where

$$\bar{\Omega}_k(\tau) = \frac{1}{n}\sum_{i=(k-1)n+1}^{kn}\Omega_i \tag{3.1.2}$$

is the cluster average of a τ-second portion of the signal beginning at sample k. The Allan deviation $\sigma(\tau)$, the square root of the Allan variance, can be interpreted as the standard deviation[1] for a new signal where each sample is the average of a cluster of the original signal Ω_k. Figure 3.1 illustrates the calculation of the Allan variance graphically. The red lines indicate the average value of a cluster.

For small values of τ, high-frequency measurement noise will dominate, whereas for large values of τ, low-frequency biases will dominate [8]. τ therefore represents the

[1] In the special case where the cluster consists of one sample of Ω_k, $\sigma(\tau)$ is equal to the standard deviation of Ω_k.

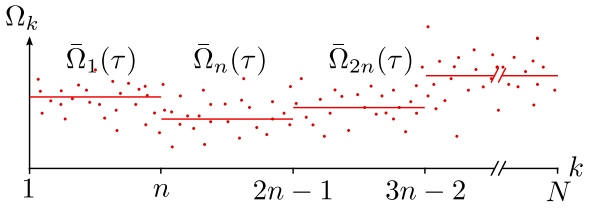


Figure 3.1: Diagram of the calculation of the simple Allan variance for Ω_k.

correlation time of the noise source.

Since Ω_k is a finite-length signal, the maximum correlation time for which the Allan variance can be calculated is limited to $n = N/2$. In this case there will only be two clusters. The statistical error in the estimate of the Allan variance for such a large correlation time will be high due to the limited data used in the calculation. The uncertainty in the calculation can be reduced by determining the Allan variance for a sliding window of length τ instead of simply subdividing the original signal. This technique is known as the overlapping Allan variance [39; 38] and will be used from here onwards. It is defined as

$$\sigma^2(\tau) = \frac{1}{2(N-2n)} \sum_{k=1}^{N-2n} (\bar{\Omega}_{k+n}(\tau) - \bar{\Omega}_k(\tau))^2. \tag{3.1.3}$$

The Allan variance is related to the power spectral density (PSD) of a signal as follows:

$$\sigma^2(\tau) = 4 \int_0^\infty S(f) \frac{\sin^4(\pi f \tau)}{(\pi f \tau)^2} \, \mathrm{d}f, \tag{3.1.4}$$

where $S(f)$ is the double-sided power spectral density (PSD) of a random process Ω_k [39; 38; 8].

When $\sigma(\tau)$ is plotted on a log-log scale, certain PSD values can be read directly from the graph by examining the slope of $\sigma(\tau)$ at various points. The most important noise types for the purposes of MEMS sensor modelling will be discussed in the following section.

3.1.2 Angular (or Velocity) Random Walk

White noise occurs in the measurements of most sensors. In the case of rate gyroscopes, the white noise is known as angular random walk (ARW), since the angular rate is integrated in order to obtain angle measurements. Similarly, white noise in the acceleration measurements is integrated to become velocity random walk (VRW). In the case of other sensors, the general term measurement random walk (MRW) may be used. The integration of the measurements of these sensors is very frequently

performed in IMUs causing the white noise to be named by the effect produced on the integrated states. The double-sided PSD of white noise is given by

$$S_\Omega(f) = N^2, \tag{3.1.5}$$

where N is the ARW or VRW coefficient. Substituting Equation (3.1.5) into Equation (3.1.4) yields the corresponding Allan variance function [38]:

$$\sigma^2(\tau) = \frac{N^2}{\tau}. \tag{3.1.6}$$

$\sigma(\tau) = \frac{N}{\sqrt{\tau}}$ will have slope of $-1/2$ on a log-log plot. As a result, the sensor's white noise can be readily distinguished and the value of N can be read directly off the graph where $\tau = 1$.

3.1.3 Bias Instability

Bias instability, also commonly referred to as flicker noise, is a flat segment of the Allan variance curve. In the case of a fully modelled inertial sensor with active bias estimation, the bias instability represents the maximum stability achievable [38].

The PSD for bias instability is

$$S_\Omega(f) = \begin{cases} \frac{B^2}{2\pi}\frac{1}{f} & \text{if } f \leq f_0 \\ 0 & \text{if } f > f_0 \end{cases} \tag{3.1.7}$$

where B is the bias instability coefficient. Substituting Equation (3.1.7) into Equation (3.1.4) yields the corresponding Allan variance function [38]:

$$\sigma^2(\tau) \approx \frac{2B^2 \ln(2)}{\pi}. \tag{3.1.8}$$

3.1.4 Rate Random Walk

The ARW is a random walk on the integrated signal. In contrast the rate random walk (RRW) is a random walk on the original signal. The PSD for RRW is

$$S_\Omega(f) = \left(\frac{K}{2\pi}\right)^2 \frac{1}{f^2}, \tag{3.1.9}$$

where K is the RRW coefficient. Substituting Equation (3.1.9) into Equation (3.1.4) yields the corresponding Allan variance function [38]:

$$\sigma^2(\tau) = \frac{K^2\tau}{3}. \tag{3.1.10}$$

$\sigma(\tau) = K\sqrt{\frac{\tau}{3}}$ will have slope of $1/2$ on a log-log plot. K can be read directly off the graph where $\tau = 3$.

3.1.5 Correlated Noise

A hump-shaped Allan variance curve is commonly exhibited by certain sensors, such as GPS receivers [40; 41]. This characteristic of the Allan variance curve is given the general term "correlated noise" [38]. First-order Gauss-Markov processes are most frequently used to model correlated noise [42]. A Gauss-Markov process features an exponential autocorrelation

$$R(T) = \sigma_{gm}^2 e^{\frac{-\tau_{gm}}{|T|}} \tag{3.1.11}$$

and is expressed in the time domain as follows:

$$\dot{b} = \frac{-b}{\tau_{gm}} + \omega_{gm}, \tag{3.1.12}$$

where τ_{gm} is the correlation time of the process. ω_{gm} represents the zero mean Gaussian-distributed white noise with covariance σ_{gm}^2 [42]. This differential equation is solved by integration:

$$\begin{aligned} b &= \int \dot{b}\, dt \\ &= \int \left(\frac{-b}{\tau_{gm}} + \omega_{gm} \right) dt. \end{aligned} \tag{3.1.13}$$

Euler integration can then be used to obtain a discrete equivalent to Equation (3.1.13):

$$b_{k+1} = \left(1 - \frac{-T_s}{\tau_{gm}} \right) b_k + T_s w_k, \tag{3.1.14}$$

where q_c is the amplitude of the correlation noise and T_s represents the sampling time of the sensor. The Allan variance of a Gauss-Markov process is given by [38; 40]

$$\sigma^2(\tau) = \frac{(q_c \tau_{gm})^2}{\tau} \left[1 - \frac{\tau_{gm}}{2\tau} \left(3 - 4e^{-\frac{\tau}{\tau_{gm}}} + e^{-\frac{2\tau}{\tau_{gm}}} \right) \right], \tag{3.1.15}$$

which forms a hump with slopes of $\pm 1/2$ on either side of the peak.

3.1.6 Other Noise Terms

The following noise terms that can also be identified using Allan variance plots are quantisation noise, sinusoidal noise and rate random run (RRR).

Quantisation noise, despite its name, is not related to the noise that results from the bit quantising that occurs at the output of a digital MEMS sensor. Instead, quantisation noise is the noise caused by time quantising at the output of rate integrating sensors [38; 43]. Therefore, it is not present in the Allan variance plot for a rate gyro or accelerometer. Bit quantisation noise, produced at the output of rate gyros and accelerometers, is not distinguishable from an Allan variance plot. Bit quantisation

is essentially a random constant[2] added to the output signal. The Allan variance is invariant to the addition of random constants to the signal, which causes the effects not to be visible on the plot [43].

Sinusoidal noise is occasionally present in MEMS sensors [43]. A single frequency sinusoidal wave has a PSD of

$$S_\Omega(f) = \frac{1}{2}\Omega_0^2 \left[\delta(f - f_0) + \delta(f + f_0)\right], \tag{3.1.16}$$

where $\delta(f)$ is the Dirac delta function, Ω_0 is the amplitude of the sinusoidal wave and f_0 is its frequency in Hertz. Substituting Equation (3.1.16) into Equation (3.1.4) yields the corresponding Allan variance function [38]:

$$\sigma^2(\tau) = \Omega_0^2 \left(\frac{\sin^2(\pi f_0 \tau)}{\pi f_0 \tau}\right)^2. \tag{3.1.17}$$

In theory, sinusoidal noise manifests as sharp descending peaks in the Allan variance plot. In practice, sinusoidal noise, if present, is generally obscured by other, more dominant noise sources [38]. Alternative techniques, such as autocorrelation, can be used to identify this noise component more reliably [43].

Rate random ramps (+1 slope of $\sigma(\tau)$) are not frequently observed on the outputs of MEMS inertial sensors. Detailed explanations for these noise types can be found in [38], [43] and [39] and are therefore not discussed here.

The sum of the Allan variances corresponding to each of the noise sources present on a particular sensor [38; 44] form the total Allan variance for the sensor:

$$\sigma^2(\tau) = \sigma_N^2(\tau) + \sigma_B^2(\tau) + \sigma_K^2(\tau) + \ldots \tag{3.1.18}$$

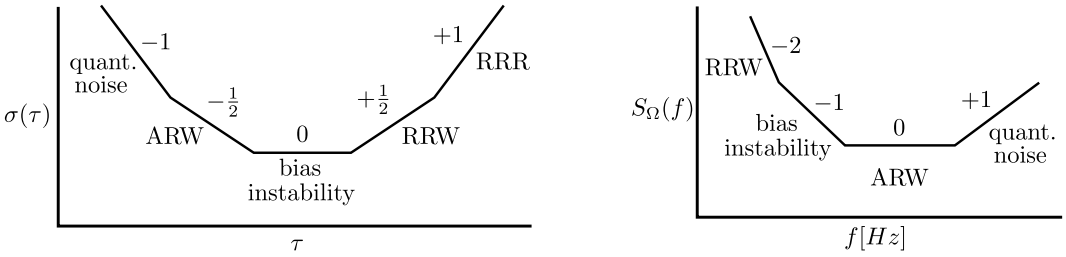
Figure 3.2a shows the slopes corresponding to each of the noise sources that an Allan variance curve is capable of identifying. Furthermore, Figure 3.2b shows the corresponding PSD plot.

Estimation accuracy of the Allan variance curve depends on the correlation time and number of data points in the dataset [38; 40]. The percentage error is given by

$$\delta(n) = \frac{1}{2\left(\sqrt{\frac{N}{n}} - 1\right)}. \tag{3.1.19}$$

Caution is needed when analysing the long correlation time region of an Allan variance curve. Estimation error may cause misrepresention of the true stochastic behaviour of measurements, especially in small datasets.

[2] Allan variance datasets are captured while the sensor is stationary. Therefore, if significant bit-quantisation is present, neighbouring samples will likely be binned within the same bit quantisation level. Taking the difference of the signal and the signal shifted by one time step will result in a variance of zero, and hence no effect will be observed on the Allan variance curve.



(a) Generic Allan variance curve.

(b) Generic PSD curve.

Figure 3.2: PSD and Allan variance plots for generic sensors.

3.2 Rate Gyroscope

3.2.1 Background

Angular rate is measured by rate gyroscopes (gyros). Inertial navigation systems usually integrate the measurements output from rate gyros to obtain angular measurements of the vehicle. Various types of gyroscopes exist, including fibre-optic, ring-laser and microelectromechanical systems (MEMS) gyros. The helicopter in the ESL uses a MEMS inertial measurement unit (IMU).

MEMS gyros and accelerometers are widely used in mobile phones, automobile airbag systems and vehicle navigation systems. The benefits of MEMS inertial sensors are that they have a low power consumption, a minute size and are cost effective. However, MEMS inertial sensors are significantly more complex to model than high-end inertial sensors, such as ring laser and fiber-optic gyros [45]. This is because MEMS sensors are often strongly affected by various factors, such as changing temperatures and vibration [45; 43]. The use of MEMS sensors on a small rotory-wing UAV requires a thorough examination of the influences of these factors on the inertial sensors.

The ADIS16405 is the inertial measurement unit used in the ESL helicopter and contains three-axis rate gyros and accelerometers. The original analog gyro and accelerometer data is filtered and sampled internally. These filtering and sampling processes are examined before analysing the noise characteristics of the gyros.

3.2.2 Acquisition of IMU Data

The IMU sensors in the ESL are presently sampled at a rate of 50 Hz. This rate matches the rate at which servo-motors require commands to be sent. Furthermore, this is also the rate at which the propagation updates of the Kalman estimator are currently executed [5]. The effects of the internal filtering and sampling on the Allan variance curve need to be taken into consideration to ensure that the bias stability

region of the sensors is unaffected during the process.

Currently, the IMU used in the ESL is the ADIS14605. Triaxial inertial sensors, featured in the ADIS14605, are filtered with an analog low-pass filter (LPF) at 330 Hz and sampled internally at 819.2 Hz. The digital signals are then filtered using a digital Bartlett window Finite Impulse Response (FIR) filter and downsampled at 50 Hz. The cut-off frequency of the Bartlett-window filter is chosen to prevent aliasing. This frequency was selected to be approximately 11 Hz in order to satisfy the Nyquist sampling theorem [3]. Groenewald [7] designed the original IMU breakout board (CANSense) for the ESL. He concludes that an inertial sensor bandwidth of 10 Hz is sufficient, as the inner loops of the Massachusetts Institute of Technology's more aggressively controlled X-Cell helicopter has a bandwidth of 12.5 Hz. Figure 3.3 demonstrates the filtering process performed on the gyroscopes and accelerometers in the IMU.

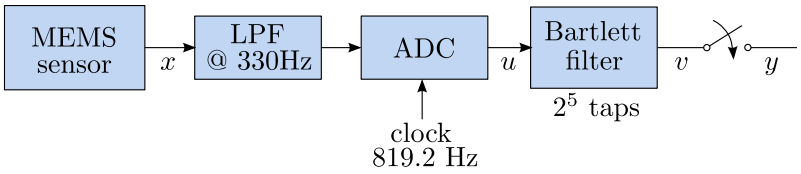
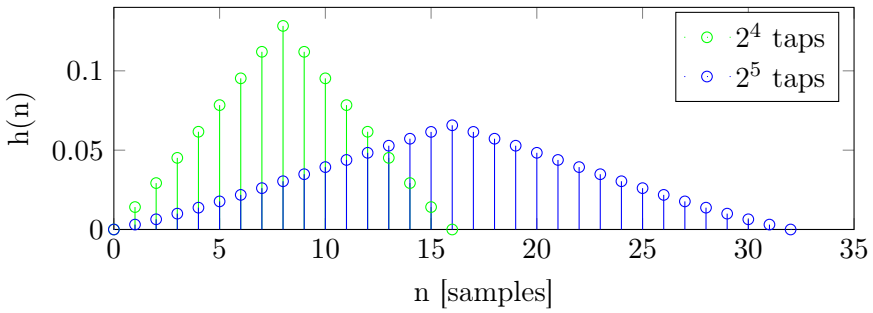


Figure 3.3: Block diagram adapted from [46] shows the filtering and sampling processes in the ADIS16405 IMU.

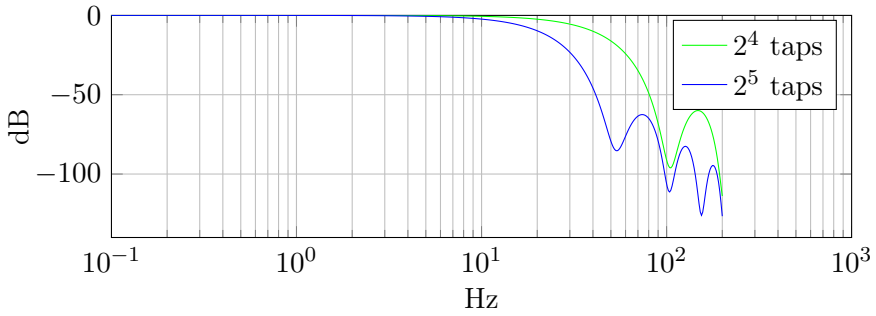
All causal low-pass filters create some degree of phase lag in the smoothed signal. The Bartlett-window filter is chosen to have 2^5 taps in order to obtain a cut-off frequency of 11 Hz [47]. The impulse response of the filter is a triangular impulse response that is shifted backward in time to ensure causality. The peak of the impulse response therefore occurs after the 16^{th} sample. At a sampling rate of 819.2 Hz, this corresponds to a phase delay of 19.5 ms. This is within one sample period of the Kalman filter, which is satisfactory. Figure 3.4 shows the impulse and frequency responses of the 2^5 tap Bartlett-window filter currently used. In addition, a 2^4-tap version is also depicted that would have a cutoff frequency of 16 Hz.

Figure 3.5 shows Allan variance plots at each stage of the filtering process performed on the gyroscopes in the IMU. The original noise data was simulated using band-limited white noise and a random walk using realistic PSD for an ADIS16405 gyroscope. The Bartlett window serves to reduce high frequency noise, as indicated

[3] The Bartlett-window filter has a gradual frequency roll off, as do all realisable filters. Therefore, the cut-off region was stipulated to be well within the Nyquist requirement, which states that the cutoff frequency should be half the sampling frequency.



(a) Impulse response of Bartlett window



(b) FFT of Bartlett window FIR filter

Figure 3.4: Impulse response of Bartlett window and resultant FFT of Bartlett window FIR filter.

by the curved region of signal v. The dotted lines indicate that variance would increase[4] at the output of the IMU if the Bartlett-window filter is set to 2^4 taps (y_2) or removed entirely (y_3). This is intuitively correct because aliasing occurs when the Nyquist sampling theorem is disregarded. y_1 represents the Allan variance curve that would result using the currently selected parameters for the ADIS16405 IMU onboard the helicopter. Downsampling has the effect of removing the high frequency region of the Allan variance curve. y_1 overlaps the original 819.2 Hz signal x, demonstrating that the Allan variance is not worsened by the filtering and sampling process performed in the IMU. The choice of cut-off frequency and sampling rate of the IMU is therefore satisfactory and no changes need to be made.

3.2.3 Noise Analysis

Nine continuous hours of IMU data, sampled at 50 Hz, were captured in order to characterise the noise of the gyroscope and accelerometer. This dataset was recorded indoors in order to have better control over the ambient temperature. Ideally temperatures should be constant when Allan variance analysis is performed in order to

[4] If no filtering is performed prior to downsampling to 50 Hz then the Allan variance will increase by a factor of $\frac{819.2}{50}$ [43].

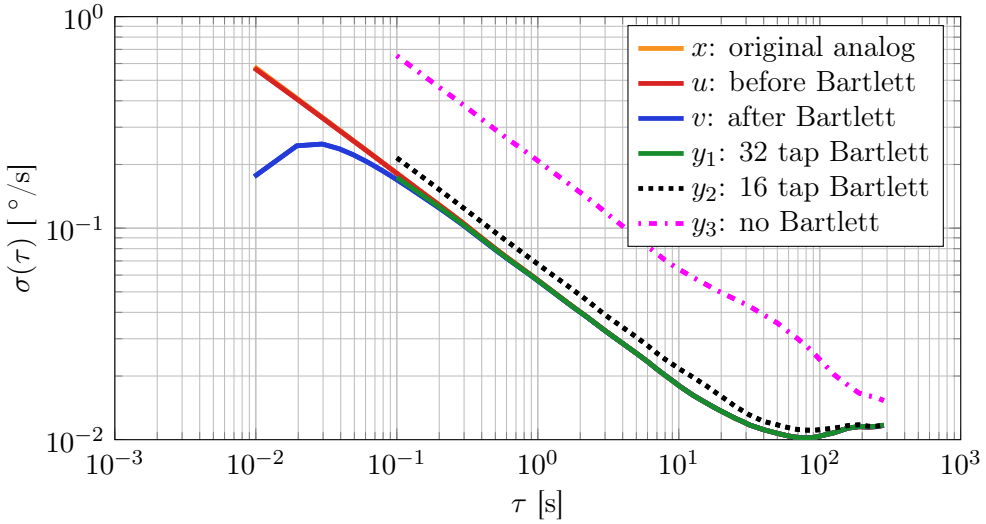


Figure 3.5: Allan variance of MEMS sensors within ADIS16405 IMU.

prevent systematic errors being interpreted as low frequency biases [43]. The complete dataset with the internally-measured gyro temperature is shown in Figure 3.6. As can be seen in Figure 3.6 the temperature of the gyros stabilises after approximately 1.5 hours. The ADIS16405 features internal temperature compensation to help remove systematic errors resulting from temperature fluctuations. However, gyro X still appears significantly affected by the change in temperature[5].

A five hour subset of the dataset, which began three hours after recording started, was selected for Allan variance analysis in order to prevent systematic errors due to temperature changes. Furthermore, the external electromagnetic interference observed on the magnetometer readings for this dataset was minimal for this period. Figure 3.7 shows the Allan variance curves for the subset of the complete indoor gyro dataset. The ARW, bias stability and RRW slopes are clearly visible in Figure 3.7. The three gyros show strong similarities to each other in their Allan variance curves.

Figure 3.8 shows the Allan variance curve of the gyro displayed in its datasheet [47]. Although MEMS sensors are known to exhibit large changes in their noise levels under different conditions [45], the Allan variance curves in Figure 3.7 match the datasheet Allan variance curves quite closely. The bias stability region of the actual gyro data ranges from a correlation time of approximately 30 to 90 seconds, depending on the gyro axis. This is reasonably close to the theoretical 100 seconds shown in the datasheet.

[5]The mean of each gyro signal has been subtracted, hence the gyro X readings are not zero to begin with.

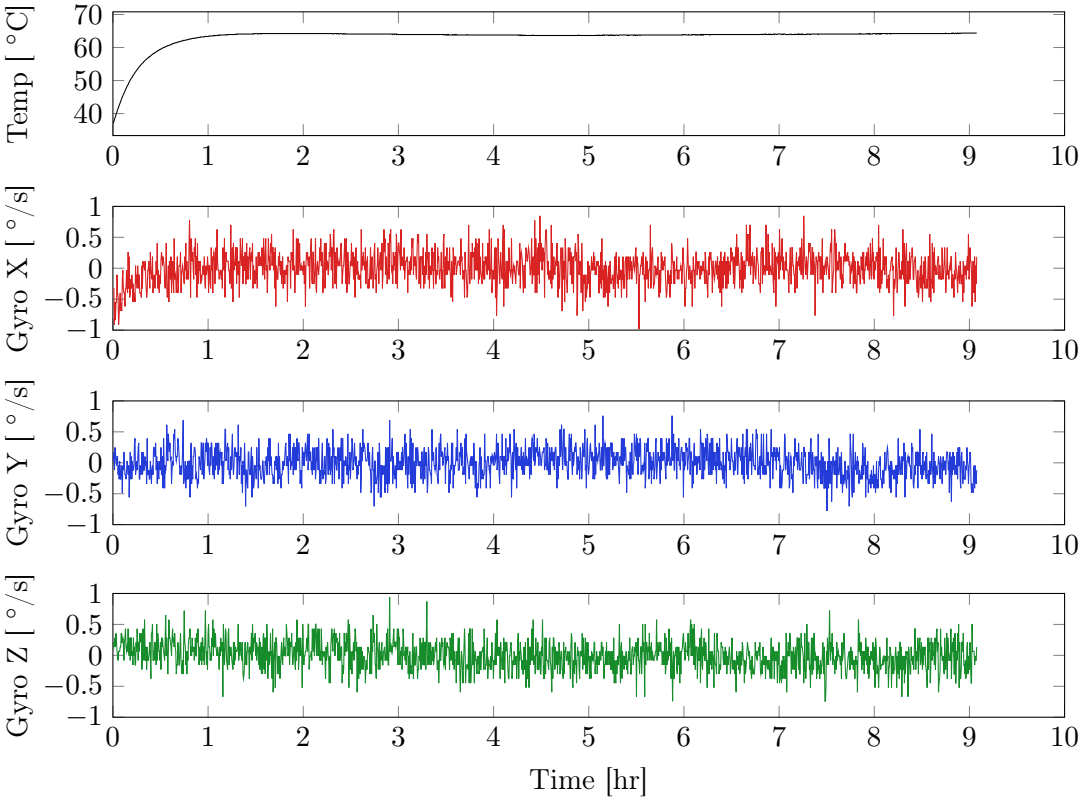


Figure 3.6: Gyro data and temperature for indoor dataset.

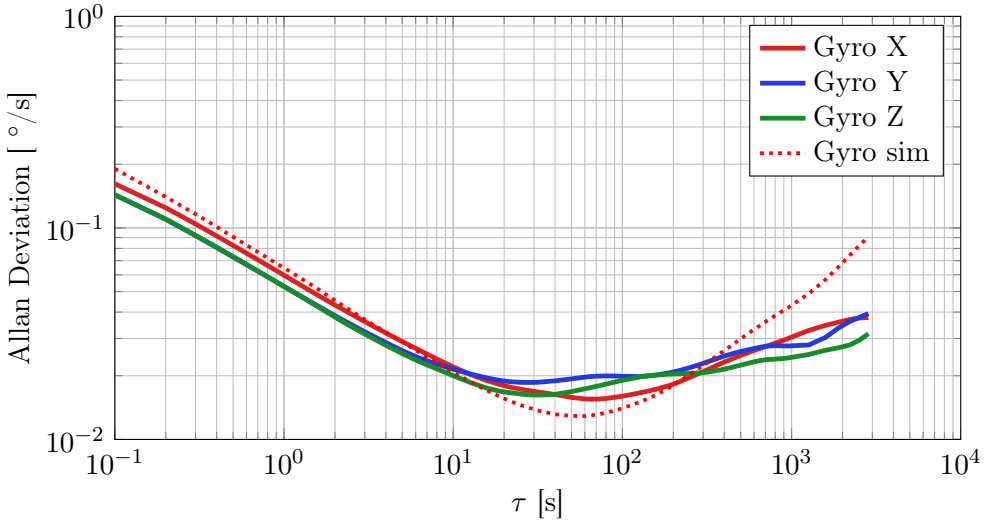


Figure 3.7: Allan variance of actual and simulated gyro data.

Figure 3.9 shows a comparison of the Allan variance curves for the indoor and outdoor data captures, both with engines off. The higher frequency noise components of both datasets demonstrate a strong resemblance. However, the low frequency

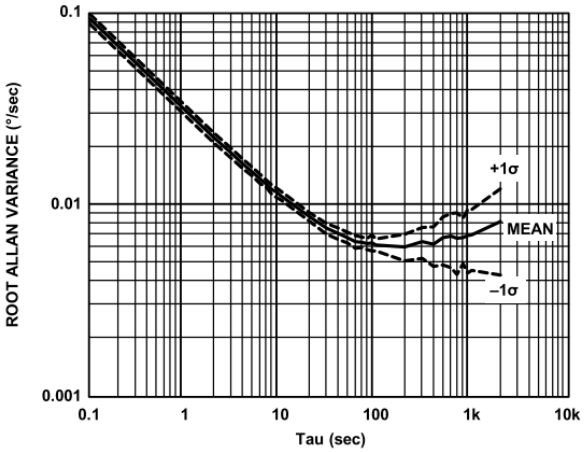


Figure 3.8: Allan variance of ADIS16405 gyro shown in its datasheet [47].

content of the outdoor dataset is noticeably different to the indoor dataset. The correlation times of the bias stability region for the outdoor dataset is shorter for each of the gyros. Gyro X has shifted from approximately 70 seconds to 10 seconds. This amounts to a shift of an order of magnitude from the theoretical bias stability correlation time of 100 seconds (see Figure 3.5).

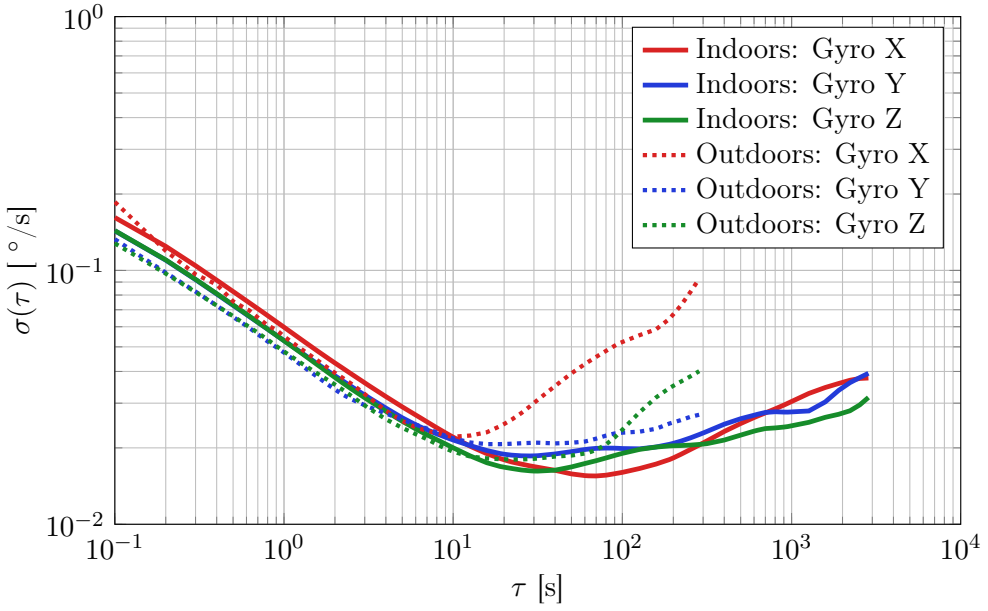


Figure 3.9: Comparison of Allan variance for the indoor and outdoor datasets.

The increase in low frequency noise could be caused by the change in temperature of the IMU while the outdoor dataset was being captured. Although the overlapping Allan variance technique makes very efficient use of the data available, significant

uncertainty still exist in the low frequency region. The ambient outdoor temperature will change, thus obtaining a very long dataset outdoors with a constant temperature is not feasible.

The internal temperature compensation of the ADIS16405 may not be performing as well for the X-gyro compared to the Y and Z-axis gyros. This surmise can be made as a result of the significant correlation between temperature change and bias of the X-axis gyro during the warm-up period of the indoor dataset (Figure 3.6), as well as the noticeable deviation in the magnitude of the RRW between indoor and outdoor datasets.

3.2.4 Simulated Noise Models

Allan variance analyses of the gyro datasets reveal that the stochastic biases are dominated by ARW and RRW (Figures 3.7, 3.9). The gyros can therefore be modelled as white noise with the addition of white noise integrated. This is conventionally used to simulate RRW [48; 39; 44]:

$$\begin{aligned} \tilde{w} &= w + b + n_1 \\ \dot{b} &= n_2, \end{aligned} \tag{3.2.1}$$

where w is the true angular rate within the gyroscope's local reference frame and n_1 and n_2 are zero-mean Gaussian white noise processes [48]. Band-limited white noise blocks are used to simulate the white noise (ARW) component in Simulink. White noise has a constant PSD, which is calculated from the ARW parameter as shown in Equation (3.1.5). The RRW is modelled in the same way, followed by an integrator. The PSD of the band-limited white noise block for the RRW is calculated using Equation (3.1.5) by substituting K in place of N. Figure 3.10 depicts the complete Simulink model of the simulated gyroscopes. The input signal is low-pass filtered at 10 Hz to prevent aliasing. The outputs of the band-limited white noise blocks are low-pass filtered in order to model the spectral characteristics of the higher frequency ARW and lower frequency RRW.

Table 3.2 shows the Allan variance parameters for the indoor gyro datasets as well as the simulated gyro observed in the Allan variance curves (see Figure 3.8).

Table 3.2: Gyroscope Allan variance parameters

Gyroscope axis	**X**	**Y**	**Z**	**Simulation**
ARW $[^\circ/s/\sqrt{Hz}]$	5.31×10^{-2}	5.45×10^{-2}	6.01×10^{-2}	6.45×10^{-2}
RRW $[^\circ/s/s/\sqrt{Hz}]$	1.81×10^{-3}	2.32×10^{-3}	2.10×10^{-3}	2.21×10^{-3}

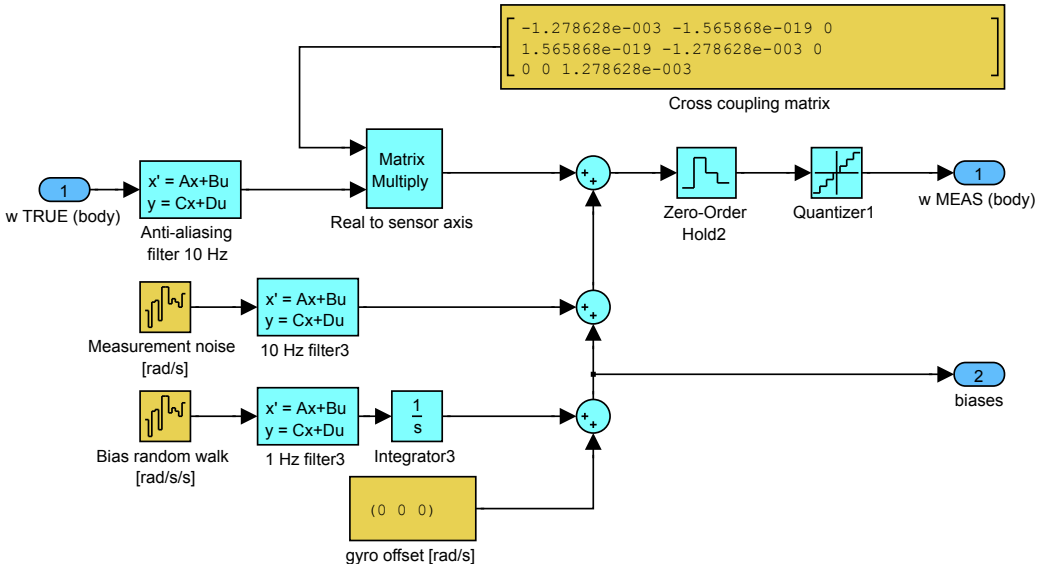


Figure 3.10: Simulink model of simulated gyroscopes.

A comparison of the PSD of the actual and simulated gyros is shown below in Figure 3.11. The white noise regions match very closely, as seen in the Allan variance plots.

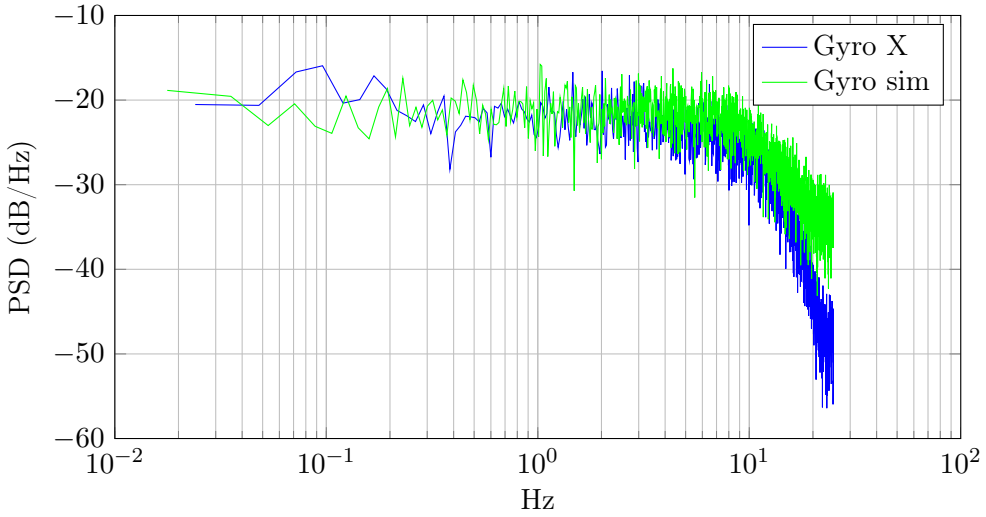


Figure 3.11: PSD of actual and simulated gyro data.

The effects of the Bartlett-window filter are clearly visible with frequencies above the 10 Hz region attenuated. The simulated gyro models the true stochastic behaviour of the actual gyro sufficiently for the purposes of this project.

3.3 Accelerometer

3.3.1 Background

Accelerometers are used to measure acceleration relative to free fall. When placed on a stationary surface, an accelerometer measures the acceleration due to the normal force, which will be referred to here as the static acceleration. Inertial navigation systems extract the dynamic component of acceleration by adding gravitational acceleration. Velocity measurements are obtained by rotating the dynamic acceleration into an inertial reference frame and integrating. Position measurements can then be obtained by integrating the velocity measurements.

3.3.2 Noise Analysis

Allan variance analysis of an accelerometer is very similar to that of a rate gyroscope. As described in Section 3.1, the spectral components observed in the sensor data have very similar characteristics.

Figure 3.12 below shows the nine hour accelerometer dataset that was recorded. The same five hour subset of the data was selected for Allan variance analysis. This minimised possible systematic biases introduced by temperature changes and electromagnetic disturbances. The accelerometer data shows a noticeable random walk component as well as significant bit quantisation. This is noticeably different from the gyroscope readings. Bit quantisation is common to MEMS accelerometers [45]. Bit quantisation effects, however, are not visible in the Allan variance curves, as discussed in Section 3.1.

Below Figure 3.13 shows the Allan variance of the accelerometer dataset where strong VRW and RRW slopes are present. The Y-axis accelerometer deviates somewhat from the other two accelerometers, but the same general trend is exhibited.

Figure 3.14 shows the Allan variance curve of the accelerometer displayed in its datasheet [47]. The datasheet plots the Allan variance in units of g, which correspond to 9.81 m/s, and describes the order of magnitude difference in the graphs. The Allan variance of the actual accelerometer resembles the datasheet very closely in the magnitude of the VRW and RRW regions. The correlation time of the bias stability of the actual accelerometer is approximately 100 s, whereas the theoretical accelerometer's bias stability is 25 s. Considering that MEMS sensors are very sensitive to environmental changes, this discrepancy is seen as relatively minor.

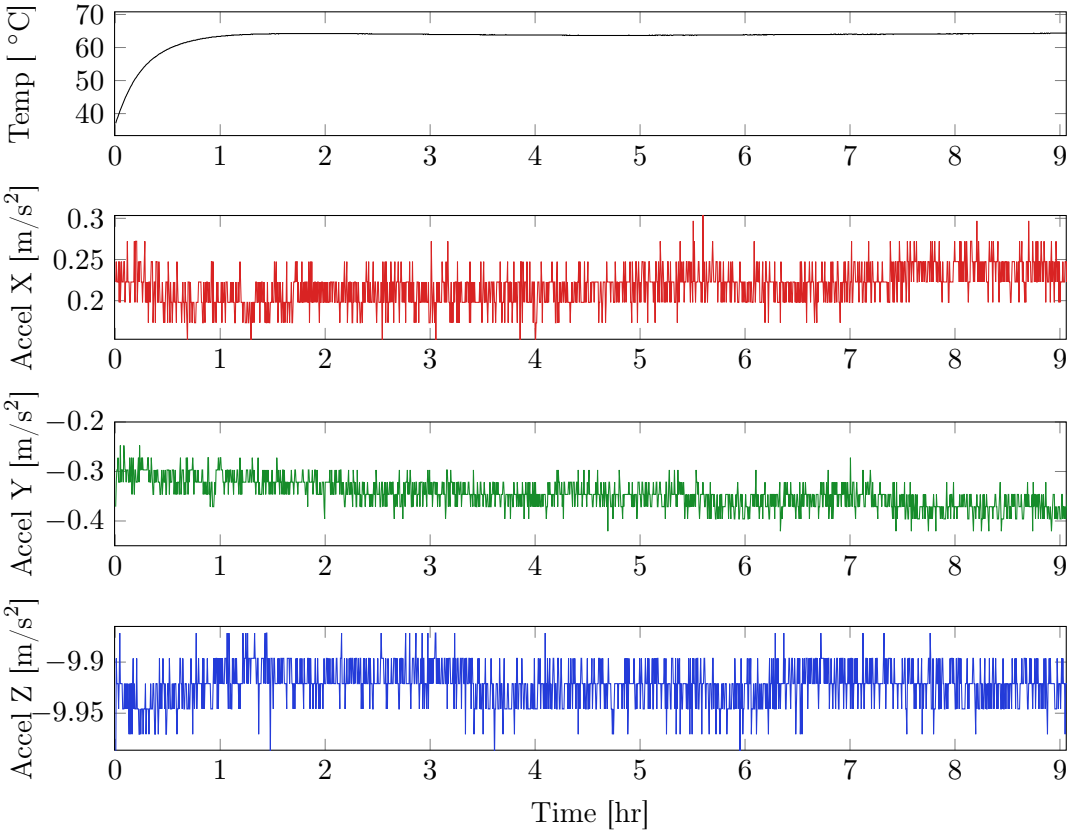


Figure 3.12: Accelerometer data and temperature for indoor dataset.

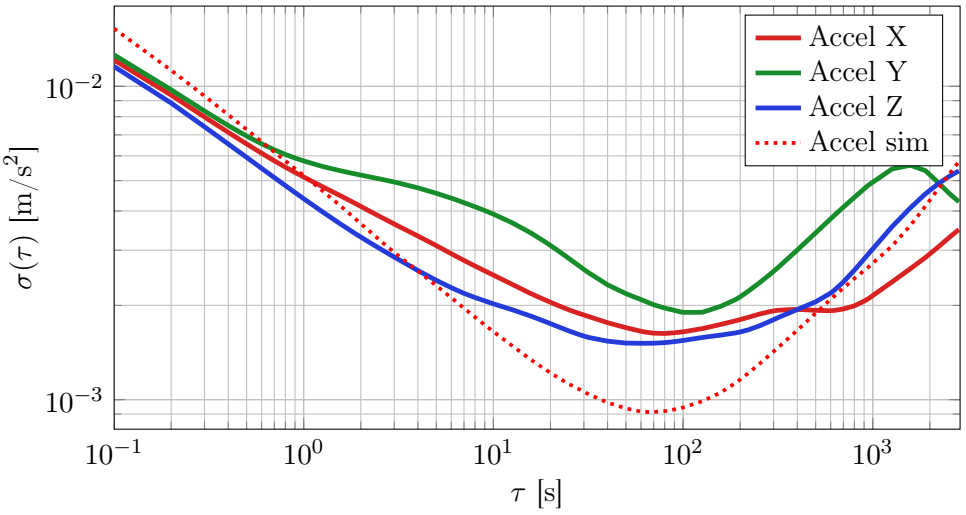


Figure 3.13: Allan variance of actual and simulated accelerometer data.

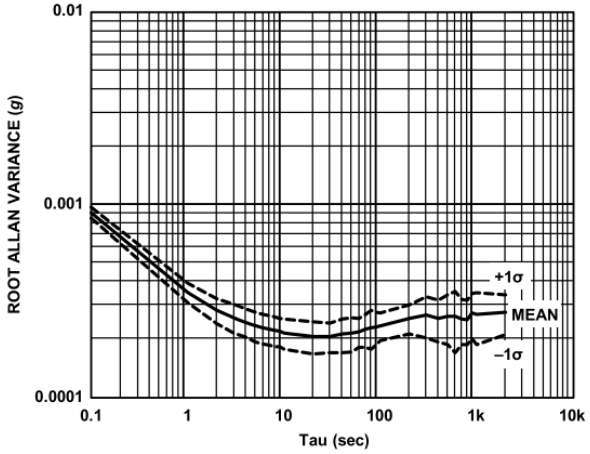


Figure 3.14: Allan variance of ADIS16405 accelerometer shown in its datasheet [47].

3.3.3 Simulated Noise Models

As in the case of the rate gyros, the accelerometer noise is dominated by velocity random walk and rate random walk slopes. The accelerometers can therefore also be modelled as the addition of white noise and integrated white noise processes:

$$\begin{aligned} \tilde{\alpha} &= \alpha + b + n_1 \\ \dot{b} &= n_2, \end{aligned} \tag{3.3.1}$$

where α is the true proper acceleration within the accelerometer's local reference frame and n_1 and n_2 are zero-mean Gaussian white noise processes. The PSD values of the VRW and RRW band-limited white noise blocks are calculated from the Allan variance parameters in Table 3.3 using Equation (3.1.5).

Table 3.3: Accelerometer Allan variance parameters

Accelerometer axis	**X**	**Y**	**Z**	**Simulation**
VRW $[m/s^2/\sqrt{Hz}]$	4.4×10^{-3}	4.8×10^{-3}	4.2×10^{-3}	5.1×10^{-3}
RRW $[m/s^2/s/\sqrt{Hz}]$	1.5×10^{-4}	2.7×10^{-4}	1.8×10^{-4}	1.4×10^{-4}

The complete Simulink model of the simulated accelerometer is depicted in Figure A.1 of Appendix A. The output signals are quantised at 1.28×10^{-3} m/s^2 to resemble the actual accelerometers' data.

Figure 3.15 shows a comparison of the PSD of actual and simulated accelerometer data. The shape of the accelerometer PSD plot closely resembles the gyro PSD.

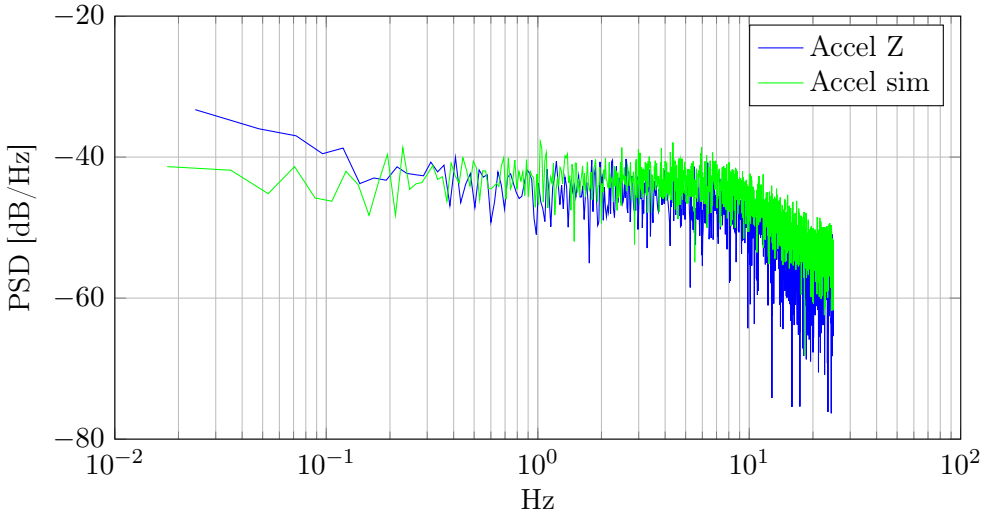


Figure 3.15: PSD of actual and simulated accelerometer data.

3.4 Magnetometer

3.4.1 Background

Magnetometers are used to measure the strength and direction of the magnetic field. The magnetometer used in the ESL helicopter is the Honeywell HMC2003. Three magneto-resistive permalloy strips, whose resistance alters with variations in the surrounding magnetic field strength, are contained in the HMC2003 [49].

The current magnetic field vector in Stellenbosch is shown in Table 3.4, obtained from the World Magnetic Model 2010. The earth's magnetic field for a given location changes in strength and direction over time. For this reason the World Magnetic Model is updated every six years.

Table 3.4: Magnetic reference vector at Stellenbosch in 2013.

Direction	**North**	**East**	**Down**	**Magnitude**
Field strength [*Gauss*]	0.093904	-0.041366	-0.236304	0.2576

3.4.2 Calibration

The magnetic field vector is subject to a wide variety of possible disturbances. These disturbances result from soft iron effects, hard iron effects and electric currents [8]. The development of an accurate model of the magnetic field surrounding a helicopter

is therefore a notoriously difficult task. Time-varying magnetic disturbances are created by rotor blades and other moving parts, as well as electric cables.

Magnetometer calibration can be classified into two distinct categories: absolute and relative calibration [8]. Absolute calibration involves orientating the vehicle at various known attitudes using a calibration table [8]. However, magnetic disturbances originating from the calibration table will result in accuracy of the calibration only at the location at which calibration is performed, which is a disadvantage [8].

Relative calibration requires no knowledge of the absolute orientation of the vehicle during the calibration process. Instead, the vehicle is rotated in 360 degrees along each of the rotation axes in turn, during which magnetometer readings are taken. If the magnetometer is perfectly calibrated the readings $\mathbf{X}_c$ will lie on a sphere of radius equal to the length of the magnetic field reference vector. An uncalibrated magnetometer's readings $\mathbf{X}_u$ will trace the surface of an ellipse translated from the origin. Calibration involves calculating the calibration matrix $\mathbf{A}$ and offset vector $\mathbf{b}$ that map the surface of the ellipse to that of the sphere:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \mathbf{A} \begin{bmatrix} X_u \\ Y_u \\ Z_u \end{bmatrix} + \mathbf{b}. \tag{3.4.1}$$

Least-squares can be used to find $\mathbf{A}$ and $\mathbf{b}$. Figure 3.16 shows the magnetometer readings before (red) and after (blue) calibration. The calibrated readings lie on the surface of the sphere, which indicate that the sensor is calibrated correctly.

3.4.3 Noise Analysis

Once calibrated, a stochastic model of the noise on the magnetometer readings can be determined. A 23 minute dataset of magnetometer readings was recorded at the Helderberg Radio Flier's (HRF) airfield, away from any electric cables and metallic objects that could create magnetic disturbances. Figure 3.17 below shows the magnetometer data captured.

Brief, periodic spikes are present on the magnetometer readings, despite efforts to minimise magnetic disturbances. Figure 3.18 shows a magnified subset of the X-axis magnetometer revealing the periodic spikes in the signal. The spikes result from the close proximity of the radio antenna to the magnetometer. The dataset was recorded with an active radio link to ensure conditions were similar to those experienced during flight.

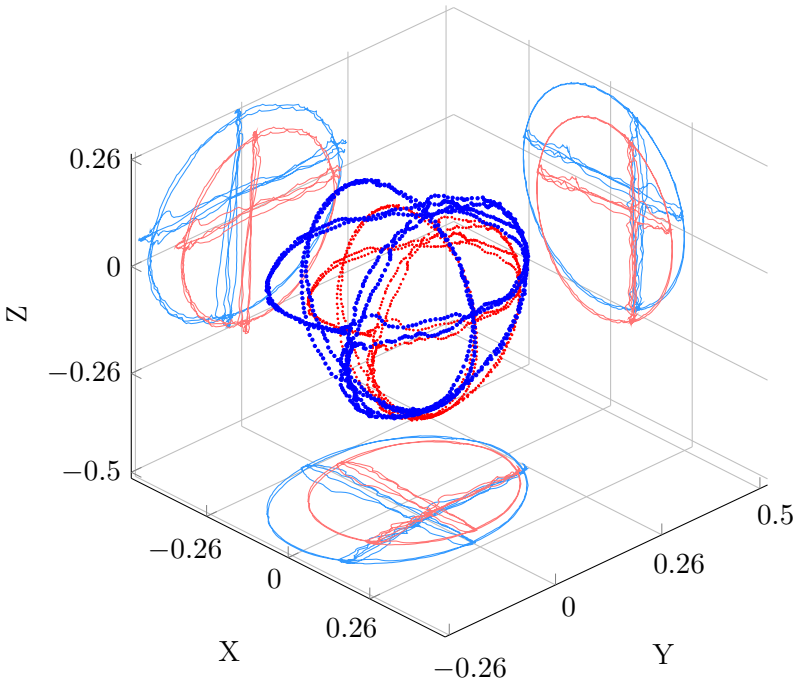


Figure 3.16: Before (red) and after (blue) magnetometer calibration. Measurements expressed in units of Gauss.

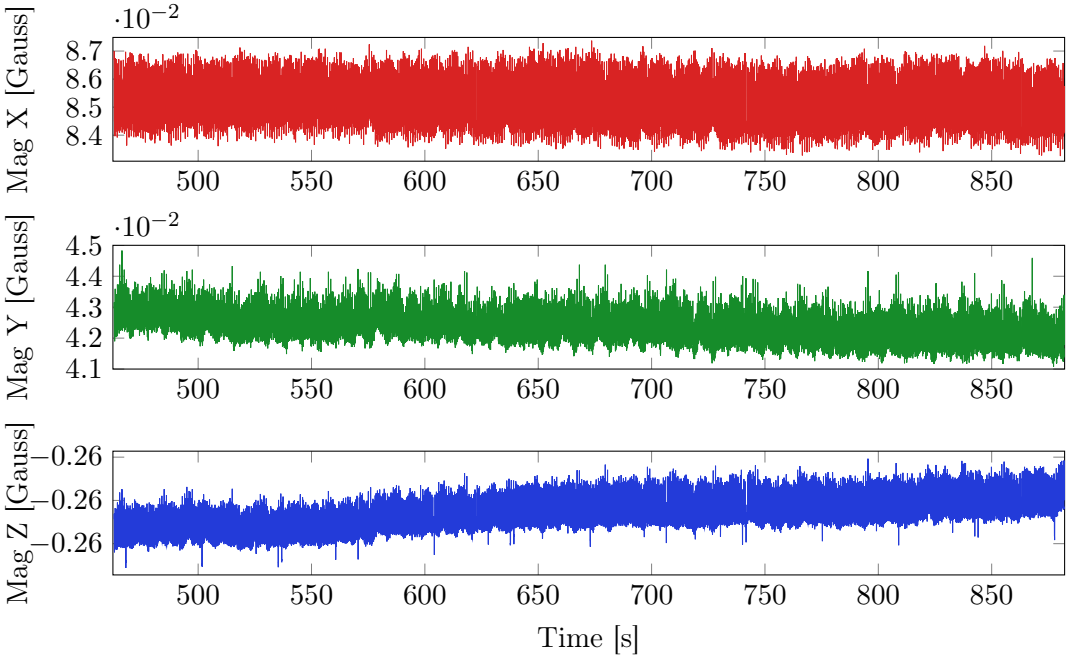


Figure 3.17: Magnetometer data.

The angle between the measured magnetic vector before a spike **a** and at the peak of a spike **b** is calculated to quantify the severity of the spikes on the measured

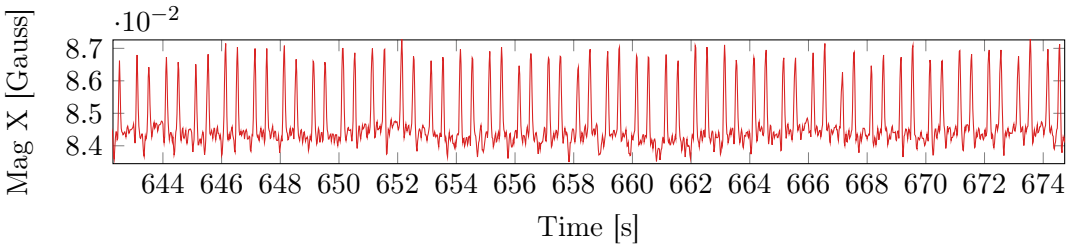


Figure 3.18: Close-up of magnetometer data.

magnetic vector. The dot product can be used to determine the angle as follows:

$$\theta = \arccos\left(\frac{\mathbf{a}\cdot\mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}\right) = 0.47^{\circ} \tag{3.4.2}$$

where $\mathbf{a} = \begin{bmatrix} 0.084 & 0.041 & -0.261 \end{bmatrix}^T$ and $\mathbf{b} = \begin{bmatrix} 0.087 & 0.042 & -0.263 \end{bmatrix}^T$. Evidently, the effect of the spikes on the measured magnetic field vector is small. The overall effect on the attitude estimates will be negligible because the spikes are brief and the Kalman estimator filters out high frequency fluctuations in the measurements, as explained in Section 2.1.

The Allan variance technique is most commonly used in the analysis of precision clocks and inertial measurement units, as discussed in Section 3.1. However, it is also widely used to model various other sensors, including magnetometers [8; 50; 51; 52]. Figure 3.19 shows the Allan variance of the magnetometer dataset. The magnetometer exhibits sharp, distinct peaks and troughs in the measurement random walk region of the Allan variance curve. This is in contrast to the smooth, well-defined slopes of the gyros and accelerometers. Very little estimation error is contained in this section of the Allan variance curve. This is due to the large number of clusters used in the Allan variance calculation. The peaks are not a result of the dataset recording being too short. Instead, they are a result of the periodic spikes observed in the data, as expected from the discussion on sinusoidal noise types in Section 3.1.

Ang [52] and Bijker [8] both indicate Allan variance plots for the HMC2003 magnetometer. Definite measurement random walk, bias stability and rate random walk slopes are observed in both cases, as is seen in Figure 3.19. The bias stability region depicted in [8], located at approximately $\sigma(1.5) = 7 \times 10^{-5}$, coincides very closely with that of Figure 3.19. This indicates that the noise levels of the magnetometer within the helicopter are characteristic of the HMC2003 in general.

Figure 3.20 shows the autocorrelation of the Y-axis magnetometer. The spikes observed in the time-series plot of the dataset have a distinct fundamental period every 375 samples, which correspond to a frequency of 2/15 Hz.

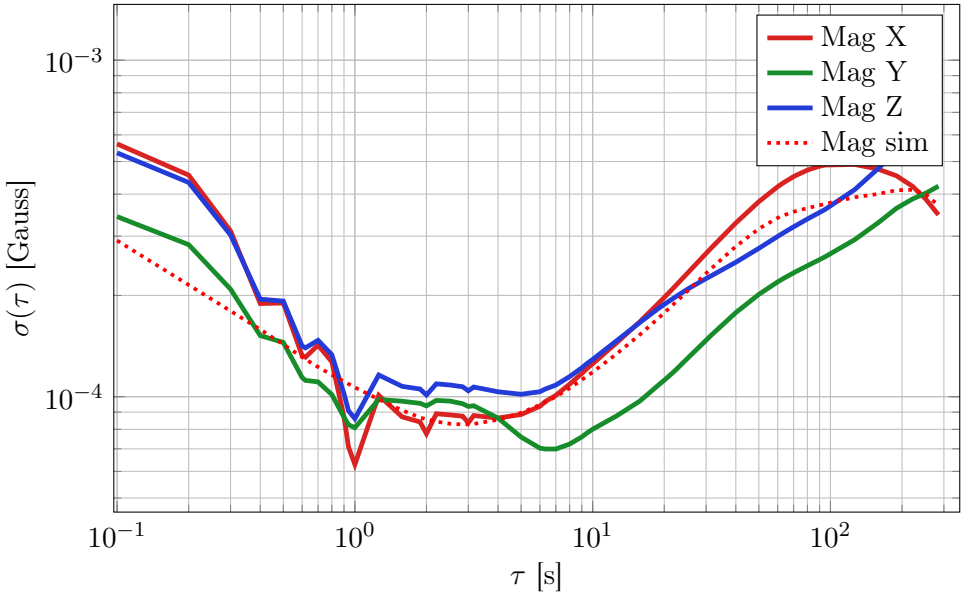


Figure 3.19: Allan variance of magnetometer.

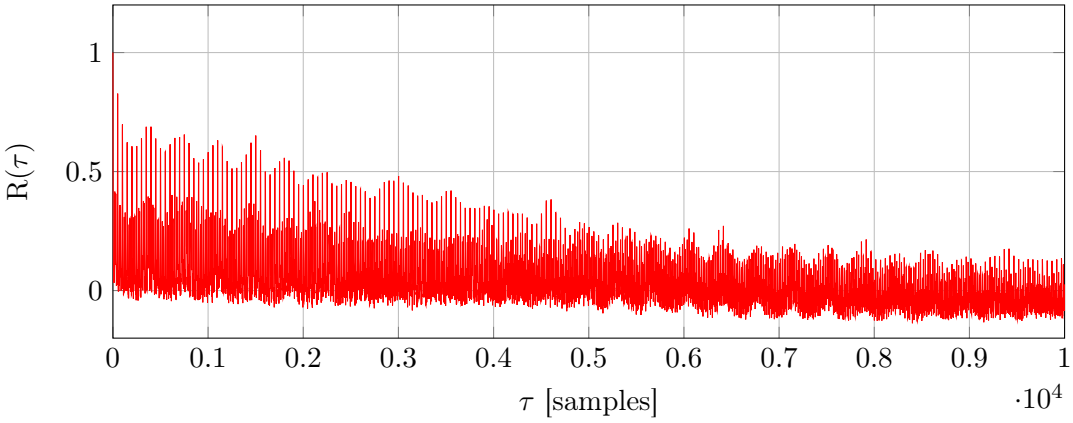


Figure 3.20: Autocorrelation of the Y-axis magnetometer.

3.4.4 Simulated Noise Models

The model of the magnetometer used for simulation accounts for hard iron effects, soft iron effects and constant electric currents. These effects are modelled by inverting Equation (3.4.1) as explained in [8].

The stochastic characteristics of the sensor are also modelled. The Allan variance curves of the magnetometer show definite measurement random walk and rate random walk slopes. This allows the stochastic characteristics of the sensor to be modelled in the same manner as the inertial sensors. Table 3.5 shows the Allan variance parameters obtained from the Allan variance curve and the parameters chosen for the simulated magnetometer. The Allan variance of the readings from the simulated

magnetometer are plotted alongside the Allan variance of the actual magnetometer readings in Figure 3.19. The spectral characteristics of the simulated magnetometer

Table 3.5: Magnetometer Allan variance parameters

Magnetometer axis	**X**	**Y**	**Z**	**Simulation**
MRW $[Gauss/\sqrt{Hz}]$	1.32×10^{-4}	1.05×10^{-4}	1.35×10^{-4}	1×10^{-4}
RRW $[Gauss/s/\sqrt{Hz}]$	7.9×10^{-5}	4.7×10^{-5}	6.8×10^{-5}	6×10^{-5}

match the actual sensors very closely, with the exception of the brief periodic spikes. As, the spikes cause a negligible influence on the estimates, the development of an accurate simulation model of the spikes is unnecessary.

Figure A.2 in Appendix A depicts the complete Simulink model of the simulated magnetometers.

3.5 GPS

3.5.1 Background

GPS position measurements are plagued by a multitude of error sources. These errors include: multipath; refraction in the ionosphere and troposphere; dilution of precision[6]; satellite ephemeris errors[7]; as well as frequency instabilities in GPS satellite clocks and receiver clocks [53; 54]. These manifest as slow moving biases (i.e. random walks) in the GPS position measurements.

Differential GPS (DGPS) is a method used to remove these biases from the measurements. The use of an additional, stationary GPS receiver is required in DGPS. This base station is known to be stationary. Thus, any non-zero signals recorded arise from noise. The biases in GPS position are generally spatially correlated. These offsets can therefore be transmitted to the vehicle's GPS receiver and corrections made to its measurements, provided the two receivers are in close proximity.

GPS satellites transmit pseudo-random coded sequences on two carrier frequencies, L1 and L2 [54]. L1 is centered on 1575.42 MHz and L2 is centered on 1227.60 MHz. These frequencies lie within the L-band of the frequency spectrum. The majority of GPS receivers, such as those operating in single-point or DGPS mode, determine

[6]Dilution of precision (DOP) is a measure of the accuracy of a GPS measurement as a function of the positions of the GPS satellites. The measurement of the GPS position will become better conditioned when the satellites are spread out in the sky. Therefore, a lower DOP will result as well as better accuracy.

[7]Discrepancies between the actual and predicted GPS satellite locations.

the range to each satellite by sliding a local copy of the coded sequence against the sequence received from the satellite. Once the two sequences line up, the range is calculated based on the delay between the two sequences and the speed of light.

Carrier phase analysis of the transmitted signals can be used in order to improve the accuracy of GPS position measurements. Rather than using only the coded sequences to determine the delay between the local and transmitted coded sequences, the receiver should compare the L1 signal phase to the local coded sequence. If the local and transmitted carrier frequencies align in phase, the range can then be precisely determined to within an integer wavelength of the true range [54]. This is known as L1-Int. The wavelength for the L1 signal is 19cm. The range measurements will therefore be known precisely within a multiple of 19cm of the true range.

Real-Time Kinematic (RTK) GPS systems combine DGPS and carrier phase analysis to provide highly precise position measurements of the receiver relative to a base station. The Novatel OEMV-1G offers ALIGN mode, which is a form of RTK, whereby accurate position measurements are calculated relative to a moving base station [55]. However, the absolute position accuracy is limited to the absolute position accuracy of the base station receiver [54].

The satellite carrier signals can also be used to measure a vehicle's velocity using the doppler effect to improve the accuracy of position measurements. The Novatel receiver, when operating in BESTVEL mode, estimates velocity using the doppler velocity measurements as well as differentiated position measurements. The receiver selects the form of velocity estimates that are most accurate [56]. The general term "differential GPS" will be used as RTK mode is a form of differential GPS. Analysis of the noise characteristics of single-point and RTK modes of operation will be performed and stochastic models derived in the following section.

3.5.2 Noise analysis

Single-Point GPS

GPS position noise is composed of two primary sources of noise: white noise and a slow random walk. A high-end GPS, such as the Novatel, that operates in single-point mode exhibits very little white noise on the position measurements. A random walk component resulting from ionospheric refractions and clock biases is unavoidable. This is seen in Figure 3.21, which shows a single-point Novatel position dataset captured for 23 minutes. The vehicle position is known to a high precision, but low accuracy, when operating in single-point mode [54]. The Down position measurements are noticeably less accurate than the North and East measurements. This

is a result of dilution of precision. The discontinuity seen in the Down position

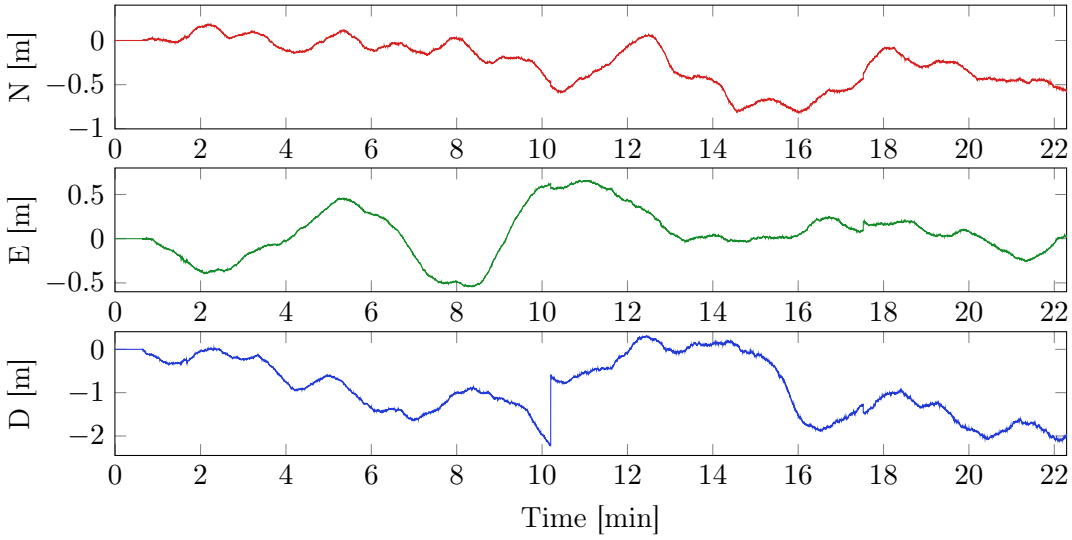


Figure 3.21: Single-point GPS position.

measurements is characteristic of single-point GPS.

GPS receiver datasheets generally describe the accuracy of the GPS position as a circular error probability (CEP) or spherical error probability (SEP). CEP is given as the radius of the circle in the horizontal plane within which 50% of the position measurements are expected to lie [57]. Similarly, SEP is the radius of the sphere within which 50% of the measurements lie. However, CEP and SEP do not describe the rate of measurement drift. PSD and Allan variance can instead be used to distinguish separate noise components and determine the bandwidth of the measurement drift [40].

Figure 3.22 shows the Allan variance of the single-point GPS position. As expected, from Figure 3.21, the single-point position measurements exhibit a significant random walk. The Allan variance curve reveals no obvious white noise component. This is because the Novatel performs significant filtering of measurements and single-point GPS position measurements are dominated by low frequency drift.

The single-point velocity measurements contain a distinct white noise component, as seen in Figure 3.23. The Down velocity has significantly more noise than the North and East velocity measurements, which is similar to the position measurements. The single-point velocity dataset is shown in Appendix A.

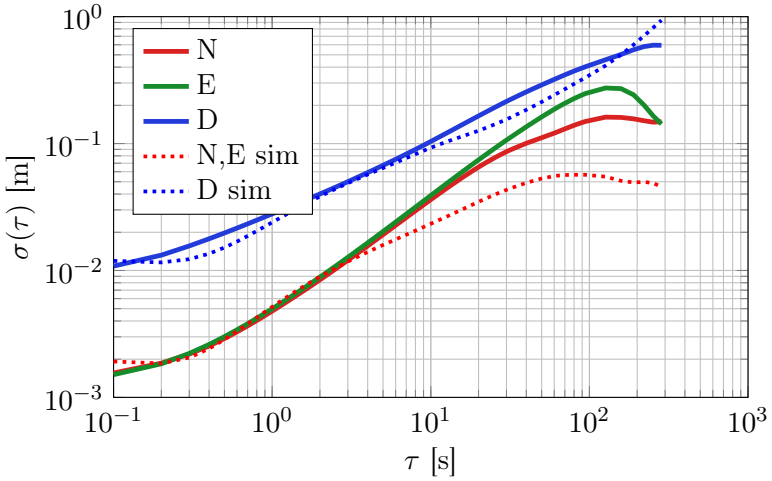


Figure 3.22: Allan variance of single-point GPS position.

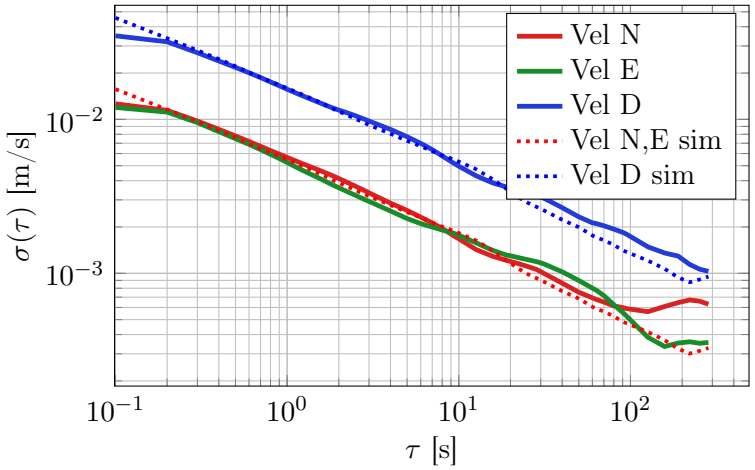


Figure 3.23: Allan variance of single-point GPS velocity.

Differential GPS

A 25 minute dataset was captured using the RTK mode of operation, whereby L1-Int was achieved and at least 13 satellites were present throughout the dataset. The dataset is shown in Appendix A. The Allan variance of the dataset is shown in Figure 3.24 and the Allan Variance parameters are listed in Appendix A. The hump observed below is characteristic of differential GPS [40]. Internal filtering is being performed by the differential GPS, resulting in correlation in the measurements. Correlation manifests as a hump in the Allan variance curve, as explained in Section 3.1.5. MRW and RRW are also present, although the MRW is difficult to quantify due to the dominance of the correlated noise.

The differential GPS velocity shows a slight overall improvement in noise levels over

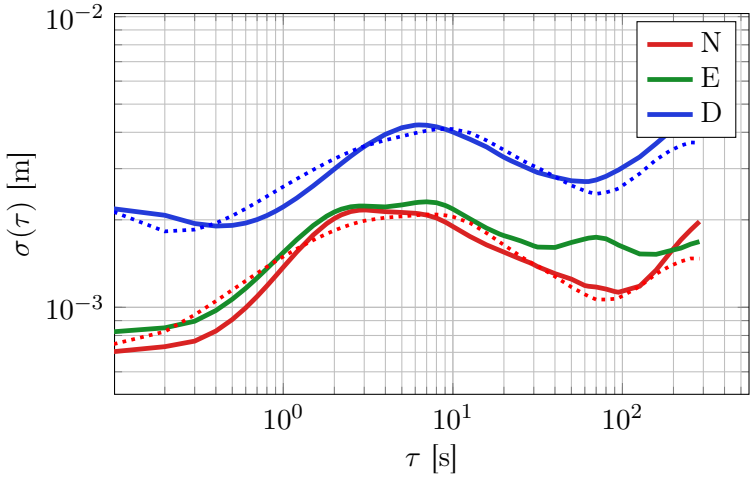


Figure 3.24: Allan variance of differential GPS position.

single-point GPS velocity. This is demonstrated in Figure 3.25. Interestingly, the North and East velocity Allan variances do not overlap. The North velocity readings exhibit stronger noise than the East velocity. The Down velocity noise is reduced most of all by the use of differential GPS. The noise spectrum is essentially white, as with single-point GPS.

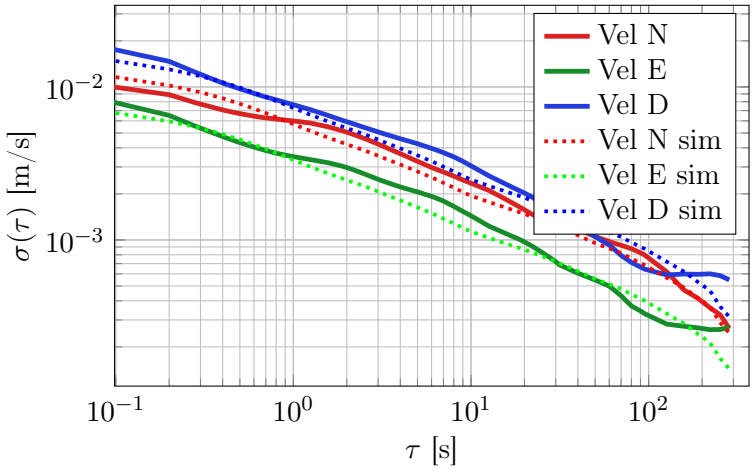


Figure 3.25: Allan variance of differential GPS velocity.

3.5.3 Simulated Noise Models

Single-Point GPS

The noise on the GPS measurements can be simulated within the NED reference frame because the helicopter has small range and low velocity. The slope of the Down position measurements is almost precisely $+1/2$ and is well modelled as an RRW. The North and East measurements have a spectral component whose Allan variance slope lies in between a pure RRW and RRR. For simplicity these measurements are modelled as an RRW too. Figure 3.22 shows the Allan variance curves of the simulated single-point position measurements alongside the Allan variance of the actual measurements.

GPS velocity measurements can be simulated simply as band-limited white noise, with a cut-off frequency larger than the sampling frequency of the GPS (10 Hz). The Allan variance of the simulated GPS velocity readings closely matches the actual readings (see Figure 3.23).

The Simulink model is based on the same model used for the inertial sensors and is therefore omitted here. The Simulink model and the Allan variance parameters used for simulation of the position and velocity measurements are listed in Appendix A.

Differential GPS

In contrast to all the other sensors analysed so far, the differential GPS position measurements are dominated by correlated noise and only have small white noise and RRW components. A first-order Gauss-Markov process is used to model the correlated noise region. The correlation time and noise amplitude of the Gauss-Markov process for each of the North, East and Down measurements can be identified directly from the Allan variance plot, as explained in [38] and [40]. The North and East measurements indicate a correlation time of approximately 3 seconds, whilst the Down measurements is 4 seconds.

The MRW and RRW slopes are not easily quantifiable from the Allan variance plot. This is due to the dominance of correlated noise. The MRW and RRW parameters were therefore determined by trial and error until the simulated Allan variance curves matched the actual plots. Comparison of the Allan variance of the simulated and actual data is shown in Figure 3.24.

A discrete noise shaping filter is used to model the Gauss-Markov process in Simulink, based on Equation (3.1.14). The MRW and RRW components are modelled as per normal and the complete Simulink diagram is shown in Appendix A.

The differential GPS velocity measurements are essentially composed only of white noise and can therefore be simulated as band-limited white noise in Simulink.

Novatel Align

There is no stationary base station available to a helicopter at sea. However, Novatel Align can be used to provide highly accurate relative position and velocity measurements. The same satellites are used by both receivers when calculating their respective positions in Align mode. This ensures that the drift in their measurements is highly correlated, allowing accurate relative measurements to be calculated. Their absolute position measurements drift in a similar manner to that of single-point GPS as no absolute correction is made. Align mode can therefore be simulated as two single-point receivers exhibiting identical random walks. The Simulink model is shown in Appendix A.

GPS Delay

The GPS data captured in this thesis exhibits measurement delay. These delays have since been attributed to hardware issues in the onboard computer (OBC) and have been rectified. Time limitations, however, prevented repetition of the flight tests shown in Chapter 8.3. The decision was therefore made to quantify the latency and to remove the delay from the signals.

The first flight test conducted consisted of a series of step tests. The heave controller of the helicopter has a relatively high bandwidth, enabling clear spikes in the accelerometer data to be observed. Figure 3.26 compares the DGPS position and accelerometer readings for one of the heave steps.

The latency is calculated as the time difference between the steps in GPS and accelerometer measurements. GPS packets arrive at 5 or 10 Hz. Therefore, the average delay of the GPS measurements lies between the observed step in GPS reading (i.e. maximum latency) and the previous GPS packet arrival (i.e. minimum latency). Three heave steps were performed. The minimum and maximum GPS position latencies are shown in Table 3.6. The more heave steps performed, the more tightly the GPS latency can be determined. This is because the true latency is bound within the smallest maximum latency and the largest minimum latency[8], denoted by the bounds in the table.

The accelerometers also have some latency associated with their readings, which is at least 20 milliseconds due to the 32-tap Bartlett filter, as discussed in Section 3.2.

[8]This assumes that the latency is constant.

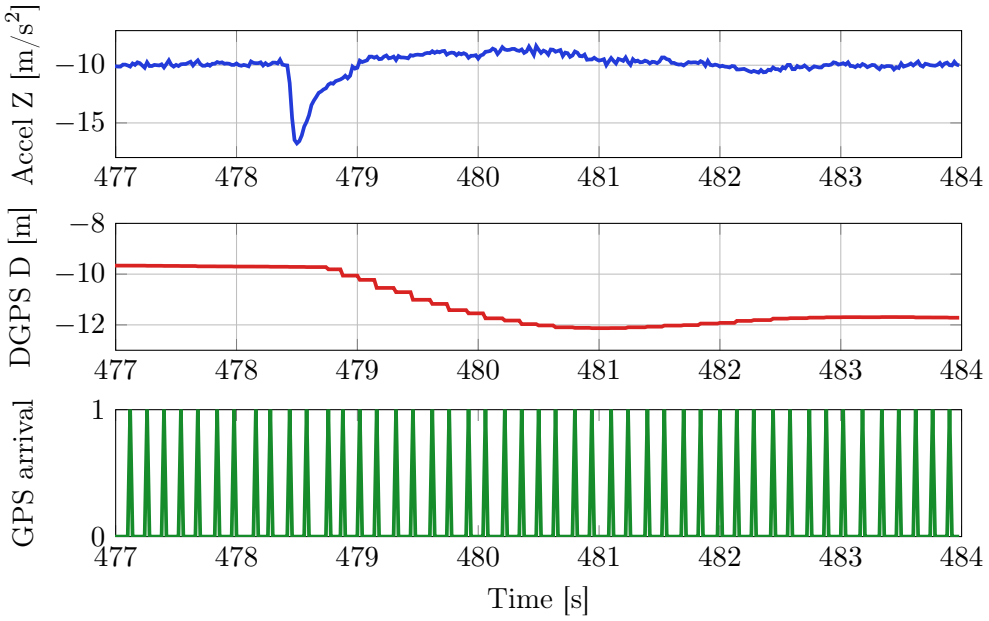


Figure 3.26: GPS position latency.

Table 3.6: GPS latency observed from heave steps in flight test

Heave step number	**Max. Position Latency** [ms]	**Min. Position Latency** [ms]	**Max. Velocity Latency** [ms]	**Min. Velocity Latency** [ms]
1	360	220	360	220
2	320	140	440	320
3	360	200	360	200
Bounds	**320**	**220**	**360**	**320**

Therefore, the latency of the GPS position is 270 ± 70 ms and the latency of the GPS velocity is 340 ± 40 ms. These values take the uncertainty in latency, due to the 20 ms sampling period of the accelerometers, into account.

3.6 Chapter Summary

Accurate state estimation requires an in-depth understanding of the noise characteristics of each sensor used in the estimation process. Thorough noise analyses were performed in this chapter for the gyroscope, accelerometer, magnetometer and GPS sensors used in this thesis. The Allan variance technique proved particularly useful in distinguishing the types of noise present on sensor measurements. However, autocorrelation was found to characterise periodic noise components, such as the periodic

spike in the magnetometer data, more easily. The GPS data exhibited measurement latency attributed to hardware problems. These problems were subsequently rectified, but due to time constraints the data captures could not be repeated. Instead, the latencies were quantified and removed from the data.

Simulation models of each of the sensors were created in Simulink based on the results of the noise analyses. These models will be used Chapter 7 to conduct realistic simulations of the estimator.

In the next chapter noise models for each of the relative sensors will developed.

Chapter 4

Relative Sensors

A helicopter requires relative sensors in order to determine the relative position and attitude of a ship. The various options available will be investigated in this chapter. The sensor options are short-listed based on their respective advantages and disadvantages. Thereafter, the short-listed sensors are modelled mathematically and stochastic noise models are developed. A full performance analysis of the sensors is conducted in Chapter 7.

4.1 Sensor Options

Monocular vision systems used for determining the position and rotation of the landing platform relative to a helicopter were developed by two previous students in the ESL, de Jager [11] and Swart [3]. Although [3] successfully demonstrated the monocular vision system in a flight test, a formal analysis of other sensors and combinations will ensure that the current sensor choice is optimal.

The feasibility of alternative relative state sensors is therefore investigated and the most feasible sensors are short-listed. Each sensor obtains measurements in a local reference frame. The general transformation of each sensor's measurements into a common relative reference frame follows.

4.1.1 Available Relative Sensors

Many sensors are capable of measuring the states of the ship deck relative to the helicopter. However, only the most important of these are summarised here.

Monocular Vision

Monocular vision is widely used for sensing the location of a landing platform relative to an unmanned helicopter [58; 59]. Information is lost when an image of a three-

dimensional scene is projected onto a two-dimensional image plane. This is explained further in Section 4.2. However, the ambiguity associated with the projection can be resolved by placing a known pattern on the landing platform [60]. This technique allows measurement of the relative translation and rotation of the ship deck.

Stereo Vision

As an alternative to monocular vision, stereo vision involves the use of two cameras to resolve the ambiguity of the projection of the scene. Consequently, the three dimensional structure of the landing platform can be obtained without the use of a known pattern on the platform. This enables a helicopter to land on any suitably-flat landing platform. However, the task of identifying the location of the landing platform is difficult without placing a known pattern on the landing platform, and would require GPS or other sensors to locate it. Furthermore, navy ship decks generally contain very little visual texture, which makes identification of corresponding points in each image difficult. Therefore, this project will investigate stereo vision using a known pattern on the deck. The additional knowledge of the baseline distance between the two cameras should improve the measurement accuracy compared to the monocular vision sensor.

Single Laser Rangefinder

Highly accurate distance measurements with excellent range and angle resolution are provided by laser rangefinders [61]. Accurate relative height measurements may be obtained by placing a downward-pointing laser rangefinder beneath an unmanned helicopter. Combining the laser distance measurements with previous state estimates enable relative height measurements to be calculated.

Multiple Laser Rangefinders

A laser rangefinder can be mounted to a rotating mechanism beneath the helicopter. The rapid rotation of the mechanism will allow the laser beam to trace a cone shape. The intersection of this cone and the landing platform (a plane) will generate an ellipse. The size and eccentricity of the ellipse can be used to measure the relative roll, pitch and height of the deck [13]. The scanning laser sensor, unlike the single laser rangefinder, relies very little on previously estimated states to obtain new measurements. Alternatively, multiple laser rangefinders can be used, as explained in Section 4.5.

Ultrasonic Rangefinder

Ultrasonic pulses transmitted by ultrasonic rangefinders and echoes are received if obstacles are located sufficiently close to the sensor. The time of flight of these pulses is used to determine the range of the obstacles. The beam width of an ultrasonic sensor is much larger than that of a laser rangefinder. Ultrasonic sensors are cost effective, however, their range resolution and angular resolution are insufficient for landing a helicopter on a ship deck. For this reason ultrasonic sensors will not be analysed further in this project.

Radar

Although in principle a radar resembles ultrasonic sensors, electromagnetic pulses are transmitted rather than sound waves. Radar beams have significant side lobes that can be minimised by careful antenna design, yet cannot be entirely removed. Similarly to ultrasonic sensors, the main lobe width is large, which limits the angular resolution. Thus a great deal of uncertainty is associated with radar distance measurements, which makes radar unsuitable for this task.

4.1.2 Short-Listed Sensors

The set of relative sensors that will be modelled for simulation are listed in Table 4.1. The CaspaPX is a camera developed specifically for the Gumstix microcomputer. This camera has been successfully used by Swart [3] for a similar purpose, that of measuring the relative states of the ship deck from the helicopter. For this reason the same camera will also be used here. Two CaspaPX camera units will be assumed for simulation of stereo vision.

Table 4.1: Relative sensor short-list.

Sensor type	**Model**
Monocular Vision	Gumstix CaspaPX
Stereo Vision	
Single Laser Rangefinder	SICK LMS-111 equivalent
Multiple Laser Rangefinders	

The SICK LMS-111 is a one-dimensional scanning laser sensor that has been installed on an all-terrain vehicle (ATV) in the ESL for research on simultaneous localisation and mapping (SLAM). This thesis seeks, in part, to determine the optimal set of sensors to use. However, physical implementation of all sensors is beyond the scope of this thesis. The laser sensor models described in this thesis will have the same

accuracy and noise characteristics as the LMS-111. Laser sensors of similar accuracy could be purchased if laser-based relative measurements are deemed necessary. In order to determine the optimal sensor suite, the performance of various combinations of sensors is analysed in Section 7.2.

4.2 Monocular Vision

Several previous projects in the ESL have investigated the use of monocular vision systems for the purpose of landing a helicopter. de Jager [11] and Swart [3] placed a number of easily identifiable markers on the landing surface. A single (monocular) camera mounted on the helicopter and facing downward was used to capture images of the markers. By matching the observed marker locations and the previously known 3D marker locations, the relative translation and orientation (i.e. the pose) of the helicopter and ship deck can be calculated. Figure 4.1 illustrates the concept of monocular vision-based pose estimation.

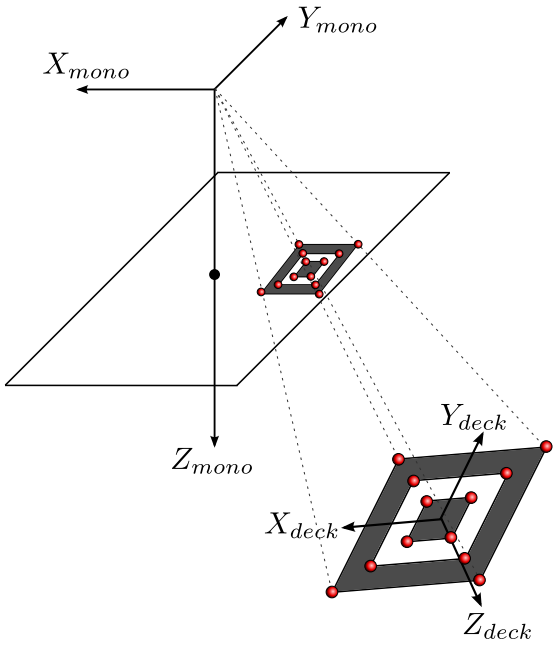
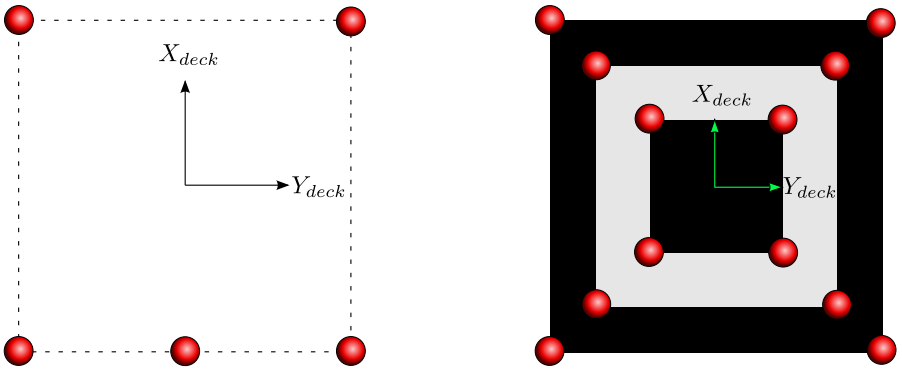


Figure 4.1: Monocular vison-based pose estimation.

[11] placed five markers in an asymmetric pattern on the landing surface to uniquely determine the relative pose. In contrast, [3] placed a set of three concentric squares on the landing surface. Image processing techniques are used to determine the corners of these square. Since the pattern is symmetrical, the pose that is measured is not unique. This was not considered a problem as the helicopter would be aligned to the

desired heading prior to landing. Diagrams of the patterns that have been used for monocular vision-based landing in the ESL are illustrated in Figure 4.2. Red points represent the markers used to estimate the pose.



(a) Pattern used by de Jager [11].

(b) Pattern used by Swart [3].

Figure 4.2: Patterns used for monocular vision pose estimation.

The singular value decomposition-based pose estimation algorithm used by [3] is highly flexible with regard to the marker pattern that can be used. In addition to the observed versatility, the algorithm has been successfully tested in flight. This pose estimation algorithm was therefore selected for simulation of the monocular vision sensor in this thesis.

This section begins with an explanation of the simulation of pattern marker images. Measurements of the pose of the camera relative to the pattern, and hence the landing platform, can then be determined from these simulated images. In order to determine the noise in the monocular vision measurements, a Monte Carlo simulation of the pose estimation algorithm is performed thereafter.

4.2.1 Simulating an Image of the Markers

The task of performing image processing to obtain the image coordinates of the markers is out of the scope of this project as previous projects in the ESL have investigated it thoroughly [11; 3]. As such, it is assumed that the image coordinates of the markers are already known.

The pinhole camera model is used without lens distortion. One of the goals of this thesis is to establish the performance of each available sensor in measuring the helicopter and ship deck states. The calculation of the relative translation and orientation of the helicopter and ship deck from the known image coordinates of the markers is of importance in this project and explained below.

In order to simulate the images of the markers, the translation $\mathbf{p}_{mono}$, and the Euler rotation $\boldsymbol{\theta}_{mono}$, of the deck relative to the camera, within the camera reference frame, are determined using (2.2.5). However, $\mathbf{p}_{sensor}$ and $\boldsymbol{\theta}_{sensor}$ are substituted for $\mathbf{p}_{mono}$ and $\boldsymbol{\theta}_{mono}$:

$$\begin{aligned} \mathbf{p}_{mono} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{offset}}(\mathbf{p}_{rel} - \mathbf{p}_{offset}) \\ \mathbf{DCM}_{\boldsymbol{\theta}_{mono}} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{offset}}\mathbf{DCM}_{\boldsymbol{\theta}_{rel}}. \end{aligned} \tag{4.2.1}$$

The Euler angles can then be extracted from the DCM matrix as follows, where $d_{j,k}$ is the j^{th} row and k^{th} column of $\mathbf{DCM}_{\boldsymbol{\theta}_{mono}}$:

$$\boldsymbol{\theta}_{mono} = \begin{bmatrix} \arctan2(d_{2,3}, d_{3,3}) \\ -\arcsin(d_{1,3}) \\ \arctan2(d_{1,2}, d_{1,1}) \end{bmatrix}. \tag{4.2.2}$$

Thereafter, the markers on the deck $\{\mathbf{x}^i_{deck}\}, i \in [1, \ldots, N_{markers}]$ are mapped from the deck's local reference frame into the camera reference frame as follows:

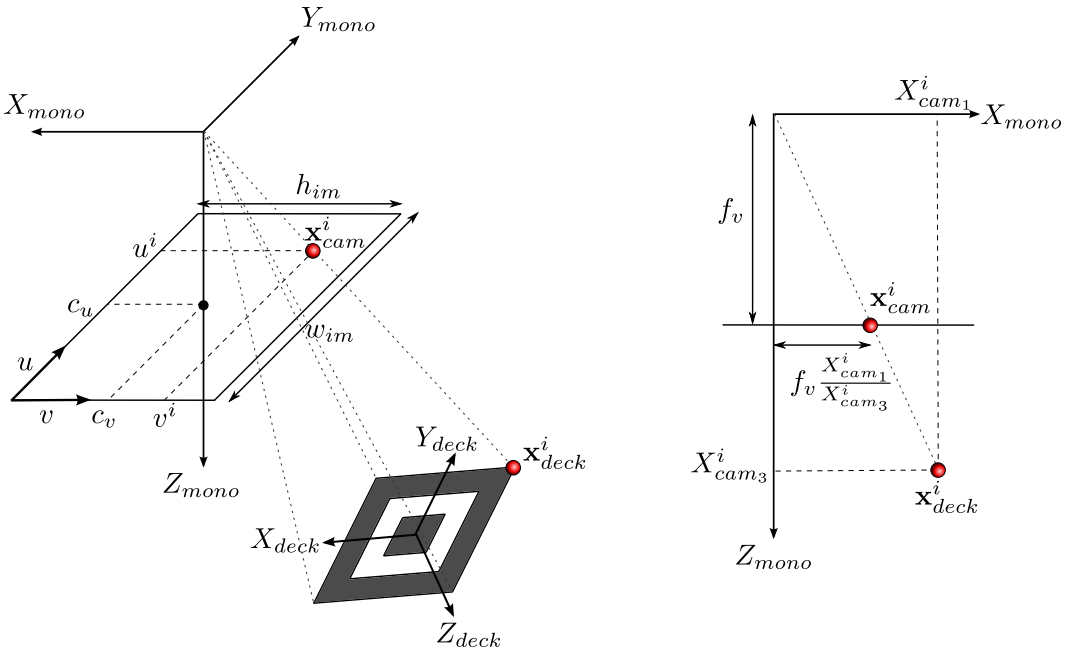
$$\begin{aligned} \mathbf{x}^i_{cam} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{mono}}\mathbf{x}^i_{deck} + \mathbf{p}_{mono} \\ &= \begin{bmatrix} \mathbf{DCM}^T_{\boldsymbol{\theta}_{mono}}\mathbf{p}_{mono} \end{bmatrix} \begin{bmatrix} \mathbf{x}^i_{deck} \\ 1 \end{bmatrix}. \end{aligned} \tag{4.2.3}$$

Since the markers lie upon the deck they have a height of zero within the deck reference frame. Thus, $x^i_{deck_3} = 0$, which permits (4.2.3) to be simplified. $h_{j,k}$ is the element of $\mathbf{DCM}^T_{\boldsymbol{\theta}_{mono}}$ at row j, column k:

$$\mathbf{x}^i_{cam} = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & p_{mono_1} \\ h_{2,1} & h_{2,2} & h_{2,3} & p_{mono_2} \\ h_{3,1} & h_{3,2} & h_{3,3} & p_{mono_3} \end{bmatrix} \begin{bmatrix} x^i_{deck_1} \\ x^i_{deck_2} \\ x^i_{deck_3} \\ 1 \end{bmatrix} = \begin{bmatrix} h_{1,1} & h_{1,2} & p_{mono_1} \\ h_{2,1} & h_{2,2} & p_{mono_2} \\ h_{3,1} & h_{3,2} & p_{mono_3} \end{bmatrix} \begin{bmatrix} x^i_{deck_1} \\ x^i_{deck_2} \\ 1 \end{bmatrix}. \tag{4.2.4}$$

Points within the camera reference frame are projected onto the image plane and converted into pixel coordinates, according to the pin-hole camera model (Figure 4.3). Below f_u and f_v is the focal length of the camera along the u and v image axes, and c_u and c_v is the center of the image along the u and v image axes:

$$\begin{aligned} v^i &= -f_v \; X^i_{cam_1}/X^i_{cam_3} + c_v \\ u^i &= f_u \; X^i_{cam_2}/X^i_{cam_3} + c_u. \end{aligned} \tag{4.2.5}$$



(a) 3D pinhole camera model.

(b) X-Z plane of pinhole camera model.

Figure 4.3: Projection of 3D points into 2D image coordinates. Modelled using pin-hole camera model.

These equations can be linearised by using homogeneous coordinates, otherwise known as projective coordinates. These coordinates are used to represent affine transformations as linear transformations [60]. The equations are formed into a camera projection matrix, $\mathbf{P}_{im}$:

$$\mathbf{P}_{im} = \begin{bmatrix} -f_v & 0 & c_v \\ 0 & f_u & c_u \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.2.6}$$

such that

$$w^i \begin{bmatrix} v^i \\ u^i \\ 1 \end{bmatrix} = \mathbf{P}_{im}\mathbf{x}^i_{cam}, \tag{4.2.7}$$

where $w^i = X^i_{cam_3}$; $f_u = w_{im}$; $f_v = h_{im}$; $c_u = {}^{w_{im}}/_2$ and $c_v = {}^{h_{im}}/_2$. The width is $w_{im} = 640$ and the height of the image is $h_{im} = 480$, measured in pixels.

4.2.2 Monocular Vision Measurements

Equations (4.2.4) and (4.2.7) can be combined to form a $(2N_{markers} \times 9)$ matrix $\mathbf{A}$. Rows $2i-1$ and $2i$ of $\mathbf{A}$ are given by [3]:

$$\begin{bmatrix} -f_v\ x_0 & -f_v\ y_0 & 0 & 0 & (c_v - v^i)x_0 & (c_v - v^i)y_0 & -f_v & 0 & (c_v - v^i) \\ 0 & 0 & f_u\ x_0 & f_u\ y_0 & (c_u - u^i)x_0 & (c_u - u^i)y_0 & 0 & f_u & (c_u - u^i) \end{bmatrix},$$

where $x_0 = x^i_{deck_1}$ and $y_0 = x^i_{deck_2}$, such that

$$\mathbf{A}\mathbf{x} = \mathbf{0}, \tag{4.2.8}$$

$$\mathbf{x} = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{2,1} & h_{2,2} & h_{3,1} & h_{3,2} & p_{mono_1} & p_{mono_2} & p_{mono_3} \end{bmatrix}^T.$$

Singular value decomposition (SVD) can thereafter be used to decompose $\mathbf{A}$ into rotation matrices $\mathbf{U}$ and $\mathbf{V}$, and a diagonal matrix $\mathbf{S}$, which consists of singular values in descending order along the diagonal [62]:

$$\mathbf{U}\ \mathbf{S}\ \mathbf{V}^T = \mathbf{A}. \tag{4.2.9}$$

The last column of $\mathbf{V}$ corresponds to the smallest singular value of $\mathbf{S}$ and is the null vector of $\mathbf{A}$. It is the solution $\tilde{\mathbf{x}}$ to the homogeneous equation (4.2.8). $\tilde{\mathbf{x}}$ will be known up to a scale factor of $\mathbf{x}$. In the absence of quantisation errors[1], $\tilde{\mathbf{x}} = \lambda\mathbf{x}$. The smallest singular value of $\mathbf{S}$ would therefore equal zero precisely. However, quantisation and finite word length exist in practice. As a result, $\tilde{\mathbf{x}} \approx \lambda\mathbf{x}$, such that $\tilde{\mathbf{x}}$ minimises $\|\mathbf{A}\tilde{\mathbf{x}}\|$ subject to $\|\tilde{\mathbf{x}}\| = 1$. This constraint prevents the trivial solution $\tilde{\mathbf{x}} = \mathbf{0}$.

Position and attitude measurements of the deck relative to the helicopter, $\tilde{\mathbf{p}}_{mono} = [\,T_1\ \ T_2\ \ T_3\,]^T$ and $\tilde{\boldsymbol{\theta}}_{mono} = [\,\phi\ \ \theta\ \ \psi\,]^T$, can then be extracted [3]:

$$\psi = \arctan2(\tilde{x}_3, \tilde{x}_1) \qquad T_1 = \frac{\tilde{x}_7}{\lambda}$$

$$\theta = -\arctan\left(\frac{\tilde{x}_5\ \cos(\psi)}{\tilde{x}_1}\right) \qquad T_2 = \frac{\tilde{x}_8}{\lambda}$$

$$\phi = \arcsin\left(\frac{\tilde{x}_6\ \cos(\psi)}{\tilde{x}_1}\right) \qquad T_3 = \frac{\tilde{x}_9}{\lambda}$$

$$\lambda = \frac{\tilde{x}_1}{\cos(\theta)\cos(\psi)}$$

These sensor measurements can then be transformed into position and attitude measurements of the deck within the relative reference frame by rearranging Equation

[1] Mapping the deck marker coordinates to integer pixel coordinates results in quantisation error.

(2.2.5):

$$\begin{aligned} \tilde{\mathbf{p}}_{rel} &= \mathbf{DCM}_{\boldsymbol{\theta}_{offset}} \tilde{\mathbf{p}}_{mono} + \mathbf{p}_{offset} \\ \mathbf{DCM}_{\tilde{\boldsymbol{\theta}}_{rel}} &= \mathbf{DCM}_{\boldsymbol{\theta}_{offset}} \mathbf{DCM}_{\tilde{\boldsymbol{\theta}}_{mono}}. \end{aligned} \tag{4.2.10}$$

Thereafter, the Euler angles can be extracted from the DCM matrix as follows, where $\tilde{d}_{j,k}$ is the j^{th} row and k^{th} column of $\mathbf{DCM}_{\tilde{\boldsymbol{\theta}}_{rel}}$:

$$\tilde{\boldsymbol{\theta}}_{rel} = \begin{bmatrix} \arctan2(\tilde{r}_{2,3}, \tilde{r}_{3,3}) \\ -\arcsin(\tilde{r}_{1,3}) \\ \arctan2(\tilde{r}_{1,2}, \tilde{r}_{1,1}) \end{bmatrix}. \tag{4.2.11}$$

4.2.3 Noise Analysis

Analysis of the physical monocular vision sensor noise is very difficult in practice. This is due to the capturing of many thousands of photos using the camera at various, precisely-known positions and orientations relative to the pattern [3]. Therefore, the simulation technique explained in [3] will be used instead.

This method involves the simulation of 10000 random rotations of the camera for varying height ratios and radius ratios. The height ratio h is defined as the height of the camera above the deck divided by the length of a side of the pattern on the deck d. The radius ratio r is defined as the horizontal radial distance of the camera to the origin of the deck divided by d. These lengths are depicted in Figure 4.4.

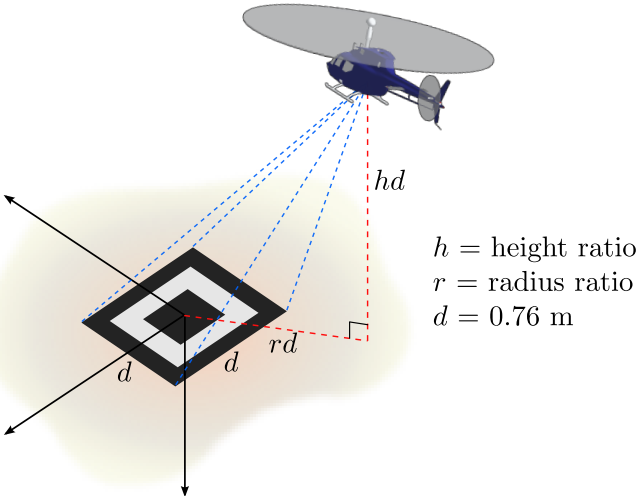


Figure 4.4: Definition of height and radius ratios.

The rotations were generated using a uniform random distribution in the range of -45° to 45° for each of the roll, pitch and yaw angles. Simulations were performed at 10 uniformly-spaced height ratios between 1 and 10 and 10 radius ratios between 0 and 5. The pattern side length is 0.78 m [3].

For each combination of height and radius ratio the root mean square error (RMSE) is calculated for each measured state:

$$RMSE = \sqrt{\left(\sum_{i=1}^{n} \frac{x_i - \tilde{x}_i}{n}\right)^2}. \tag{4.2.12}$$

The root mean square error (RMSE) of a variable x is related to the standard deviation of the error as follows:

$$\begin{aligned} RMSE(\hat{x}) &= \sqrt{MSE(\hat{x})} \\ &= \sqrt{(E[\hat{x} - x])^2 + E[(\hat{x} - E[\hat{x}])^2]} \\ &= \sqrt{\mu_{\hat{x}}^2 + \sigma_{\hat{x}}^2}, \end{aligned} \tag{4.2.13}$$

where $\mu_{\hat{x}}$ is the bias in the error and $\sigma_{\hat{x}}$ is the standard deviation of the error. Therefore, if the estimates of x are unbiased, the RMSE will equal the standard deviation of the error. If biases are present, the standard deviation alone will not reveal this form of error, and so the RMSE is used instead.

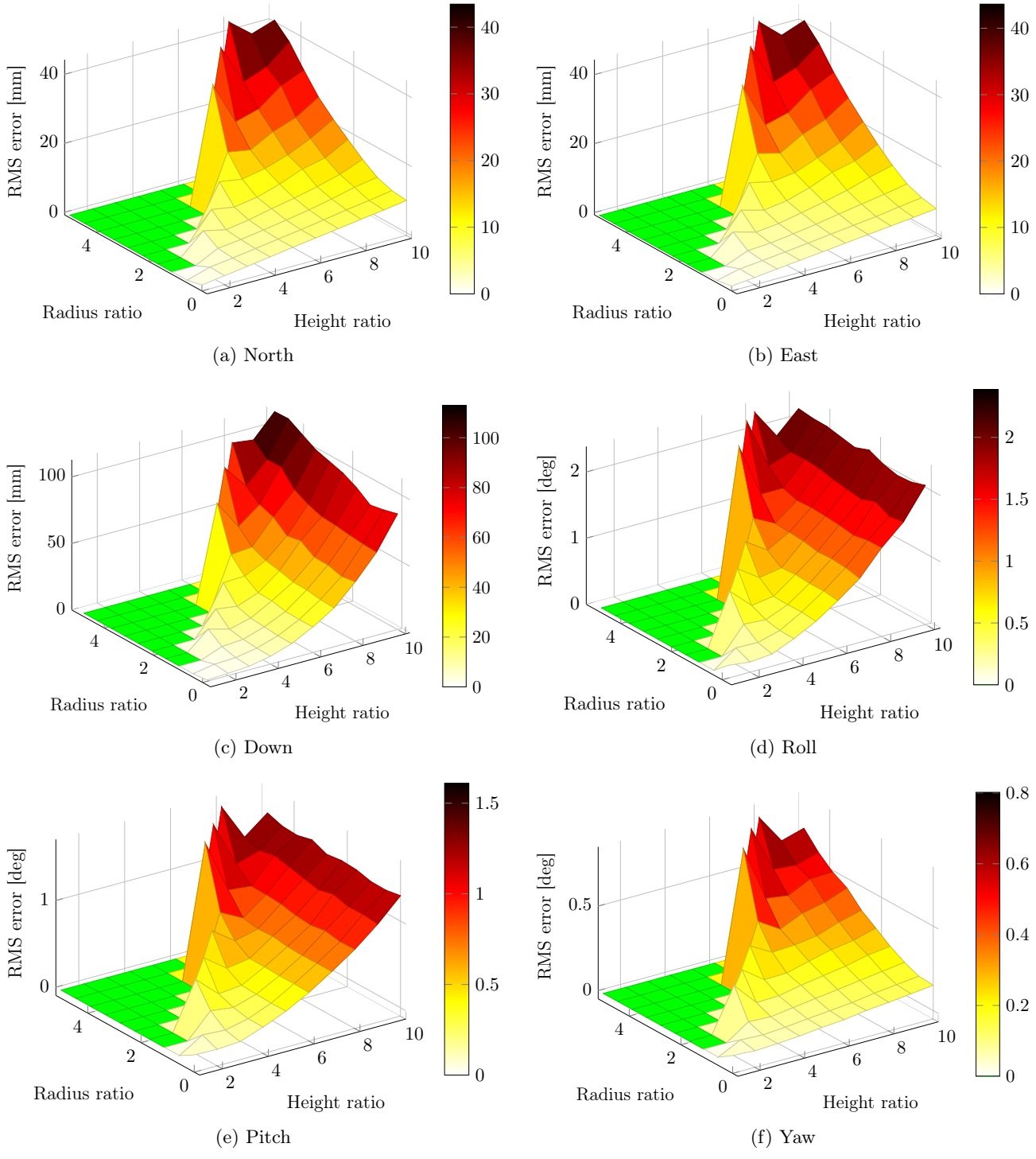
Figure 4.5 shows the RMSE in the monocular vision measurements as a function of the height ratio and radius ratio. As seen below, the green regions indicate that measurements were not obtained for the given height and radius ratios, since the deck pattern was not visible within the image plane.

The Down measurement is significantly less accurate than the North and East. This was to be expected of vision-based measurement as distance objects occupy fewer pixels in the image. Therefore, little change results in the image when motion occurs far from the camera. As height increases the accuracy of the roll and pitch degrades severely. Yaw angle is less affected by distance and so is measured more accurately than the roll and pitch. The results match those of [3] very closely, confirming the validity of the noise models.

A look up function can be implemented to determine the uncertainty in the monocular vision measurements as a function of the previous height ratio and radius ratio estimates. Second degree polynomials of the form

$$f(h, r) = a_{00} + a_{10}h + a_{01}r + a_{02}r^2 + a_{11}hr + a_{20}h^2 \tag{4.2.14}$$

are fitted to each of the measured states to prevent the need to store large arrays. Appendix B lists the coefficients of the look up functions.



(a) North (b) East

(c) Down (d) Roll

(e) Pitch (f) Yaw

Figure 4.5: RMS errors of monocular vision measurements

Noise covariance matrices for monocular vision position and attitude measurements can be constructed as follows:

$$\mathbf{R}_{\tilde{\mathbf{p}}_{mono}} = \begin{bmatrix} (\frac{f_N(h,r)}{1000})^2 & 0 & 0 \\ 0 & (\frac{f_E(h,r)}{1000})^2 & 0 \\ 0 & 0 & (\frac{f_D(h,r)}{1000})^2 \end{bmatrix}$$

$$\mathbf{R}_{\tilde{\boldsymbol{\theta}}_{mono}} = \begin{bmatrix} (f_\phi(h,r)\frac{\pi}{180})^2 & 0 & 0 \\ 0 & (f_\theta(h,r)\frac{\pi}{180})^2 & 0 \\ 0 & 0 & (f_\psi(h,r)\frac{\pi}{180})^2 \end{bmatrix}.$$

These noise covariance matrices are used in the estimation of relative states, which is described in Section 5.2.2. If the monocular vision sensor reference frame is perfectly aligned with the relative reference frame, then $\mathbf{R}_{\tilde{\mathbf{p}}_{rel}} = \mathbf{R}_{\tilde{\mathbf{p}}_{mono}}$ and $\mathbf{R}_{\tilde{\boldsymbol{\theta}}_{rel}} = \mathbf{R}_{\tilde{\boldsymbol{\theta}}_{mono}}$. In general, there is an attitude offset that needs to be taken into account:

$$\mathbf{R}_{\tilde{\mathbf{p}}_{rel}} = \mathbf{J}_{pos}\ \mathbf{R}_{\tilde{\mathbf{p}}_{mono}}\ \mathbf{J}_{pos}^T$$
$$\mathbf{R}_{\tilde{\boldsymbol{\theta}}_{rel}} = \mathbf{J}_{att}\ \mathbf{R}_{\tilde{\boldsymbol{\theta}}_{mono}}\ \mathbf{J}_{att}^T.$$

The calculation of the Jacobians is shown in Appendix D.9.

4.3 Stereo Vision

Stereo vision is an alternative method of camera-based pose estimation, whereby two side-by-side cameras are used to take photographs of the deck pattern from different perspectives, simultaneously. As illustrated in Figure 4.6, a 3D point observed in the left image will have the same vertical coordinate as in the right image. However, the horizontal coordinate of the observed points in their respective images would be different. This difference in pixel location is known as the disparity, measured in pixels, and is a function of the distance of the point from the cameras and the baseline distance between the cameras. Knowledge of the stereo camera geometry can be used to triangulate the point in 3D space [60].

4.3.1 Simulating an Image of the Markers

Similarly to the monocular vision sensor, the image processing procedure used to identify the deck pattern markers in the images is considered out of the scope of this project. The assumption is therefore made that the marker locations within the images are already known. Furthermore, the assumption is also made that the cameras' image planes have been rectified[2] and have a stereo baseline distance b.

[2]Image rectification is the process of aligning the image planes of the left and right cameras so that they are coplanar and vertically aligned. This procedure forms part of the stereo camera calibration procedure and is thoroughly explained in [63] and [60].

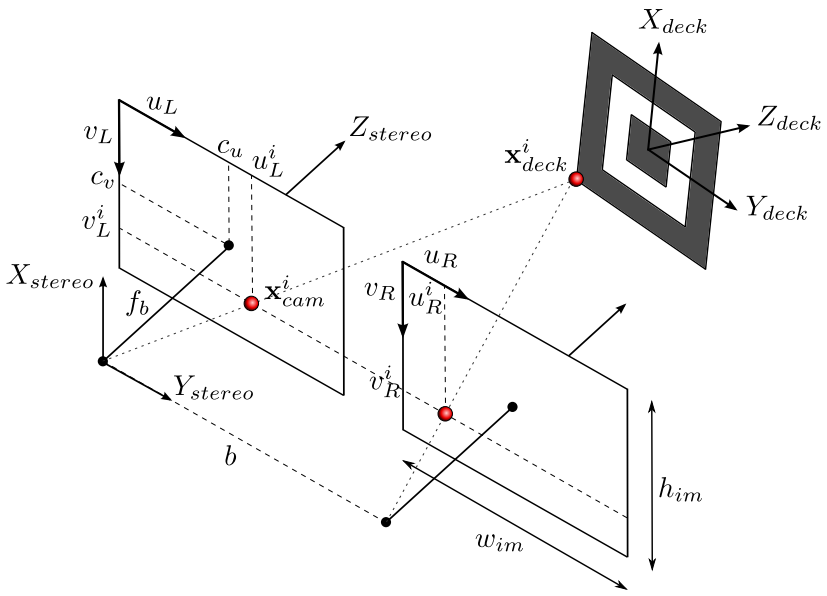


Figure 4.6: Reconstruction of 3D marker locations using stereo vision.

The true deck states within the stereo vision sensor's reference frame are determined by Equation (2.2.5):

$$\begin{aligned} \mathbf{p}_{stereo} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{offset}}(\mathbf{p}_{rel} - \mathbf{p}_{offset}) \\ \mathbf{DCM}_{\boldsymbol{\theta}_{stereo}} &= \mathbf{DCM}^T_{\boldsymbol{\theta}_{offset}}\mathbf{DCM}_{\boldsymbol{\theta}_{rel}}. \end{aligned} \tag{4.3.1}$$

Additionally, Equation (4.2.3) is used to transform the deck markers into the monocular vision sensor's reference frame and can be used in the same way for the stereo vision sensor. The left camera is chosen to be at the same location as the monocular vision sensor for convenience of notation.

The deck marker projected into the left and right image planes are therefore:

$$\begin{aligned} w^i_L \begin{bmatrix} v^i_L \\ u^i_L \\ 1 \end{bmatrix} &= \mathbf{P}_{im}\ \mathbf{x}^i_{cam} \\ w^i_R \begin{bmatrix} v^i_R \\ u^i_R \\ 1 \end{bmatrix} &= \mathbf{P}_{im}\ (\mathbf{x}^i_{cam} - \begin{bmatrix} 0 \\ b \\ 0 \end{bmatrix}). \end{aligned} \tag{4.3.2}$$

Coordinates are rounded off to the nearest integer value as images are discretised into pixels.

4.3.2 Stereo Vision Measurements

The locations of the markers within the camera reference frame can be triangulated using image coordinates of the markers and knowledge of the epipolar geometry. The image coordinates of the markers are first mapped onto the left image plane $[\,x_L^i \;\; y_L^i \;\; 0\,]^T$ and right image plane $[\,x_R^i \;\; y_R^i \;\; 0\,]^T$ within the corresponding camera reference frames:

$$\begin{aligned} y_L^i = y_R^i &= \frac{v_R^i - \frac{h_{im}}{2}}{-f_v} \\ x_L^i &= \frac{u_L^i - \frac{w_{im}}{2}}{f_u} \\ x_R^i &= \frac{u_R^i - \frac{w_{im}}{2}}{f_u} \end{aligned} \tag{4.3.3}$$

and then triangulated within the left camera reference frame:

$$\tilde{\mathbf{x}}_{cam}^i = \begin{bmatrix} \frac{f_b \;\; b}{x_L^i - x_R^i} & \frac{x_L^i \;\; b}{x_L^i - x_R^i} & \frac{y^i \;\; b}{x_L^i - x_R^i} \end{bmatrix}^T. \tag{4.3.4}$$

SVD can then be used to find the rigid transformation that relates the camera reference frame to the deck reference frame in a similar procedure to the monocular vision sensor. Let $x_0^i = x_{deck_1}^i$ and $y_0^i = x_{deck_2}^i$:

$$\tilde{\mathbf{x}}_{cam}^i = \begin{bmatrix} h_{1,1} & h_{1,2} & p_{stereo_1} \\ h_{2,1} & h_{2,2} & p_{stereo_2} \\ h_{3,1} & h_{3,2} & p_{stereo_3} \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \tag{4.3.5}$$

(4.3.5) can be rearranged in the following way:

$$\tilde{\mathbf{x}}_{cam}^i = \begin{bmatrix} x_0^i & y_0^i & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & x_0^i & y_0^i & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & x_0^i & y_0^i & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} h_{1,1} \\ h_{1,2} \\ h_{2,1} \\ h_{2,2} \\ h_{3,1} \\ h_{3,2} \\ p_{rel_1} \\ p_{rel_2} \\ p_{rel_3} \end{bmatrix} = \mathbf{a}^i \; \mathbf{x}. \tag{4.3.6}$$

Matrix $\mathbf{A}$ can be constructed for the complete set of N visible markers:

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{4.3.7}$$

where

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}^1 \\ \mathbf{a}^2 \\ \vdots \\ \mathbf{a}^N \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} \tilde{\mathbf{x}}^1_{cam} \\ \tilde{\mathbf{x}}^2_{cam} \\ \vdots \\ \tilde{\mathbf{x}}^N_{cam} \end{bmatrix}. \tag{4.3.8}$$

The solution to (4.3.7) is found in a different manner to (4.2.8), since it is non-homogenous. SVD can be used to find a least-squares solution

$$\mathbf{x} = \mathbf{A}^{+}\mathbf{b} \approx \mathbf{V}\ \mathbf{S}_{inv}\ \mathbf{U}^T\ \mathbf{b}, \tag{4.3.9}$$

where $\mathbf{A}^{+}$ is the pseudoinverse of $\mathbf{A}$. $\mathbf{S}_{inv}$ is calculated by inverting the diagonal elements of $\mathbf{S}$:

$$\mathbf{S}_{inv} = \begin{cases} 1/\mathbf{s}_{i,j} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

In order to solve (4.3.7) uniquely, $\mathbf{A}$ must be of rank 9. Three or more deck markers must therefore be visible in both camera views in order for this condition to be met.

Equations (2.2.5) and (4.2.11) can then be used to determine the deck's position and attitude within the relative reference frame.

4.3.3 Noise Analysis

Noise analysis of the stereo vision sensor is performed in the same manner as the monocular vision sensor. The stereo baseline b was chosen as 1 metre, as larger baselines are impractical for the size of the ESL helicopter.

Figure 4.7 illustrates the Monte Carlo simulations for each of the stereo vision state measurements. The stereo vision triangulation method requires the observation of markers in both cameras' image planes. The region of height ratios and radius ratios for which the deck pattern is visible is thus a subset of the monocular vision method. In particular, a blindspot occurs when the helicopter is hovering slightly above the deck pattern. The blindspot region can be reduced by choosing lenses with a larger field of view or by reducing the size of the deck pattern. However, this will reduce the measurement accuracy of the sensor. This is due to the increase in quantisation noise, since a single pixel will correspond to a larger region in 3D space. An increase in measurement accuracy can be achieved by increasing the resolution of the cameras.

However, this results in longer image processing times. An increase in the baseline distance b would also improve the measurement accuracy. This would be impractical for the size of the XCell helicopter used in the ESL and would also increase the blindspot region.

Table 4.2 below compares the accuracy of the monocular vision and stereo vision sensors at a height ratio of 6 and a radius ratio of 0. Stereo vision shows a noticeable improvement in accuracy over monocular vision, particularly in relative position measurement. This corresponds to a height of 4.68 m above the deck for the pattern used by Swart [3].

Table 4.2: Comparison of monocular and stereo vision.

Sensor type	**N** [mm]	**E** [mm]	**D** [mm]	**Roll** [°]	**Pitch** [°]	**Yaw** [°]
Monocular vision	5.6	4.2	19.2	0.72	0.46	0.09
Stereo vision	1.2	0.8	17.5	0.62	0.44	0.11

Two-dimensional second order polynomials are fitted to the noise plots of each of the measured states in the same manner as that of the monocular vision sensor, as shown in Appendix B.

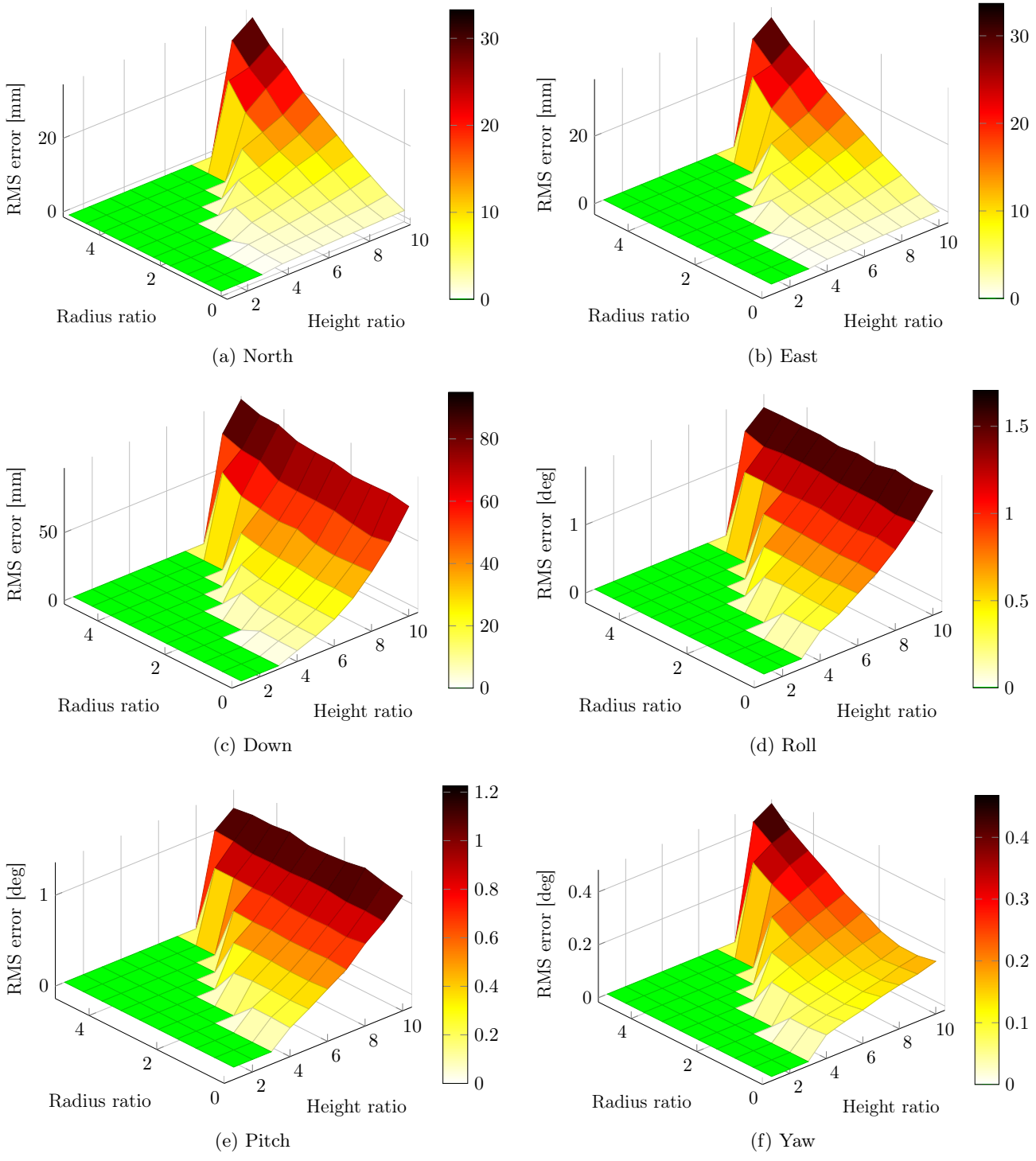
4.4 Single Laser Rangefinder

A single laser rangefinder can be used to provide relative height measurements of the ship deck. The deck can translate and rotate in three dimensions. However, only one distance value is returned by the sensor, which means that previous deck estimates need to be used to calculate the relative height of the helicopter above the ship deck.

In order to simulate laser rangefinder measurements, the ship deck is modelled as a plane in 3D space. The intersection of the laser beam and the plane is determined thereafter. In addition, realistic measurement noise is added to the distance readings.

4.4.1 Deck Plane Equation

The equation of the plane used to model the ship deck is first calculated.



(a) North (b) East

(c) Down (d) Roll

(e) Pitch (f) Yaw

Figure 4.7: RMS errors of stereo vision measurements

The position $\mathbf{p}_{laser}$ and attitude $\boldsymbol{\theta}_{laser}$ of the ship deck within the laser reference frame are determined using Equation (2.2.5), with $\boldsymbol{\theta}_{sensor} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$:

$$\begin{aligned} \mathbf{p}_{laser} &= \mathbf{p}_{rel} - \mathbf{p}_{offset} \\ \mathbf{DCM}_{\boldsymbol{\theta}_{laser}} &= \mathbf{DCM}_{\boldsymbol{\theta}_{rel}}. \end{aligned} \tag{4.4.1}$$

The ship deck plane will have a normal

$$\mathbf{n}_{laser} = \frac{\left(\mathbf{DCM}^T_{\boldsymbol{\theta}_{laser}} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}\right) \times \left(\mathbf{DCM}^T_{\boldsymbol{\theta}_{laser}} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}\right)}{\left\| \left(\mathbf{DCM}^T_{\boldsymbol{\theta}_{laser}} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}\right) \times \left(\mathbf{DCM}^T_{\boldsymbol{\theta}_{laser}} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}\right) \right\|}$$

within the laser sensor's reference frame. The deck plane equation is formed as follows:

$$n_{laser_1} x + n_{laser_2} y + n_{laser_3} z = d_{laser},$$

where

$$d_{laser} = \mathbf{n}_{laser} \cdot \mathbf{p}_{laser}.$$

The deck plane equation can be used to simulate measurements of the laser rangefinders.

4.4.2 Simulation of Laser Rangefinder Measurement

The laser beam is pointed downward from the helicopter in order to obtain deck height estimates. The distance measurement can be simulated by finding the distance from the laser rangefinder to the deck in the direction of the laser beam. The laser rangefinder is located at the origin of the laser reference frame, by definition, and the laser beam will have a direction $\mathbf{m}_{laser} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$. The noise-free measured distance can then be found:

$$\begin{aligned} d &= \frac{\left(\mathbf{p}_{laser} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}\right) \cdot \mathbf{n}_{laser}}{\mathbf{m}_{laser} \cdot \mathbf{n}_{laser}} \\ &= \frac{\mathbf{p}_{laser} \cdot \mathbf{n}_{laser}}{n_{laser_3}}. \end{aligned} \tag{4.4.2}$$

However, the actual laser beam diverges and so a conical beam model is used to simulate the laser beam more realistically. The laser rangefinder has a beam divergence θ_{beam} of 0.015 radians [64]. The closest point of intersection of the resultant cone-shaped beam and the ship deck is the distance measured by a laser rangefinder.

Figure 4.8 illustrates the conical beam model. The true distance d and the closest distance $d_{closest}$ to the deck are indicated. θ_{deck} represents the scalar angle between the laser beam direction $\mathbf{m}_{laser}$ and deck normal $\mathbf{n}_{laser}$. The laser beam originates from

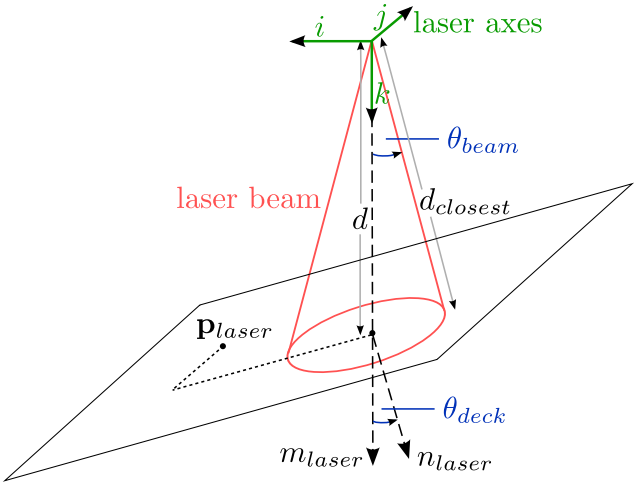


Figure 4.8: Conical laser beam model.

the laser reference frame origin. $\mathbf{p}_{laser}$ is shown to emphasise the laser rangefinder's point of intersection with the deck as an arbitrary point on the deck.

θ_{deck} can be determined using the dot product:

$$\begin{aligned} \mathbf{n}_{laser} \cdot \mathbf{m}_{laser} &= \|\mathbf{m}_{laser}\| \ \|\mathbf{n}_{laser}\| \cos(\theta_{deck}) \\ \theta_{deck} &= \arccos\Big(\frac{\mathbf{n}_{laser} \cdot \mathbf{m}_{laser}}{\|\mathbf{m}_{laser}\| \ \|\mathbf{n}_{laser}\|}\Big). \end{aligned} \tag{4.4.3}$$

$d_{closest}$ is calculated as follows:

$$\begin{aligned} d_{closest} &= \frac{d \ \sin(\pi/2 - \theta_{deck})}{\sin(\pi/2 - \theta_{beam} + \theta_{deck})} \\ &= \frac{\cos(\theta_{deck})}{\cos(\theta_{beam} - \theta_{deck})}. \end{aligned} \tag{4.4.4}$$

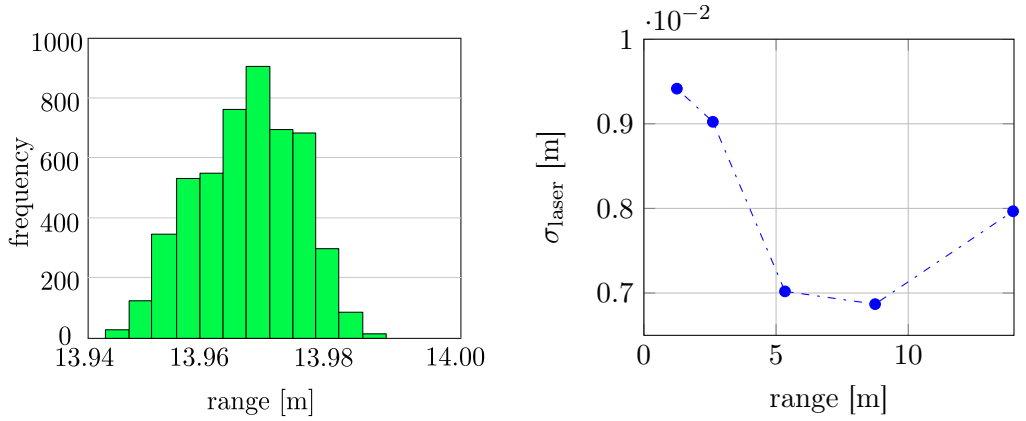
If the laser rangefinder were $d = 10$ m above the ship deck and oriented at an angle of $\theta_{deck} = \pi/2$ relative to the deck, the difference in distance between d and $d_{closest}$ will be 13.7 cm. It is therefore important to simulate laser rangefinder readings using the conical beam model, especially in the case of large θ_{deck} and d.

The laser rangefinder distance measurement can now be determined. Let v_{laser} be the noise on the laser rangefinder measurement:

$$\begin{aligned} \tilde{d}_{intersect} &= d_{closest} + v_{laser} \\ &= \frac{\cos(\theta_{deck})}{\cos(\theta_{beam} - \theta_{deck})} + v_{laser}. \end{aligned} \tag{4.4.5}$$

A practical test of the LMS111 laser sensor was performed in order to determine the sensor noise. The LMS111 was placed at various distances from a flat wall and 5000

laser readings were recorded for each distance. Five distances were tested in the range of 1.2 to 14 meters. Figure 4.9a shows a histogram of the measurements for a specific range of distances. The laser measurements are seen to be approximately Gaussian-distributed and therefore be modelled as such. Figure 4.9b shows the



(a) Histogram of rangefinder readings.

(b) Measurement noise of laser rangefinder.

Figure 4.9: Laser rangefinder noise characteristics.

standard deviation of the laser rangefinder readings as a function of the range to the wall. Despite a large range of distances tested, the laser rangefinder accuracy is essentially unaffected. Therefore, the noise on the laser rangefinder readings can be modelled as a range-independent Gaussian distribution, with $\sigma_{laser} = 8 \times 10^{-3}$ m:

$$v_{laser} \in \mathcal{N}(0, \sigma^2_{laser}).$$

The LMS111 datasheet states that the noise on the sensor has a standard deviation of 12 mm, which is very close to that observed in practice. The noise may vary for different surfaces because the energy of the reflection depends on the surface properties [64].

4.4.3 Measurement of Relative Down State

A single laser rangefinder outputs a single relative distance measurement. If the deck and helicopter were both level, this distance measurement could be used to update the relative Down state estimate directly. Since this will not generally be the case, previous relative state estimates are used to resolve the laser sensor's relative Down measurement.

The helicopter and ship are orientated at arbitrary attitudes relative to one another. Thus, the laser sensor is used to correct the latest estimated relative Down estimate.

The laser is used to measure the difference in distance between the measured point of intersection of the laser beam and the deck, and the point of intersection of the laser and previously estimated ship deck. If the deck attitude remains unchanged between the previous estimate and the laser measurement being taken, this difference will be equal to the difference between the current relative Down state and the latest relative Down state estimate. This difference, $\tilde{\delta}$, is subsequently added to the latest relative Down estimate to obtain a relative Down measurement. This measurement is the final measurement supplied by the laser rangefinder sensor.

The LMS111 laser scanner used in the ESL outputs measurements every 20 ms. The ship deck motion recordings presented in [4] reveal that the roll and pitch underwent a maximum change of $3°/s$ in heavy swell. This amounts to a heave of 7 cm due to the translation of the deck from the pivot point of the ship [4]. As a result a 7 cm bias would be introduced into the relative Down laser measurement. However, this would occur very briefly under worst-case conditions.The assumption of constant deck attitude is thus deemed satisfactory.

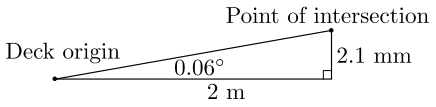


Figure 4.10: Change in deck attitude between laser measurements.

The point of intersection measured in the laser reference frame is $\begin{bmatrix} 0 \\ 0 \\ \tilde{d}_{intersect} \end{bmatrix}$, which can be mapped into the relative reference frame:

$$\tilde{\mathbf{p}}_{intersect} = \begin{bmatrix} 0 \\ 0 \\ \tilde{d}_{intersect} \end{bmatrix} + \mathbf{p}_{offset}. \tag{4.4.6}$$

The normal of the deck plane within the relative reference frame can be obtained using the relative attitude estimates:

$$\hat{\mathbf{n}}_{rel} = \frac{\left(\mathbf{DCM}^T_{\hat{\boldsymbol{\theta}}_{rel}} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}\right) \times \left(\mathbf{DCM}^T_{\hat{\boldsymbol{\theta}}_{rel}} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}\right)}{\left\| \left(\mathbf{DCM}^T_{\hat{\boldsymbol{\theta}}_{rel}} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}\right) \times \left(\mathbf{DCM}^T_{\hat{\boldsymbol{\theta}}_{rel}} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}\right) \right\|}. \tag{4.4.7}$$

The equation for the estimated plane is therefore

$$\hat{n}_{rel_1} x + \hat{n}_{rel_2} y + \hat{n}_{rel_3} z = \hat{d}_{rel}, \tag{4.4.8}$$

where

$$\hat{d}_{rel} = \hat{\mathbf{n}}_{rel} \cdot \hat{\mathbf{p}}_{rel}. \tag{4.4.9}$$

The measured point of intersection $\tilde{\mathbf{p}}_{intersect}$ of the laser beam and the deck can be projected vertically onto the estimated deck plane within the relative reference

frame. The estimated height of the deck at this location is

$$\hat{p}_{intersect_3} = \frac{\hat{d}_{rel} - \hat{n}_{rel_1} \cdot \tilde{p}_{intersect_1} - \hat{n}_{rel_2} \cdot \tilde{p}_{intersect_2}}{\hat{n}_{rel_3}}. \tag{4.4.10}$$

The ship deck's Down state within the relative reference frame p_{rel_3} can be updated using measurement $\tilde{p}_{rel_3}$:

$$\begin{aligned} \tilde{\delta} &= \tilde{p}_{intersect_3} - \hat{p}_{intersect_3} \\ \tilde{p}_{rel_3} &= \hat{p}_{rel_3} + \tilde{\delta}, \end{aligned} \tag{4.4.11}$$

where $\hat{p}_{rel_3}$ is the most recent Down estimate. The relative attitude cannot be measured using only one laser rangefinder.

Figure 4.11 shows the geometry of the single laser rangefinder measurement.

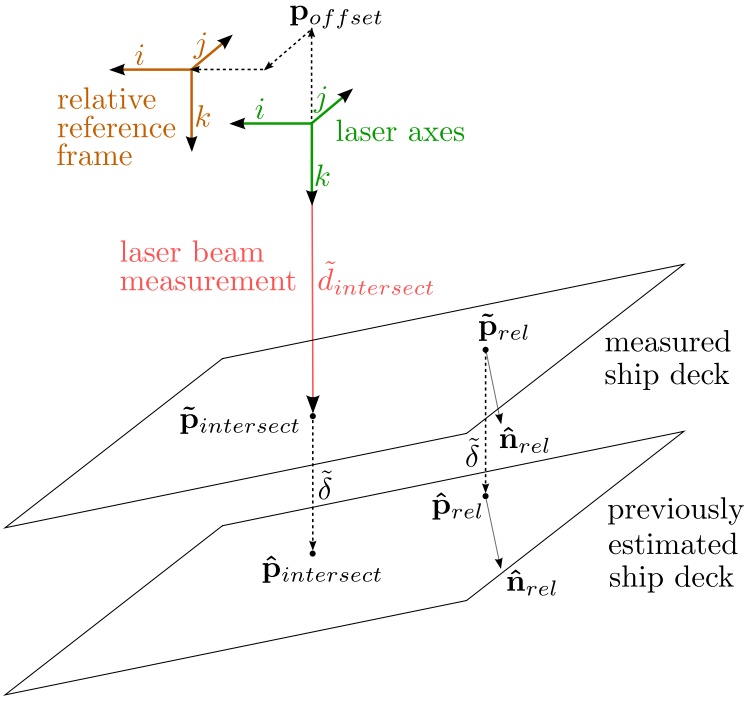


Figure 4.11: Measuring relative Down position using a single laser rangefinder.

4.4.4 Noise Analysis

Errors in the laser sensor relative Down measurements result from Gaussian-distributed white noise on the laser rangefinder readings, biases resulting from the conical shape of the laser beams that are not completely corrected for and noise in the relative attitude estimates used to calculate the deck plane equation.

The measurement noise is dependent on the distance of the center of the laser beam d and the angle between the laser beam and the deck θ_{deck}. 20 000 simulations of distance readings were performed at 10 distances ranging from 1 to 20 metres. In addition, simulations were performed at 10 angles ranging from 0 to 45 degrees. Gaussian noise with a standard deviation of 0.34 degrees, which resembles that observed in practice, was added to the simulated relative Euler angle estimates.

The relative Down position measurement $\tilde{p}_{rel_3}$ is calculated by adding the previously estimated relative Down estimate $\hat{p}_{rel_3}$ to the measured delta value $\tilde{\delta}$, as shown in Equation (4.4.11):

$$\tilde{p}_{rel_3} = \hat{p}_{rel_3} + \tilde{\delta}.$$

The variance on the measurement is

$$\sigma^2_{\tilde{p}_{rel_3}} = \sigma^2_{\hat{p}_{rel_3}} + \sigma^2_{\tilde{\delta}}.$$

$\sigma^2_{\hat{p}_{rel_3}}$ is obtained from the weighted mean of previous relative estimates, which is

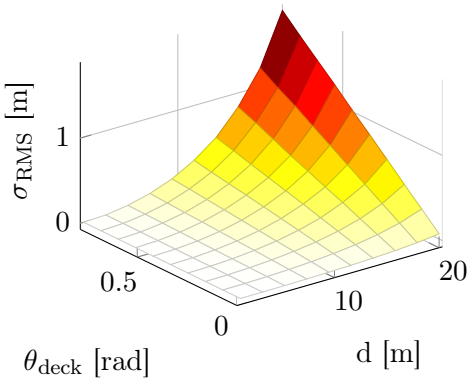


Figure 4.12: Noise on $\tilde{p}_{intersect_3}$ measurement.

explained in Section 5.2.2. Figure 4.12 shows the RMS error on the measured point of intersection of the laser beam and the deck, $\tilde{p}_{intersect_3}$, as a function of d and θ_{deck}. Imperfect relative estimates prevent the true distance measurements from being extracted when inverting the conical beam model. It is thus demonstrated that noise increases as d and θ_{deck} increase together. This is expected when using the conical beam model.

A second order polynomial surface is fitted to the noise plot in the same manner as performed for the vision sensors. The parameters of the fitted surface are shown in Appendix B.

4.5 Multiple Laser Rangefinders

Three or more laser rangefinders originating from the same location and pointing in different directions can be used to measure the Down, roll and pitch states of the deck relative to the helicopter.

The scope of this project is limited to the selection of the sensors necessary to land the helicopter on a ship deck and not the physical implementation of the sensors. As such, the derivation of the sensor model is kept general. This enables the multiple laser sensor to be practically implemented in different ways. For example, this could be done using a set of four single laser-rangefinders mounted onto the helicopter or a single scanning laser rangefinder that effectively generates hundreds of laser beams, as implemented by [13].

4.5.1 Simulation of Laser Rangefinder Measurements

Laser rangefinder measurements are calculated similarly to the single laser model. The calculations are shown for laser beam $i \in (1, N)$, where N is the number of laser beams.

The laser beams are pointed at equal angles along the surface of an imaginary cone. Laser beam i will have a direction

$$\mathbf{m}^i_{laser} = \frac{1}{\sqrt{1+\cot^2\alpha}} \begin{bmatrix} \sin(\psi^i_{laser}) \\ \cos(\psi^i_{laser}) \\ \cot(\alpha) \end{bmatrix},$$

where $\psi^i_{laser} = \frac{i2\pi}{N}$ is the angle, in radians, of laser beam i. The N laser beams trace the surface of a cone. α is the angle between the surface of the cone and its center, as shown in Figure 4.13 below. The true distance from laser rangefinder i to the ship

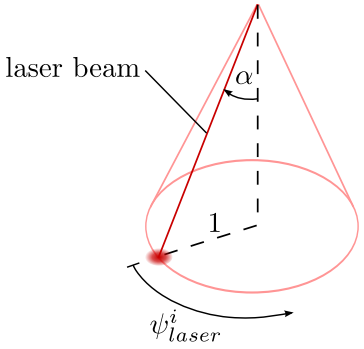


Figure 4.13: Multiple laser beams trace the surface of a cone.

deck can then be calculated:

$$\begin{aligned} d^i &= \frac{\left(\mathbf{p}_{laser} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}\right) \cdot \mathbf{n}_{laser}}{\mathbf{m}^i_{laser} \cdot \mathbf{n}_{laser}} \\ &= \frac{\mathbf{p}_{laser} \cdot \mathbf{n}_{laser}}{\mathbf{m}^i_{laser} \cdot \mathbf{n}_{laser}} \end{aligned} \tag{4.5.1}$$

Equations (4.4.3) and (4.4.5) are then used to simulate the distance measurement $\tilde{d}^i_{intersect}$ by applying the conical beam model to the true distance d^i and adding Gaussian measurement noise.

4.5.2 Measurement of Relative Down, Roll and Pitch States

The relative Down measurement can be calculated in a similar manner to that of the single laser rangefinder. The equation of the deck plane is then calculated in order to measure the relative roll and pitch states. This is followed by the calculation of the relative roll and pitch measurements using the measured deck plane. The choice of the tapering of the cone shape traced by the multiple laser rangefinder sensor is then discussed.

Down Position Measurement

Distance measurements returned by the laser rangefinder contain biases due to the conical beam model used to model the laser beams. The latest estimated deck attitude states are used to correct for the biases in the distance measurements. Doing so prevents these biases affecting the relative Down, roll and pitch measurements. This is achieved by inverting the conical beam model using an estimation of the transform:

$$\hat{d}^i = \frac{\tilde{d}^i_{intersect} \cos(\theta_{beam} - \hat{\theta}_{deck})}{\cos(\hat{\theta}_{deck})}. \tag{4.5.2}$$

Since the laser beam directions are known for the distance measurements, their corresponding points of intersection with the ship deck within the relative reference frame can be determined:

$$\tilde{\mathbf{p}}^i_{intersect} = \hat{d}^i \ \mathbf{m}^i_{laser} + \mathbf{p}_{offset}. \tag{4.5.3}$$

The multiple measured points of intersection can be averaged to reduce the presence of noise:

$$\bar{\mathbf{p}}_{intersect} = \frac{1}{N} \sum_{i=1}^{N} \tilde{\mathbf{p}}^i_{intersect}. \tag{4.5.4}$$

The average measured point of intersection of the laser beam and the deck can then be projected vertically onto the previously estimated deck plane (see Figure 4.14),

which is calculated using Equation (4.4.8). The height of the deck at this location is

$$\hat{p}_{intersect_3} = \frac{\hat{d}_{rel} - \hat{n}_{rel_1} \cdot \bar{p}_{intersect_1} - \hat{n}_{rel_2} \cdot \bar{p}_{intersect_2}}{\hat{n}_{rel_3}}. \tag{4.5.5}$$

The ship deck's Down state can be updated using measurement $\tilde{p}_{rel_3}$:

$$\begin{aligned} \tilde{\delta} &= \bar{p}_{intersect_3} - \hat{p}_{intersect_3} \\ \tilde{p}_{rel_3} &= \hat{p}_{rel_3} + \tilde{\delta} \end{aligned} \tag{4.5.6}$$

where $\hat{p}_{rel_3}$ is the most recent relative Down estimate.

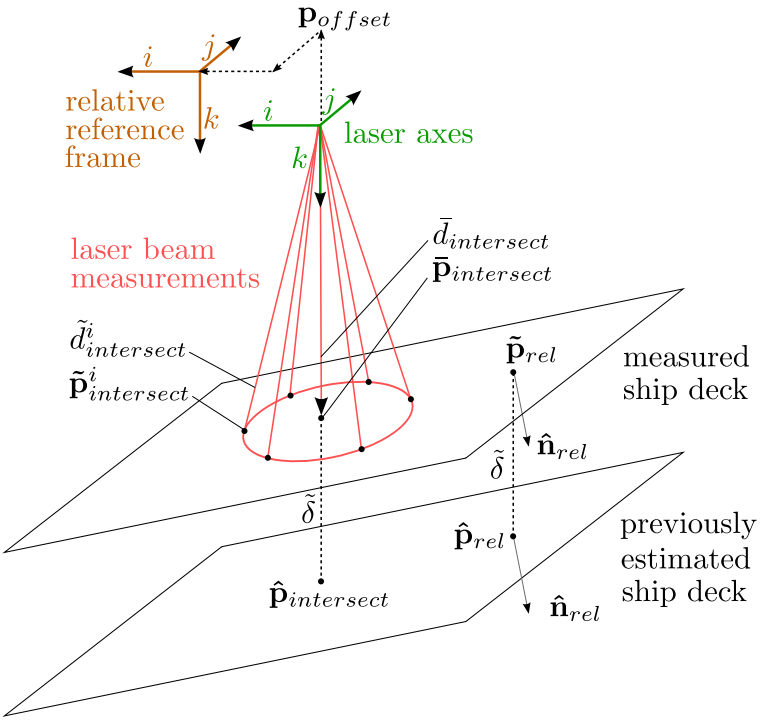


Figure 4.14: Measuring relative Down position using multiple laser rangefinders.

Determine Equation of Deck using Points of Intersection

The deck plane equation can be measured using three or more laser beam distance values.

The measured deck plane equation within the relative reference frame can be written as follows, assuming that no noise is present:

$$\tilde{\mathbf{p}}^i_{intersect} \cdot \mathbf{n}_{rel} = d_{rel}. \tag{4.5.7}$$

The above equation can be rearranged by grouping the unknowns together in one vector:

$$\begin{bmatrix} \tilde{\mathbf{p}}^{i}_{intersect}{}^{T} & -1 \end{bmatrix} \begin{bmatrix} \mathbf{n}_{rel} \\ d_{rel} \end{bmatrix} = 0. \tag{4.5.8}$$

Now let

$$\mathbf{A} = \begin{bmatrix} \tilde{\mathbf{p}}^{1}_{intersect}{}^{T} & -1 \\ \tilde{\mathbf{p}}^{2}_{intersect}{}^{T} & -1 \\ \vdots & \vdots \\ \tilde{\mathbf{p}}^{N}_{intersect}{}^{T} & -1 \end{bmatrix} \tag{4.5.9}$$

and

$$\mathbf{x} = \begin{bmatrix} \mathbf{n}_{rel} \\ d_{rel} \end{bmatrix}. \tag{4.5.10}$$

If no measurement noise is present:

$$\mathbf{A}\mathbf{x} = \mathbf{0}. \tag{4.5.11}$$

SVD can then be used to determine $\mathbf{x}$ up to a scale factor, as performed in Section 4.2.2, provided that four or more points of intersection occur with the deck:

$$\mathbf{U}\ \mathbf{S}\ \mathbf{V}^{T} = \mathbf{A}. \tag{4.5.12}$$

The last column of $\mathbf{V}$ contains the coefficients of the deck plane equation. The measured deck normal $\tilde{\mathbf{n}}_{rel}$ can be transformed into a unit vector by normalising:

$$\tilde{\mathbf{n}}_{rel} = \frac{\begin{bmatrix} V_{14} & V_{24} & V_{34} \end{bmatrix}^{T}}{\left\| \begin{bmatrix} V_{14} & V_{24} & V_{34} \end{bmatrix}^{T} \right\|} \tag{4.5.13}$$

$$\tilde{d}_{rel} = \frac{V_{44}}{\left\| \begin{bmatrix} V_{14} & V_{24} & V_{34} \end{bmatrix}^{T} \right\|}. \tag{4.5.14}$$

If the normal vector is pointing upward (i.e. $\tilde{n}_{rel_3}$ is negative), $\tilde{\mathbf{n}}_{rel}$ and $\tilde{d}_{rel}$ must both be multiplied by -1.

Attitude Measurement

The roll and pitch of the ship deck can also be measured, as three or more laser rangefinder distance measurements are available. Yaw cannot be measured using the laser sensor. Therefore, the estimated deck yaw $\hat{\psi}$ is first removed from $\tilde{\mathbf{n}}_{rel}$ in order to produce $\tilde{\mathbf{n}}_{ny}$, the measured deck normal without yaw[3]:

$$\tilde{\mathbf{n}}_{ny} = \begin{bmatrix} \cos\hat{\psi} & \sin\hat{\psi} & 0 \\ -\sin\hat{\psi} & \cos\hat{\psi} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tilde{\mathbf{n}}_{rel}. \tag{4.5.15}$$

The roll and pitch can then be extracted from $\tilde{\mathbf{n}}_{ny}$ as follows:

$$\begin{aligned} \tilde{\phi} &= -\arcsin(\tilde{n}_{ny_2}) \\ \tilde{\theta} &= \arcsin\left(\frac{\tilde{n}_{ny_1}}{\cos\tilde{\phi}}\right). \end{aligned} \tag{4.5.16}$$

Choosing α

The choice of α controls the accuracy of the multiple laser sensor. A large α results in a large ellipse being projected onto the deck. A large ellipse results in accurate measurement of the relative attitude. However, if α is too large, the helicopter has to be very close to the deck surface for the entire laser cone to intersect the deck surface. The width of the ship deck described in [4] is 16 metres, which is its shorter dimension. Therefore, the value of α is chosen to be 0.38 radians. This ensures that the cone intersects the 16 metre wide deck at a height of 20 metres, provided the helicopter is directly above the origin of the deck and the helicopter and ship are level. Figure 4.15 illustrates the geometry of the laser beam and ship deck.

4.5.3 Noise Analysis

The noise analysis of the multiple laser rangefinder sensor is performed in the same way as the single laser rangefinder. The noise observed in the relative Down position measurement noise is not displayed here as it is almost identical to that of the single laser rangefinder.

The multiple laser sensor relies on previous relative yaw estimates to measure the roll and pitch states. Gaussian white noise with a standard deviation of 0.3° was added to the simulated relative yaw estimates, as this is roughly the noise level observed

[3] Appendix C explains this procedure.

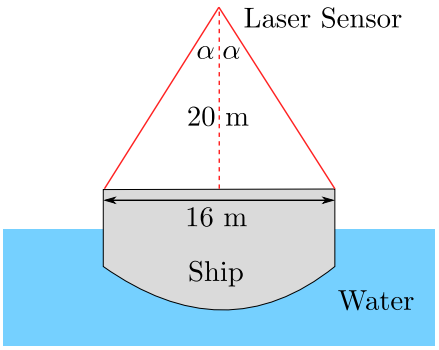
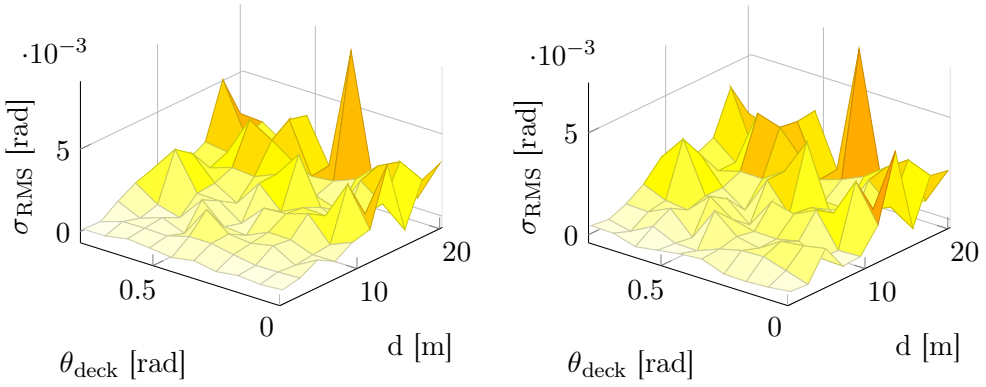


Figure 4.15: Choosing α such that the laser sensor can observe the entire deck at 20 m altitude.

when using the difference of the helicopter and ship estimates to calculate the relative estimates.

Figure 4.16 shows the RMS error in the roll and pitch measurements. For a fixed α and d, increasing θ_{deck} results in an increase in the semi-major axis of the ellipse. As the condition of matrix $\mathbf{A}$ improves, so do the accuracy of the roll and pitch measurements.



(a) Noise on roll measurement.

(b) Noise on pitch measurement.

Figure 4.16: Noise on roll and pitch laser measurements.

Second order polynomials are fitted to the noise plots of each of the measured states in the same manner as performed for the other relative state sensors. The fitted surfaces are shown in Appendix B.

4.6 Chapter Summary

Relative sensors are required to autonomously land a helicopter on a ship deck. This chapter short-listed and modelled the most promising relative sensors. These included monocular vision, stereo vision, a single laser rangefinder and a sensor incorporating multiple laser rangefinders. The monocular and stereo vision sensors provide measurements of the relative position and attitude states. In contrast, the single laser sensor only provides a relative Down measurement and the multiple laser sensor provides relative Down, roll and pitch measurements. Furthermore, the laser sensors require preliminary relative estimates in order to calculate these measurement. This is because the laser range measurements alone are insufficient in measuring the translation and rotation of the vehicles. As such, the laser sensors need to be used in conjunction with another relative sensor capable of measuring the complete relative vehicle states.

The monocular vision sensor was modelled similarly to [3]. This involves the use of singular value decomposition to obtain the relative pose of a predetermined pattern on the deck. The models of the stereo vision and laser sensors were derived in this thesis. Noise models of each of the sensors were developed using Monte Carlo simulation. The accuracy of the vision sensors is primarily dependent on the height and radial distance that the helicopter hovers relative to the pattern on the ship deck. In comparison, the laser sensor accuracy was found to be determined by the relative distance and angle of the deck. These noise models will be used in the following chapter to determine the optimal weighting of sensor measurements when estimating the relative and ship states.

Chapter 5

Estimator Structure

In order to obtain accurate relative and ship state estimates, sensor measurements need to be fused in an optimal manner. This chapter begins with a discussion of the important factors that need to be taken into account when designing an optimal estimator. These factors are explained using a suboptimal, "naïve" estimator design as an example. The final design of the estimator follows. A discussion on how the estimator can be used for varying degrees of ship deck instrumentation, ranging from fully instrumented to uninstrumented decks, concludes this chapter.

5.1 Design Considerations of the Estimator

Several sets of states need to be estimated for use by the helicopter's control system. These include the helicopter's states, the relative states of the helicopter and ship deck, and the ship's states. An explanation of these states follows below:

Helicopter's absolute position, velocity and attitude: This set of states is used for commanding the helicopter's position and orientation in the world. For example, these states are used when navigating between way-points and hovering.

Relative position, velocity and attitude: Knowledge of relative states is required by the control system in order to regulate the rate of descent of the helicopter relative to the ship deck when landing. These states are also used to determine whether conditions allow for safe landing of the helicopter. If the relative roll, pitch or velocity between the helicopter and ship is too great, the landing should be aborted. A landing should be reattempted once conditions are more suitable.

Ship's absolute position and attitude: Estimates of the absolute position and attitude of the ship can be used to predict the optimal time to land.

Predicting the future motion of the ship deck is important. This is to ensure that the helicopter lands without excessive force. In rough swells the ship deck will undergo rapid rolling, pitching and heaving. [4] indicates that the heli-pad, located at the stern of a South African Navy Patrol Corvette, is capable of peak-to-peak heave amplitudes of almost nine metres. In addition, roll and pitch angles of just under eight degrees in each direction are observed. [3] forms predictions of the deck heave up to 20 seconds into the future. Based on the predicted trajectory, a safe landing trajectory and time of touch down is then identified.

The estimator used within the ESL is currently only capable of estimating the helicopter's absolute position, velocity and attitude states. Ship motion prediction is performed directly on raw monocular vision measurements [3]. This technique, although simple, does not allow for multiple sensor measurements to be optimally combined.

An overall estimation framework therefore needs to be designed that permits multiple relative sensor measurements to be combined in an optimal sense. If a ship state estimator is available, its estimates need to be combined with measured ship states to improve the measurements. To highlight these points, a discussion of a naïve estimator design follows.

5.1.1 A Naïve Estimator Design

The manner in which relative states are estimated is important. One method would be to estimate the helicopter and ship states using independent estimators. The ship's position and attitude could be updated by adding the relative measurements, such as those obtained from the vision and laser sensors, to the helicopter estimates. Relative estimates could then be calculated by taking the generalised difference[1] of the estimated helicopter and ship states.

This approach is problematic due to the loss of accuracy of the relative measurements when those measurements are added to the comparatively uncertain helicopter estimates. The variance of the ship's Kalman filter measurement would be larger than both the variance of the relative measurement and the helicopter states. Assume for demonstration purposes that the addition of the states is linear[2]:

$$\tilde{\mathbf{x}}_{deck} = \hat{\mathbf{x}}_{heli} + \tilde{\mathbf{x}}_{rel}, \tag{5.1.1}$$

[1] A simple subtraction of the helicopter and ship states cannot actually be performed for either the translational or rotational states, as the relationship between the states is a non-linear transform. Thus, the term "vector difference" is used. Section 5.2 describes the transformations in detail.

[2] This could be described as the "vector addition".

where $\tilde{\mathbf{x}}_{deck} \sim \mathcal{N}\left(0, \sigma^2_{\tilde{\mathbf{x}}_{deck}}\right)$, $\hat{\mathbf{x}}_{heli} \sim \mathcal{N}\left(0, \sigma^2_{\hat{\mathbf{x}}_{heli}}\right)$ and $\tilde{\mathbf{x}}_{rel} \sim \mathcal{N}\left(0, \sigma^2_{\tilde{\mathbf{x}}_{rel}}\right)$ are the measured ship deck position, estimated helicopter position and relative measurement, respectively. Variance in the measured deck position would be larger than the estimated helicopter position:

$$\sigma^2_{\tilde{\mathbf{x}}_{deck}} = \sigma^2_{\hat{\mathbf{x}}_{heli}} + \sigma^2_{\tilde{\mathbf{x}}_{rel}}. \tag{5.1.2}$$

The assumption can be made that the ship estimator is at least as accurate as the helicopter estimator, as the same estimator and sensors can be placed on the ship. Due to the high uncertainty in the measurement updates, the ship estimator would give low weighting to the measurement updates and little benefit would be gained from the highly accurate relative sensors.

Furthermore, the vector difference of the helicopter and ship states is calculated to determine relative estimates. This serves only to further increase the uncertainty:

$$\hat{\mathbf{x}}_{rel} = \hat{\mathbf{x}}_{heli} - \hat{\mathbf{x}}_{deck}, \tag{5.1.3}$$

$$\sigma^2_{\hat{\mathbf{x}}_{rel}} = \sigma^2_{\hat{\mathbf{x}}_{heli}} + \sigma^2_{\hat{\mathbf{x}}_{deck}}. \tag{5.1.4}$$

The use of multiple relative sensors would not significantly decrease the relative estimate variance $\sigma^2_{\hat{\mathbf{x}}_{rel}}$, due to the noise in the measurements. As described above, the noise in the ship measurements would be larger than the noise in the ship estimator. The uncertainty in the helicopter state estimates would, therefore, have a very strong impact on the performance of the overall system.

With these factors in mind, an alternative framework for estimation is proposed.

5.2 Proposed Estimator Structure

The following estimator design is proposed based on the discussion on the naïve estimator design presented above. An overview of the design is given to introduce the overall concept, followed by detailed explanations on the calculation of relative and ship estimates.

5.2.1 Design Overview

In order to derive maximum accuracy from the relative state sensors, the relative measurements should not be added to uncertain estimates. Previously, the relative measurements were used to update the ship deck estimates and the difference of the helicopter and ship states was found to determine the relative estimates. An alternative method of calculating the relative estimates follows.

Firstly, the vector difference between the helicopter and ship deck estimates is found, which will be referred to as the initial relative estimate $\hat{\mathbf{x}}_{rel}$. The weighted mean of the relative sensor measurements and the initial relative estimate, producing the final relative estimate $\bar{\mathbf{x}}_{rel}$, is then calculated. Thereafter, the final relative estimate is fed to the control system to perform autonomous landing.

A major disadvantage of the naïve estimator structure is that accurate relative measurements are added to estimated absolute helicopter states to produce more inaccurate ship measurements. In the proposed estimator, these ship measurements $\tilde{\mathbf{x}}_{ship}$ are filtered using a low pass filter (LPF). This is in order to reduce the variance. $\tilde{\mathbf{x}}_{ship}^{LPF}$ will be used to denote the filtered ship measurements.

The weighted mean of $\tilde{\mathbf{x}}_{ship}^{LPF}$ and the initial ship estimates $\hat{\mathbf{x}}_{ship}$ can be taken in order to further reduce the variance. The resultant final ship state estimate, $\bar{\mathbf{x}}_{ship}$, is subsequently stored. The recent history of final ship state estimates can be used to perform ship motion prediction[3].

A more detailed explanation of the estimation of the relative and ship states follows.

5.2.2 Relative State Estimation

Two forms of relative measurements are available. These include the initial relative estimates, obtained from the vector difference of the helicopter and ship estimators, and the relative sensor measurements. The weighted mean enables fusion of these separate measurements in a unified manner. The initial estimate and various sensor measurements are weighted by their respective covariance matrices. This ensures that more certain measurements are weighted more heavily than less certain measurements. This enables initial relative estimates, if available, to be treated as any another relative measurement.

In order to calculate the initial relative estimate, estimates of the deck states are required. These are obtained by transforming the initial ship position estimate to the deck position using the initial ship attitude estimate:

$$\hat{\mathbf{p}}_{deck} = \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{ship}}^{T} \mathbf{p}_{ship \to deck} + \hat{\mathbf{p}}_{ship}. \tag{5.2.1}$$

The ship deck attitude estimate is the same as the ship attitude estimate. The initial relative position and attitude estimates are then calculated using Equation (2.2.3):

$$\begin{aligned} \hat{\mathbf{p}}_{rel} &= \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{heli}}^{T} (\hat{\mathbf{p}}_{deck} - \hat{\mathbf{p}}_{heli}) \\ \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{rel}} &= \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{heli}}^{T} \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{deck}}. \end{aligned} \tag{5.2.2}$$

[3]Ship motion prediction has been investigated by Swart [3] and Bellstedt and is out of the scope of this project.

The corresponding covariance matrices are calculated by linearising Equation (5.2.2), as shown in Appendix D.6.

Each relative sensor provides a relative position measurement $\tilde{\mathbf{p}}^i_{rel}$ and attitude measurement $\tilde{\boldsymbol{\theta}}^i_{rel}$. The measurements have covariance matrices $\mathbf{R}^i_{\tilde{\mathbf{p}}_{rel}}$ and $\mathbf{R}^i_{\tilde{\boldsymbol{\theta}}_{rel}}$ associated, as explained in Chapter 4. The weighting factor of sensor i's measurement is

$$\mathbf{W}^i = {\mathbf{R}^i_{\tilde{\mathbf{p}}_{rel}}}^{-1}. \tag{5.2.3}$$

Larger variances receive a smaller weighting. The final relative estimate, calculated as the weighted arithmetic mean, is then

$$\bar{\mathbf{p}}_{rel} = \left(\sum_{i=1}^{N} \mathbf{W}^i\right)^{-1} \left(\sum_{i=1}^{N} \mathbf{W}^i \cdot \tilde{\mathbf{p}}^i_{rel}\right) \tag{5.2.4}$$

and the corresponding covariance matrix of the final relative estimate is

$$\mathbf{R}_{\bar{\mathbf{p}}_{rel}} = \left(\sum_{i=1}^{N} \mathbf{W}^i\right)^{-1}. \tag{5.2.5}$$

The weighted arithmetic mean $\bar{\boldsymbol{\theta}}_{rel}$ and covariance matrix $\mathbf{R}_{\bar{\boldsymbol{\theta}}_{rel}}$ of the attitude measurements are calculated similarly and are omitted here.

The single and multiple laser sensors only provide measurements for a subset of states that the camera and relative state estimates provide. Their measurement vectors and covariance matrices can be modified in order to ensure that all the measurements and covariance matrices are of the same dimensions. The missing diagonal elements of the covariance matrices are set to a large number[4] to prevent the corresponding missing laser measurement from influencing the mean measurement. The missing off-diagonal elements are set to zero to prevent coupling between the existing and missing measurements. The single laser sensor's relative position covariance matrix will, therefore, be

$$\mathbf{R}_{\tilde{\mathbf{p}}_{rel}} = \begin{bmatrix} 10^6 & 0 & 0 \\ 0 & 10^6 & 0 \\ 0 & 0 & \sigma^2_{\tilde{p}_{rel_3}} \end{bmatrix}, \tag{5.2.6}$$

where $\sigma^2_{\tilde{p}_{rel_3}}$ is the variance of the single laser sensor's position measurement $\tilde{p}_{rel_3}$, as calculated in Section 4.4.4. This sensor is not capable of measuring any of the attitude states and has been excluded from the calculation of the final relative attitude estimate. The multiple laser sensor produces both relative position and attitude

[4] In practice any value that is far greater than the other variances is sufficient. A value of one million is used here.

measurements. The relative position covariance matrix will be identical to that of Equation (5.2.6), with the exception of $\sigma^2_{\tilde{p}_{rel_3}}$, which will have a different value, as calculated in Section 4.5.3. The attitude covariance matrix is:

$$\mathbf{R}_{\tilde{\boldsymbol{\theta}}_{rel}} = \begin{bmatrix} \sigma^2_{\tilde{\phi}} & 0 & 0 \\ 0 & \sigma^2_{\tilde{\theta}} & 0 \\ 0 & 0 & 10^6 \end{bmatrix}, \tag{5.2.7}$$

where $\sigma^2_{\tilde{\phi}}$ and $\sigma^2_{\tilde{\theta}}$ are the variances of the multiple laser sensor's relative roll and pitch measurements, as calculated in Section 4.5.3. The missing elements of the single laser sensor's position vector, as well as the multiple laser sensor's position and attitude measurement vectors, are set to zero. For example, the laser position measurement vector will be:

$$\mathbf{p}_{rel} = \begin{bmatrix} 0 & 0 & \tilde{p}_{rel_3} \end{bmatrix}^T. \tag{5.2.8}$$

Equation (5.2.4) is the weighted mean of independent Gaussian-distributed random variables. The weighted mean corresponds to the maximum-likelihood estimate of the relative states [65]. The variance of the final relative estimate is smaller than any of the individual relative measurement's variances. Therefore, the relatively uncertain initial relative estimates will increase the certainty in the final relative estimate. This is in contrast to the naïve estimator, whereby the certainty would decrease. The weighted mean is also flexible, enabling any number of relative measurements to be combined.

5.2.3 Final Ship State Estimation

Two sources of ship measurements are potentially available. Final relative measurements can be transformed to the ship estimation point using helicopter state estimates. If the ship deck is instrumented with an IMU and GPS receiver, ship states can be estimated aboard the ship and transmitted to the helicopter. The former will be referred to as ship measurements and the latter, initial ship estimates. If these are both available, the weighted mean can be taken in order to improve the final ship estimate.

Calculating Ship Measurements

Final relative measurements are transformed into deck measurements using heli-

copter estimates:

$$\begin{aligned}\tilde{\mathbf{p}}_{deck} &= \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{heli}} \cdot \bar{\mathbf{p}}_{rel} + \hat{\mathbf{p}}_{heli} \\ \mathbf{DCM}_{\tilde{\boldsymbol{\theta}}_{deck}} &= \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{heli}} \cdot \mathbf{DCM}_{\bar{\boldsymbol{\theta}}_{rel}}.\end{aligned} \tag{5.2.9}$$

Since estimates are used to transform the measurements from the relative to inertial reference frames, variance increases. The calculation of the resultant covariance matrices is shown in Appendix D.7.

The ship measurements can then be obtained by transforming the ship deck measurements to the ship estimation point, by rearranging Equation (2.2.4):

$$\begin{aligned}\tilde{\mathbf{p}}_{ship} &= \tilde{\mathbf{p}}_{deck} - \mathbf{DCM}^{T}_{\tilde{\boldsymbol{\theta}}_{deck}} \cdot \mathbf{p}_{ship \to deck} \\ \tilde{\boldsymbol{\theta}}_{ship} &= \tilde{\boldsymbol{\theta}}_{deck}.\end{aligned} \tag{5.2.10}$$

The transformation of the deck measurements further increases the variance in the corresponding position states. The attitude states contain the same variance as that of the deck measurements. This is because the attitude states are unchanged when transforming from one position of the ship to another. The ship position measurement covariance matrix is calculated in Appendix D.8.

Filtering the Ship Measurements

A large ship, such as a SA Navy Corvette, features a significant amount of inertia. A Fast Fourier transform of the data captured by [4] reveal that the bandwidth of the ship motion has a cut-off frequency of approximately 0.2 Hz, as shown in Figure 5.1. Any frequency content of the ship measurements exceeding 0.2 Hz can therefore be attributed to sensor noise and noise in the helicopter estimates, rather than actual ship motion. The noise can be removed by applying a LPF to the ship measurements. This will result in a decrease of the variance of the measurements. The extent to which it will decrease depends on the frequency spectrum of the LPF, as explained below.

Since the final ship estimates are used to perform ship motion prediction, it is crucial that the filter does not remove any of the frequency content of the actual ship dynamics. Furthermore, phase lag needs to be minimised, preferably to zero. This is to ensure that the filtered ship measurements do not introduce a lag into the motion prediction results. A phase lag would result in the helicopter touching down later than desired.

A linear phase lag is not required. This, therefore, permits infinite impulse response (IIR) filters to be used. An elliptic filter, which is a form of IIR filter, was chosen.

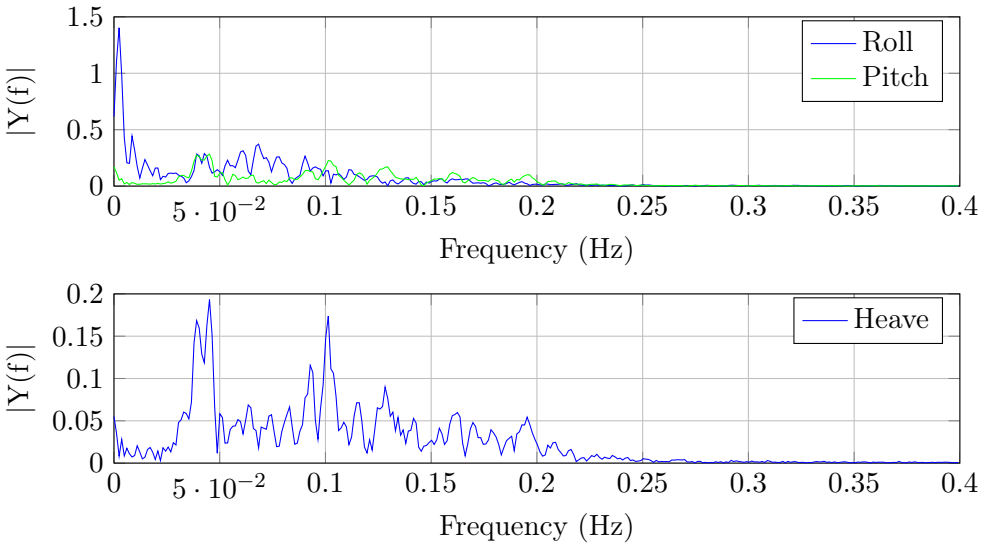


Figure 5.1: Single-sided frequency spectrums of roll, pitch and heave of a South African Navy Corvette.

This filter features the steepest roll-off between passband and stopband for any filter of a given order and amount of ripple. A steep roll-off is a desirable property as it enables as much noise as possible to be removed whilst preserving the actual ship motion in the measurements.

A 3^{rd} order elliptic filter with the following form was designed:

$$y[n] = \sum_{k=1}^{3} a_k y[n-k] + \sum_{l=0}^{3} b_l x[n-l], \tag{5.2.11}$$

whereby x is a signal representing the incoming ship measurements and y is a signal representing the filtered ship measurements. A 3 dB cut-off frequency of 3.5 Hz was chosen as a compromise between noise suppression and phase lag in the pass band. 0.01 dB ripple in the pass band and 20 dB of stop band suppression were selected. The coefficients chosen are listed in Table 5.1. Figure 5.2 shows its Bode plot and an analysis of its performance is done in Section 7.2.3.

The variance of the filtered measurements is determined by calculating the equivalent noise bandwidth B_{equiv} of the filter. B_{equiv} represents the cut-off frequency of the ideal (i.e. brick wall) LPF that would have the same total noise power of the actual filter, as measured in Hertz:

$$B_{equiv} = \frac{1}{2\pi} \int_0^{\infty} \left| \frac{H(e^{jw})}{H_{max}} \right|^2 \mathrm{d}w, \tag{5.2.12}$$

where H_{max} is the maximum amplitude of $H(e^{jw})$, the frequency response of the LPF. H_{max} is chosen to be unity in the filter design process. $H(e^{jw})$ and the brick-

Table 5.1: Filter coefficients

$\mathbf{a_0}$	1.0	$\mathbf{b_0}$	0.03503846221
$\mathbf{a_1}$	−2.194803315	$\mathbf{b_1}$	−0.0059717365
$\mathbf{a_2}$	1.7128222220	$\mathbf{b_2}$	−0.0059717365
$\mathbf{a_3}$	−0.459885455	$\mathbf{b_3}$	0.03503846222

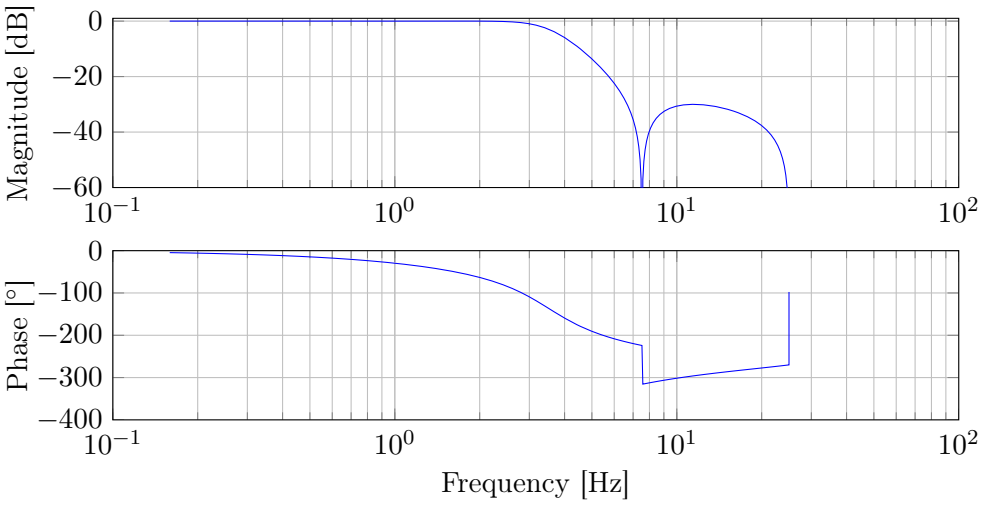


Figure 5.2: Bode plot of filter.

wall filter, $H_{equiv}(e^{jw})$, are shown in Figure 5.3. The frequency response $H(e^{jw})$ of

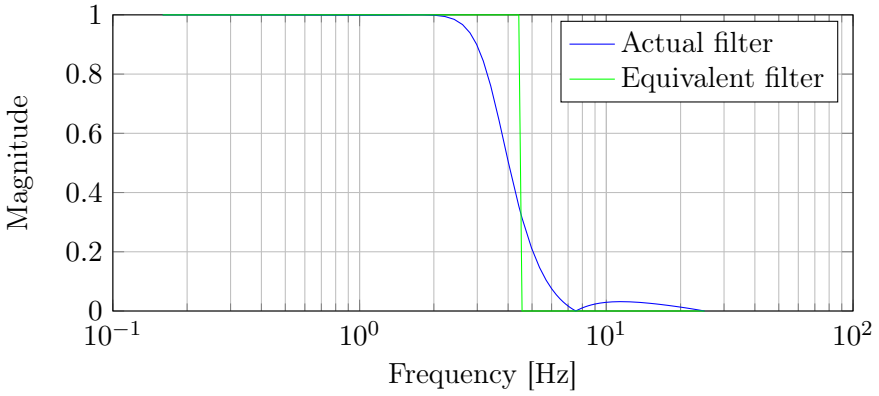


Figure 5.3: Frequency response of the actual LPF and equivalent brick wall filter.

the filter can be calculated from the filter coefficients:

$$H(e^{jw}) = \frac{\sum_{k=0}^{3} b_k e^{-jwk}}{\sum_{l=0}^{3} a_l e^{-jwl}}. \tag{5.2.13}$$

Due to the complexity of the integral in Equation (5.2.12), numerical integration is performed. This is achieved by subdividing the frequency spectrum (0 to Nyquist frequency, which is 25 Hz) into 10 000 bins. In addition, the range is subdivided into 1 000 levels, forming a fine grid of blocks. The integral of $g(w) = \left|\frac{H(e^{jw})}{H_{max}}\right|^2$ is the area under the $g(w)$ curve. This is equal to the number of blocks that have a height less than $g(w)$. The equivalent noise bandwidth of the chosen LPF is thus determined to be

$$B_{equiv} = 4.54\,\text{Hz}.$$

The frequency spectrum of the filtered signal is the product of the input signal's and filter's frequency spectrum:

$$y(t) = x(t) \otimes h(t) \;\Leftrightarrow\; Y(e^{jw}) = X(e^{jw})H(e^{jw}). \tag{5.2.14}$$

The input signal's frequency spectrum is white Gaussian-distributed noise with a variance of σ_x^2 and bandwidth equal to the Nyquist frequency B_{nyq} (25 Hz). The input signal's amplitude n_0 is calculated as follows:

$$\begin{aligned}
\sigma_x^2 &= 2\int_0^{\infty} X(f)\,\mathrm{d}f \\
&= 2\int_0^{B_{nyq}} n_0\,\mathrm{d}f \\
&= 2\,B_{nyq}\,n_0. \\
\\
\therefore n_0 &= \frac{\sigma_x^2}{2\,B_{nyq}}.
\end{aligned} \tag{5.2.15}$$

Therefore, the frequency spectrum of the filtered signal is

$$Y(f) = \begin{cases} n_0\,H_{max} & \text{if } f \le B_{nyq} \\ 0 & \text{otherwise.} \end{cases} \tag{5.2.16}$$

The variance of the filtered signal can then be calculated:

$$\begin{aligned}
\sigma_y^2 &= 2\int_0^{\infty} Y(f)\,\mathrm{d}f \\
&= 2\int_0^{\infty} X(f)\,H(f)\,\mathrm{d}f \\
&= 2\int_0^{B_{nyq}} n_0\,H_{max}\,\mathrm{d}f \\
&= 2\,B_{nyq}\,n_0.
\end{aligned} \tag{5.2.17}$$

The factor by which the variance of the ship measurements is reduced is

$$\frac{\sigma_y^2}{\sigma_x^2} = \frac{2\,B_{equiv}\,n_0}{2\,B_{nyq}\,n_0} = \frac{B_{equiv}}{B_{nyq}} = \frac{4.54}{25} \approx 0.182. \tag{5.2.18}$$

This is a significant reduction in the variance of the ship measurements. The resultant position and attitude covariance matrices will be

$$\mathbf{R}_{\tilde{\mathbf{p}}_{ship}^{LPF}} = \frac{B_{equiv}}{B_{nyq}} \mathbf{R}_{\tilde{\mathbf{p}}_{ship}} \tag{5.2.19}$$

and

$$\mathbf{R}_{\tilde{\boldsymbol{\theta}}_{ship}^{LPF}} = \frac{B_{equiv}}{B_{nyq}} \mathbf{R}_{\tilde{\boldsymbol{\theta}}_{ship}}. \tag{5.2.20}$$

Now that the ship measurements have been filtered and their covariance matrices calculated, the final ship estimates can be calculated using the weighted mean.

Weighted Mean of Ship Measurements

If the estimates from the ship's state estimator are transmitted to the helicopter, the weighted mean of these initial ship estimates $\hat{\mathbf{x}}_{ship}$ and the filtered ship measurements $\tilde{\mathbf{x}}_{ship}^{LPF}$ can be taken to produce a final ship estimate $\bar{\mathbf{x}}_{ship}$.

The initial ship estimates' position $\mathbf{R}_{\hat{\mathbf{p}}_{ship}}$ and attitude $\mathbf{R}_{\hat{\boldsymbol{\theta}}_{ship}}$ covariance matrices are obtained directly from the ship estimator. The final ship position and attitude estimates are calculated in the same manner as that of the relative state estimates. The weighting factor of either the ship measurement or initial ship estimate is given by:

$$\mathbf{W}^i = \mathbf{R}_{\tilde{\mathbf{p}}_{ship}}^{i\ -1}. \tag{5.2.21}$$

The final ship estimate, calculated as the weighted arithmetic mean, is then

$$\bar{\mathbf{p}}_{ship} = \left(\sum_{i=1}^{2} \mathbf{W}^i \right)^{-1} \left(\sum_{i=1}^{2} \mathbf{W}^i \cdot \tilde{\mathbf{p}}_{ship}^i \right) \tag{5.2.22}$$

and the corresponding covariance matrix of the final ship estimate will be

$$\mathbf{R}_{\bar{\mathbf{p}}_{ship}} = \left(\sum_{i=1}^{2} \mathbf{W}^i \right)^{-1}. \tag{5.2.23}$$

The weighted arithmetic mean $\bar{\boldsymbol{\theta}}_{ship}$ and covariance matrix $\mathbf{R}_{\bar{\boldsymbol{\theta}}_{ship}}$ of the attitude measurements are calculated similarly and are thus omitted here.

Choosing the Point of Ship State Estimation

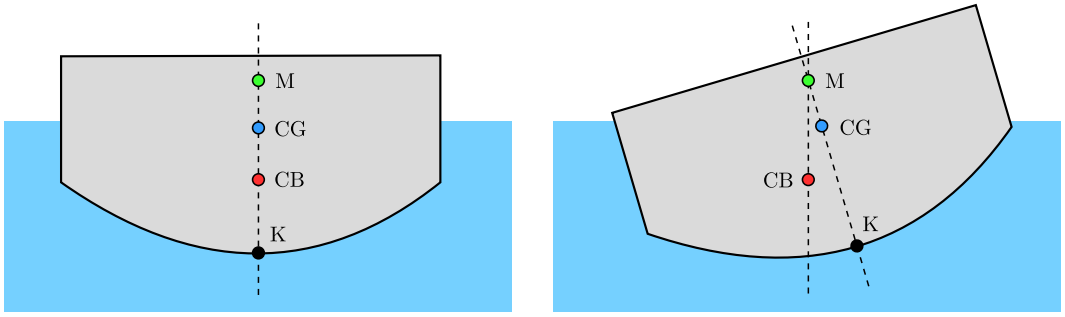
This project provides the estimation framework upon which ship motion prediction can be performed. Ship motion prediction is out of the scope of this thesis. Further studies are therefore required in order to determine the final choice of the ship estimation point. This leaves the choice of the value of the position offset between the ship estimation point and the ship deck, $\mathbf{p}_{ship \to deck}$, to be determined.

However, it is important to note that the point of estimation is not an arbitrary choice. If this point is not chosen to be the pivot point of the ship, then states will be coupled and harmonics will be observed on the ship estimates. This would make the prediction of ship motion more difficult.

There are several different points that can be investigated, which include:

- Centre of gravity (CG)
- Centre of bouyancy (CB)
- Metacentre (M)
- Recording point used by [4] to collect ship data

A discussion of the relationship between the locations of CG, CB and the metacentre for a ship follows. The metacentre remains at a fixed location relative to the ship. However, CB shifts laterally depending on the tilt of the ship. This is because it lies at a fixed height above the keel (K) and vertically beneath the metacentre. Figure 5.4 shows these points and how they change depending on the ship's roll.



(a) Ship in upright position (b) Ship exhibiting a roll

Figure 5.4: Relationship between locations of CG, CB and metacentre for a ship.

In this study, for purpose of convenience, the point of estimation was chosen to coincide with the recording point of the ship estimates used in [4]. The offset of the

deck origin from this point is therefore

$$\mathbf{p}_{ship \to deck} = \begin{bmatrix} 58.35 & 0.525 & 7.775 \end{bmatrix}^T.$$

5.3 Differing Levels of Ship Instrumentation

The estimation structure discussed this far is flexible. Combining relative measurements using the weighted mean allows for any number of such measurements to be combined. This thesis aimed to investigate the feasibility of landing an unmanned helicopter on a ship while only transmitting the ship GPS measurements to the helicopter's estimator (i.e. a partially instrumented ship) or without transmitting any information from the ship to the helicopter (i.e. an uninstrumented ship). This section explains the estimator configuration for each scenario. A discussion of the potential advantages and disadvantages for each option, will follow.

5.3.1 Fully Instrumented Ship

The fully instrumented ship is the complete estimation structure proposed in Section 5.2. There are two ways to achieve a fully instrumented ship. One option is that a full IMU, magnetometer and DGPS be installed on the ship and the measurements transmitted to the helicopter. These readings are then inputted into the same form of Kalman estimator used by the helicopter to estimate its own states. The helicopter uses a kinematic estimator, which is vehicle-independent, as explained in Section 6.1.

If the ship already contains a full state estimator and is able to transmit its states up to the helicopter, there is no need to rig the ship with additional instruments. A lower bandwidth communication link could then be used. This is because less data is required to represent the estimated states than the sensor readings used to calculate the estimates. Furthermore, it is likely that a navy ship will feature a far more advanced estimator and set of sensors than those featured on the helicopter. A model of the ship dynamics will most likely be taken into account in order to improve its estimates.

This is, however, completely dependent on the ship and out of the scope of this project. The "worst-case scenario" ship estimator will therefore be assumed. This comprises the same estimator, IMU, GPS and magnetometer used on the helicopter.

Figure 5.5 shows the complete estimation process for the fully instrumented ship deck. The assumption is made that at least one of the relative state sensors is available for this configuration.

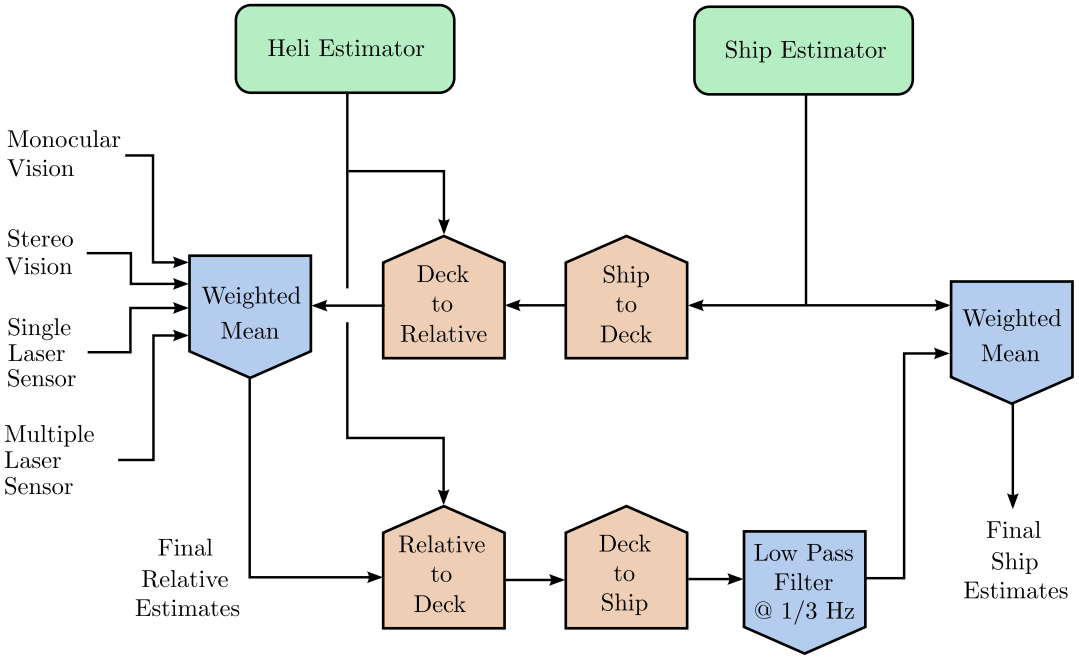


Figure 5.5: Complete estimation process for the instrumented ship. Red and blue blocks represent an increase and decrease in noise, respectively.

5.3.2 Partially Instrumented Ship

The helicopter landing procedure is more robust against communications failure if the reliance on a high bandwidth communications link is decreased. The bandwidth requirements of the communications link decreases when only the ship's GPS position and velocity readings are transmitted to the helicopter.

In order to achieve this configuration with the proposed estimator structure, the initial relative position estimate can be calculated using the ship's GPS position, in place of the ship's estimated position. The final relative position estimate is then calculated in the same way as before. The final relative attitude estimates are the weighted mean of the available relative sensor measurements, as no initial ship attitude states are available.

As before, the final ship position estimates can be calculated as the weighted mean of the filtered ship measurements and the GPS position measurements. A single GPS is not capable of measuring the attitude of a vehicle[5]. This means that the final ship attitude estimate will be the filtered ship attitude measurements. Figure 5.6 demonstrates the estimation configuration.

[5]The GPS can provide a heading estimate based on the assumption that the vehicle is heading in the direction of its velocity vector. However, this assumption is not necessarily true in the case of a ship. If the ship is not translating the heading will be unknown. However, roll and pitch are the more important of the attitude states for landing purposes.

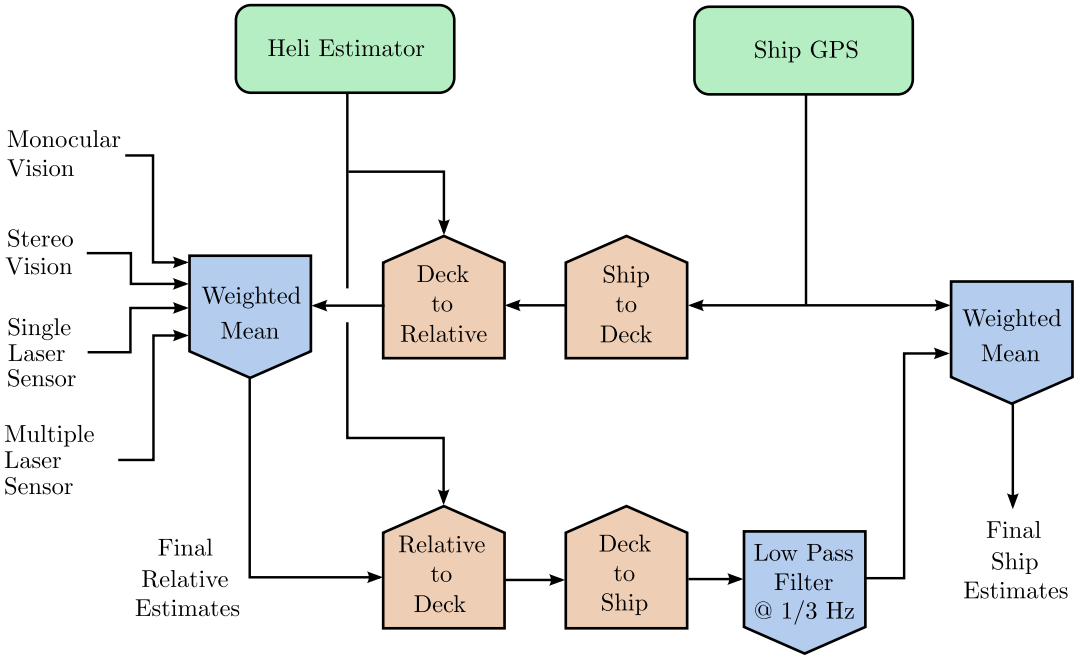


Figure 5.6: Estimation process for the partially instrumented ship. Red and blue blocks represent an increase and decrease in noise, respectively.

5.3.3 Uninstrumented Ship

Requiring a communications link exposes the helicopter to the risk of crashing if the link fails at a crucial moment. GPS is a notoriously weak and sporadic signal, and the possibility of jamming is not remote. Thus, the ability to land on the deck using only the sensors located on the helicopter is desirable. Furthermore, the cost of the entire system is minimised when additional sensors or communications links do not need to be installed or fitted to the deck.

The proposed estimator structure can be configured to handle the absence of initial ship estimates. The final relative estimates can thereby be calculated as the weighted mean of solely the relative sensor measurements. The filtered ship measurements become the final ship estimates. This is illustrated in Figure 5.7.

The use of this structure means that no ship velocity estimates are available for ship motion prediction. The relative state sensors available within the scope of this project are only capable of providing position and attitude measurements. Velocity estimates of the ship can be approximated by differentiating the ship position estimates if required.

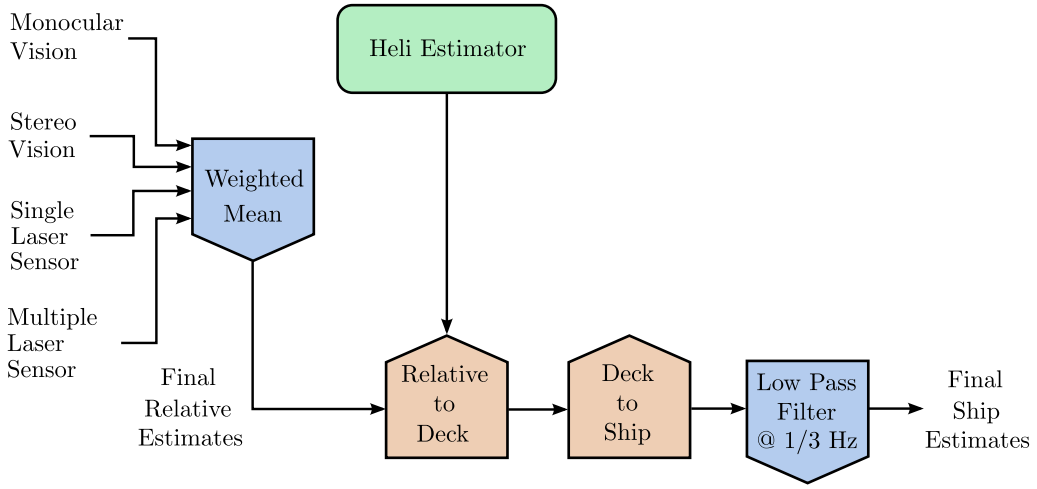


Figure 5.7: Estimation process for the uninstrumented ship. Red and blue blocks represent an increase and decrease in noise, respectively.

5.4 Chapter Summary

The design of the overall structure for estimation of the relative states and ship states was explained in this chapter. The weighted mean of sensor measurements was used to obtain optimal estimates of the various vehicle states. The measurements were weighted according to their respective variances. The estimation accuracy was shown to increase when more sensors are used.

The estimator structure was designed to handle various levels of ship deck instrumentation, including fully instrumented, partially instrumented and uninstrumented ship decks. Fully instrumented ship decks enable more accurate estimation of the relative and ship states. Uninstrumented ship decks, however, are more robust to electronic warfare and communications failures.

The following chapter focuses on the component of the estimator used to estimate the helicopter's states.

Chapter 6

Helicopter Estimator

This chapter presents an overview of the estimator currently being used in the ESL. A detailed discussion of two widely used attitude determination algorithms follows. Although they have been used previously within the ESL, these algorithms have not yet been documented and analysed.

Modifications to the existing ESL helicopter estimator are then proposed. Active gyro bias estimation is incorporated into the attitude estimator, which enables non-deterministic, time-varying biases to be estimated and removed.

The stochastic sensor models developed in Chapter 3 are then used to accurately determine process and measurement noise covariance matrices. This prevents the need to tune the Kalman filters.

Methods to compensate for latency in the GPS measurements are discussed. Finally, observability tests are performed to ensure that the modifications made to the filters do not render the states unobservable.

6.1 Current ESL Estimator

The estimator currently used in the ESL is a full-state strap-down kinematic estimator initially developed by Hough [9]. A kinematic model is an exact representation of the system's dynamics and as such contains no uncertainty [34]. A kinematic estimator can be used for any vehicle, without requiring process models to be developed for each vehicle. Instead, the driving inputs of the Kalman filters are directly obtained from the strap-down gyroscope and accelerometer sensors [11; 9].

The translational and rotational states are decoupled and estimated using separate estimators.

6.1.1 Position and Velocity Estimator

A steady state linear Kalman filter is used to estimate the position and velocity states. Accelerometer readings $\boldsymbol{\alpha}$ are rotated from the helicopter body reference frame into the inertial reference frame and the static, gravitational component of acceleration is added. The resultant dynamic acceleration measurements serve as the driving input to the Kalman filter. GPS position and velocity measurements are used to update the states. In the absence of GPS measurement updates the error in the states would be unbounded [11]. Figure 6.1 illustrates the estimator structure.

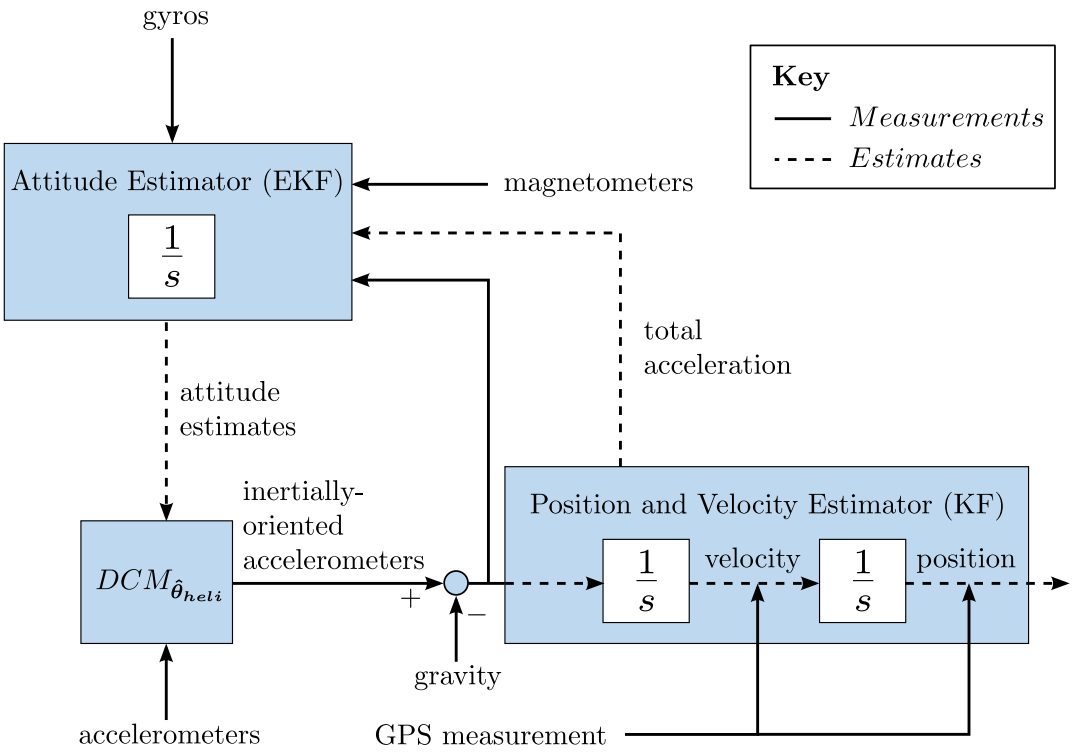


Figure 6.1: Current ESL Helicopter Estimator

The states of the system are the position and velocity of the helicopter within the NED reference frame:

$$\mathbf{x} = \begin{bmatrix} P_N & P_E & P_D & V_N & V_E & V_D \end{bmatrix}^T. \tag{6.1.1}$$

The driving input $\mathbf{u}$ is given by

$$\mathbf{u} = \begin{bmatrix} \alpha_N \\ \alpha_E \\ \alpha_D \end{bmatrix} = \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{heli}} \cdot \begin{bmatrix} \alpha_x \\ \alpha_y \\ \alpha_z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}. \tag{6.1.2}$$

The dynamic equations of the continuous-time system are given as follows:

$$\begin{bmatrix} \dot{P}_N \\ \dot{P}_E \\ \dot{P}_D \\ \dot{V}_N \\ \dot{V}_E \\ \dot{V}_D \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{A}} \cdot \begin{bmatrix} P_N \\ P_E \\ P_D \\ V_N \\ V_E \\ V_D \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{B}} \cdot \underbrace{\begin{bmatrix} \alpha_N \\ \alpha_E \\ \alpha_D \end{bmatrix}}_{\mathbf{u}} + \mathbf{w}_t. \tag{6.1.3}$$

Discretisation of these equations for implementation in the discrete Kalman filter is thoroughly explained in [11]. The discretised state transition and input matrices are:

$$\begin{bmatrix} P_N \\ P_E \\ P_D \\ V_N \\ V_E \\ V_D \end{bmatrix}_{k+1} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & \triangle T & 0 & 0 \\ 0 & 1 & 0 & 0 & \triangle T & 0 \\ 0 & 0 & 1 & 0 & 0 & \triangle T \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{\Phi}} \cdot \begin{bmatrix} P_N \\ P_E \\ P_D \\ V_N \\ V_E \\ V_D \end{bmatrix}_k + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \triangle T & 0 & 0 \\ 0 & \triangle T & 0 \\ 0 & 0 & \triangle T \end{bmatrix}}_{\mathbf{\Gamma}} \cdot \underbrace{\begin{bmatrix} \alpha_N \\ \alpha_E \\ \alpha_D \end{bmatrix}}_{\mathbf{u}_k} + \mathbf{w}_k. \tag{6.1.4}$$

GPS measurements of the states are given by:

$$\mathbf{y} = \mathbf{C} \cdot \mathbf{x}_k + \mathbf{v}_k, \tag{6.1.5}$$

where $\mathbf{C} = \mathbf{I}_{6\times 6}$ and $\mathbf{v}_k$ is the noise of the GPS measurements.

6.1.2 Attitude Estimator

The helicopter attitude is represented using Euler 3-2-1 angles. Euler angles provide a more intuitive representation of a vehicle's attitude than quaternions. The singu-

larity incurred during a 90° pitch is considered outside of the flight envelope of the helicopter used in this project [11].

The helicopter attitude dynamics are nonlinear, which requires an extended Kalman filter to be used. Gyroscope readings $\mathbf{u} = [\,p \;\; q \;\; r\,]^T$ form the driving inputs to the attitude estimator. The continuous nonlinear dynamic equations are as follows, whereby $\mathbf{x} = [\,\phi \;\; \theta \;\; \psi\,]^T$ are the Euler angles:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \cdot \begin{bmatrix} p \\ q \\ r \end{bmatrix}}_{\mathbf{f}(\mathbf{x},\mathbf{u})} + \mathbf{w}_t, \tag{6.1.6}$$

where $\mathbf{w}_t$ is zero mean Guassian distributed white noise with a covariance of $\mathbf{Q}$. These nonlinear equations are linearised using a first order Taylor series expansion and discretised [11]:

$$\mathbf{\Phi}_k = \begin{bmatrix} 1 + \triangle T \cdot t_\theta(c_\theta q - s_\phi r) & \triangle T \cdot \sec\theta^2(s_\phi q + c_\phi r) & 0 \\ -\triangle T \cdot (s_\phi q + c_\phi r) & 1 & 0 \\ \triangle T \cdot \sec\theta(c_\phi q - s_\phi r) & \triangle T \cdot \sec\theta \cdot t_\theta(s_\phi q + c_\phi r) & 1 \end{bmatrix}_{\hat{\mathbf{x}},\mathbf{u}} \tag{6.1.7}$$

$$\mathbf{\Gamma}_k = \triangle T \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi \sec\theta & c_\phi \sec\theta \end{bmatrix}_{\hat{\mathbf{x}}}, \tag{6.1.8}$$

where $s_\alpha = \sin(\alpha), c_\alpha = \cos(\alpha)$ and $t_\alpha = \tan(\alpha)$. The linearised system equations are then substituted into (2.1.1):

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{\Gamma}_k \mathbf{u}_k + \mathbf{w}_k. \tag{6.1.9}$$

The outputs are given by:

$$\mathbf{y}_k = \mathbf{C} \cdot \mathbf{x}_k + \mathbf{v}_k, \tag{6.1.10}$$

where $\mathbf{C} = \mathbf{I}_{3\times 3}$ and $\mathbf{v}_k$ is the noise of the attitude measurements.

6.2 Attitude Measurement Algorithms

Measurement of a helicopter's attitude is considerably more difficult than measurement of its position and velocity. There are no low-cost sensors capable of directly

measuring the vehicle's attitude [11]. This is known as the attitude determination problem.

Two commonly used techniques for measuring the attitude of a vehicle are the TRIAD and the tilt-heading algorithms. These algorithms seek to determine the rotation matrix that maps reference vectors within the inertial reference frame to measurements of these vectors within the vehicle's body reference frame:

$$\begin{aligned} \mathbf{DCM}_{\boldsymbol{\theta}} \cdot \mathbf{B} &= \mathbf{b} \\ \mathbf{DCM}_{\boldsymbol{\theta}} \cdot \mathbf{E} &= \mathbf{e}. \end{aligned} \tag{6.2.1}$$

$\boldsymbol{\theta}$ is the vector of Euler angles representing the relative rotation between the two reference frames[1]. $\mathbf{B}$ and $\mathbf{E}$ are independent, normalised reference vectors within the inertial reference frame. $\mathbf{b}$ and $\mathbf{e}$ are corresponding measurements of those vectors within the body reference frame. The magnetic field vector and the gravity vector are the reference vectors most commonly used for aircraft navigation. These vectors are fixed within the inertial reference frame and can be measured or calculated within the vehicle's body reference frame.

If measurement noise is not present, it can be shown that [66]:

$$\begin{aligned} \mathbf{b} \cdot \mathbf{e} &= (\mathbf{DCM}_{\boldsymbol{\theta}} \cdot \mathbf{B}) \cdot (\mathbf{DCM}_{\boldsymbol{\theta}} \cdot \mathbf{E}) \\ &= \mathbf{E}^T \cdot \mathbf{DCM}_{\boldsymbol{\theta}}^T \cdot \mathbf{DCM}_{\boldsymbol{\theta}} \cdot \mathbf{B} \\ &= \mathbf{E}^T \cdot \mathbf{B} \\ &= \mathbf{E} \cdot \mathbf{B}. \end{aligned} \tag{6.2.2}$$

The TRIAD algorithm is, however, capable of determining an analytic solution in the presence of measurement noise.

6.2.1 TRIAD Algorithm

The TRIAD algorithm was the first satellite attitude determination algorithm to be published [66] and is one of the simplest methods available. It derives its name from its calculation of a triad of orthogonal vectors within each of the reference frames. These vector triads are used to calculate the rotation matrix that relates the reference frames:

$$\mathbf{DCM}_{\tilde{\boldsymbol{\theta}}} = \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \mathbf{s}_3 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & \mathbf{t}_3 \end{bmatrix}^{-1}, \tag{6.2.3}$$

whereby

$$\mathbf{DCM}_{\tilde{\boldsymbol{\theta}}} \cdot \mathbf{s}_i = \mathbf{t}_i, \qquad i = 1, 2, 3. \tag{6.2.4}$$

[1] $\boldsymbol{\theta} = \boldsymbol{\theta}_{heli}$.

The magnetic field vector $\mathbf{B}$ and total acceleration within the inertial reference frame $\mathbf{E}$ are chosen as reference vectors. These vectors correspond to the magnetometer readings $\mathbf{b}$ and accelerometer readings $\mathbf{e}$ within the body reference frame. Total acceleration within the inertial reference frame is calculated, where $\mathbf{v}_k$ and $\mathbf{v}_{k-1}$ are estimated velocity readings obtained from the linear Kalman filter. This is done immediately after a GPS measurement update, when the estimates will be most accurate. $\triangle T_{GPS}$ is the time between the GPS packet arrivals [11] and $\mathbf{g}$ is the gravity vector within the inertial reference frame:

$$\mathbf{E} = \frac{\mathbf{v}_k - \mathbf{v}_{k-1}}{\triangle T_{GPS}} + \mathbf{g}. \tag{6.2.5}$$

The TRIAD algorithm forms $\{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3\}$ from $\mathbf{B}$ and $\mathbf{E}$, and $\{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3\}$ from $\mathbf{b}$ and $\mathbf{e}$ as follows:

$$\begin{aligned} \mathbf{s}_1 &= \frac{\mathbf{b}}{\|\mathbf{b}\|} \\ \mathbf{s}_2 &= \frac{\mathbf{b} \times \mathbf{e}}{\|\mathbf{b} \times \mathbf{e}\|} \\ \mathbf{s}_3 &= \mathbf{s}_1 \times \mathbf{s}_2 \end{aligned} \tag{6.2.6}$$

and

$$\begin{aligned} \mathbf{t}_1 &= \frac{\mathbf{B}}{\|\mathbf{B}\|} \\ \mathbf{t}_2 &= \frac{\mathbf{B} \times \mathbf{E}}{\|\mathbf{B} \times \mathbf{E}\|} \\ \mathbf{t}_3 &= \mathbf{t}_1 \times \mathbf{t}_2. \end{aligned} \tag{6.2.7}$$

$\{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3\}$ and $\{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3\}$ are both orthonormal. This leads to a convenient computationally-efficient analytic solution of the rotation matrix:

$$\begin{aligned} \mathbf{DCM}_{\tilde{\theta}} &= \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \mathbf{s}_3 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & \mathbf{t}_3 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \mathbf{s}_3 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & \mathbf{t}_3 \end{bmatrix}^{T}. \end{aligned} \tag{6.2.8}$$

Equation (4.2.11) can be used to extract the Euler angles from the DCM.

6.2.2 Tilt-Heading Algorithm

As another solution to the attitude determination problem, the tilt-heading algorithm is proposed. The tilt-heading algorithm only uses the magnetometer measurements in the calculation of the heading (yaw) attitude state, rather than all three attitude states. Accurate sensor models for magnetometers are very difficult to determine, especially in close proximity of a navy ship that could contain numerous

unmodelled magnetic field sources. The magnetometer readings should, therefore, be permitted to influence as few states as possible.

The inaccurate knowledge of the magnetic environment manifests as biases in the magnetometer sensor readings. Since the vision sensor is used to update the relative states, rather than the helicopter's absolute (inertial) states[2], the only other sensor available within the scope of this project that is capable of providing a heading measurement is the GPS. However, the GPS provides heading measurements based on the vehicle's horizontal velocity, which is almost stationary when the helicopter is hovering or landing. Furthermore, the direction that a helicopter is traveling does not necessarily indicate its heading. This necessitates the use of the magnetometer as an absolute heading sensor.

The tilt-heading algorithm calculates the gravity vector within the body reference frame [11]:

$$\begin{aligned}\mathbf{g}^B = \begin{bmatrix} g_x & g_y & g_z \end{bmatrix}^T &= \hat{\mathbf{a}}^B_{dyn} - \tilde{\mathbf{a}}^B \\ &= \mathbf{DCM}_{\hat{\boldsymbol{\theta}}} \cdot \hat{\mathbf{a}}^I_{dyn} - \tilde{\mathbf{a}}^B \\ &= \mathbf{DCM}_{\hat{\boldsymbol{\theta}}} \cdot \frac{\mathbf{v}_k - \mathbf{v}_{k-1}}{\triangle T_{GPS}} - \tilde{\mathbf{a}}^B.\end{aligned} \tag{6.2.9}$$

The roll and pitch of the vehicle are then determined from the inclination of $\mathbf{g}^B$:

$$\begin{aligned}\theta &= \arctan\left(\frac{-g_x}{\sqrt{g_y^2 + g_z^2}}\right) \\ \phi &= \arctan\left(\frac{g_y}{g_z}\right).\end{aligned} \tag{6.2.10}$$

Since the magnetometer is mounted to the vehicle in a strapped-down configuration, the effects of roll and pitch need to be removed before the magnetometer's reading can be compared to the reference magnetic field vector. Let $\mathbf{b} = \begin{bmatrix} b_x & b_y & b_z \end{bmatrix}^T$ and $\mathbf{B} = \begin{bmatrix} B_N & B_E & B_D \end{bmatrix}^T$ be the magnetometer readings within the body reference frame and the reference magnetic field vector respectively. Let $\bar{\mathbf{b}} = \begin{bmatrix} \bar{b}_x & \bar{b}_y & \bar{b}_z \end{bmatrix}^T$ be the result of removing the vehicle's tilt from $\mathbf{b}$:

$$\begin{aligned}\bar{\mathbf{b}} &= \mathbf{T}_\theta^{-1} \cdot \mathbf{T}_\phi^{-1} \cdot \mathbf{b} \\ &= \begin{bmatrix} \cos\theta & \sin\phi\sin\theta & \sin\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \cdot \mathbf{b}.\end{aligned} \tag{6.2.11}$$

[2] [11] used the vision sensor measurements to update the helicopter's states because the assumption could be made that the landing surface was stationary and level. In that case, motion relative to the deck pattern was equivalent to the absolute motion of the helicopter.

Now that the magnetometer measurements have been rotated into the same plane as the North and East components of the reference magnetic field vector, the heading can be calculated:

$$\psi = \arctan\left(\frac{\bar{b}_x}{\bar{b}_y}\right) - \arctan\left(\frac{B_N}{B_E}\right). \tag{6.2.12}$$

6.3 Gyro Bias Estimation

Kalman filters assume that process and measurement noise are zero mean Gaussian white noise [28]. Unmodelled slowly drifting biases can, therefore, have a significant impact on the accuracy of Kalman filters.

Systematic and turn-on biases in gyroscopes can be removed through calibration and zeroing of the gyro readings during the initialisation of the helicopter's estimator. Stochastic gyro biases, however, can only be determined through active bias estimation.

The noise analyses performed in Chapter 3 reveal that the gyro sensor noise is dominated by white noise and rate random walk. The white noise component is already taken into account by the attitude estimator. Rate random walk, however, requires the appending of additional states to shape white noise into the required frequency spectrum [28; 67]. Bias estimation can be performed by removal of the bias state estimates from the incoming gyro readings prior to their inclusion into the Kalman filter. These unbiased driving inputs should then exhibit only white noise.

Figure 6.2 indicates the concept of bias estimation for a single axis for illustration purposes. White noise w_2 and rate random walk w_1 are added to the true angular rate p to simulate noisy gyro readings $\tilde{p}$.

In this simplified, single-axis attitude system, two Kalman gains $\mathbf{K}$ are determined: one per state. The attitude measurement $\tilde{\phi}$ is assumed to be unbiased, with only white Gaussian noise present. Gyro biases cannot be measured directly. Instead, the error between the measured attitude and the currently estimated attitude is used to calculate a Kalman gain that will enable the bias value to be estimated.

The complete attitude estimator structure for three axis bias estimation is illustrated in Figure 6.3:

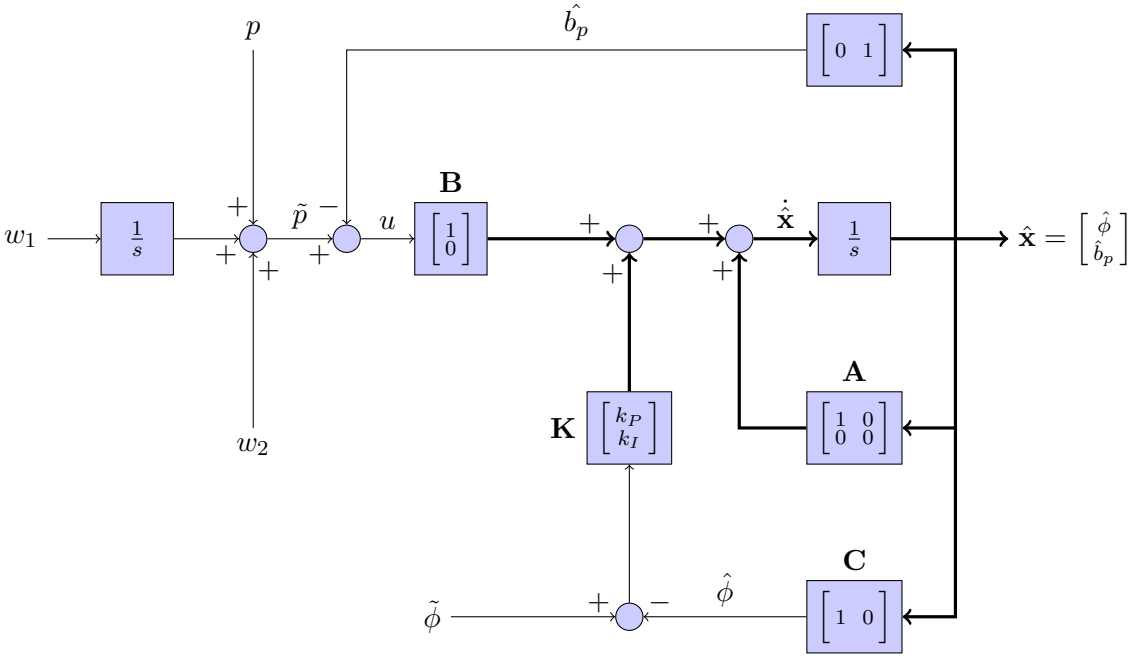


Figure 6.2: Block diagram of bias estimator for a single attitude state

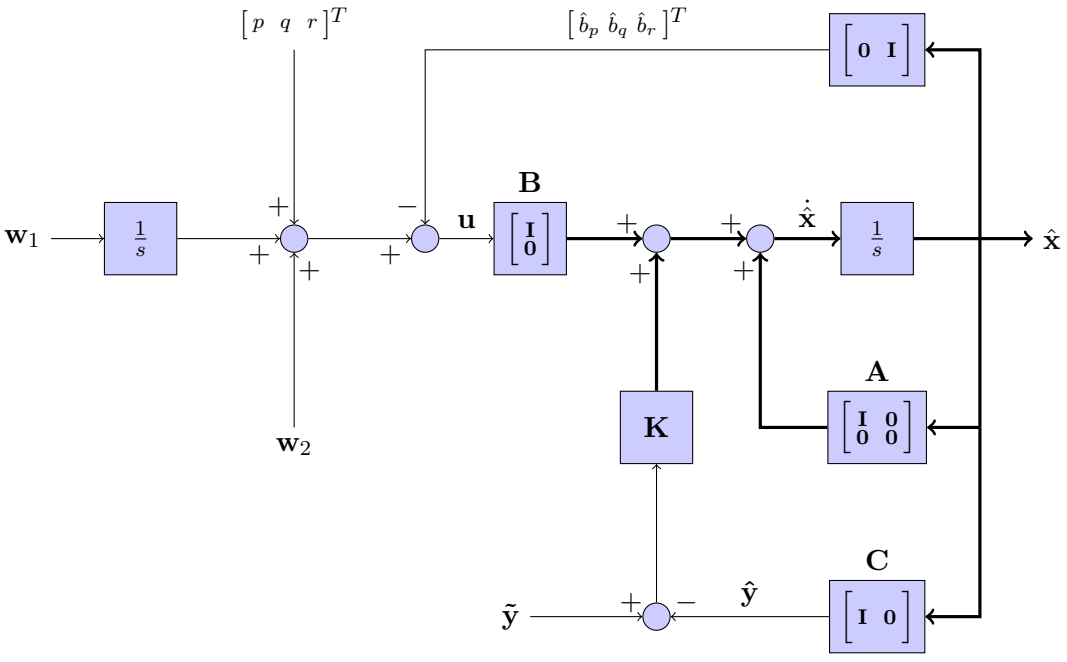


Figure 6.3: Block diagram of attitude estimator for three-axis bias estimation

The non-linear continuous-time attitude dynamics model with bias estimation is as follows, where b_p, b_q and b_r are the biases on the three gyro axes:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{b_p} \\ \dot{b_q} \\ \dot{b_r} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & \sin\phi\cos\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} p - b_p \\ q - b_q \\ r - b_r \end{bmatrix}}_{\mathbf{f(x,u)}} + \mathbf{w}_t. \tag{6.3.1}$$

The output matrix is modified to accommodate the additional states:

$$\mathbf{C} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix}. \tag{6.3.2}$$

Although the gyro measurements are dominated by white noise and rate random walk, a small degree of flicker noise can also be identified from the test results in Section 3.2. As explained in Section 3.2, flicker noise can be modelled using a first order Gauss-Markov model [41]. The Kalman filter is fundamentally based on the Gauss-Markov noise model, which allows this form of noise to be easily incorporated into the filter [41]. The correlation time of the flicker noise is fairly small[3] for relatively low-cost IMUs, such as the ADIS16405 used in this thesis [68]. As a result, the bandwidth of the Kalman filter would be insufficient to track this moderately high frequency bias. Higher frequency biases are harder to distinguish from white noise. Thus, the benefit of estimating high frequency biases is not worth the increase in complexity and computation associated with the increased Kalman filter states.

This gyro bias estimation technique has not been implemented previously in the ESL. However, it has been successfully demonstrated in the flight of autonomous vehicles at S-Plane Automation[4]. Furthermore, the gyro bias estimation technique has also been implemented and tested in a satellite system for a two-axis low-cost gyro [68]. Section 7.1.2 presents the results of the implementation of bias estimation in this thesis.

[3] 10 to 70 seconds, as seen in Section 3.2.
[4] www.s-plane.co.za

6.4 Noise Covariance Matrices

A distinct characteristic of the kinematic estimator structure is that process noise is essentially composed entirely of sensor noise. Therefore, unlike many other systems, process noise can easily be measured in the same manner as sensor measurement noise. This section will show the derivation of the process and measurement noise covariance matrices for the position and velocity Kalman filter, following by the attitude Kalman filter.

6.4.1 Position and Velocity Estimator

6.4.1.1 Process Noise

The process noise of the helicopter's linear Kalman filter is time-varying zero-mean Gaussian distributed white noise:

$$\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k). \tag{6.4.1}$$

The kinematic structure of the estimator causes the process noise to enter the state estimates exclusively via the driving inputs. The inputs[5] only directly drive the velocity states.

$\mathbf{u}_k$ is calculated by rotating the accelerometer readings into the inertial reference frame and deducting gravity, as shown in Equation (6.1.2). Noise is therefore introduced by measurement noise on the accelerometers as well as uncertainty in the Euler estimates used to rotate the accelerometer readings. A minor degree of noise is also introduced by numerical integration in the Kalman filter, although its contribution is negligible.

The noise covariance matrix, which corresponds to these sources of uncertainty, is given by:

$$\mathbf{N} = \mathbf{I}_{6\times 6} \cdot \begin{bmatrix} \sigma^2_{\alpha_x} & \sigma^2_{\alpha_y} & \sigma^2_{\alpha_z} & \sigma^2_{\phi_{heli}} & \sigma^2_{\theta_{heli}} & \sigma^2_{\psi_{heli}} \end{bmatrix}^T. \tag{6.4.2}$$

The variance of the accelerometer noise is calculated from the velocity random walk coefficient N, which is listed in Table 3.3:

$$\sigma^2_{\alpha_x} = \sigma^2_{\alpha_y} = \sigma^2_{\alpha_z} = 2BN. \tag{6.4.3}$$

B is the equivalent bandwidth of the velocity random walk, which is approximately 10 Hz, as illustrated in Figure 3.10.

[5] Process noise is present in the position state estimates, but only as a result of the integration of velocity state estimates during propagation updates.

Time-varying attitude estimate variances $\sigma^2_{\boldsymbol{\theta}_{heli}}$ are obtained from the diagonal elements of the state error covariance matrix $\mathbf{P}$ of the attitude estimator.

The process noise covariance matrix $\mathbf{Q}_k$ is determined from $\mathbf{N}$ as follows:

$$\mathbf{Q}_k = \mathbf{J} \cdot \mathbf{N} \cdot \mathbf{J}^T. \tag{6.4.4}$$

The driving inputs are a nonlinear function of the accelerometer readings and Euler angle estimates. This transformation needs to be linearised[6] in order to determine $\mathbf{Q}_k$ as a linear function of $\mathbf{N}$. Equation (6.1.2) is linearised by calculating the Jacobian, which is a first order Taylor series expansion approximating the nonlinear transformation:

$$\begin{aligned}
\mathbf{J} &= \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{w}} \\
&= \frac{\partial (\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u})}{\partial \mathbf{w}} \\
&= \frac{\partial (\mathbf{B}\mathbf{u})}{\partial \mathbf{w}} \\
&= \frac{\partial (0, 0, 0, u_1, u_2, u_3)}{\partial (\alpha_x, \alpha_y, \alpha_z, \phi_{heli}, \theta_{heli}, \psi_{heli})} \\
&= \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\frac{\partial u_1}{\partial \alpha_x} & \frac{\partial u_1}{\partial \alpha_y} & \frac{\partial u_1}{\partial \alpha_z} & \frac{\partial u_1}{\partial \phi_{heli}} & \frac{\partial u_1}{\partial \theta_{heli}} & \frac{\partial u_1}{\partial \psi_{heli}} \\
\frac{\partial u_2}{\partial \alpha_x} & \frac{\partial u_2}{\partial \alpha_y} & \frac{\partial u_2}{\partial \alpha_z} & \frac{\partial u_2}{\partial \phi_{heli}} & \frac{\partial u_2}{\partial \theta_{heli}} & \frac{\partial u_2}{\partial \psi_{heli}} \\
\frac{\partial u_3}{\partial \alpha_x} & \frac{\partial u_3}{\partial \alpha_y} & \frac{\partial u_3}{\partial \alpha_z} & \frac{\partial u_3}{\partial \phi_{heli}} & \frac{\partial u_3}{\partial \theta_{heli}} & \frac{\partial u_3}{\partial \psi_{heli}}
\end{bmatrix}.
\end{aligned} \tag{6.4.5}$$

The partial derivatives of $\mathbf{J}$ are explained in Appendix D.

6.4.1.2 Measurement Noise

The position and velocity state measurements are obtained directly from GPS readings. GPS position noise features several different spectral components, including white noise, measurement random walk and Gauss-Markov noise. These components are modelled in Section 3.5.2 to simulate realistic GPS readings.

In order to account for these spectral components in the Kalman filter, additional states would need to be appended onto the Kalman filter [69; 70]. The magnitude of

[6] A linear transformation preserves the Gaussian properties of the noise, which is required by the Kalman filter.

correlated noise in differential GPS position measurements is small[7]. Appending the additional states would significantly increase the size of the Kalman filter without achieving significant improvements in the estimation performance. The decision was, therefore, taken to approximate the GPS position measurement noise as white noise for the purposes of the Kalman filter. The GPS velocity readings only exhibit white noise and so do not require appending extra states.

Measurement noise covariance matrices are constructed from the variance of the noise in the GPS readings:

$$\mathbf{R} = \mathbf{I}_{6x6} \cdot \begin{bmatrix} \sigma_{P_N}^2 & \sigma_{P_E}^2 & \sigma_{P_D}^2 & \sigma_{V_N}^2 & \sigma_{V_E}^2 & \sigma_{V_D}^2 \end{bmatrix}^T . \tag{6.4.6}$$

The standard deviations of the GPS position and velocity readings were determined from the same datasets as used for the differential GPS Allan variance analysis in Section 3.5.2. Table 6.1 lists the values obtained.

Table 6.1: GPS noise values for use in Kalman filter

σ_{P_N} [m]	σ_{P_E} [m]	σ_{P_D} [m]	σ_{V_N} [m/s]	σ_{V_E} [m/s]	σ_{V_D} [m/s]
4.0×10^{-3}	4.2×10^{-3}	9.1×10^{-3}	1.41×10^{-2}	1.10×10^{-2}	2.39×10^{-2}

6.4.2 Attitude Estimator

6.4.2.1 Process Noise

The attitude Kalman filter is driven by gyro measurements. A nonlinear transform, $\mathbf{f}(\mathbf{x}, \mathbf{u})$, is used to propagate the estimator states, as shown in Equation (6.1.6). Process noise consists primarily of gyro measurement noise and uncertainty in the current Euler angle estimates. A small degree of noise is introduced through mathematical approximations. This includes the linearisation of the kinematic equations and the Euler integration of the states.

A covariance matrix representing the sources of the noise affecting the driving inputs is given by

$$\mathbf{N} = \mathbf{I}_{9x9} \cdot \begin{bmatrix} \sigma_{\hat{\phi}}^2 & \sigma_{\hat{\theta}}^2 & \sigma_{\hat{\psi}}^2 & \sigma_p^2 & \sigma_q^2 & \sigma_r^2 & \sigma_{b_p}^2 & \sigma_{b_q}^2 & \sigma_{b_r}^2 \end{bmatrix}^T . \tag{6.4.7}$$

[7] In contrast, single-point GPS position readings exhibit significant random walk. If a single-point GPS should be used, then noise shaping of the position measurements would be strongly advised.

The noise characteristics for each axis of the gyro are approximately the same, as seen in Figure 3.6 and Table 3.2:

$$\begin{aligned} \sigma_p^2 = \sigma_q^2 = \sigma_r^2 = \sigma_{ARW}^2 \\ \sigma_{b_p}^2 = \sigma_{b_q}^2 = \sigma_{b_r}^2 = \sigma_{RRW}^2. \end{aligned} \tag{6.4.8}$$

σ_{ARW}^2 and σ_{RRW}^2 are calculated from the Allan variance coefficients as follows:

$$\begin{aligned} \sigma_{ARW}^2 &= 2B_{ARW}N \\ \sigma_{RRW}^2 &= 2B_{RRW}K, \end{aligned} \tag{6.4.9}$$

where B_{ARW} and B_{RRW} are 10 Hz and 1 Hz respectively. N and K are the ARW and RRW coefficients shown in Table 3.2.

The Jacobian and process noise covariance matrices are calculated in the same manner as Equations (6.4.5) and (6.4.4):

$$\begin{aligned} \mathbf{J} &= \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{w}} \\ &= \frac{\partial(f_1, f_2, f_3, f_4, f_5, f_6)}{\partial(\hat{\phi}, \hat{\theta}, \hat{\psi}, p, q, r, b_p, b_q, b_r)} \\ &= \begin{bmatrix} \frac{\partial f_1}{\partial \hat{\phi}} & \frac{\partial f_1}{\partial \hat{\theta}} & \frac{\partial f_1}{\partial \hat{\psi}} & \frac{\partial f_1}{\partial p} & \frac{\partial f_1}{\partial q} & \frac{\partial f_1}{\partial r} & \frac{\partial f_1}{\partial \hat{b}_p} & \frac{\partial f_1}{\partial \hat{b}_q} & \frac{\partial f_1}{\partial \hat{b}_r} \\ \frac{\partial f_2}{\partial \hat{\phi}} & \frac{\partial f_2}{\partial \hat{\theta}} & \frac{\partial f_2}{\partial \hat{\psi}} & \frac{\partial f_2}{\partial p} & \frac{\partial f_2}{\partial q} & \frac{\partial f_2}{\partial r} & \frac{\partial f_2}{\partial \hat{b}_p} & \frac{\partial f_2}{\partial \hat{b}_q} & \frac{\partial f_2}{\partial \hat{b}_r} \\ \frac{\partial f_3}{\partial \hat{\phi}} & \frac{\partial f_3}{\partial \hat{\theta}} & \frac{\partial f_3}{\partial \hat{\psi}} & \frac{\partial f_3}{\partial p} & \frac{\partial f_3}{\partial q} & \frac{\partial f_3}{\partial r} & \frac{\partial f_3}{\partial \hat{b}_p} & \frac{\partial f_3}{\partial \hat{b}_q} & \frac{\partial f_3}{\partial \hat{b}_r} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \end{aligned} \tag{6.4.10}$$

$$\mathbf{Q}_k = \mathbf{J} \cdot \mathbf{N} \cdot \mathbf{J}^T. \tag{6.4.11}$$

The calculation of the Jacobian, $\mathbf{J}$, is shown in Appendix D. The result can be verified by examining the individual elements of $\mathbf{Q}_k$. For example, according to the equation above,

$$\begin{aligned} \mathrm{Q}_{k22} &= \mathrm{J}_{21}{}^2\mathrm{N}_{11} + \mathrm{J}_{22}{}^2\mathrm{N}_{22} + \mathrm{J}_{23}{}^2\mathrm{N}_{33} + \mathrm{J}_{24}{}^2\mathrm{N}_{44} + \ldots + \mathrm{J}_{29}{}^2\mathrm{N}_{99} \\ &= (\frac{\partial f_2}{\partial \hat{\phi}})^2\sigma_{\hat{\phi}}^2 + (\frac{\partial f_2}{\partial \hat{\theta}})^2\sigma_{\hat{\theta}}^2 + (\frac{\partial f_2}{\partial \hat{\psi}})^2\sigma_{\hat{\psi}}^2 + (\frac{\partial f_2}{\partial p})^2\sigma_{ARW}^2 + \ldots + (\frac{\partial f_2}{\partial b_r})^2\sigma_{RRW}^2 \\ &= (-q\sin\hat{\phi} - r\cos\hat{\phi})^2\sigma_{\hat{\phi}}^2 + (\cos\hat{\phi})^2\sigma_{ARW}^2 + (-\sin\hat{\phi})^2\sigma_{ARW}^2 \\ &\quad + (-\cos\hat{\phi})^2\sigma_{RRW}^2 + (\sin\hat{\phi})^2\sigma_{RRW}^2 \\ &= (-q\sin\hat{\phi} - r\cos\hat{\phi})^2\sigma_{\hat{\phi}}^2 + \sigma_{ARW}^2 + \sigma_{RRW}^2. \end{aligned}$$

Since

$$(-q\sin\hat{\phi} - r\cos\hat{\phi})^2\ \sigma^2_{\hat{\phi}}\ \geq\ 0,$$

$$\sigma^2_{ARW} > 0$$

and

$$\sigma^2_{RRW} > 0,$$

the minimum noise bound for Q_{k22} will always be greater than zero. The sensor noise should never become zero regardless of sensor orientation, which supports the statement that the minimum noise bound will indeed always be greater than zero.

6.4.2.2 Measurement Noise

Attitude measurements are obtained using either the TRIAD or tilt-heading algorithms. In each case, the measurements produced are nonlinear functions of the current state, magnetometer or accelerometer measurements. These noise sources can be assumed to be Gaussian-distributed white noise[8]. Since the EKF requires the measurement noise to have a Gaussian distribution, the nonlinear function needs to be linearised.

The measurement noise covariance matrix is determined in a similar manner to the process noise covariance matrix:

$$\mathbf{R}_k = \mathbf{J}\cdot\mathbf{N}\cdot\mathbf{J}^T, \tag{6.4.12}$$

where $\mathbf{N}$ is the noise covariance matrix corresponding to the noise sources.

Appendices D.3.2 and D.3.1 provide detailed explanations of this process and show the measurement noise matrix calculation for each algorithm.

6.5 Observability Tests

A system is observable if its initial state can be determined using measurements of its outputs over a finite interval. If a system is not observable, a Kalman filter's estimates of the state vector and error covariance matrix will not converge to steady-state values. Observability is, therefore, integral to the design of an estimator.

[8]In practice, other spectral components are expected as well, however these components will have a minimal influence on the behaviour of the filter.

Observability of linear time-invariant systems is simply determined by calculating the rank of the observability matrix:

$$\mathbf{O} = \begin{bmatrix} \mathbf{C} \\ \mathbf{C\Phi} \\ \vdots \\ \mathbf{C\Phi}^{n-1} \end{bmatrix}, \tag{6.5.1}$$

where n is the number of states in the system. If the rank of the observability matrix is equal to n, the system is observable.

Substitution of the linear Kalman filter's discrete state transition matrix $\mathbf{\Phi}$ and output matrix $\mathbf{C}$ into Equation (6.5.1) yields a rank of six. The linear Kalman filter is therefore observable[9].

In contrast, the attitude estimator is nonlinear. Observability cannot simply be determined by the substitution of the linearised matrices into Equation (6.5.1) [71]. This test for global observability does not apply to nonlinear systems in general as $\mathbf{\Phi}$ and $\mathbf{C}$ can be a function of the current state. Instead, local observability is determined at the current state by calculating the Lie derivatives [71]. The observability matrix is constructed from the Lie derivatives, as explained in Appendix F:

$$\mathbf{O}(\mathbf{x}) = \begin{bmatrix} \nabla L_f^0 \mathbf{h}(\mathbf{x}) \\ \nabla L_f^1 \mathbf{h}(\mathbf{x}) \\ \vdots \\ \nabla L_f^{n-1} \mathbf{h}(\mathbf{x}) \end{bmatrix}. \tag{6.5.2}$$

A nonlinear system is locally weakly observable at $\mathbf{x}$ if $\mathbf{O}(\mathbf{x})$ has rank n. The system is locally weakly observable if $\mathbf{O}(\mathbf{x})$ has rank n for all $\mathbf{x}$.

The general form of the observability calculation for nonlinear systems is represented by Equation (6.5.2). In the case of the helicopter's EKF, the measurements are in fact a linear function of the states:

$$\mathbf{y} = \mathbf{h}(\mathbf{x}) = \mathbf{C}\mathbf{x}. \tag{6.5.3}$$

In the case of the six state attitude estimator, calculation of the observability matrix must be performed as $\mathbf{C}$ is not an identity matrix. A trajectory needs to be chosen because the observability of the system is dependent on the particular state.

[9] A system that has the identity matrix as its output matrix $\mathbf{C}$ is observable [71].

The rank condition of observability provides only a "yes" or "no" interpretation of observability. It is useful, however, to obtain a quantitative measure of the degree of observability. The condition of the observability matrix, $\delta(\mathbf{x})$, is frequently used [72; 71]. The condition is calculated as a ratio of the maximum and minimum eigenvalues, as explained in Appendix F:

$$\delta(\mathbf{x}) = \frac{\left|\lambda_{\max}[\mathbf{O}^T\mathbf{O}, \mathbf{x}]\right|}{|\lambda_{\min}[\mathbf{O}^T\mathbf{O}, \mathbf{x}]|}. \tag{6.5.4}$$

First, the observability as a function of attitude is determined. This is achieved by setting the angular rates and gyro biases to zero. The resultant observability matrix is

$$\begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & \mathbf{G} \\ \mathbf{0}_{12x3} & \mathbf{0}_{12x3} \end{bmatrix}, \tag{6.5.5}$$

where

$$\mathbf{G} = \begin{bmatrix} -1 & -\sin\phi\tan\theta & -\cos\phi\tan\theta \\ 0 & -\cos\phi & \sin\phi \\ 0 & -\sin\phi\sec\theta & -\cos\phi\sec\theta \end{bmatrix}. \tag{6.5.6}$$

Observability is, therefore, independent of roll and heading angles. The rows of $\mathbf{G}$ are linearly independent regardless of ϕ. Pitch angle, however, does have an influence on observability. Due to the presence of the $\sec\theta$ terms, an increase in the pitch angle decreases the condition of $\mathbf{G}$. The system becomes singular as θ approaches $\pi/2$. This is expected due to the Euler 3-2-1 representation of attitude. Figure 6.4 demonstrates the influence of the pitch angle on the degree of observability. The rank of Equation (6.5.2) is six for all attitude angles, apart from the singularity.

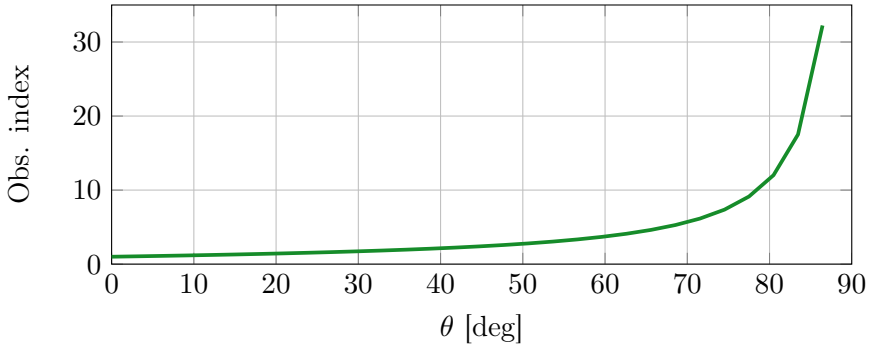


Figure 6.4: Observability index for pitch angle variation.

The observability is then determined as a function of the vehicle's angular velocity. This is achieved by setting Euler angles and bias states to zero. Angular rates are varied one at a time from zero to 45°/s. In all cases, the rank of Equation (6.5.2) is six. Inspection of the degree of observability reveals that large angular rates degrade the estimation accuracy of the EKF, as shown below in Figure 6.5.

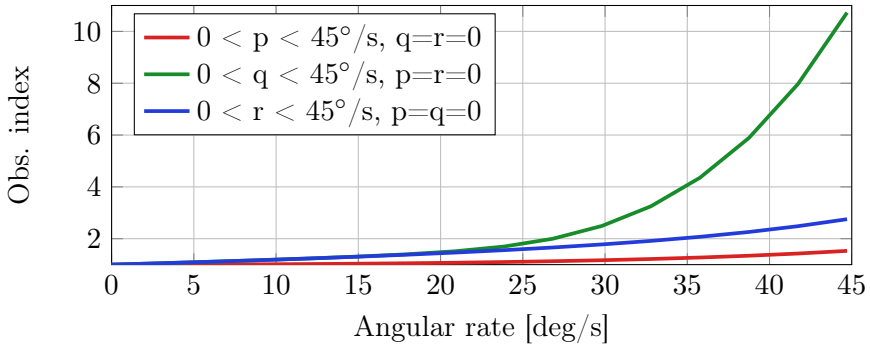


Figure 6.5: Observability index as a function of angular rate.

In comparison, the three-state EKF is completely observable regardless of the current state. The attitude states are measured directly. No hidden states exist. Thus, the degree of observability remains one irrespective of the system's states. This can be checked by substituting the three-state EKF's state transition functions Equation (6.1.6) and output matrix $\mathbf{C} = \mathbf{I}_{3x3}$ into Equation (6.5.2).

The flight envelope of the helicopter is restricted to near hover, with relatively slow angular rates, when landing on the ship. The benefits of gyro bias estimation therefore out-weighs the slight degradation in degree of observability of the EKF.

6.6 Chapter Summary

This chapter provided an overview of the estimator currently being used by the helicopter in the ESL. Two attitude determination algorithms, which have previously been implemented and not yet documented, are explained in detail.

It has been demonstrated that time-varying gyro biases have significantly impacted the accuracy of the helicopter's attitude estimates in past ESL projects [9]. A strategy for estimating and removing these biases was presented in this chapter.

The use of a kinematic model of the helicopter resulted in the process noise being equivalent to accelerometer and gyro sensor noise. This enabled the process noise covariance matrices to be accurately determined by using the results of the statistical

analyses performed in Chapter 3. Therefore, the Kalman filters required no manual tuning of their parameters.

Finally, observability tests were performed to determine whether the system states could be estimated. The observability index was used to quantify the degree of observability as a function of the system state. This allowed for greater insight than the "yes" or "no" information obtained from the rank observability condition conventionally used.

Thorough analyses of the performance of the estimator will be discussed in the following chapter.

Chapter 7

Analysis of Estimator

The helicopter estimator and the relative and ship state estimator are the components that form the complete estimator. This chapter opens with a performance analysis of the helicopter estimator. An analysis of the various relative sensor configurations follow. This is in order to determine the optimal sensor suite for an autonomous ship deck landing.

7.1 Helicopter Estimator

The tilt-heading algorithm is currently used to measure the helicopter's attitude. In this section a thorough sensitivity analysis is performed to identify the effects of biases on tilt-heading measurements in comparison to the TRIAD algorithm measurements. This analysis enables the selection of an attitude measurement algorithm. In addition, this analysis will allow the quantification of the degree to which sensor calibration errors influence the final estimates. More accurate calibration can then be used for sensors that are particularly influential to the estimation accuracy.

The performance of the gyro bias estimation algorithm will be tested thereafter. Incorporating bias estimation diminishes the observability of the estimator. This is explained in Chapter 6.5. It is therefore important to determine whether bias estimation is beneficial.

Furthermore, the sensitivity of the helicopter estimator to inaccuracy in the Allan variance noise analysis is determined. Errors in the Allan variance parameters would result in the Kalman filters providing suboptimal estimates. Finally, the accuracy of the variance estimated in the state error covariance matrix is then compared to the actual variance of the Kalman filter estimates. Ensuring that the Kalman filter variance estimates are accurate is important when taking the weighted mean of the relative measurements. If the variance is underestimated, the Kalman filter estimates

will be overly trusted in relation to the relative sensors.

7.1.1 Attitude Measurement Algorithm Comparison

The attitude measurement algorithm is an integral part of the helicopter's estimator. Attitude measurements are used to correct any drift in the attitude estimates resulting from the propagation updates. Attitude estimates are used to rotate the accelerometer readings into the inertial reference frame, which are then used to drive the position and velocity estimator. The velocity estimates are subsequently used by the attitude estimator to calculate the next attitude estimates. Therefore, any biases in the attitude measurements would influence the attitude estimates as well as the position and velocity estimates.

The Kalman filter assumes that its measurement updates are unbiased. Low-cost accelerometers are, however, known to exhibit substantial random walks, as shown in Chapter 3.3. The magnetic field surrounding the vehicle is notoriously difficult to model. As such, biases resulting from unmodelled magnetic disturbances are common. Furthermore, the magnetic and gravity reference vectors are determined using a lookup function for a given geographical coordinate. These reference vectors are averages taken over broad regions. As such, the true local magnetic or gravity reference vector could deviate slightly from the lookup values. These discrepancies would manifest as biases in the attitude measurements.

7.1.1.1 Method to Test Bias Sensitivity

This section examines the sensitivity of the TRIAD and tilt-heading algorithms to each of these potential sources of bias. This is achieved by initialising the helicopter to a hover state[1] and running the estimator using simulated, biased, noise-free[2] measurements. The elements of each bias vector are perturbed individually in the form of a one-at-a-time sensitivity test. The magnitudes of the bias values are set to realistic quantities. This permits a rough comparison of the influence of each bias on the attitude measurements despite the differences of the bias source units.

Table 7.1 lists the values of the accelerometer, magnetometer rotation, gravity reference vector and magnetic reference vector biases. The bias values that are provided are considered large, yet plausible, magnitudes. Further explanations of the choice of these values are provided in Appendix G. Four non-zero factors of each of the bias

[1]The helicopter hovers at 5° roll and 0° pitch. Yaw can be assumed zero for convenience, although its value is not important.

[2]Eliminating sensor noise for this test prevents sensor noise from appearing as biases. The contribution of biases can then be clearly distinguished.

Table 7.1: Magnitude of biases tested

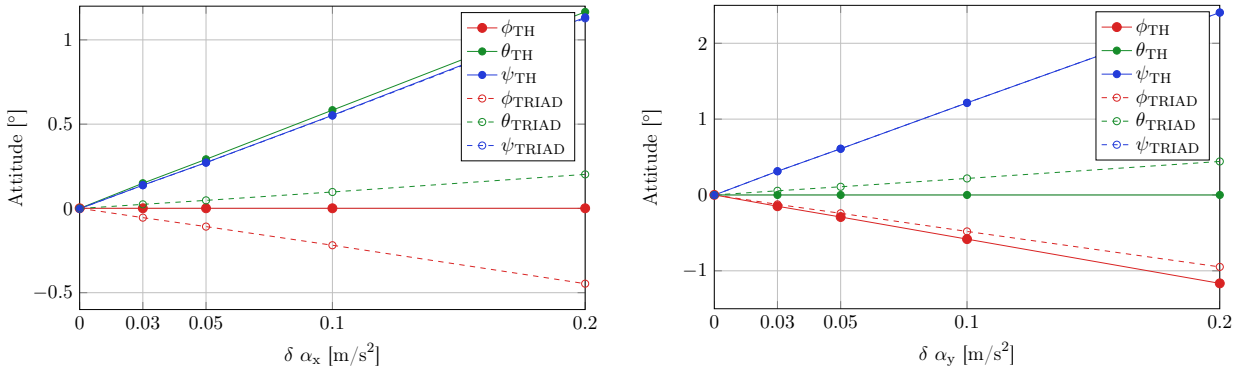
Bias Source	Magnitude of bias	Unit
Accelerometer	0.1	m/s^2
Magnetometer Rotation	5	$^\circ$
Gravity Reference Vector	0.03	m/s^2
Magnetic Field Reference Vector	7.6×10^{-4}	*Gauss*

magnitudes in Table 7.1 are tested. These factors are $\{\frac{1}{4}, \frac{1}{2}, 1, 2\}$. The testing of multiple values enables nonlinear effects to be identified.

Once the estimator reaches a steady state the error in the attitude estimates, relative to unbiased measurements, is recorded.

7.1.1.2 Accelerometer Biases

Equation (6.2.10) indicates that the roll measurements determined using the tilt-heading algorithm are a function of the y and z accelerometer readings. Pitch measurements are a function of the x and y accelerometer readings. Roll and pitch measurements are therefore unaffected by biases in the x and z axes, respectively. TRIAD uses every accelerometer axis to calculate attitude measurements. Thus, a bias in any axis will cause a bias in each of the attitude states. This is confirmed below in Figure 7.1, as certain attitude states exhibit zero error despite biases in certain accelerometer axes.



(a) Bias in x accelerometer

(b) Bias in y accelerometer

Figure 7.1: Attitude estimation error due to bias in individual components of accelerometer readings.

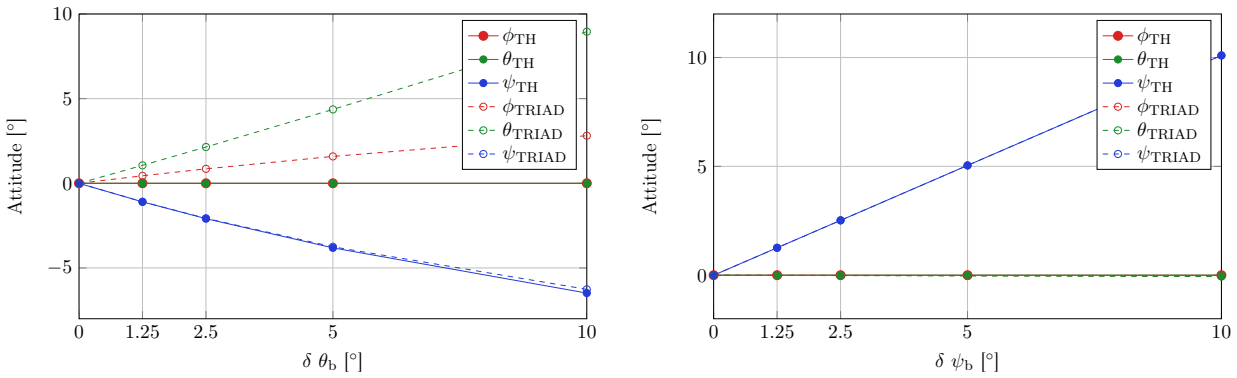
7.1.1.3 Magnetometer Biases

The TRIAD algorithm normalises the magnetometer measurements, causing only an error in the orientation of the vector to introduce a bias. Thus, biases are simulated by rotating the true magnetic field:

$$\tilde{\mathbf{b}} = \mathbf{DCM}_{\delta\boldsymbol{\theta}_b} \cdot \mathbf{b}, \tag{7.1.1}$$

where $\delta\boldsymbol{\theta}_b = [\,\delta\phi_b \;\; \delta\theta_b \;\; \delta\psi_b\,]^T$ forms the angular bias and $\mathbf{b}$ represents the true magnetic field in the body reference frame.

The error in attitude estimates are measured when the magnetic field is rotated in the pitch and yaw directions, respectively. The error in attitude estimates are shown below in Figures 7.2a and 7.2b.



(a) Bias in θ_b

(b) Bias in ψ_b

Figure 7.2: Simulating magnetometer biases by rotating magnetic field.

The tilt-heading algorithm is far more immune to disturbances in the magnetic field. The tilt-heading algorithm does not make use of magnetic field measurements when calculating roll and pitch measurements, which is unlike the TRIAD algorithm. This is verified in Figure 7.2a above. Naturally, the attitude measurement algorithms are equally susceptible to biases in the yaw component of the magnetic field. This is confirmed in Figure 7.2b above.

Biases in the gravitational and magnetic field reference vectors can be calibrated out and are not analysed here. The complete set of sensitivity analysis results are shown in Appendix G. The sensitivity analysis was performed at forward flight conditions too, where the helicopter has a roll of 5° and pitch of −20°. The results for this case are almost identical to those during hover conditions and are therefore not shown here.

The sensitivity analysis of the effects of accelerometer and magnetometer biases on the attitude measurement algorithms confirm that the tilt-heading algorithm is less susceptible to biases, particularly magnetometer biases. The tilt-heading algorithm is therefore chosen for use in the helicopter's estimator. All subsequent analyses are conducted using this algorithm.

7.1.2 Gyro Bias Estimation

Biases are defined as constant or slowly changing errors. This section examines the performance of gyro bias estimation for constant and slow, time-varying biases. Constant biases may arise from a calibration error. In contrast, slow, time-varying biases can be attributed to temperature changes and stochastic biases, which are described in Chapter 3.2.

7.1.2.1 Constant Biases

The simulated helicopter was made to fly along the trajectory shown below in Figure 7.3 in order to test the gyro bias estimation. The estimated attitude with 3-axis active bias estimation is overlaid for comparison. Figure 7.4 shows the bias values used in the test, as well as the estimated biases.

As can be seen below, bias estimation performs remarkably well, as the bias estimates converge quickly to the true values.

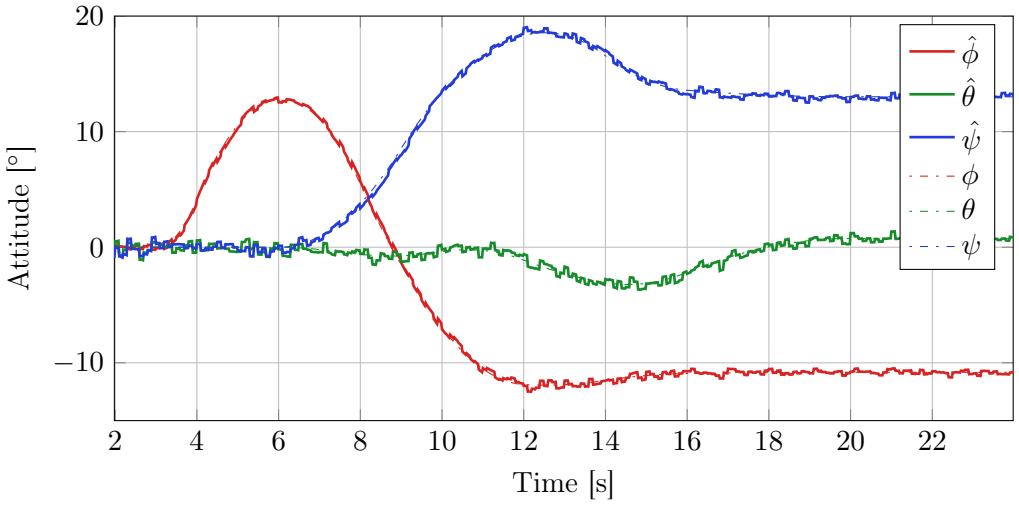


Figure 7.3: Attitude trajectory.

Figure 7.5 below shows the distribution of attitude state innovation with and without bias estimation. The solid line indicates the case in which bias estimation is

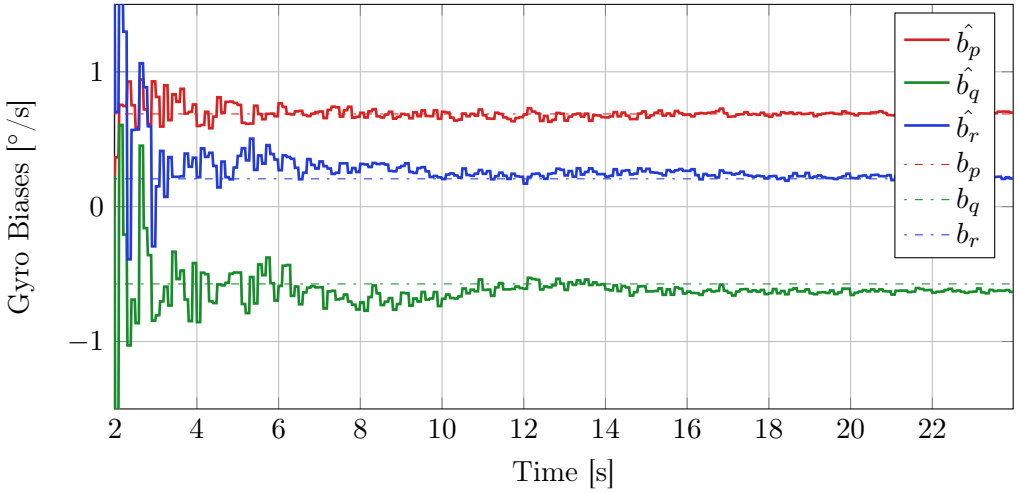


Figure 7.4: Constant gyro biases.

performed and the dotted line indicates the case where bias estimation is not performed. The innovation values were recorded over the steady-state portion of the trajectory that is shown in Figure 7.3. The simulation length was extended to 20 minutes to gather sufficient innovation values.

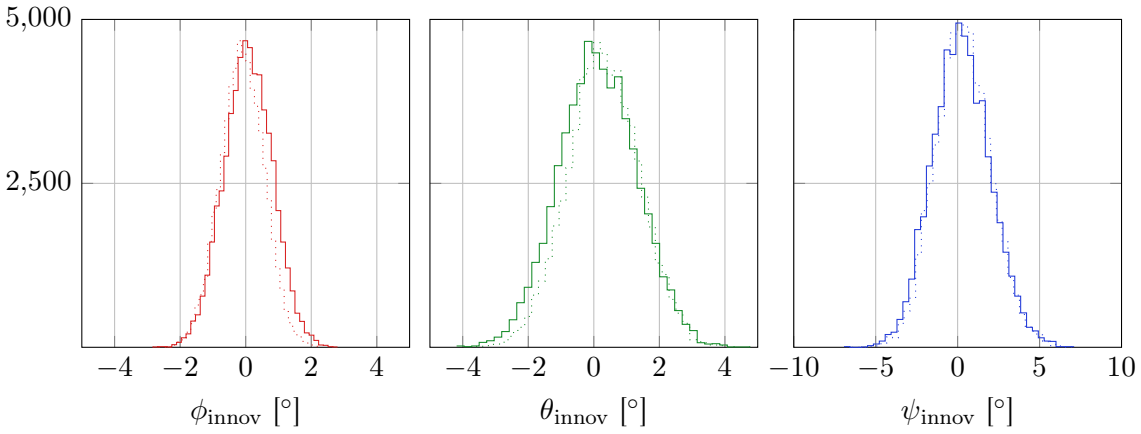


Figure 7.5: Histogram of innovation for attitude states. Solid line represents having bias estimation included, dotted line, without.

The innovations are Gaussian distributions whether bias estimation is performed or not. The mean values of the innovations of both attitude estimators are listed below in Table 7.2. Gyro bias estimation clearly minimises bias in the Kalman filter innovations, and hence, in the attitude estimates.

Table 7.2: Mean innovation of attitude estimates

	Mean state innovation [°]		
With bias estimation	0.0033	0.0138	0.0590
Without bias estimation	-0.1703	0.1929	0.1659

7.1.2.2 Time-Varying Biases

Real gyroscopes exhibit systematic and stochastic biases. This is explained in Chapter 3.2. A 20 minute section of data was extracted from the gyro dataset shown in Figure 7.6 below. During this period, the temperature of the gyros changed from 37° to 60°. Therefore, both systematic and stochastic biases are present in this subset of the dataset.

In order to test the bias estimation on real gyros, the simulated gyro biases were replaced by real gyro data. The simulated angular random walk and rate random walk noises were disabled as the actual gyro data already contains these noise components. The same attitude trajectory that is shown in Section 7.1.2.1 was used in the simulation.

Figure 7.6 below shows that the active bias estimation was able to track the trend in the actual gyro readings. The trends in the gyro readings correspond to the gyro biases. The bias estimates were initialised to zero prior to running the experiment.

Some error, however, is still present. This is particularly exhibited in the latter 10 minute period of the p bias component. The error is due to the Kalman filter converging on bias estimates several minutes after initialising. Thereafter, the Kalman filter begins to trust the process model more. The Kalman filter, therefore, reacts slowly to subsequent changes in the biases.

The bias estimation technique explained in Chapter 7.1.2 models the biases as constants. The estimates are permitted to change slowly, due to non-zero process noise for the bias states. This accounts for the slow reaction of the EKF to changes in the actual biases (refer to Figure 7.6).

7.1.3 Sensitivity to Noise Modelling Inaccuracies

The process and measurement noise covariance matrices are constructed using noise values that were determined using Allan variance analysis in Chapter 3. Significant effort has been made to model the sensors realistically. However, the noise

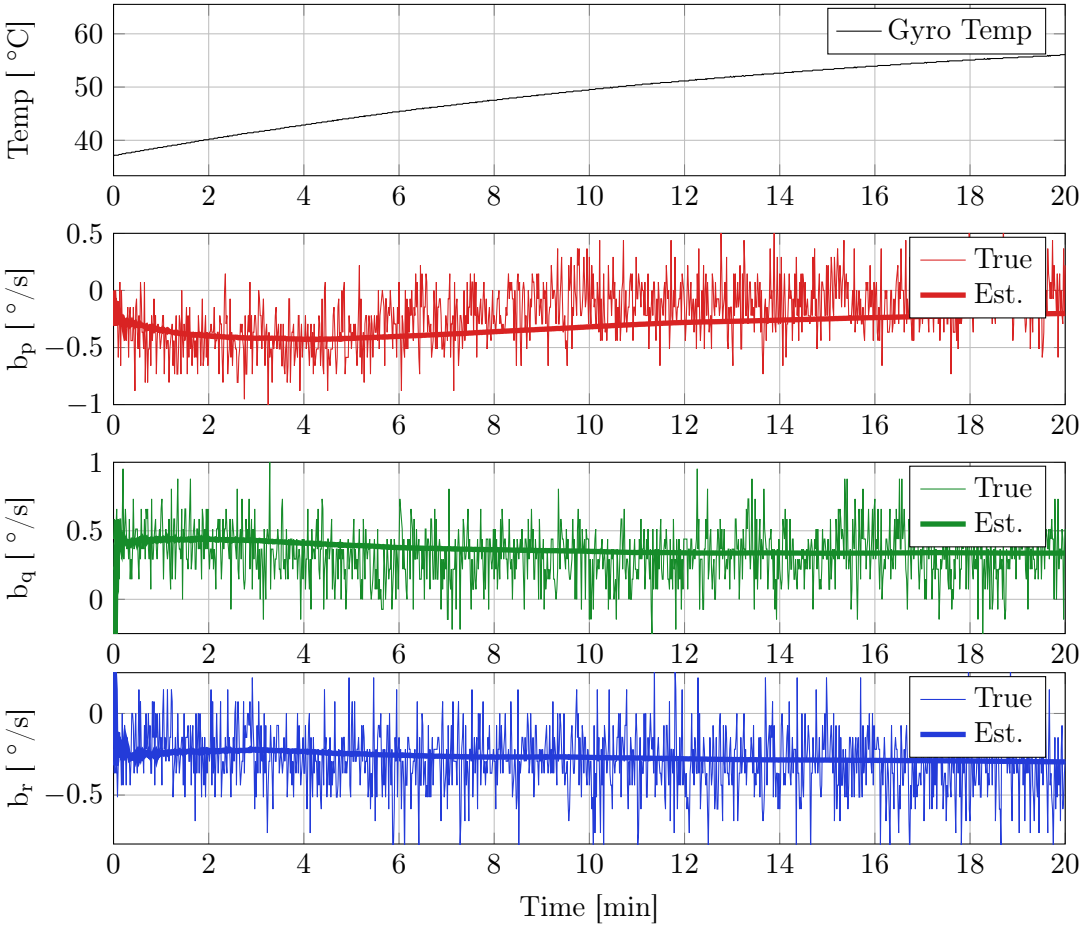


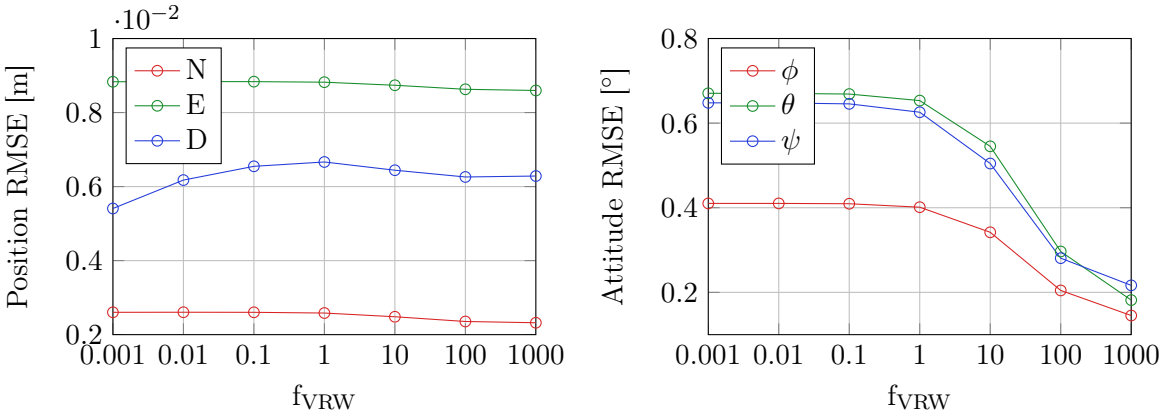
Figure 7.6: Estimating biases in real gyro data.

characteristics of the sensors, especially the accelerometers and gyros, are affected by temperature change, vibration and other environmental factors. Furthermore, the Allan variance technique requires vast quantities of data to measure the noise parameters precisely. Therefore, it is important to determine the sensitivity of the estimates to variation in the Allan variance parameters.

In this experiment, the magnitude of the VRW, ARW and RRW Allan variance parameters used in the Kalman filters' process and measurement noise covariance matrices are varied. The noise parameters of the simulated gyro and accelerometer sensors remain unchanged from the values listed in Chapter 3. Let f_{VRW}, f_{ARW} and f_{RRW} be factors multiplied by the original VRW, ARW and RRW Allan variance parameters in the noise matrices. The factors are varied one at a time in order-of-magnitude increments. A factor of 1 indicates the original Allan variance value. The true helicopter trajectory is the same as that shown above in Figure 7.3.

7.1.3.1 Sensitivity to VRW

The effects of variation in VRW on RMS estimation error are shown in Figure 7.7. The increase of f_{VRW} results in an improvement in attitude estimation. In contrast, the position estimates show minor sensitivity to variation in VRW.



(a) Position (b) Attitude

Figure 7.7: Sensitivity of position and attitude estimation error to VRW parameter.

Attitude estimation accuracy can be improved by trusting the propagation updates more. This is indicated below in Figures 7.7 and 7.8. Increasing f_{VRW} leads to the attitude measurements being trusted less, which is equivalent to propagation updates being trusted more.

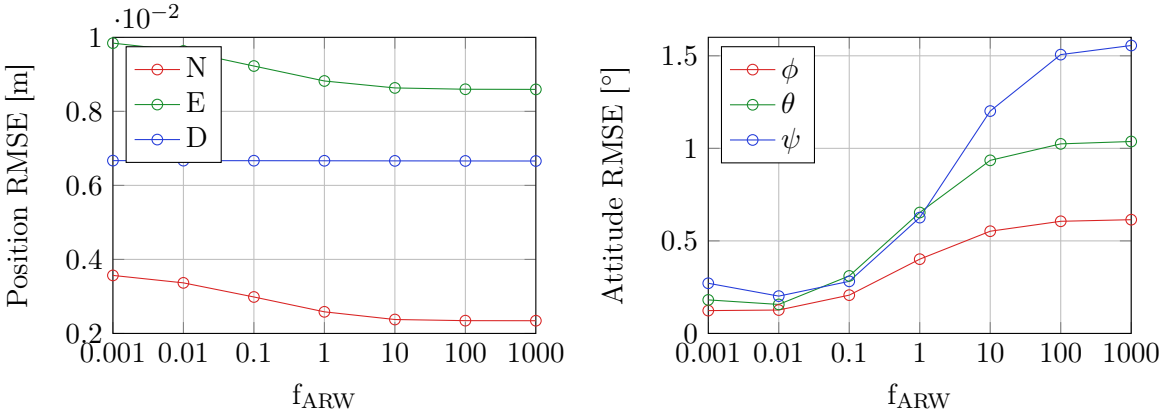
7.1.3.2 Sensitivity to ARW

The effects of variation in ARW on the RMS estimation error is demonstrated in Figure 7.8. Interestingly, the decrease of f_{ARW} results in an improvement in attitude estimation at the expense of position estimation accuracy, and vice versa. Therefore, $f_{ARW} = 1$ represents a compromise between position and attitude estimation error.

7.1.3.3 Sensitivity to RRW

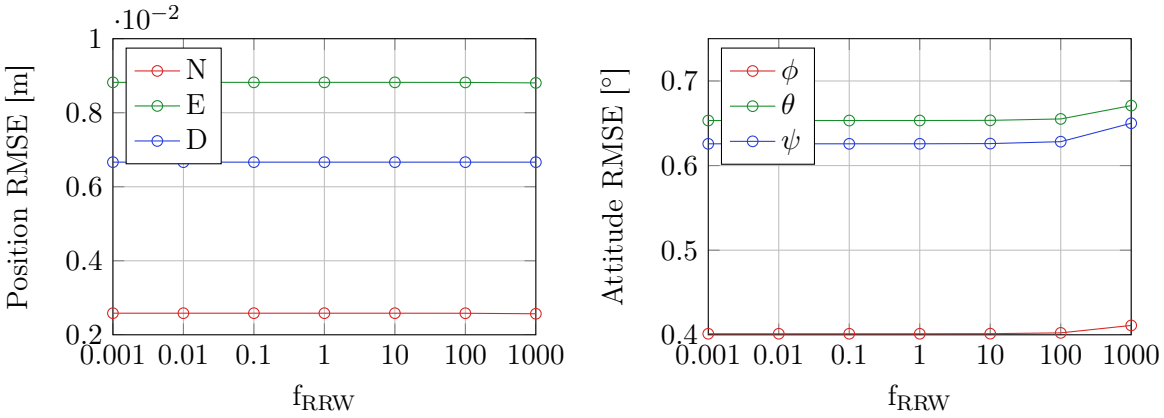
Estimation results are minimally impacted by variation in f_{RRW}, as shown in Figure 7.9. This is because the value of the RRW parameter, measured using Allan variance analysis, is very small in comparison to ARW.

Variations that occur, by as much as three orders of magnitude, result in little change in the EKF process noise covariance matrix. This insensitivity is convenient, as the



(a) Position (b) Attitude

Figure 7.8: Sensitivity of position and attitude estimation error to ARW parameter.



(a) Position (b) Attitude

Figure 7.9: Sensitivity of position and attitude estimation error to RRW parameter.

RRW component of gyro noise requires very large datasets in order to be accurately measured.

7.1.4 Accuracy of Estimated Variance

A Kalman filter's gains are essentially determined as a ratio of the process and measurement noise covariance matrices. If these matrices are both scaled by the same constant factor, the Kalman gains will remain the same. However, the magnitude of a Kalman filter's state error covariance matrix elements is dependent on the magnitude of the values of the process and measurement noise covariance matrices, in addition to their ratio.

In most Kalman filtering situations, the magnitude of the state error covariance matrix elements is not important. Optimal filtering can be performed regardless of

whether this matrix is out by a scale factor. In this project, the state error covariance matrix is used in subsequent stages of the estimation process. Relative state estimation is performed by taking the weighted mean of the deck and helicopter estimates and relative measurements. The weighting requires the variance of the state estimates to be calculated correctly in order to weight the measurements optimally.

The helicopter was simulated in a stationary state for a 5 minute period in order to check the accuracy of the variance estimates. The actual variance of each of the states was determined by calculating the variance on the state estimates. The estimated variances are the diagonal elements of the state error covariance matrices immediately after measurement updates. Table 7.3 below compares the actual and estimated variances for two sets of Allan variance parameters. These are the original and the optimum Allan variance parameters.

Table 7.3: Comparison of Actual and Estimated Variance.

	$f_{VRW} = 1,\ f_{ARW} = 1$		$f_{VRW} = 10,\ f_{ARW} = 0.1$	
	Actual	**Estimated**	**Actual**	**Estimated**
$\boldsymbol{\sigma}_N^2\ [\mathrm{m}^2]$	1.03×10^{-5}	4.14×10^{-5}	1.05×10^{-5}	3.90×10^{-5}
$\boldsymbol{\sigma}_E^2\ [\mathrm{m}^2]$	8.31×10^{-6}	4.01×10^{-5}	8.32×10^{-6}	3.77×10^{-5}
$\boldsymbol{\sigma}_D^2\ [\mathrm{m}^2]$	3.67×10^{-5}	2.76×10^{-5}	3.63×10^{-5}	3.74×10^{-5}
$\boldsymbol{\sigma}_{V_N}^2\ [\mathrm{m}^2/\mathrm{s}^2]$	3.04×10^{-4}	2.82×10^{-5}	2.51×10^{-4}	1.69×10^{-5}
$\boldsymbol{\sigma}_{V_E}^2\ [\mathrm{m}^2/\mathrm{s}^2]$	9.59×10^{-5}	3.76×10^{-5}	8.10×10^{-5}	2.83×10^{-5}
$\boldsymbol{\sigma}_{V_D}^2\ [\mathrm{m}^2/\mathrm{s}^2]$	1.90×10^{-4}	1.70×10^{-4}	3.04×10^{-4}	1.00×10^{-4}
$\boldsymbol{\sigma}_{\phi}^2\ [\mathrm{rad}^2]$	3.02×10^{-5}	2.70×10^{-5}	4.90×10^{-6}	1.76×10^{-5}
$\boldsymbol{\sigma}_{\theta}^2\ [\mathrm{rad}^2]$	1.40×10^{-4}	3.73×10^{-5}	1.91×10^{-6}	2.76×10^{-5}
$\boldsymbol{\sigma}_{\psi}^2\ [\mathrm{rad}^2]$	1.33×10^{-4}	1.32×10^{-4}	1.42×10^{-5}	7.25×10^{-5}

As indicated, the actual and estimated variances are, overall, very similar. All estimated variances are within an order of magnitude of the actual variances. The estimated variances are, therefore, sufficiently accurate for use in the weighted mean.

7.2 Sensor Selection

The relative sensors that are available were discussed in Chapter 4. Thereafter, an explanation of how these relative measurements are incorporated into the final

estimates of the relative state and ship state estimates followed in Chapter 5.2. This section will initially identify the combinations of sensors that can be used. A subset will be selected for analysis thereafter. This subset will be evaluated upon relative and ship state estimation performance for a simulated ship deck landing of the helicopter.

The analysis in this section is performed with $f_{ARW} = 0.1$ and $f_{VRW} = 10$. These settings result in a substantial improvement in helicopter and ship estimation accuracy. The estimator would in practice be run using the optimal settings, as the results for these settings are more meaningful than the default values.

7.2.1 Sensor Combinations

Three degrees of deck instrumentation are examined in this project. These include fully instrumented, partially instrumented and uninstrumented ship decks. There are numerous combinations of relative sensors that can be chosen for each of these degrees of deck instrumentation. However, the number of combinations needs to be reduced in order to make the sensor selection task tractable. Two sensor configurations are examined for each deck instrumentation level: "complete" and "minimal".

The most complete set of sensors possible for the specific deck instrumentation level form complete configurations. Decisions were made between stereo or monocular vision sensors. In addition, a choice was made between single and multiple lasers.

In contrast, minimal configurations consist of the smallest set of sensors that can be chosen for the specific deck instrumentation level, provided that the required states can be measured. In several cases, however, multiple complete and minimal combinations were possible. The most useful combinations are included in the tests. Table 7.4 lists the configurations that were chosen.

7.2.2 Relative State Estimation

The vehicle trajectories that were chosen to simulate a helicopter landing on a heaving deck are presented in Appendix H. These trajectories feature sinusoidal disturbances to demonstrate the estimation performance more clearly. The helicopter and ship initial state estimates are plotted.

Novatel Align mode is simulated for the fully and partially instrumented ship deck cases. The latency in the Novatel GPS measurements is negligible relative to the update period of the helicopter's Kalman filters. Zero GPS latency will, therefore, be assumed.

Table 7.4: Sensor combinations available

Level of ship deck instrumentation	**Minimal Sensor Configurations**		**Complete Sensor Configurations**	
	ID	**Sensors**	**ID**	**Sensors**
Fully instrumented	M1	Ship estimates Align GPS Monocular vision	C1	Ship estimates Align GPS Monocular vision Multiple lasers
	M2	Ship estimates Align GPS		
Partially instrumented	M3	Align GPS Monocular vision	C2	Align GPS Monocular vision Multiple lasers
Uninstrumented	M4	Monocular vision	C3	Stereo vision Multiple lasers
			C4	Monocular vision Multiple lasers
			C5	Monocular vision Single laser

7.2.2.1 Initial Relative Estimates

The initial relative estimates are available for the fully instrumented deck sensor configurations. These estimates are calculated using Equation (5.2.2).

Table 7.5: RMS position and attitude error of the initial relative estimates

$\hat{\mathbf{p}}_{rel}$			$\hat{\boldsymbol{\theta}}_{rel}$		
N [mm]	**E [mm]**	**D [mm]**	ϕ **[deg]**	θ **[deg]**	ψ **[deg]**
13.4	23.4	33.6	0.047	0.049	0.068

The RMS error in the initial relative estimates is displayed in Table 7.5. Down estimates are noisier due to the dilution of precision inherent in the GPS measure-

ments. The yaw angle estimate is less accurate than the roll and pitch angles. This is expected, since the tilt-heading algorithm uses the roll and pitch measurements to calculate the yaw measurement.

7.2.2.2 Comparison of Monocular and Stereo Vision

The relative state measurements obtained using monocular vision are exhibited in Figure 7.10. The accuracy of the roll and pitch measurements increases as the helicopter nears the deck pattern. As expected, the yaw measurements exhibit very little sensitivity to hover height. Prior to touch down, a loss of monocular vision measurements occurs. This is because the camera is too close to the pattern. A minimum of four deck pattern markers are required to calculate measurements.

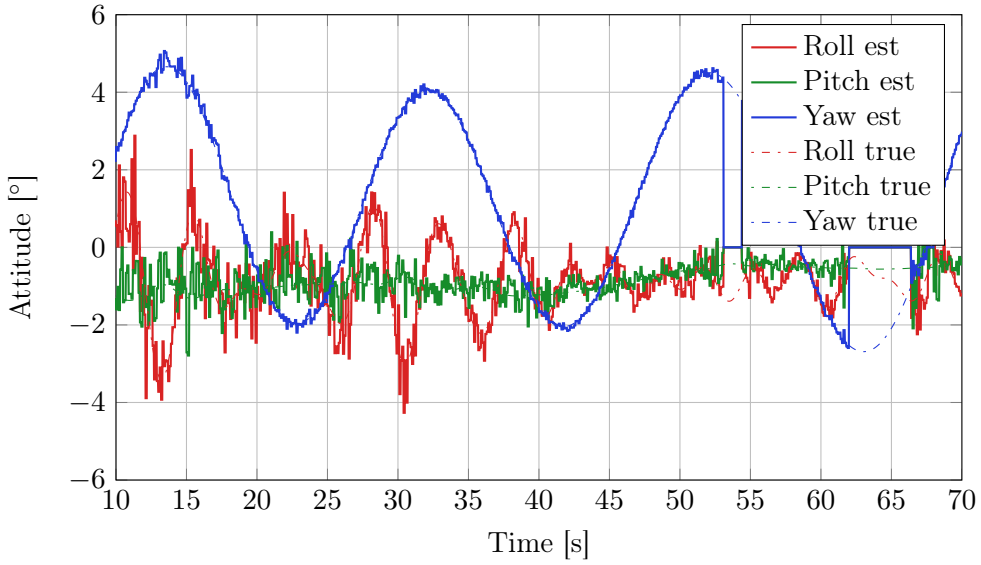


Figure 7.10: Monocular vision attitude measurements.

The performance of the stereo vision sensor is presented in Figure 7.11. The measurements are slightly less noisy than those of the monocular vision sensor. The blind spot, however, significantly reduces the state space of relative positions and attitudes within which stereo measurements are available. This blind spot is most problematic just before touchdown, when the helicopter is nearer the pattern.

7.2.2.3 Performance Using Complete Sensor Configuration

The relative estimation performance of the most complete sensor configuration, C1, is shown in Figure 7.12. The multiple laser sensor is able to provide very accurate

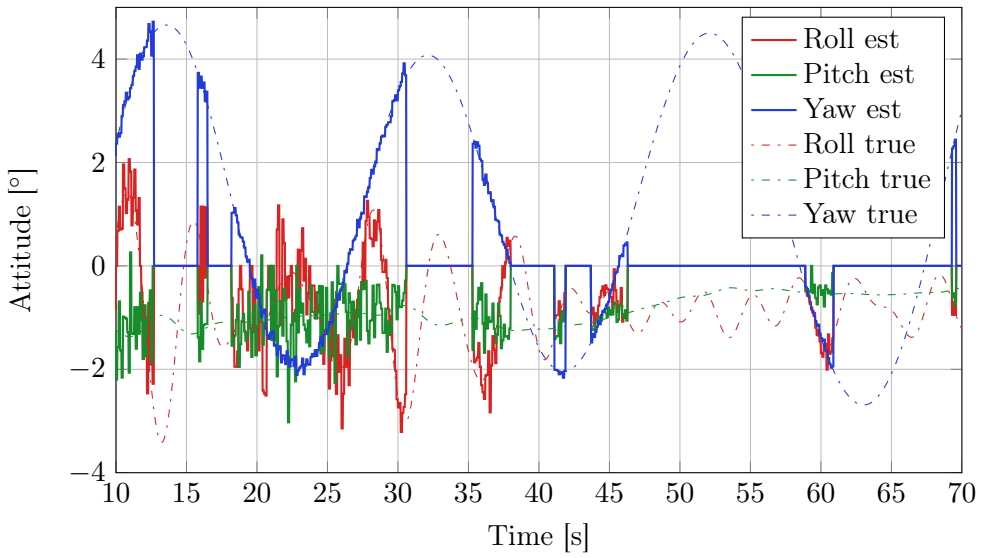


Figure 7.11: Stereo vision attitude measurements.

roll and pitch measurements. This significantly improves the estimation performance when the helicopter is hovering well above the deck.

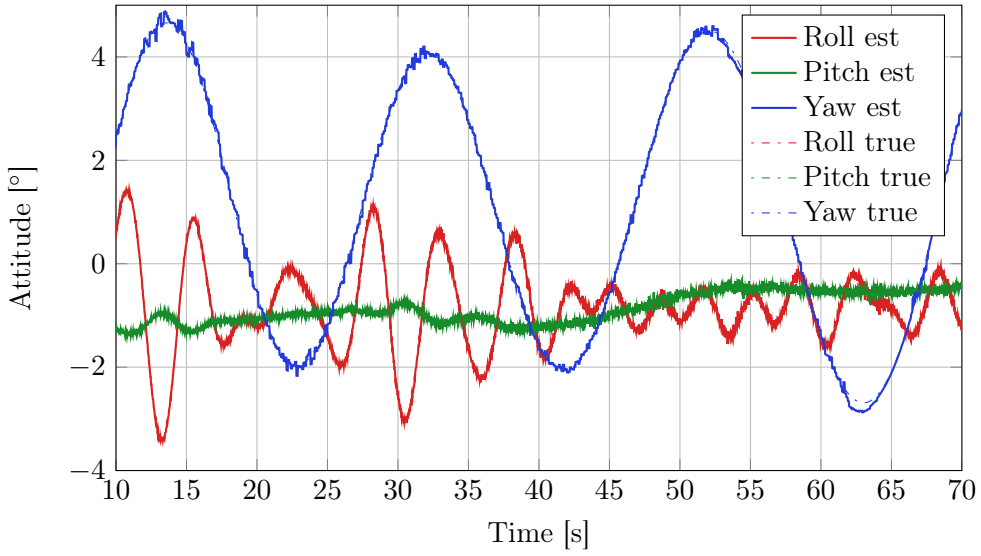


Figure 7.12: C1 relative attitude measurements.

The accuracy of the laser sensor actually improves with an increase in hover height. The area of the ellipse, traced by the multiple laser sensor, increases when hover height increases. A larger ellipse improves the condition of Equation (4.5.9). As a result, SVD is able to determine the relative roll and pitch more accurately.

The initial relative estimates are also hover height independent. The loss of vision measurements prior to touchdown is, consequently, no longer a problem. When vision and laser measurements are available, they improve the final relative estimates. When unavailable, the uncertainty in the final relative estimates merely increases.

7.2.2.4 Comparison of Sensor Configurations

The final relative position RMS estimation error is shown below in Figure 7.13. The choice of sensors does not greatly affect the final relative position estimation error. The difference in position accuracy, between using only monocular vision (M4) and using all available sensors (C1), is minimal. However, these RMS error statistics exclude the trajectory segments in which no measurements were obtained.

Combining pure monocular vision (M4) with a single laser rangefinder produces C5. The addition of the single laser provides only a minor improvement in the Down estimation accuracy. This result is trajectory-dependent. If the helicopter were to hover at a greater altitude above the deck, the laser would provide a greater improvement in Down estimates.

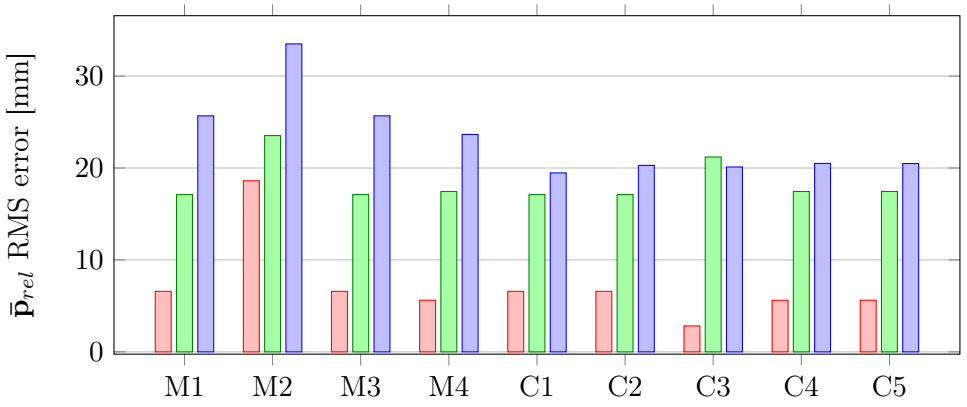


Figure 7.13: RMS error of final relative position estimates.

The final relative attitude RMS estimation error is indicated below in Figure 7.14. A vision-only or vision and Align sensor configuration results in a substantial final relative roll and pitch error. The error was significantly reduced by supplementing the vision sensor with initial relative estimates or the multiple laser sensor.

Stereo vision and multiple lasers (C3) provide the best roll and pitch accuracy. However, this is partly due to the missing measurements in several segments of the trajectory. Relative attitude estimates are not improved by augmenting monocular vision

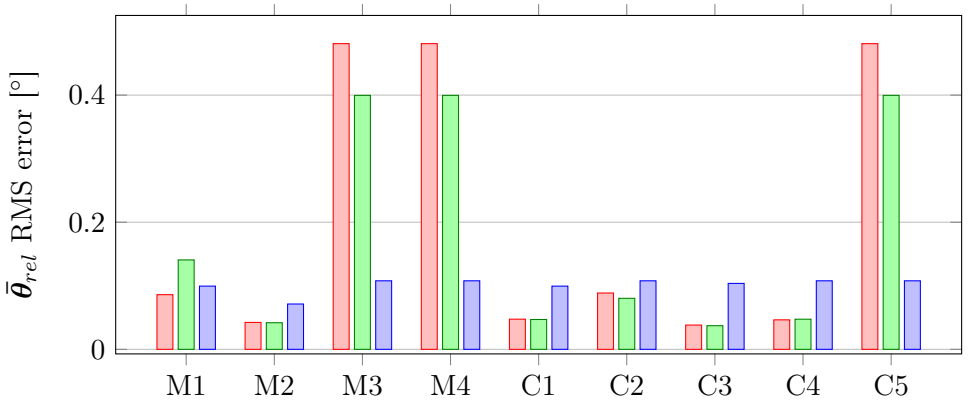


Figure 7.14: RMS error of final relative attitude estimates.

with a single laser rangefinder. This is because the relative Down measurement is not involved in the final attitude estimates calculation.

7.2.3 Ship State Estimation

Ship position is an absolute state. As such, drift in the GPS position measurements results in a drift in the final ship position estimates. GPS drift cannot be corrected for because it is the only absolute position sensor available to the estimator. Fortunately, the frequency of GPS drift is very low. GPS drift is acceptable for motion prediction because the higher frequency components are of greater importance for ship motion prediction.

This drift is significantly larger than the error produced by any of the other sensors. In order to compare the accuracy of each sensor configuration, the simulated rate random walk in the helicopter and ship GPS position measurements is disabled.

The RMS error in the final ship position and attitude estimates is indicated by Figures 7.15 and 7.17 below. The accuracy of the ship position and attitude states is highly dependent on the accuracy of the relative roll and pitch estimates. This is because the offset of the ship deck from the point of estimation of the ship motion is large. Small errors in relative attitude measurement will result in large ship position estimation errors, as indicated by Equation (2.2.4). A comparison of sensor combinations C4 and C5 reveals that the inclusion of the multiple laser sensor significantly improves measurement of the ship's roll, pitch and heave states.

The accuracy of the helicopter and initial ship attitude estimates is very important in the fully instrumented ship deck (M1, M2 and C1). This leads to accurate initial relative estimates, as well as highly accurate deck attitude estimates. This is because the deck attitude is calculated from the helicopter and initial relative estimates. The

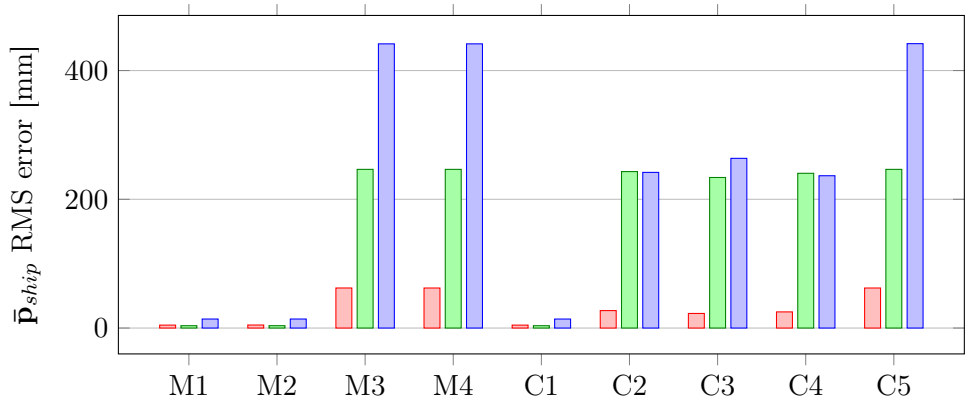


Figure 7.15: RMS error of final ship position estimates.

helicopter attitude estimates are therefore used twice in this calculation. Since the deck attitude error is so small, amplifying it by the position offset between deck and ship center results in a significantly smaller error than the other sensor options. The ship attitude estimates obtained from C1 are shown below in Figure 7.16.

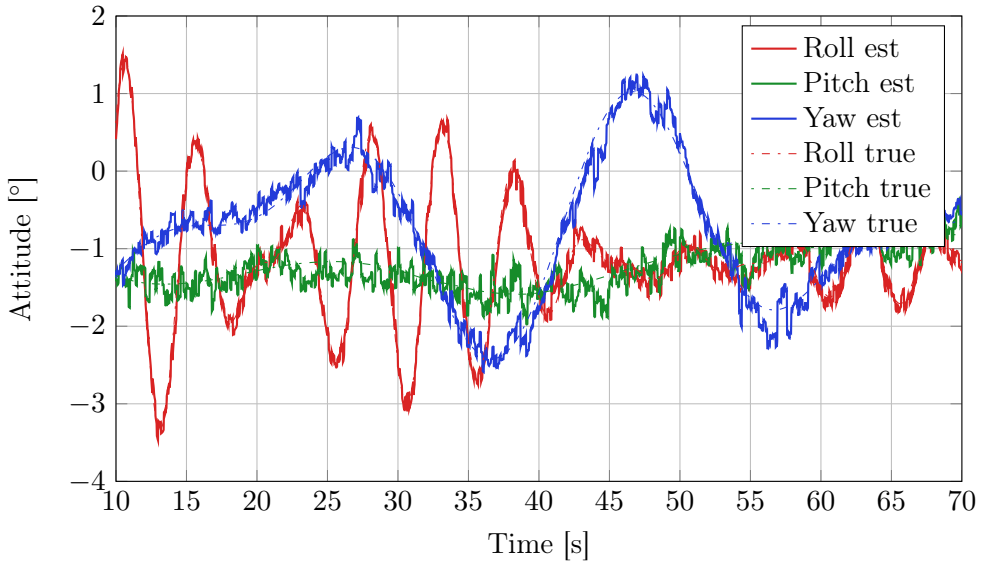


Figure 7.16: C1 ship attitude measurements with Align enabled.

Figure 7.17 below demonstrates that stereo vision (C3) exhibits more error than monocular vision (C4). This is due to the more frequent loss in stereo measurements. The error increases when only initial relative estimates are available. Monocular vision measurements are available for most of the trajectory. Therefore, C4 has the benefit of the averaging of the measurements of the two sensors over a longer time period.

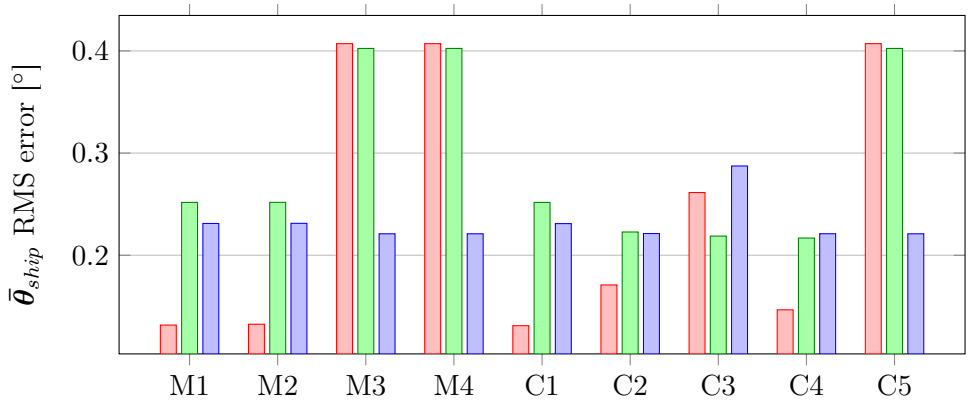


Figure 7.17: RMS error of final ship attitude estimates.

The ship position measurements, which are obtained using only the monocular vision sensor (M4), are shown in Figure 7.18 below. As expected from the relative position results, accuracy is enhanced as the helicopter nears the deck.

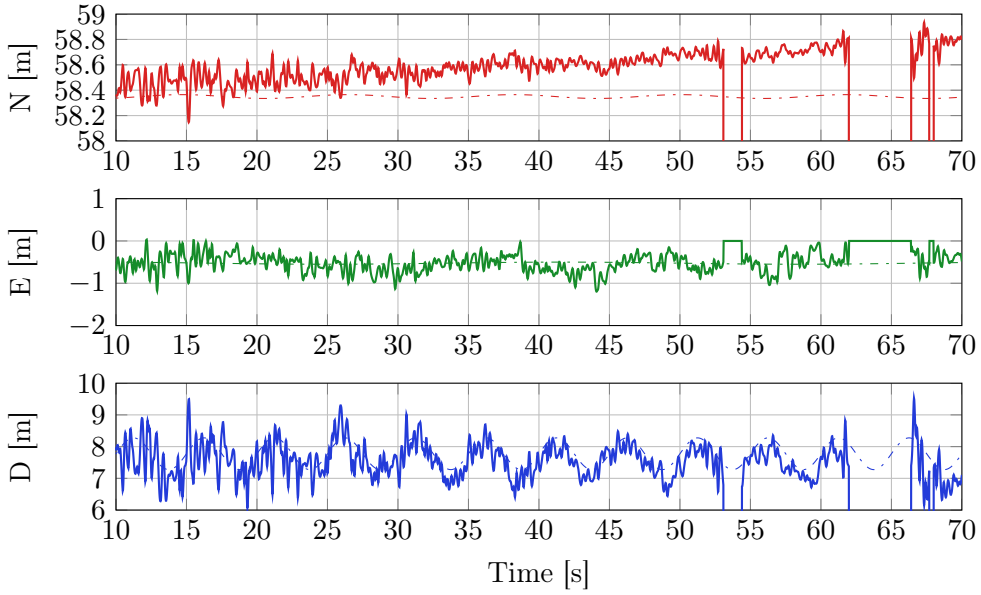


Figure 7.18: M4 ship position measurements with Align enabled.

The corresponding ship attitude estimates are presented in Figure 7.19. Significant noise is present in the roll and pitch measurements when the hover height is large, but reduces substantially as the height decreases.

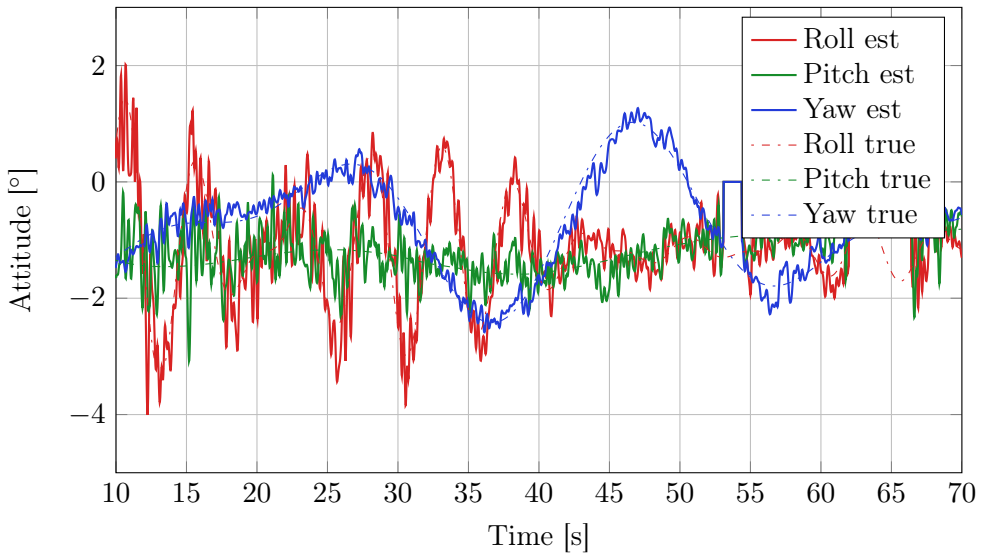


Figure 7.19: M4 ship attitude measurements with Align enabled.

7.2.4 Discussion of Results

Single-point GPS receivers exhibit large amplitude low-frequency drift in their position measurements. However, a pair of nearby GPS receivers operating in Novatel Align mode would permit accurate relative position measurements to be obtained. This is explained in Chapter 3.5. This chapter reveals that the estimation structure described in Chapter 5.2 enables accurate relative estimates to be obtained despite the drift in the helicopter and ship GPS position measurements.

The choice of sensors for relative and ship motion estimation is primarily dependent on the hover height of the helicopter above the deck. If the ship deck is heaving violently due to large swell, hovering well above the deck would be necessary during the ship motion prediction stage. In this situation, the multiple laser rangefinder would complement the monocular vision sensor well. The multiple laser rangefinder would provide accurate roll, pitch and Down measurements. The vision sensor provides accurate North, East and yaw measurements. The combination of the multiple laser rangefinder and monocular vision sensor would enable the helicopter to accurately estimate and predict the ship states from a safe altitude.

Stereo vision is slightly more accurate than monocular vision. The blind spot inherent to the stereo geometry is a major issue when the helicopter rolls and pitches relatively close to the deck. The stereo blindspot could be minimised by increasing the field-of-view of each camera. Depth accuracy would be lost, however, unless the image resolution is also increased. Monocular vision has a smaller blindspot and thus better suited for the task.

A monocular-vision-only configuration exhibits far more error than fully instrumented and multiple laser-based configurations in this analysis. However, the accuracy of monocular vision could be significantly improved by increasing the image resolution. This is confirmed in Figure 7.20 below, which shows the monocular vision RMS error as the width and height of the image is scaled upward by a factor. Measurement error diminishes rapidly as resolution increases.

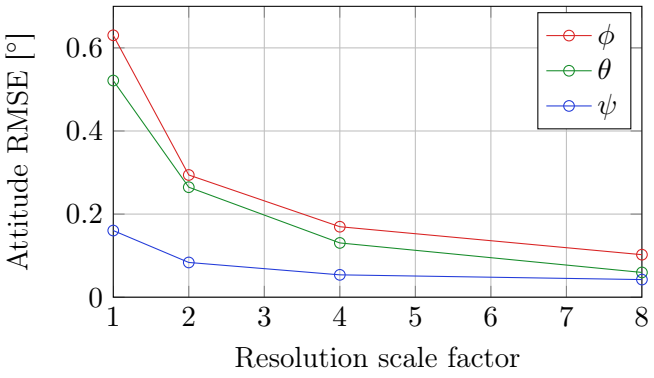


Figure 7.20: Monocular vision attitude estimation error for an increase in image resolution.

The increase in processing requirements could be comfortably achieved with the advent of small, powerful and cost-effective microcomputers, such as the Gumstix. Hence, a monocular vision sensor could prove adequate for estimating the relative and ship state estimates.

7.3 Chapter Summary

A detailed analysis of the helicopter estimator was conducted in this chapter. Two attitude determination algorithms that have been used previously within the ESL were initially discussed. These algorithms include the TRIAD and tilt-heading algorithms. The sensitivity of each of these algorithms to biases in sensor measurements was examined. The tilt-heading algorithm was seen to be less sensitive to biases.

An analysis of gyro bias estimation followed. Constant gyro biases were first tested, followed by random walk biases, which were captured from actual gyro data. In both cases the estimator successfully estimated the biases. Bias estimation was shown to reduce attitude estimation error in the presence of gyro biases.

The sensitivity of the helicopter estimator to variation in the Kalman filter's Allan variance parameters was then conducted. The values of ARW, VRW and RRW used in the process and measurement noise covariance matrices were varied one at a time.

The resultant position and attitude estimation error was recorded. The attitude estimates were substantially improved. This was achieved by decreasing the ARW by an order of magnitude and by increasing the VRW by an order of magnitude. The estimator was insensitive to the value of RRW.

The variance estimated by the Kalman filters was then compared to the actual variance in their estimates. The estimates were shown to closely resemble the actual variances. This demonstrated that the Kalman filters were performing correctly.

An analysis of the relative and ship state estimators followed. A subset of the possible sensor combinations was chosen for the analysis. Complete and minimal sensor configurations were selected for each of the three degrees of deck instrumentation.

The accuracy of vision-based sensors was shown to be hover height dependent. It was reasoned that monocular vision could prove adequate and cost-effective for relative and ship state estimation if the image resolution is increased. If a large hover height is desired, the use of the multiple laser sensor would significantly improve accuracy. Final selection of whether the multiple laser sensor is required, therefore, depends on the height at which the helicopter hovers when performing ship motion prediction.

The next chapter discusses the hardware implementation of the estimator and shows the results of flight tests that were performed.

Chapter 8

Hardware, Integration and Flight Testing

In order to meet the computational demands of the new estimator designed in this thesis, a new hardware solution had to be developed. This chapter describes the implementation of the new estimator and the integration of the existing avionics. Results obtained from flight tests performed with the new estimator and hardware conclude the chapter.

8.1 Avionics System

The onboard computer (OBC) developed in the ESL is a modular, PIC-based avionics package. Communication between modules occurs over a CAN bus. Current increases in computational demands have lead to this system reaching its memory and processing power limits. In addition, the CAN bus is becoming saturated, which results in packet loss. A completely new avionics system was required in order to be able to implement the new estimator.

Behrens' development of a new avionics system occurred concurrently with this thesis. Although the system was not fully completed in time for this project, certain key components were available. These components include the Gumstix processor, a real-time operating system and the CAN to Ethernet converter board used to interface with the existing hardware.

8.1.1 The Gumstix

The Gumstix computer-on-module (COM) has an ARM-based architecture. It has several orders of magnitude more processing power than the dsPICs used by the existing OBC. Processing-intensive control and estimation techniques can thus be employed, whereby, previously, a steady-state Kalman filter was used to estimate position and velocity states. The covariance matrix of the attitude estimator was

approximated as a diagonal matrix to minimise the processing requirements. The Gumstix, however, permits full, time-varying KFs and EKFs to be run with ease.

In order for the Gumstix to be made suitable for a real-time application such as control and estimation, a real-time operating system is required.

8.1.2 Real-Time Operating System

Microcontrollers, such as the PIC microcontrollers used in the OBC, can be programmed to satisfy real-time constraints. This is achieved by means of timers and interrupts. However, as the system grows in complexity, so does the task of the programmer. Furthermore, the processing requirements may increase due to unforeseen functionality demands. In this case, printed circuit boards often have to be redesigned and code rewritten in order to accommodate a new microcontroller.

In contrast, operating systems abstract away hardware differences. Code can therefore be reused with little to no modification. Device drivers are often prewritten for the operating system. This minimises the need for masses of low-level code to be written.

Nonetheless, there are several disadvantages to using operating systems. There are large overheads in terms of memory and storage requirements. Guaranteeing real-time objectives are met is often more challenging than for a simple microcontroller design. Higher-priority kernel processes may interrupt the user's processes indefinitely. Microcontrollers, in comparison, give the user complete control over resource usage.

Behrens compiled a Linux distribution for ARM microprocessors, known as the Yocto Project, for the Gumstix. The operating system was made real-time by compiling it with the Preempt RT patch. This would enable user programs to preempt kernel processes.

8.1.3 Interfacing with Existing Avionics Hardware

The new avionics system is designed to be modular. The modules communicate amongst each other via Ethernet instead of the CAN bus. However, the system was designed to be backwards compatible with the existing OBC and its associated IMU board, servo motor board and other hardware. In order to achieve this, Behrens developed a CAN to Ethernet converter (CAN2Eth). CAN packets sent from the OBC are encapsulated within User Datagram Protocol (UDP) datagrams by CAN2Eth.

These packets are then sent over Ethernet and received by the Gumstix. The reverse

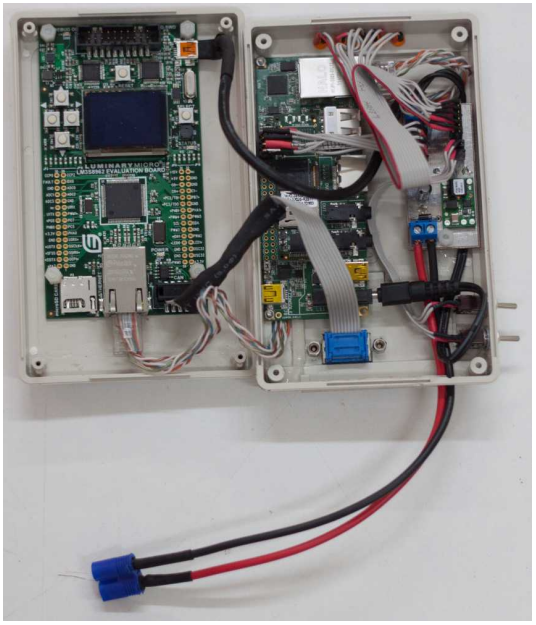


Figure 8.1: CAN2Ethernet board (left) and Gumstix (right) in the Gumstix flight box.

procedure is performed when sending data from the Gumstix to the CAN bus. The use of UDP is ideal due to its low overhead. Figure 8.1 shows CAN2Eth and the Gumstix housed within the flight box.

8.1.4 Toolchain and Development Workflow

The estimator's code was initially developed and tested on a desktop version of Ubuntu. This enabled sophisticated Integrated Development Environments (IDEs) and network analysis tools, such as Wireshark, to be used. Once working, the code was cross-compiled for the ARM architecture. This reduced compilation timer[1] and minimised the number of write cycles incurred by the SD card, which has a limited lifespan. It also had the advantage that the same G++ compiler was used to compile the code for both devices.

The PC was assigned a static IP in the CAN2Ethernet board's network address range. Thereby, all the same UDP packets from the OBC, HIL and servo boards are received by the PC that the Gumstix would formerly receive. Software testing of the Gumstix can be performed in this way using the exact same code that would be compiled on the Gumstix.

[1]It takes approximately 8.5 minutes to compile the estimator code natively on the Gumstix. In comparison, it takes approximately 30 seconds to cross-compile within VirtualBox on an i5 with 3GB RAM.

This workflow is unidirectional, with all code changes made on the PC and then copied over to the Gumstix. Software bugs are therefore minimised, which results in more reliable code.

A Linux-based avionics system that communicates over Ethernet offers a host of tools for software development and debugging. SSH (Secure SHell) can be used to execute commands on the Gumstix remotely. In addition, SCP (Secure CoPy) can be used to copy files between the Gumstix and PC. This enables code to be developed on a desktop PC rather than editing remotely over serial, which is both slow and tedious[2]. The estimator code developed needed to be tested by using hardware-in-the-loop simulation.

8.2 Hardware-in-the-Loop

Hardware-in-the-Loop (HIL) testing is a crucial stage of the development cycle for all aircraft and other vehicles in the ESL. Software-only testing does not uncover certain potential issues, such as communication delays and malfunctions, which are readily observed in HIL tests. Furthermore, HIL testing can determine whether there is saturation of the CAN bus and radio link, as well as any inability of the SD card to cope with the logging data rate.

The decision was made that the estimator would initially be flown alongside the existing OBC's estimator. The estimator would act as a payload instead of an active component in the avionics system. This would allow for the newly developed estimator to be tested during flight without risk to the overall system. A comparison of the new and existing estimators could then be made using the sensor data captured in flight.

As a result, the Gumstix-based estimator would not be "within the loop". Instead, the output state estimates generated during HIL testing would be logged to the Gumstix memory card to be analysed after the hardware simulation or flight test. Figure 8.2 shows the placement of the Gumstix flight box beneath the helicopter. Thereafter Figure 8.3 illustrates the HIL configuration used in this research, as well as modifications made by the author.

The Simulink HIL model used for the testing is the version used by [11] and previous students. However, the sensor simulation models have been replaced with those shown in Appendix A.1.

[2] [3] describes the potential benefit of compiling the Gumstix's Linux with a desktop image. Although this may aid development, the graphical user interface may take up significant system resources. A real-time avionics system would avoid luxuries such as a graphical user interface.

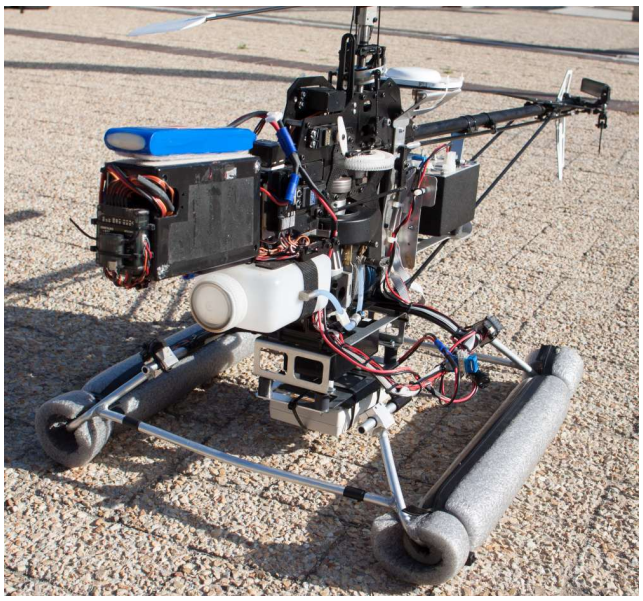


Figure 8.2: Gumstix flight box cable-tied beneath helicopter.

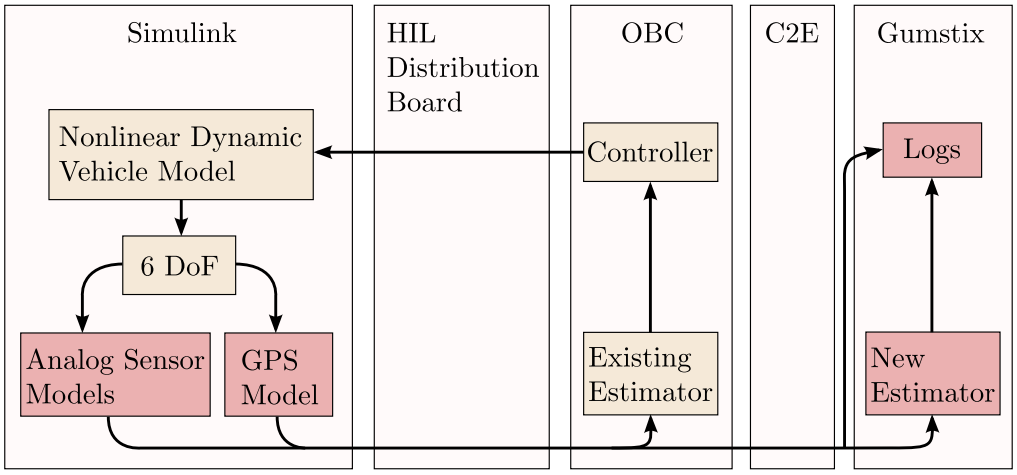


Figure 8.3: HIL configuration. Red blocks indicate modifications made by the author.

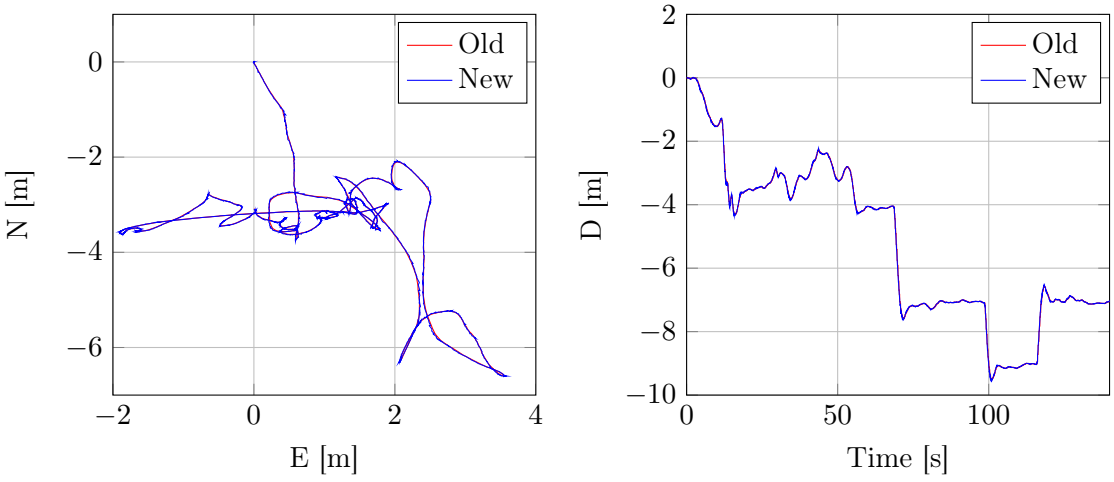
8.3 Flight Testing

A performance analysis of a vehicle's state estimator is inherently difficult to conduct. A ground truth of the vehicle's trajectory is required in order to measure error in the estimates. The ground truth needs to be significantly more accurate than the estimates being analysed. However, the sensors used to calculate the vehicle's estimates are usually the most accurate sensors available. The Vicon motion capture system is used by several projects in the literature to establish a highly accurate ground truth [67; 73]. The cost of this motion capture system is, unfortunately, well outside the budget of this project.

Despite this, flight testing is still very useful. The existing OBC's estimator has performed reliably over several years of flight tests in the ESL and will be used as a comparison to the new Gumstix-based estimator. This will enable a qualitative analysis of the estimator to be performed.

8.3.1 Flight Test One

The first flight test was held at Vergenoegd field on 5 November 2013. The estimator was evaluated whilst step testing the newly adjusted controller gains. As a result, the helicopter was flown in a geometric pattern. The trajectory is illustrated below in Figures 8.4b and 8.5.



(a) North and East estimates

(b) Down estimates

Figure 8.4: Position estimates of the new and old estimators for flight test one.

The old estimator has a complete reliance on GPS measurements, as the propagation updates were disabled. Thus, the old estimator's position and velocity estimates are equivalent to the DGPS measurements captured. In comparison, the new estimator's position and velocity estimates differ slightly from these measurements. The propagation updates have a minimal impact on the estimates as the accelerometer readings are relatively noisy.

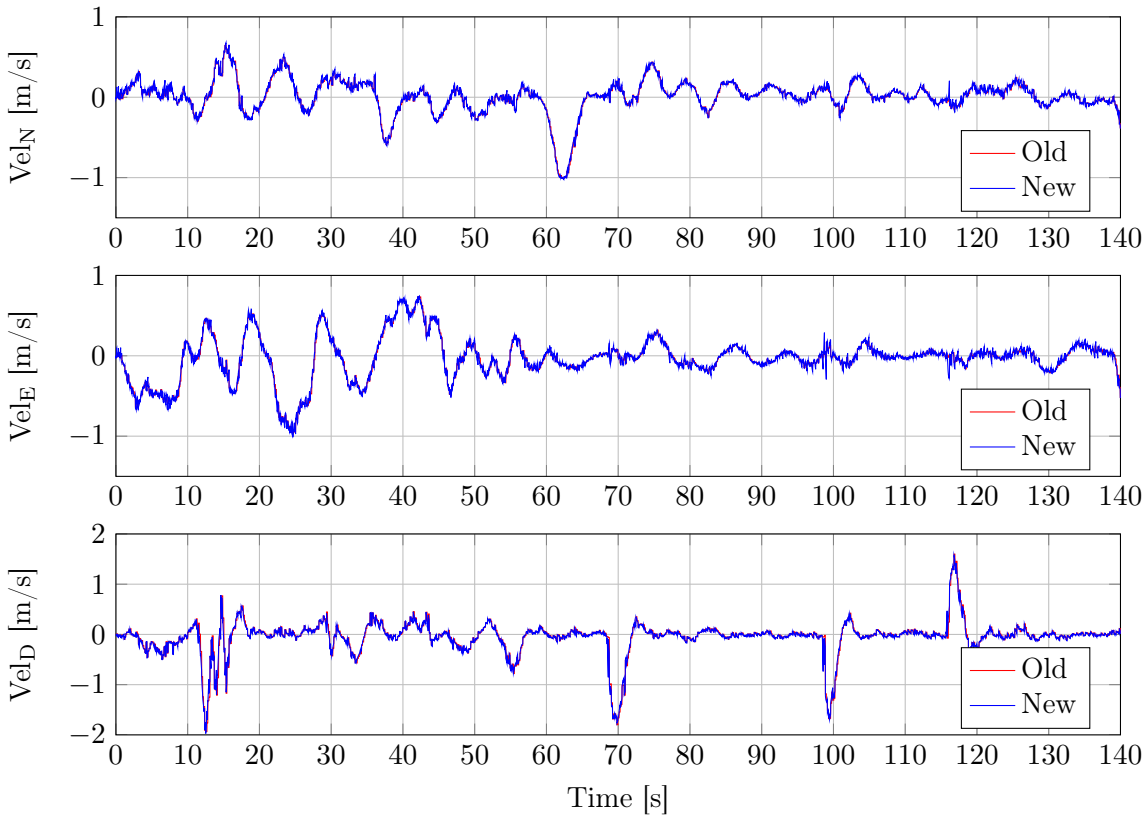


Figure 8.5: Velocity estimates of the new and old estimators for flight test one.

There are more discrepancies between the new and old attitude estimates than seen in the position and velocity estimates. These differences in attitude estimates are seen in Figure 8.6. As indicated in the software simulations, the new estimates prove to be noisier. In addition, the new estimator trusts the measurement updates more than the gyros and accelerometers.

8.3.2 Flight Test Two

A second flight test was conducted at the Helderberg Radio Flier's Club on 8 November 2013. The safety pilot flew the helicopter in a fairly arbitrary pattern in order to test the estimator more thoroughly. This pattern included two complete revolutions in heading. Figure 8.7 below shows the trajectory of the helicopter. However, safety concerns regarding strong winds prevented the pilot from flying higher and for a longer duration.

The position and velocity state estimates exhibit the same characteristics that were seen in the previous flight test. The estimator is seen to rely heavily on the DGPS measurements. However, the smoothing effects of the propagation updates are ap-

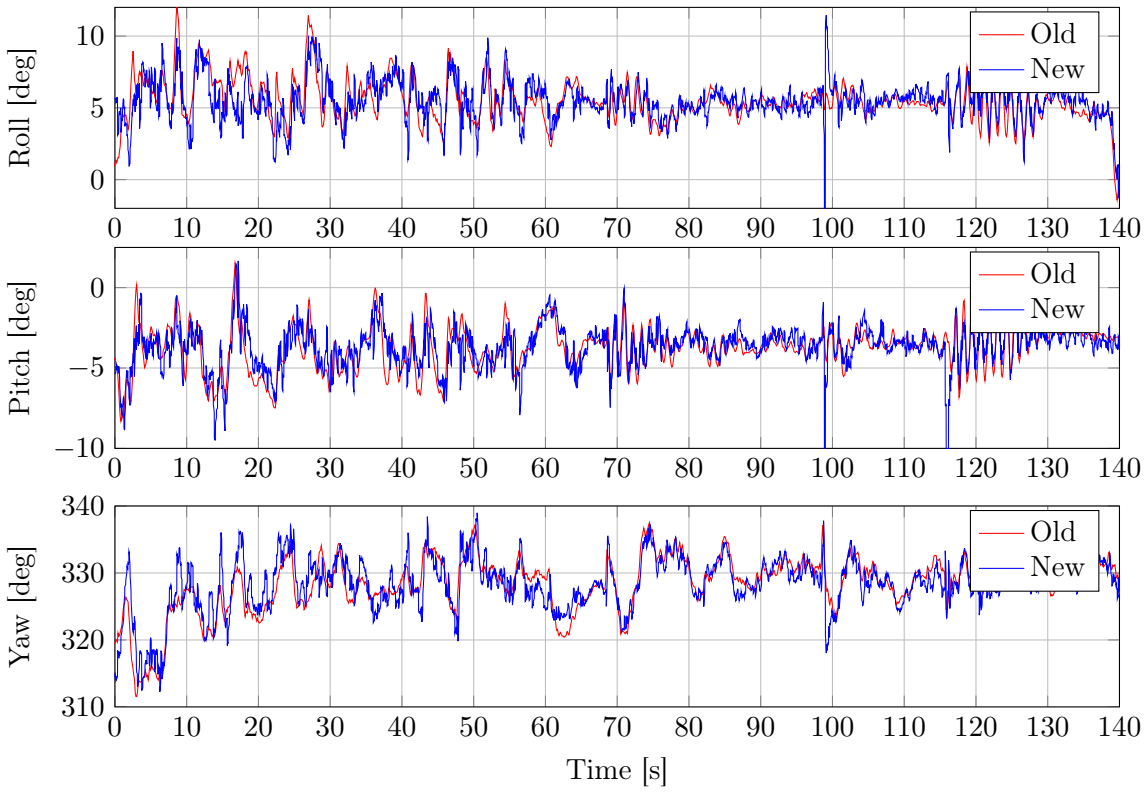
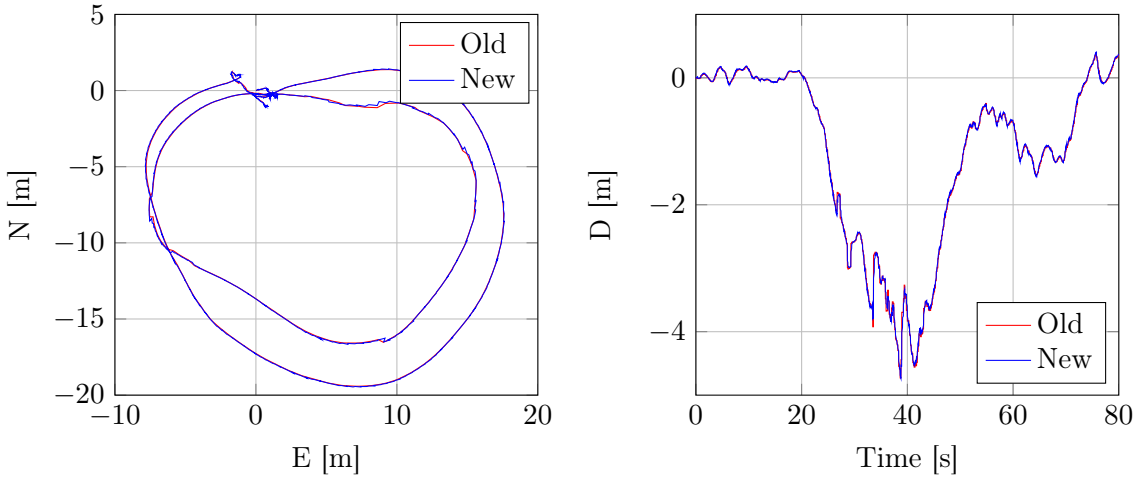


Figure 8.6: Attitude estimates of the new and old estimators for flight test one.



(a) North and East estimates

(b) Down estimates

Figure 8.7: Position estimates of the new and old estimators for flight test two.

parent during the rapid changes in the Down velocity.

Figure 8.9 shows the attitude estimates of both estimators. Here it can be noted that both estimators exhibit the same overall behaviour. However, the new estima-

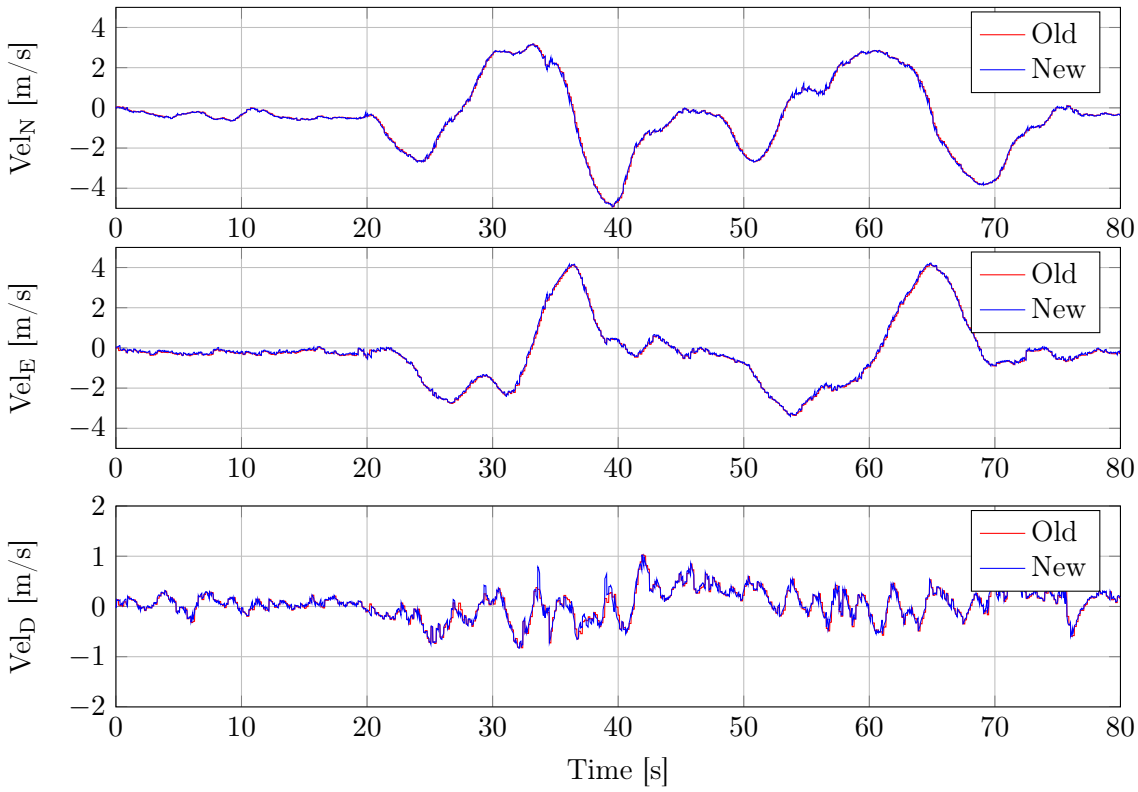


Figure 8.8: Velocity estimates of the new and old estimators for flight test two.

tor's estimates are noisier than the old estimator's. The attitude estimates are very sensitive to velocity estimation noise. This is due to the use of the derivative of the velocity estimates, which amplifies high frequency noise. The derivative of the Down velocity impulse in the 34^{th} second causes a sudden upward spike, followed by a sudden downward spike in the attitude estimates.

The tilt-heading algorithm recursively depends on previous attitude estimates. Subsequently, the roll angle estimate becomes noisy until both the velocity estimation noise subsides and the attitude estimates have converged.

The roll angle is fairly constant for the first 20 seconds of the recording. During this period the roll is approximately five and zero degrees for the new and old estimators, respectively. In addition, there is a significant difference in the pitch estimates during the 24-32 second interval and 52-59 second interval.

There is, unfortunately, no ground truth available to determine which estimator is more accurate on these occasions. The discrepancy between the estimates of the new and old estimators is too small to be distinguishable in video analyses of the flights. However, both flight test results exhibit similar characteristics to that of the simulation results. The increased noise of the new estimator compared to the old

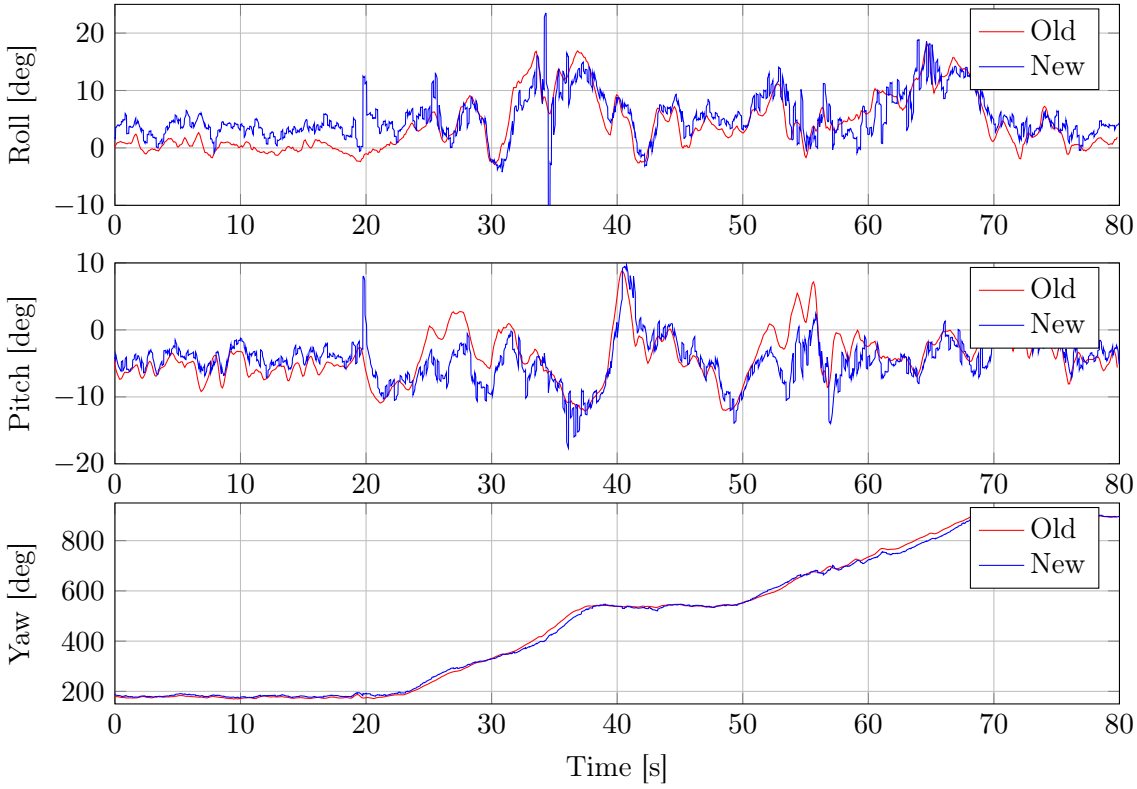


Figure 8.9: Attitude estimates of the new and old estimators for flight test two.

estimator matches that were observed in the simulations. Confidence can therefore be instilled in the noise models created in Chapter 3 and the software simulations performed in Chapter 7.

8.4 Chapter Summary

In order to perform flight testing of the estimator, the estimator needed to be integrated into the existing avionics stack. This chapter provided an overview of the hardware and software used in the implementation of the new estimator. A Gumstix microcomputer featuring a real-time Linux-based operating system was used. The communication between the OBC and Gumstix was established via a CAN to Ethernet board.

The resultant system was subsequently flight tested. Due to the lack of ground truth data available, a qualitative analysis of the estimator performance was conducted. The resultant estimates were noisier than those of the existing estimator. This was predicted by the analyses performed in Chapter 7. The conclusion was reached that the estimator is functioning correctly as the estimates generally behaved similarly.

Chapter 9

Conclusions and Recommendations

The primary objective of this thesis was to develop a unified estimation framework for landing an unmanned helicopter on a ship deck. In order to achieve this, simulation models of the sensors were developed. A flexible estimation structure, which permits various combinations of sensor measurements to be incorporated in an optimal manner, was proposed. The estimation accuracy was investigated for three different degrees of ship deck sensor instrumentation. These included ship decks that were fully instrumented, partially instrumented and those without instrumentation. The fully instrumented deck featured a complete IMU and GPS receiver placed on the deck. The partially instrumented deck featured only a GPS receiver.

Modifications were subsequently made to the helicopter's state estimator. These changes included the addition of active gyro bias estimation. The process and measurement noise covariance matrices were then calculated directly from the sensor noise analyses. The helicopter state estimator was implemented on a microcomputer and flight tested. A qualitative comparison of the newly modified estimator to that of the previous, followed. This was due to the absence of a ground truth of the helicopter states.

9.1 Conclusions

Based on the analyses conducted and the results obtained, the following conclusions were drawn:

9.1.1 Sensor Modelling

Noise analysis was performed for each of the available inertial, absolute and relative sensors. Various noise analysis techniques were used. These included Allan variance and power spectral density analysis, autocorrelation and Monte Carlo simulation.

The effect of temperature on the inertial sensors was examined. The sensor simulation models were shown to closely resemble the noise characteristics of actual sensor data captured in the laboratory and outdoors. However, the noise levels of MEMS inertial sensors are known to also depend on vibration. Since Allan variance analysis requires the sensors to be stationary, the effects of vibration on the sensors during flight could not be measured.

9.1.2 Relative and Ship State Estimation

The relative and ship state estimation accuracy was examined for three different levels of deck instrumentation (i.e. fully instrumented, partially instrumented and uninstrumented).

- If a relatively low hover height is feasible, monocular vision is sufficient for estimating all required states. Differential GPS (Novatel Align) is unnecessary.
- In large swell, a higher hover altitude may be desirable. In this case, the inclusion of the multiple laser sensor significantly improves relative and ship roll and pitch estimates.

Communication between the helicopter and ship can thus be avoided. The system is therefore robust against communications failure and electronic warfare.

9.1.3 Helicopter Estimator

Active gyro bias estimation was incorporated into the helicopter's attitude estimator. Experiments performed using simulated and actual gyro noise demonstrate that this technique is highly effective in estimating and removing constant and time-varying biases. Since the incorporation of bias estimation involves the appending of latent variables to the estimator, the degree of observability decreases. A decrease in the degree of observability results in a reduction in estimation accuracy. However, around hover conditions these effects are minimal.

A method to calculate the Kalman filter noise matrices directly from sensor noise analyses was proposed. The noise matrices were calculated by linearising the process and measurement functions. It was shown that the resultant noise matrices were within the right ballpark in Kalman filter performance. Although the estimation results were satisfactory, the subsequent tuning of the noise parameters lead to better performance. This was to be expected, as the EKF is suboptimal, thus requiring manual adjustment in practice.

9.2 Recommendations

There is the need for more extensive sensor modelling, the use of automatic differentiation for the calculation of large Jacobians and the possibility of using an unscented Kalman filter or particle filter instead of an EKF. A discussion of these recommendations follows.

9.2.1 Extensive Sensor Modeling

The sensor modeling performed in this project is a significant improvement over the previous sensor models used in HIL testing. However, the effects of vibration on the noise levels of the MEMS sensors has not yet been investigated in this study. Unfortunately the Allan variance technique cannot be used to analyse the effects of vibration of the helicopter in flight. This is because Allan variance analysis requires several hours worth of data and stationary sensors. However, the higher frequency noise components can be examined using much shorter data captures. It is recommended that an analysis be performed by means of vibration of the IMU at a similar frequency to that of the helicopter whilst in flight.

9.2.2 Automatic Differentiation for Large Jacobians

Calculating Jacobian transforms using symbolic differentiation enables exact derivatives to be calculated. For the simpler Jacobians, the analytic expressions returned by symbolic differentiation give valuable insight into the behaviour of the linearisation.

Many of the Jacobians calculated in this project are, however, enormously large. Little insight can be gained by inspecting the analytic expressions obtained. This is particularly true of the TRIAD and tilt-heading Jacobians.

For such cases, automatic differentiation is better suited. Automatic differentiation provides the exact derivative. In addition, it is computationally efficient and can be used to determine the derivative of algorithms that contain loops and condition statements, unlike symbolic differentiation. As described in Appendix E, this technique was used in this study to check the accuracy of the Jacobians. The use of automatic differentiation would, however, prove invaluable for nonlinear observability analysis and real-time calculation of the noise matrices.

9.2.3 Investigation of the UKF and Particle Filter

Kalman filters approximate process noise as Gaussian distributed noise. These Gaussian distributed random variables are propagated through the system analytically.

If the nonlinearities of a system are too severe, linearising the system may prove insufficient.

However, unlike the EKF, the Unscented Kalman filter (UKF) does not linearise the system. Instead, it propagates specially chosen points through the nonlinear function. These "sigma points" completely describe the mean and covariance of the posterior distribution up to the third order, using Taylor series expansion [74]. Conventionally, the EKF uses only a first-order approximation. The UKF is therefore both computationally efficient, due to deterministic sampling of the distribution, and more accurate than the EKF.

If the Gaussian assumption does not hold for a given system, a more general form of the Bayes filter may be necessary. The particle filter, also known as sequential Monte Carlo [75], is able to completely describe arbitrary distributions. The distribution is sampled at a large number of locations. Similarly to the UKF, these samples are propagated through the nonlinear function. The posterior distribution is then resampled.

Due to the computational requirements, particle filters are not yet common-place in real-time applications. However, the operations performed by the particle filter are highly parallelisable. Implementing the algorithm on field-programmable gate arrays (FPGAs) would therefore enable massive speed-up factors to be obtained. This could enable real-time constraints to be met.

Nonlinearities are most present when the helicopter undergoes rapid motion relative to the Kalman filter update rate. Investigating the use of the UKF or even particle filter might yield estimation improvements under these conditions.

9.2.4 Groundtruth for Helicopter Trajectory

The evaluation of the performance of an aircraft's estimator is a difficult task as a ground truth of the vehicle's trajectory is required. The sensors used to create the ground truth need to be significantly more accurate than the vehicle's estimator.

There are two possible options to obtain an accurate ground truth. Firstly, a motion capture system, such as the Vicon system, could be used. However, the Vicon system is very expensive and well outside of the budget of this research. Nonetheless, it may be possible to develop a suitable system in-house. This could involve using multi-view stereo to track markers placed on the vehicle, similarly to that done by Vicon. A Kalman filter could be used to track the helicopter states from one frame to another. This would enable sub-pixel and sub-frame estimation of the helicopter states.

Alternatively, a more accurate IMU could be used alongside the unit being tested. The estimates from each estimator could then be compared.

9.3 Chapter Summary

The goal of this research was to develop an estimation framework capable of landing an autonomous helicopter on a ship deck. This objective has been achieved and demonstrated through extensive software simulation and flight testing of the helicopter estimator. There is opportunity for future research in the modelling of the effects of vibration on sensor noise, the potential replacement of the EKF with an alternative estimator and the capturing of a groundtruth of the helicopter's states.

List of References

[1] Steele, J.: Launching a new era: Navy creates 1st unmanned helicopter unit. 2012. Available at: http://www.utsandiego.com/news/2012/Sep/09/launching-new-era-navy-creates-1st-unmanned-helico/

[2] Maxey, K.: X-47B drone completes first autonomous carrier landing. 2013. Available at: http://www.engineering.com/DesignerEdge/DesignerEdgeArticles/ArticleID/5989/

[3] Swart, A.D.: *Monocular vision assisted autonomous landing of a helicopter on a moving deck*. Master's thesis, Stellenbosch, South Africa, Mar 2013.

[4] Smit, P.E.: *Development of a 3-DOF motion simulation platform*. Master's thesis, Stellenbosch, South Africa, Mar 2010.

[5] Carstens, N.: *Development of a low-cost, low-weight flight control system for an electrically powered model helicopter*. Master's thesis, Stellenbosch, South Africa, Apr 2005.

[6] Sprague, K. *et al.*: Design and applications of an avionics system for a miniature acrobatic helicopter. In: *Digital Avionics Systems, 2001. DASC. 20th Conference*, vol. 1, pp. 3C5/1–3C5/10. Florida, 2001.

[7] Groenewald, S.: *Development of a rotary-wing test bed for autonomous flight*. Master's thesis, Stellenbosch, South Africa, 2006.

[8] Bijker, J.: *Development of an attitude heading reference system for an airship*. Master's thesis, Stellenbosch, South Africa, 2006.

[9] Hough, W.J.: *Autonomous aerobatic flight of a fixed wing unmanned aerial vehicle*. Master's thesis, Stellenbosch, 2007.

[10] Visser, B.J.: *Die presisie landing van 'n onbemande vliegtuig*. Master's thesis, Stellenbosch, South Africa, 2008.

[11] De Jager, A.M.: *The design and implementation of vision-based autonomous rotorcraft landing*. Master's thesis, Stellenbosch, South Africa, Mar 2011.

[12] Systems, T.: TRW completes first automatic takeoff and landing of U.S. Army's Hunter UAV. 2013.
Available at: http://www.sncorp.com/press_more_info.php?id=328

[13] Garratt, M. *et al.*: Systems for automated launch and recovery of an unmanned aerial vehicle from ships at sea. In: *22nd International UAV Systems Conference*. 2007.

[14] Gelb, A.: *Applied optimal estimation*. MIT Press, Cambridge, Massachusetts, 1974.

[15] Simon, D.: *Optimal state estimation: Kalman, H-infinity and nonlinear approaches*. John Wiley & Sons, Inc, 2006.

[16] Durrant-Whyte, H.: *Introduction to estimation and the Kalman filter*. 2.2. University of Sydney, 2001.

[17] Allerton, D.J. and Jia, H.: A review of multisensor fusion methodologies for aircraft navigation systems. *The Journal of Navigation*, vol. 58, no. 03, pp. 405–417, 2005.

[18] Hall, D.L. and Llinas, J.: An introduction to multisensor data fusion. *Proceedings of the IEEE*, vol. 85, no. 1, pp. 6–23, 1997.

[19] Hutchinson, C.E.: The Kalman filter applied to aerospace and electronic systems. *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 20, no. 4, pp. 500–504, 1984.

[20] Hamilton, J.: *Time series analysis*. Princeton University Press, 1994.

[21] Welch, G. and Bishop, G.: An introduction to the Kalman filter. Tech. Rep., University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 1995.

[22] Bewley, T.F.: *Advances in econometrics*, vol. 1 of *Advances in econometrics*. Cambridge University Press, 1994.

[23] Abolmaesumi, P. *et al.*: Image-guided control of a robot for medical ultrasound. *Robotics and Automation, IEEE Transactions on*, vol. 18, no. 1, pp. 11–23, February 2002.

[24] Vauhkonen, M., Karjalainen, P.A. and Kaipio, J.P.: A Kalman filter approach to track fast impedance changes in electrical impedance tomography. *Biomedical Engineering, IEEE Transactions on*, vol. 45, no. 4, pp. 486–493, 1998.

[25] Maybeck, P.S.: *Stochastic models, estimation, and control*. 1st edn. Academic Press, New York, 1979.

[26] Papoulis, A. and Pillai, S.U.: *Probability, random variables, and stochastic processes*. McGraw-Hill series in electrical engineering: Communications and signal processing. Tata McGraw-Hill, 2002.

[27] Aliev, F.A. and Ozbek, L.: Evaluation of convergence rate in the central limit theorem for the Kalman filter. *Automatic Control, IEEE Transactions on*, vol. 44, no. 10, pp. 1905–1909, October 1999.

[28] van Daalen, C.E., Jones, T. and Jacquet, C.D.: Advanced Estimation 813: Course notes, 2011.

[29] Wan, E.A. and van der Merwe, R.: The unscented Kalman filter for nonlinear estimation. In: *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. IEEE*, pp. 153–158. 2000.

[30] Julier, S.J. and Uhlmann, J.K.: New extension of the Kalman filter to nonlinear systems. 1997.

[31] Thrun, S. *et al.*: *Probabilistic robotics*, vol. 1. MIT Press Cambridge, MA, 2005.

[32] Arulampalam, M.S. *et al.*: A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *Signal Processing, IEEE Transactions on*, vol. 50, no. 2, pp. 174–188, 2002.

[33] Brown, R.G. and Hwang, P.Y.C.: *Introduction to random signals and applied Kalman filtering.* J. Wiley, New York, 1992.

[34] Tonne, K.K.: *Stability analysis of EKF-based attitude determination and control.* Ph.D. thesis, May 2007.

[35] Allan, D.W.: Statistics of Atomic Frequency Standards. *IEEE*, vol. 54, no. 2, pp. 221–230, 1966.

[36] Allan, D.W.: Millisecond pulsars rival best atomic clock stability. In: *41st Annual Frequency Control Symposium*, pp. 2–11. 1987.

[37] Kochakian, C.: Time-domain uncertainty charts (Green Charts): A tool for validating the design of IMU/instrument interfaces. In: *Proceedings of the AIAA Guidance and Control Conference.* 1980.

[38] IEEE standard specification format guide and test procedure for single-axis interferometric fiber optic gyros. 2008.

[39] Hou, H.: *Modeling inertial sensors errors using Allan variance.* Ph.D. thesis, University of Calgary, 2004.

[40] Niu, X. *et al.*: Using Allan variance to analyze the error characteristics of GNSS positioning. *GPS Solutions*, vol. 18, no. 129, pp. 231–242, 2014.

[41] Dever, C.W.: *Vehicle model-based filtering for spacecraft attitude determination.* Ph.D. thesis, MIT, Boston, 1998.

[42] du Plessis, J.: *An alternative gyroscope calibration methodology.* Ph.D. thesis, University of Johannesburg, 2013.

[43] Yuksel, Y.: Notes on stochastic errors of low cost MEMS inertial units, 2012.

[44] Woodman, O.J.: An introduction to inertial navigation. Tech. Rep. 696, University of Cambridge, Aug 2007.

[45] Yuksel, Y., El-Sheimy, N. and Noureldin, A.: Error modeling and characterization of environmental effects for low cost inertial MEMS units. *IEEE/ION Position, Location and Navigation Symposium*, pp. 598–612, 2010.

[46] Analog Devices: ADIS16407: Ten degrees of freedom inertial sensor. 2011.

[47] Analog Devices: ADIS16405: Triaxial inertial sensor with sagnetometer. 2009.

[48] Cemenska, J.: *Sensor modeling and Kalman filtering applied to satellite attitude determination*. Ph.D. thesis, University of California at Berkeley, Berkeley, 2004.

[49] Honeywell: 3-Axis magnetic sensor hybrid. 2011.

[50] Renaudin, V., Afzal, M.H. and Lachapelle, G.: New method for magnetometers based orientation estimation. *IEEE/ION Position, Location and Navigation Symposium*, , no. D, pp. 348–356, 2010.

[51] Tuthill, J. *et al.*: *Design and simulation of a nano-satellite attitude determination system*. Ph.D. thesis, Naval Postgraduate School, Dec 2009.

[52] Ang, W.T.: *Active tremor compensation in handheld instrument for microsurgery*. Ph.D. thesis, Carnegie Mellon University, 2004.

[53] Solimeno, A.: *Low-cost INS/GPS data fusion with extended Kalman filter for airborne applications*. Ph.D. thesis, Universidade Tecnica de Lisboa, Jul 2007.

[54] Coias, J.: *Attitude determination using multiple L1 GPS receivers*. Ph.D. thesis, Instituto Superior Tecnico, 2012.

[55] Novatel: APN-048: ALIGN family of heading solutions. 2011.

[56] Novatel: OEMV family firmware reference manual. 2010.

[57] Novatel: APN-029: GPS position accuracy measures. 2003.

[58] Bottasso, C.L., Leonello, D. and Milano, P.: Vision-augmented inertial navigation by sensor fusion for an autonomous rotorcraft vehicle. In: *AHS International Specialists' Meeting on Unmanned Rotorcraft*, pp. 1028–1033. 2008.

[59] Bosch, S., Lacroix, S. and Caballero, F.: Autonomous detection of safe landing areas for an UAV from monocular images. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5522–5527, October 2006.

[60] Hartley, R. and Zisserman, A.: *Multiple view geometry in computer vision*. 2nd edn. Cambridge University Press, New York, NY, USA, 2003.

[61] Visser, W.: *Automation and navigation of a terrestrial vehicle*. Master's thesis, Stellenbosch, South Africa, Mar 2012.

[62] Kalman, D.: A singularly valuable decomposition: The SVD of a matrix, Feb 2002.

[63] Brink, W.: *Stereo vision for simultaneous localisation and mapping.* Master's thesis, Stellenbosch, South Africa, 2012.

[64] SICK: Laser measurement sensors of the LMS1xx product family. 2012.

[65] James, F.: *Statistical methods in experimental physics.* 2nd edn. World Scientific, Switzerland, 2006.

[66] Markley, F.L. and Carpenter, J.R.: Linear covariance analysis and epoch state estimators. Tech. Rep. GSFC.JA.6379, NASA, Jan 2012.

[67] Hidalgo, J. *et al.*: Improving planetary rover attitude estimation via MEMS sensor characterization. *Sensors (Basel, Switzerland)*, vol. 12, no. 2, pp. 2219–35, 2012.

[68] Yadlin, R.: Attitude determination and bias estimation using Kalman filtering, 2011.

[69] Li, L.: *Separability of deformations and measurement noises of GPS time series with modified Kalman filter for landslide monitoring in real-time.* Ph.D. thesis, University of Bonn, China, May 2011.

[70] Kuhlmann, H.: Kalman filtering with coloured measurement noise for deformation analysis. In: *FIG Symposium on Deformation Measurements*, 11. Stuttgart, Germany, 2003.

[71] Letellier, C., Aguirre, L. and Maquet, J.: How the choice of the observable may influence the analysis of nonlinear dynamical systems. *Communications in Nonlinear Science and Numerical Simulation*, vol. 11, no. 5, pp. 555–576, 2006.

[72] Arrichiello, F. *et al.*: An observability metric for underwater vehicle localization using range measurements. *Sensors (Basel, Switzerland)*, vol. 13, no. 12, pp. 16191–215, 2013.

[73] Sabatini, A.M.: Kalman-filter-based orientation determination using inertial/magnetic sensors: observability analysis and performance evaluation. *Sensors (Basel, Switzerland)*, vol. 11, no. 10, pp. 9182–206, 2011.

[74] Merwe, R.V.D. and Wan, E.A.: Sigma-point Kalman filters for integrated navigation. Tech. Rep., Oregon Health and Science University, Jun 2004.

[75] Chen, Z.: Bayesian filtering: From kalman filters to particle filters, and beyond. *Statistics*, vol. 182, no. 1, pp. 1–69, 2003.

[76] Arras, K.: Introduction to mobile robotics: Error propagation, 2009.
Available at: `http://ais.informatik.uni-freiburg.de/teaching/ss09/robotics/slides/error_propagation.pdf`

[77] Radul, A.: Introduction to automatic differentiation. 2013.
Available at: `http://alexey.radul.name/ideas/2013/introduction-to-automatic-differentiation/`

[78] Neidinger, R.D.: Introduction to automatic differentiation and MATLAB object-oriented programming. *SIAM Review*, vol. 52, no. 3, pp. 545–563, 2010.

[79] Walter, S.F. and Lehmann, L.: Algorithmic differentiation in python with algopy. *Journal of Computational Science*, vol. 4, no. 5, pp. 334 – 344, 2013. ISSN 1877-7503. Available at: `http://www.sciencedirect.com/science/article/pii/S1877750311001013`

Appendices

Appendix A

Sensor Models and Data Sets

A.1 Simulink Sensor Models

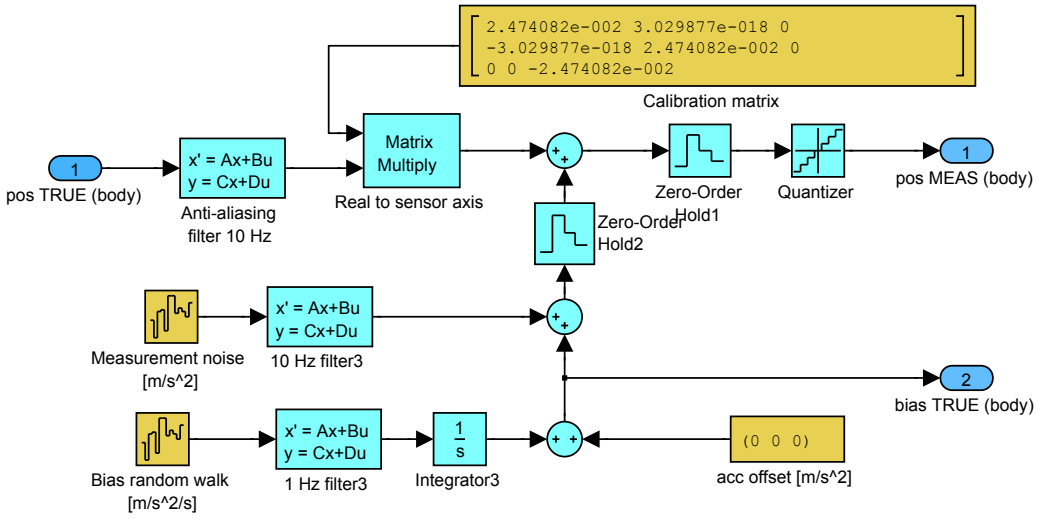


Figure A.1: Simulink accelerometer model.

Table A.1 summerises the Allan variance parameters for the actual and simulated GPS position and velocity data. The Allan variance parameters used in simulation are indicated within parentheses.

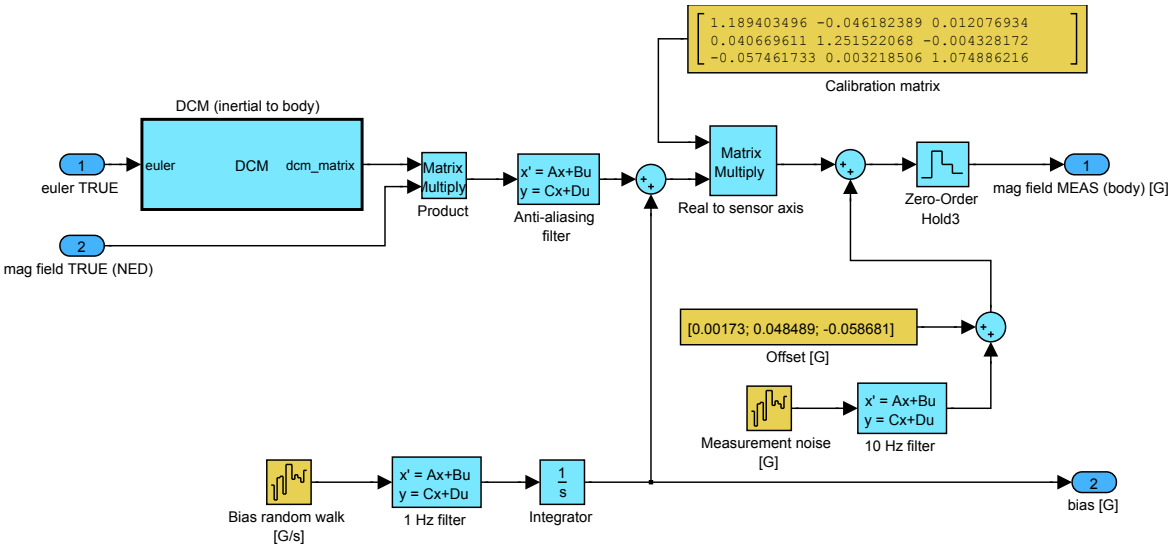


Figure A.2: Simulink magnetometer model.

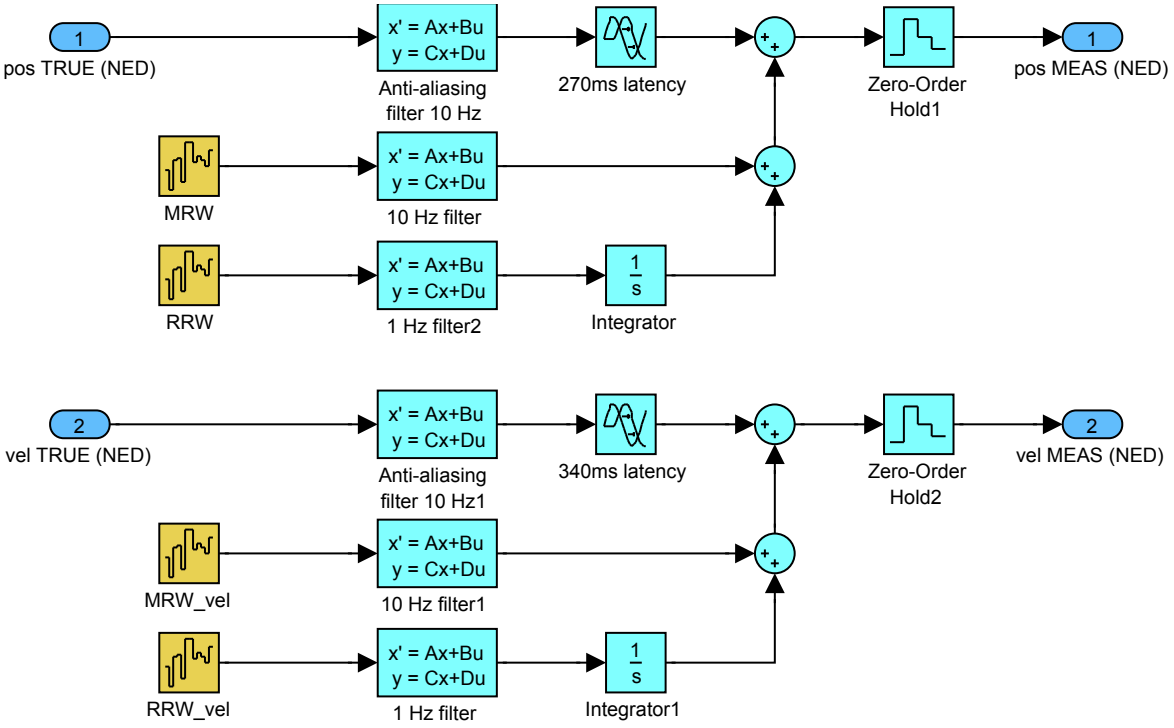


Figure A.3: Simulink single-point GPS model.

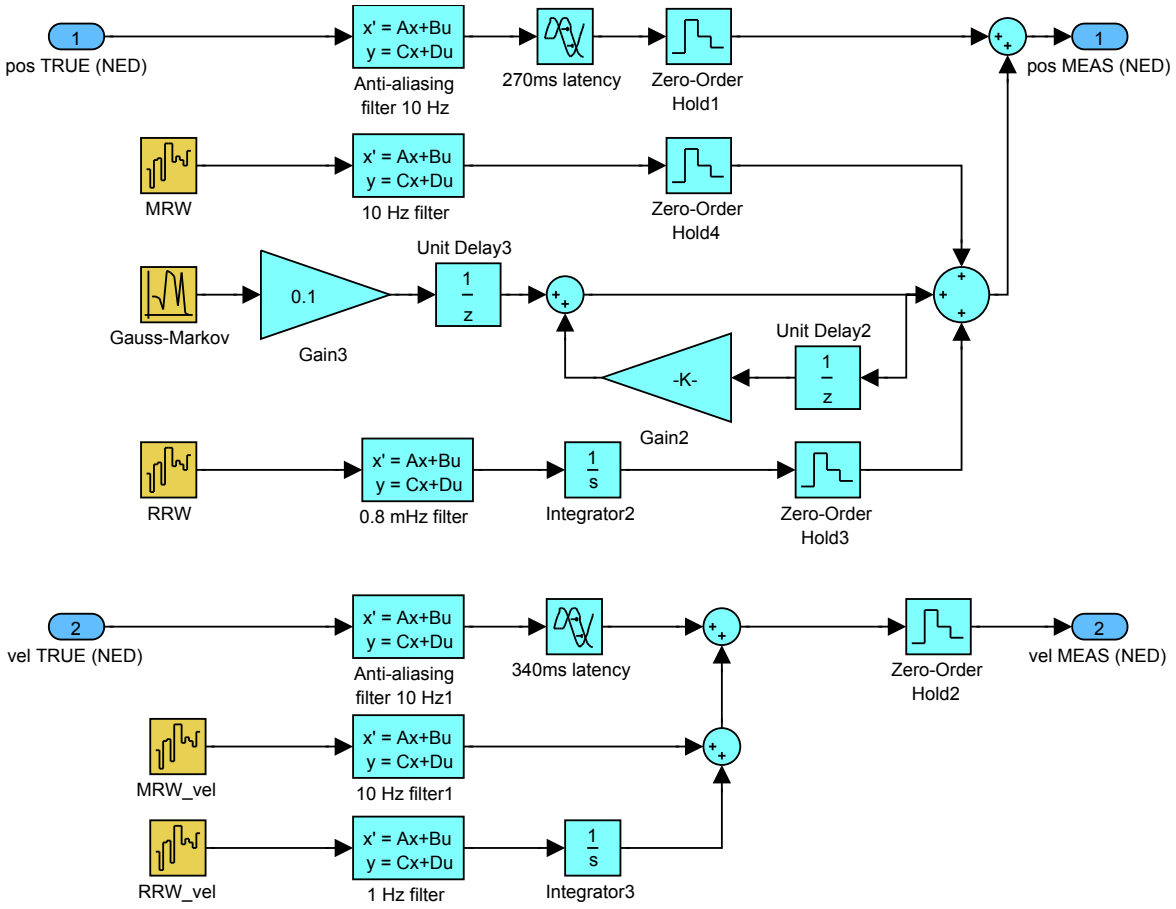


Figure A.4: Simulink DGPS model.

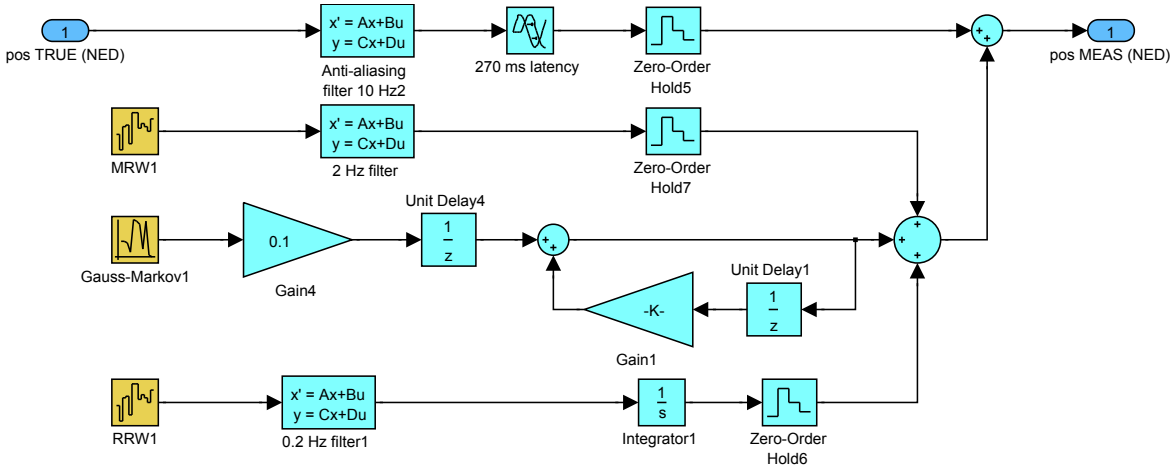


Figure A.5: Simulink Novatel Align model.

Table A.1: GPS Allan variance parameters

GPS mode	**A.V. Parameter**	**N**	**E**	**D**
Single-point Position	MRW $[m/\sqrt{Hz}]$	- (1×10^{-3})	- (1×10^{-3})	- (6×10^{-3})
	RRW $[m/s/\sqrt{Hz}]$	1.20×10^{-2} (1.20×10^{-2})	1.27×10^{-2} (1.20×10^{-2})	5.01×10^{-2} (5.50×10^{-2})
Single-point Velocity	MRW $[m/s/\sqrt{Hz}]$	5.66×10^{-3} (5.40×10^{-3})	5.11×10^{-3} (5.40×10^{-3})	1.57×10^{-2} (1.57×10^{-2})
Differential Position	MRW $[m/\sqrt{Hz}]$	2×10^{-4} (2×10^{-4})	2×10^{-4} (2×10^{-4})	8×10^{-4} (8×10^{-4})
	RRW $[m/s/\sqrt{Hz}]$	2×10^{-4} (2×10^{-4})	2×10^{-4} (2×10^{-4})	6×10^{-4} (6×10^{-4})
	G-M: τ_{gm} $[s]$	3 (3)	3 (3)	4 (4)
	G-M: σ_{gm} $[m]$	9×10^{-3} (9×10^{-3})	9×10^{-3} (9×10^{-3})	1.4×10^{-2} (1.4×10^{-2})
Differential Velocity	MRW $[m/s/\sqrt{Hz}]$	5.99×10^{-3} (6.0×10^{-3})	3.5×10^{-3} (3.5×10^{-3})	7.65×10^{-3} (7.6×10^{-3})

A.2 Sensor Datasets

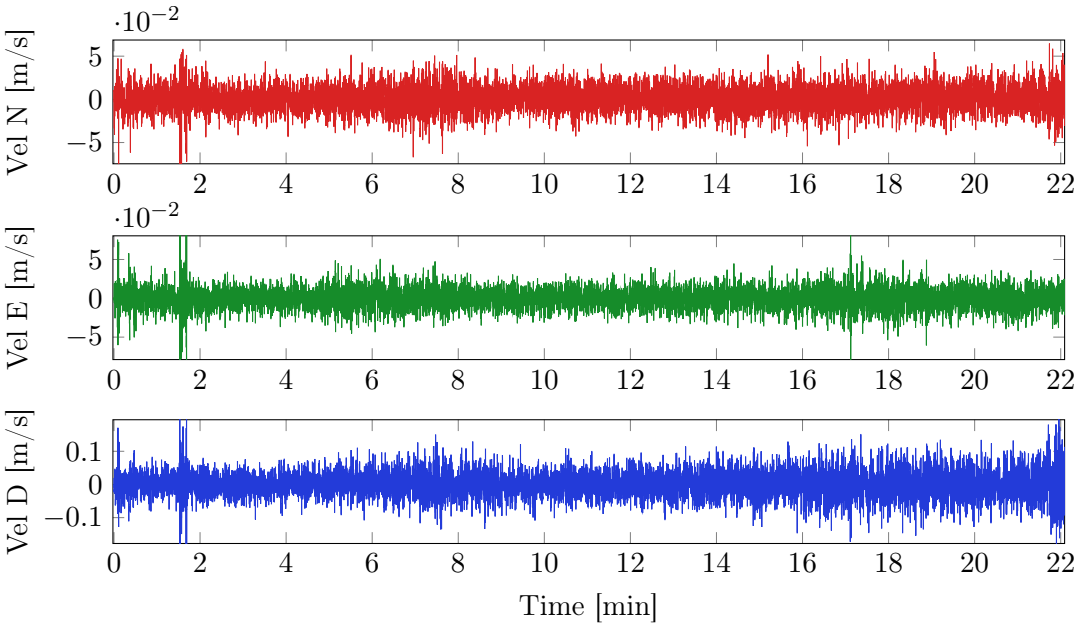


Figure A.6: Single-point GPS velocity.

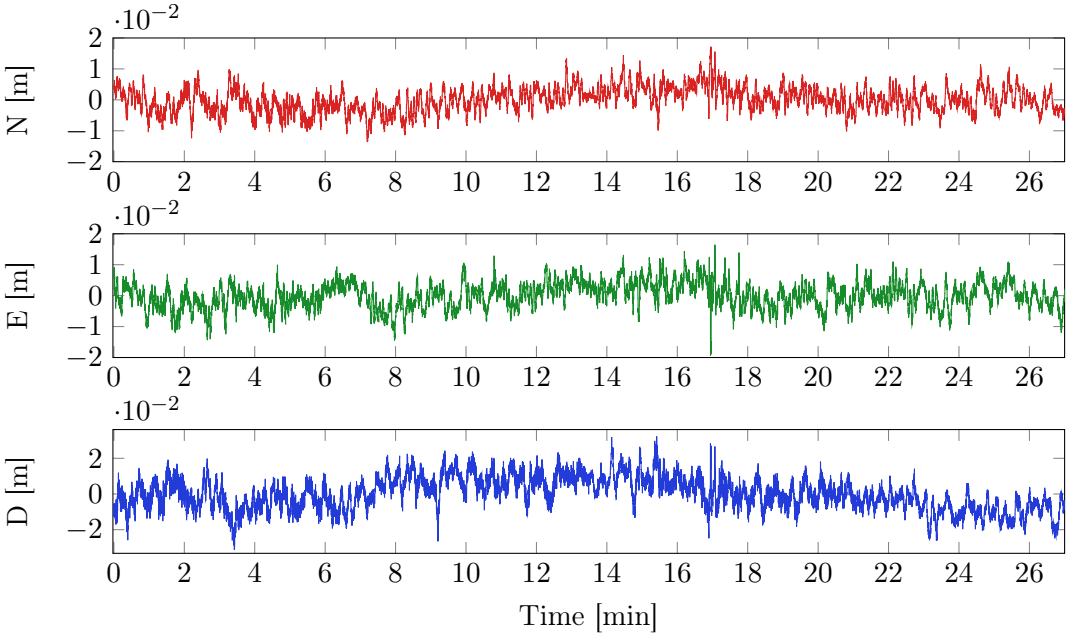


Figure A.7: Differential GPS position.

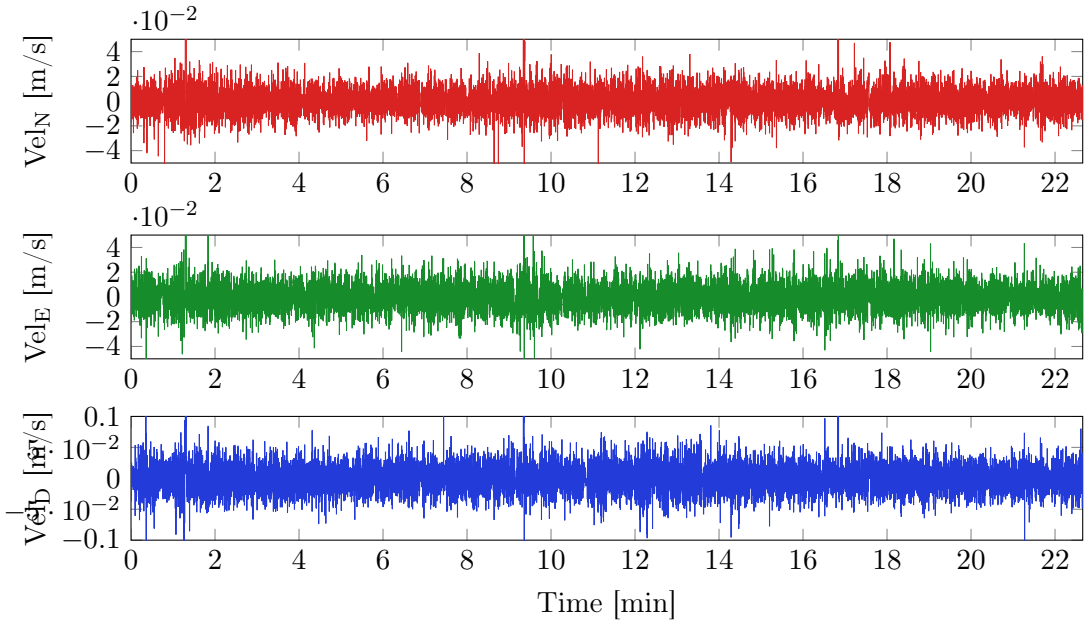


Figure A.8: Differential GPS velocity.

Appendix B

Surface Fitting of Relative Sensor Noise

Chapter 4 describes the methods used to model the vision and laser sensors. The coefficients of the polynomial functions are given here.

B.1 Monocular Vision

Table B.1 lists the coefficients of the polynomials used to describe the measurement noise for each state. The RMS error of the fitted surface for each state is indicated in the right-most column. The polynomial used to fit the surface is of the form

$$f(h, r) = a_{00} + a_{10}\, h + a_{01}\, r + a_{02}\, h^2 + a_{11}\, h\, r + a_{20}\, r^2. \tag{B.1.1}$$

Table B.1: Monocular vision RMS error surface fit coefficients.

State	$\mathbf{a_{00}}$	$\mathbf{a_{10}}$	$\mathbf{a_{01}}$	$\mathbf{a_{02}}$	$\mathbf{a_{11}}$	$\mathbf{a_{20}}$	**RMSE**
North [mm]	2.307669	-0.266904	-0.905416	0.123799	-0.148405	1.699407	2.3947
East [mm]	2.402493	-0.630289	-0.457807	0.137337	-0.141240	1.680426	2.4658
Down [mm]	15.258951	-10.521082	7.556363	1.794952	-1.448190	2.573945	5.7257
Roll [deg]	0.267321	-0.149514	0.455287	0.035594	-0.077118	0.061029	0.1592
Pitch [deg]	0.171787	-0.098261	0.317647	0.022756	-0.053950	0.044781	0.1200
Yaw [deg]	0.064965	-0.022302	0.096711	0.003724	-0.014616	0.029078	0.0527

B.2 Stereo Vision

The polynomial used to fit the simulated stereo vision noise is of the same form as (B.1.1). The coefficients are given in Table B.2.

Table B.2: Stereo vision RMS error surface fit coefficients.

State	a_{00}	a_{10}	a_{01}	a_{02}	a_{11}	a_{20}	RMSE
North [mm]	3.622393	-0.859930	-3.437343	0.067293	0.714963	0.497540	0.4101
East [mm]	3.574440	-0.884506	-3.263114	0.067676	0.725736	0.464473	0.4159
Down [mm]	24.191805	-10.967703	-0.106987	1.629752	0.006842	0.663371	1.2783
Roll [deg]	0.041177	-0.006898	0.002917	0.016976	-0.001234	0.001772	0.0167
Pitch [deg]	-0.006884	0.003950	0.003490	0.011380	-0.000745	0.001449	0.0123
Yaw [deg]	0.022538	0.013649	-0.039594	0.000089	0.004253	0.011412	0.0073

B.3 Laser Rangefinders

A polynomial of the following form is used to fit the laser measurement noise:

$$f(d, \theta_{deck}) = a_{00} + a_{10}\, d + a_{01}\, \theta_{deck} + a_{02}\, d^2 + a_{11}\, d\, \theta_{deck} + a_{20}\, \theta_{deck}^2. \tag{B.3.1}$$

Least-squares is prone to fitting the densely populated regions of data more accurately than the sparsely populated regions. As such, the region around $d = 0$, $\theta_{deck} = 1$ tends to exhibit significant fitting error. Since this is the most important region of the surface, a least-squares weighting function was introduced. This function is given by

$$w(d, \theta_{deck}) = \frac{100}{\frac{d}{d_{max}} + \frac{\theta_{deck}}{\theta_{deck\,max}}}. \tag{B.3.2}$$

Table B.3 lists the coefficients of the polynomials used to describe the measurement noise for each state. The RMS error of the fitted surface for each state is indicated.

Table B.3: Laser sensor RMS error surface fit coefficients.

State	a_{00}	a_{10}	a_{01}	a_{02}	a_{11}	a_{20}	RMSE
$p_{intersect_3}$ [m]	0.084081	-0.041482	-0.216248	0.002507	0.087492	-0.098216	0.0902
Roll [rad]	0.000089	0.000119	-0.000605	-0.000000	0.000019	0.000748	0.0011
Pitch [rad]	0.000102	0.000138	-0.000487	-0.000002	0.000005	0.000675	0.0010

Since the single and multiple laser sensors exhibit almost identical noise on the $p_{intersect_3}$ measurements, the same surface is used to approximate the measurement noise.

Appendix C

Extracting Attitude from a Normal Vector

Provided yaw is already known, roll and pitch measurements may be determined from a normal vector.

A ship deck can be modeled as a plane. Within the deck reference frame the normal of the plane will simply be the vector

$$\mathbf{n}_{deck} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T.$$

The normal expressed within the relative reference frame is

$$\begin{aligned} \mathbf{n}_{rel} &= \mathbf{DCM}_{\boldsymbol{\theta}_{rel}}^T \mathbf{n}_{deck} \\ &= \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}^T \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \end{aligned}$$

where

$$\boldsymbol{\theta}_{rel} = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T.$$

The yaw can be removed from $\mathbf{n}_{rel}$ to produce the vector $\mathbf{n}_{ny}$:

$$\begin{aligned}
\mathbf{n}_{ny} &= \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{n}_{rel} \\
&= \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} \cos\theta & \sin\theta\sin\phi & \sin\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} \sin\theta\cos\phi \\ -\sin\phi \\ \cos\theta\cos\phi \end{bmatrix}
\end{aligned}$$

.

The roll and pitch angles can then be determined from $\mathbf{n}_{ny}$ as follows:

$$\phi = -\arcsin(n_{ny_2})$$

$$\theta = \arcsin\left(\frac{n_{ny_1}}{\cos\phi}\right).$$

Appendix D

Noise Matrices

D.1 Definition of the Jacobian

Let $\mathbf{f}(x_1, x_2, \ldots, x_n)$ be a nonlinear vector function of m elements.

The Jacobian matrix $\mathbf{J}$ of $\mathbf{f}$ is a matrix of first-order partial derivatives:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}. \tag{D.1.1}$$

D.2 EKF Process Noise

D.2.1 EKF without Bias Estimation

The process model of the three-state EKF is given by:

$$\begin{aligned} \dot{\hat{\phi}} &= p + q \sin\hat{\phi} \tan\hat{\theta} + r \cos\hat{\phi} \tan\hat{\theta} \\ \dot{\hat{\theta}} &= q \cos\hat{\phi} - r \sin\hat{\phi} \\ \dot{\hat{\psi}} &= q \sin\hat{\phi} \sec\hat{\theta} + r \cos\hat{\phi} \sec\hat{\theta}. \end{aligned} \tag{D.2.1}$$

There is uncertainty in the Euler estimates and noise in the gyro measurements. A diagonal covariance matrix matrix is used to represent these sources of noise:

$$\mathbf{N} = \text{diag}(\sigma^2_{\hat{\phi}}, \sigma^2_{\hat{\theta}}, \sigma^2_{\hat{\psi}}, \sigma^2_p, \sigma^2_q, \sigma^2_r). \tag{D.2.2}$$

Process noise is in reality a nonlinear function of these noise sources. However, to preserve the Gaussian distribution, the nonlinear functions are linearised. The process noise covariance matrix is therefore calculated using first-order error propagation [76]:

$$\mathbf{Q}_k = \mathbf{J}\ \mathbf{N}\ \mathbf{J}^T. \tag{D.2.3}$$

The Jacobian is given by

$$\mathbf{J} = \begin{bmatrix} \frac{\partial\dot{\phi}}{\partial\hat{\phi}} & \frac{\partial\dot{\phi}}{\partial\hat{\theta}} & \frac{\partial\dot{\phi}}{\partial\hat{\psi}} & \frac{\partial\dot{\phi}}{\partial p} & \frac{\partial\dot{\phi}}{\partial q} & \frac{\partial\dot{\phi}}{\partial r} \\ \frac{\partial\dot{\theta}}{\partial\hat{\phi}} & \frac{\partial\dot{\theta}}{\partial\hat{\theta}} & \frac{\partial\dot{\theta}}{\partial\hat{\psi}} & \frac{\partial\dot{\theta}}{\partial p} & \frac{\partial\dot{\theta}}{\partial q} & \frac{\partial\dot{\theta}}{\partial r} \\ \frac{\partial\dot{\psi}}{\partial\hat{\phi}} & \frac{\partial\dot{\psi}}{\partial\hat{\theta}} & \frac{\partial\dot{\psi}}{\partial\hat{\psi}} & \frac{\partial\dot{\psi}}{\partial p} & \frac{\partial\dot{\psi}}{\partial q} & \frac{\partial\dot{\psi}}{\partial r} \end{bmatrix}. \tag{D.2.4}$$

The partial derivatives are as follows:

$$\begin{aligned}
\frac{\partial\dot{\phi}}{\partial\hat{\phi}} &= q\cos\hat{\phi}\tan\hat{\theta} - r\sin\hat{\phi}\tan\hat{\theta} & \frac{\partial\dot{\theta}}{\partial p} &= 0 \\
\frac{\partial\dot{\phi}}{\partial\hat{\theta}} &= q\sin\hat{\phi}\sec^2\hat{\theta} - r\cos\hat{\phi}\sec^2\hat{\theta} & \frac{\partial\dot{\theta}}{\partial q} &= \cos\hat{\phi} \\
\frac{\partial\dot{\phi}}{\partial\hat{\psi}} &= 0 & \frac{\partial\dot{\theta}}{\partial r} &= -\sin\hat{\phi} \\
\frac{\partial\dot{\phi}}{\partial p} &= 1 & \frac{\partial\dot{\psi}}{\partial\hat{\phi}} &= q\cos\hat{\phi}\sec\hat{\theta} - r\sin\hat{\phi}\sec\hat{\theta} \\
\frac{\partial\dot{\phi}}{\partial q} &= \sin\hat{\phi}\tan\hat{\theta} & \frac{\partial\dot{\psi}}{\partial\hat{\theta}} &= q\sin\hat{\phi}\sec\hat{\theta}\tan\hat{\theta} + r\cos\hat{\phi}\sec\hat{\theta}\tan\hat{\theta} \\
\frac{\partial\dot{\phi}}{\partial r} &= \cos\hat{\phi}\tan\hat{\theta} & \frac{\partial\dot{\psi}}{\partial\hat{\psi}} &= 0 \\
\frac{\partial\dot{\theta}}{\partial\hat{\phi}} &= -q\sin\hat{\phi} - r\cos\hat{\phi} & \frac{\partial\dot{\psi}}{\partial p} &= 0 \\
\frac{\partial\dot{\theta}}{\partial\hat{\theta}} &= 0 & \frac{\partial\dot{\psi}}{\partial q} &= \sin\hat{\phi}\sec\hat{\theta} \\
\frac{\partial\dot{\theta}}{\partial\hat{\psi}} &= 0 & \frac{\partial\dot{\psi}}{\partial r} &= \cos\hat{\phi}\sec\hat{\theta}
\end{aligned}$$

D.2.2 EKF with Bias Estimation

The process model of the six-state EKF is given by equation (6.3.1). This can be written out as follows:

$$\begin{aligned}
\dot{\phi} &= p - b_p + (q - b_q)\sin\hat{\phi}\tan\hat{\theta} + (r - b_r)\cos\hat{\phi}\tan\hat{\theta} \\
\dot{\theta} &= (q - b_q)\cos\hat{\phi} - (r - b_r)\sin\hat{\phi} \\
\dot{\psi} &= (q - b_q)\sin\hat{\phi}\sec\hat{\theta} + (r - b_r)\cos\hat{\phi}\sec\hat{\theta}.
\end{aligned} \tag{D.2.5}$$

The discrete process noise matrix $\mathbf{Q}_k$ is determined in a similar manner to that of the three-state EKF. The covariance matrix of the noise sources is given by

$$\mathbf{N} = \text{diag}(\sigma^2_{\hat{\phi}}, \sigma^2_{\hat{\theta}}, \sigma^2_{\hat{\psi}}, \sigma^2_p, \sigma^2_q, \sigma^2_r, \sigma^2_{\hat{b}_p}, \sigma^2_{\hat{b}_q}, \sigma^2_{\hat{b}_r}). \tag{D.2.6}$$

Linearising the process model with respect to the noise sources produces the Jacobian:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial\dot{\phi}}{\partial\hat{\phi}} & \frac{\partial\dot{\phi}}{\partial\hat{\theta}} & \frac{\partial\dot{\phi}}{\partial\hat{\psi}} & \frac{\partial\dot{\phi}}{\partial p} & \frac{\partial\dot{\phi}}{\partial q} & \frac{\partial\dot{\phi}}{\partial r} & \frac{\partial\dot{\phi}}{\partial\hat{b}_p} & \frac{\partial\dot{\phi}}{\partial\hat{b}_q} & \frac{\partial\dot{\phi}}{\partial\hat{b}_r} \\ \frac{\partial\dot{\theta}}{\partial\hat{\phi}} & \frac{\partial\dot{\theta}}{\partial\hat{\theta}} & \frac{\partial\dot{\theta}}{\partial\hat{\psi}} & \frac{\partial\dot{\theta}}{\partial p} & \frac{\partial\dot{\theta}}{\partial q} & \frac{\partial\dot{\theta}}{\partial r} & \frac{\partial\dot{\theta}}{\partial\hat{b}_p} & \frac{\partial\dot{\theta}}{\partial\hat{b}_q} & \frac{\partial\dot{\theta}}{\partial\hat{b}_r} \\ \frac{\partial\dot{\psi}}{\partial\hat{\phi}} & \frac{\partial\dot{\psi}}{\partial\hat{\theta}} & \frac{\partial\dot{\psi}}{\partial\hat{\psi}} & \frac{\partial\dot{\psi}}{\partial p} & \frac{\partial\dot{\psi}}{\partial q} & \frac{\partial\dot{\psi}}{\partial r} & \frac{\partial\dot{\psi}}{\partial\hat{b}_p} & \frac{\partial\dot{\psi}}{\partial\hat{b}_q} & \frac{\partial\dot{\psi}}{\partial\hat{b}_r} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{D.2.7}$$

$$\frac{\partial\dot{\phi}}{\partial\hat{\phi}} = (q - b_q)\cos\hat{\phi}\tan\hat{\theta} - (r - b_r)\sin\hat{\phi}\tan\hat{\theta}$$

$$\frac{\partial\dot{\phi}}{\partial\hat{\theta}} = (q - b_q)\sin\hat{\phi}\sec^2\hat{\theta} - (r - b_r)\cos\hat{\phi}\sec^2\hat{\theta}$$

$$\frac{\partial\dot{\phi}}{\partial\hat{\psi}} = 0$$

$$\frac{\partial\dot{\phi}}{\partial p} = 1$$

$$\frac{\partial\dot{\phi}}{\partial q} = \sin\hat{\phi}\tan\hat{\theta}$$

$$\frac{\partial\dot{\phi}}{\partial r} = \cos\hat{\phi}\tan\hat{\theta}$$

$$\frac{\partial\dot{\phi}}{\partial\hat{b}_p} = -1$$

$$\frac{\partial\dot{\phi}}{\partial\hat{b}_q} = -\sin\hat{\phi}\tan\hat{\theta}$$

$$\frac{\partial\dot{\phi}}{\partial\hat{b}_r} = -\cos\hat{\phi}\tan\hat{\theta}$$

$$\frac{\partial\dot{\theta}}{\partial\hat{\phi}} = -(q - b_q)\sin\hat{\phi} - (r - b_r)\cos\hat{\phi}$$

$$\frac{\partial\dot{\theta}}{\partial\hat{\theta}} = 0$$

$$\frac{\partial\dot{\theta}}{\partial\hat{\psi}} = 0$$

$$\frac{\partial\dot{\theta}}{\partial p} = 0$$

$$\frac{\partial\dot{\theta}}{\partial q} = \cos\hat{\phi}$$

$$\frac{\partial\dot{\theta}}{\partial r} = -\sin\hat{\phi}$$

$$\frac{\partial\dot{\theta}}{\partial\hat{b}_p} = 0$$

$$\frac{\partial\dot{\theta}}{\partial\hat{b}_q} = -\cos\hat{\phi}$$

$$\frac{\partial\dot{\theta}}{\partial\hat{b}_r} = \sin\hat{\phi}$$

$$\frac{\partial\dot{\psi}}{\partial\hat{\phi}} = (q - b_q)\cos\hat{\phi}\sec\hat{\theta} - (r - b_r)\sin\hat{\phi}\sec\hat{\theta}$$

$$\frac{\partial\dot{\psi}}{\partial\hat{\theta}} = (q - b_q)\sin\hat{\phi}\sec\hat{\theta}\tan\hat{\theta} + (r - b_r)\cos\hat{\phi}\sec\hat{\theta}\tan\hat{\theta}$$

$$\frac{\partial\dot{\psi}}{\partial\hat{\psi}} = 0$$

$$\frac{\partial\dot{\psi}}{\partial p} = 0$$

$$\frac{\partial\dot{\psi}}{\partial q} = \sin\hat{\phi}\sec\hat{\theta}$$

$$\frac{\partial\dot{\psi}}{\partial r} = \cos\hat{\phi}\sec\hat{\theta}$$

$$\frac{\partial\dot{\psi}}{\partial\hat{b}_p} = 0$$

$$\frac{\partial\dot{\psi}}{\partial\hat{b}_q} = -\sin\hat{\phi}\sec\hat{\theta}$$

$$\frac{\partial\dot{\psi}}{\partial\hat{b}_r} = -\cos\hat{\phi}\sec\hat{\theta}$$

D.3 EKF Measurement Noise

D.3.1 Tilt-Heading Measurement Noise

The tilt-heading algorithm is nonlinear. As such, it needs to be linearised in order to determine the measurement noise covariance matrix:

$$\mathbf{R} = \mathbf{J}\ \mathbf{N}\ \mathbf{J}^T. \tag{D.3.1}$$

Previous Euler estimates, GPS velocity, accelerometer and magnetometer readings all contribute to the noise in the resultant attitude measurements. The diagonal covariance matrix corresponding to these noise sources is given by:

$$\mathbf{N} = \text{diag}(\sigma^2_{\hat{\phi}}, \sigma^2_{\hat{\theta}}, \sigma^2_{\hat{\psi}}, \sigma^2_{\beta_1}, \sigma^2_{\beta_2}, \sigma^2_{\beta_3}, \sigma^2_{\tilde{\alpha}_x}, \sigma^2_{\tilde{\alpha}_y}, \sigma^2_{\tilde{\alpha}_z}, \sigma^2_{\tilde{M}_x}, \sigma^2_{\tilde{M}_y}, \sigma^2_{\tilde{M}_z}). \tag{D.3.2}$$

β is the acceleration calculated from the current GPS velocity and the estimated velocity from the linear KF immediately after the previous GPS measurement:

$$\boldsymbol{\beta} = \frac{\tilde{\mathbf{V}}_k - \hat{\mathbf{V}}_{k-1}}{\triangle T_{GPS}}. \tag{D.3.3}$$

The tilt-heading algorithm, shown in Chapter 6.2.2, can be summarised as follows:

$$\begin{aligned}
g_x &= \beta_1 \cos\hat{\psi}\cos\hat{\theta} + \beta_2 \sin\hat{\psi}\cos\hat{\theta} - \beta_3 \sin\hat{\theta} - \tilde{\alpha}_x \\
g_y &= \beta_1 \cos\hat{\psi}\sin\hat{\theta}\sin\hat{\phi} - \beta_2 \sin\hat{\psi}\cos\hat{\phi} + \beta_3 \sin\hat{\psi}\sin\hat{\theta}\sin\hat{\phi} - \tilde{\alpha}_y \\
g_z &= \beta_1(\cos\hat{\psi}\sin\hat{\theta}\cos\hat{\phi} + \sin\hat{\psi}\sin\hat{\phi}) + \beta_2 \sin\hat{\psi}\sin\hat{\theta}\cos\hat{\phi} + \beta_3 \cos\hat{\theta}\cos\hat{\phi} - \tilde{\alpha}_z
\end{aligned}$$

$$\begin{aligned}
\tilde{\phi} &= \arctan\left(\frac{-g_x}{\sqrt{{g_y}^2 + {g_z}^2}}\right) \\
\tilde{\theta} &= \arctan\left(\frac{g_y}{g_z}\right) \\
\tilde{\psi} &= \arctan\left(\frac{\tilde{M}_x\cos\tilde{\theta} + \tilde{M}_y\sin\tilde{\phi}\sin\tilde{\theta} + \tilde{M}_z\sin\tilde{\theta}\cos\tilde{\phi}}{\tilde{M}_y\cos\tilde{\phi} - \tilde{M}_z\sin\tilde{\phi}}\right).
\end{aligned}$$

Calculating the Jacobian of the tilt-heading algorithm is not simple. Matlab Symbolic Toolbox or Python's sympy package may be used to calculate it. However, naïvely writing the tilt-heading algorithm in symbolic algebra form and computing the Jacobian in a single step is highly inefficient. Code generated in this manner would prove near impossible to debug. It would also be very inefficient in terms of computation. By making careful use of the chain rule, several of the partial derivatives can be reused.

As such, the Jacobian was calculated by hand. Matlab Symbolic Toolbox was used to check each term. The C++ implementation of the calculation is given here. OpenCV matrices are used to perform linear algebra operations.

```
1  Mat EKF::tiltHeadingNoise (double Vel_current[3], double ...
       accelMeas[3], double magMeas[3], double eulerMeas[3], double ...
       noise_vel[3], double noise_accel, double noise_mag)
2  {
3      double phi = X.at<double>(0), theta = X.at<double>(1), psi = ...
           X.at<double>(2),
4              Mx = magMeas[0], My = magMeas[1], Mz = magMeas[2],
5              f1 = eulerMeas[0], f2 = eulerMeas[1];
6
7      double beta[3] =
8          {(Vel_current[0]-Vel_prev.at<double>(0))/TimeSinceLastUpdate,
9          (Vel_current[1]-Vel_prev.at<double>(1))/TimeSinceLastUpdate,
10         (Vel_current[2]-Vel_prev.at<double>(2))/TimeSinceLastUpdate};
11
12     Mat N =  diag ((Mat_<double>(12,1) <<
13         P.at<double>(0,0), P.at<double>(1,1), P.at<double>(2,2),
14         2*noise_vel[0]/TimeSinceLastUpdate,
15         2*noise_vel[1]/TimeSinceLastUpdate,
16         2*noise_vel[2]/TimeSinceLastUpdate,
17         noise_accel, noise_accel, noise_accel,
18         noise_mag, noise_mag, noise_mag), 12);
19
20     double gx = cos(psi)*cos(theta)*beta[0] + ...
           sin(psi)*cos(theta)*beta[1]
21                 - sin(theta)*beta[2] - accelMeas[0],
22             gy = ...
                   (cos(psi)*sin(theta)*sin(phi)-sin(psi)*cos(phi))*beta[0]
23                 + (sin(psi)*sin(theta)*sin(phi)
24                     +cos(psi)*cos(phi))*beta[1]
25                 + cos(theta)*sin(phi)*beta[2] - accelMeas[1],
26             gz = ...
                   (cos(psi)*sin(theta)*cos(phi)+sin(psi)*sin(phi))*beta[0]
27                 + (sin(psi)*sin(theta)*cos(phi)
28                     -cos(psi)*sin(phi))*beta[1]
29                 + cos(theta)*cos(phi)*beta[2] - accelMeas[2],
30
31             s = pow(gy,2),
32             t = pow(gz,2),
33             v = s+t,
34             w = sqrt(v),
35             Mx2 = cos(f2)*Mx+sin(f1)*sin(f2)*My+sin(f2)*cos(f1)*Mz,
36             My2 = cos(f1)*My-sin(f1)*Mz,
37             u1 = gy/gz,
38             u2 = -gx/w,
39             u3 = Mx2/My2,
40             d_f1_u1 = 1/(1+pow(u1,2)),
41             d_f2_u2 = 1/(1+pow(u2,2)),
```

```
42              d_f3_u3 = 1/(1+pow(u3,2)),
43              d_u1_gy = 1/gz,
44              d_u1_gz = -gy/pow(gz,2),
45              d_u2_gx = -1/w,
46              d_u2_w = gx/pow(w,2),
47              d_u3_Mx2 = 1/My2,
48              d_u3_My2 = -Mx2/pow(My2,2),
49              d_Mx2_f1 = cos(f1)*sin(f2)*My-sin(f2)*sin(f1)*Mz,
50              d_Mx2_f2 = ...
                    -sin(f2)*Mx+sin(f1)*cos(f2)*My+cos(f2)*cos(f1)*Mz,
51              d_My2_f1 = -sin(f1)*My-cos(f1)*Mz,
52              d_Mx2_Mx = cos(f2),
53              d_Mx2_My = sin(f1)*sin(f2),
54              d_Mx2_Mz = sin(f2)*cos(f1),
55              d_My2_My = cos(f1),
56              d_My2_Mz = -sin(f1),
57              d_w_v = 0.5/sqrt(v),
58              d_v_s = 1,
59              d_v_t = 1,
60              d_s_gy = 2*gy,
61              d_t_gz = 2*gz,
62
63              d_gx_x1 = 0,
64              d_gx_x2 = -cos(psi)*sin(theta)*beta[0]
65                      -sin(psi)*sin(theta)*beta[1]-cos(theta)*beta[2],
66              d_gx_x3 = -sin(psi)*cos(theta)*beta[0]
67                      +cos(psi)*cos(theta)*beta[1],
68              d_gx_x4 = cos(psi)*cos(theta),
69              d_gx_x5 = sin(psi)*cos(theta),
70              d_gx_x6 = -sin(theta),
71              d_gx_x7 = -1,
72              d_gx_x8 = 0,
73              d_gx_x9 = 0,
74              d_gx_x10 = 0,
75              d_gx_x11 = 0,
76              d_gx_x12 = 0,
77              d_gy_x1 = (cos(psi)*sin(theta)*cos(phi)
78                  +sin(psi)*sin(phi))*beta[0]
79                  +(sin(psi)*sin(theta)*cos(phi)
80                  -cos(psi)*sin(phi))*beta[1]
81                  +cos(theta)*cos(phi)*beta[2],
82              d_gy_x2 = (cos(psi)*cos(theta)*sin(phi))*beta[0]
83                  +sin(psi)*cos(theta)*sin(phi)*beta[1]
84                  -sin(theta)*sin(phi)*beta[2],
85              d_gy_x3 = (-sin(psi)*sin(theta)*sin(phi)
86                  -cos(psi)*cos(phi))*beta[0]
87                  +(cos(psi)*sin(theta)*sin(phi)
```

```
88                  -sin(psi)*cos(phi))*beta[1],
89              d_gy_x4 = cos(psi)*sin(theta)*sin(phi)
90                  -sin(psi)*cos(phi),
91              d_gy_x5 = sin(psi)*sin(theta)*sin(phi)
92                  +cos(psi)*cos(phi),
93              d_gy_x6 = cos(theta)*sin(phi),
94              d_gy_x7 = 0,
95              d_gy_x8 = -1,
96              d_gy_x9 = 0,
97              d_gy_x10 = 0,
98              d_gy_x11 = 0,
99              d_gy_x12 = 0,
100             d_gz_x1 = (-cos(psi)*sin(theta)*sin(phi)
101                 +sin(psi)*cos(phi))*beta[0]
102                 +(-sin(psi)*sin(theta)*sin(phi)
103                 -cos(psi)*cos(phi))*beta[1]
104                 -cos(theta)*sin(phi)*beta[2],
105             d_gz_x2 = cos(psi)*cos(theta)*cos(phi)*beta[0]
106                 +sin(psi)*cos(theta)*cos(phi)*beta[1]
107                 -sin(theta)*cos(phi)*beta[2],
108             d_gz_x3 = (-sin(psi)*sin(theta)*cos(phi)
109                 +cos(psi)*sin(phi))*beta[0]
110                 +(cos(psi)*sin(theta)*cos(phi)
111                 +sin(psi)*sin(phi))*beta[1],
112             d_gz_x4 = cos(psi)*sin(theta)*cos(phi)
113                 +sin(psi)*sin(phi),
114             d_gz_x5 = sin(psi)*sin(theta)*cos(phi)
115                 -cos(psi)*sin(phi),
116             d_gz_x6 = cos(theta)*cos(phi),
117             d_gz_x7 = 0,
118             d_gz_x8 = 0,
119             d_gz_x9 = -1,
120             d_gz_x10 = 0,
121             d_gz_x11 = 0,
122             d_gz_x12 = 0;
123
124     double d_f1_x1 = d_f1_u1 * (d_u1_gy*d_gy_x1 + d_u1_gz*d_gz_x1),
125             d_f1_x2 = d_f1_u1 * (d_u1_gy*d_gy_x2 + d_u1_gz*d_gz_x2),
126             d_f1_x3 = d_f1_u1 * (d_u1_gy*d_gy_x3 + d_u1_gz*d_gz_x3),
127             d_f1_x4 = d_f1_u1 * (d_u1_gy*d_gy_x4 + d_u1_gz*d_gz_x4),
128             d_f1_x5 = d_f1_u1 * (d_u1_gy*d_gy_x5 + d_u1_gz*d_gz_x5),
129             d_f1_x6 = d_f1_u1 * (d_u1_gy*d_gy_x6 + d_u1_gz*d_gz_x6),
130             d_f1_x7 = d_f1_u1 * (d_u1_gy*d_gy_x7 + d_u1_gz*d_gz_x7),
131             d_f1_x8 = d_f1_u1 * (d_u1_gy*d_gy_x8 + d_u1_gz*d_gz_x8),
132             d_f1_x9 = d_f1_u1 * (d_u1_gy*d_gy_x9 + d_u1_gz*d_gz_x9),
133             d_f1_x10 = d_f1_u1 * (d_u1_gy*d_gy_x10 + ...
                    d_u1_gz*d_gz_x10),
```

```
134             d_f1_x11 = d_f1_u1 * (d_u1_gy*d_gy_x11 + ...
                    d_u1_gz*d_gz_x11),
135             d_f1_x12 = d_f1_u1 * (d_u1_gy*d_gy_x12 + ...
                    d_u1_gz*d_gz_x12),
136
137             d_f2_x1 = d_f2_u2 * (d_u2_gx*d_gx_x1 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x1 + ...
                    d_v_t*d_t_gz*d_gz_x1)),
138             d_f2_x2 = d_f2_u2 * (d_u2_gx*d_gx_x2 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x2 + ...
                    d_v_t*d_t_gz*d_gz_x2)),
139             d_f2_x3 = d_f2_u2 * (d_u2_gx*d_gx_x3 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x3 + ...
                    d_v_t*d_t_gz*d_gz_x3)),
140             d_f2_x4 = d_f2_u2 * (d_u2_gx*d_gx_x4 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x4 + ...
                    d_v_t*d_t_gz*d_gz_x4)),
141             d_f2_x5 = d_f2_u2 * (d_u2_gx*d_gx_x5 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x5 + ...
                    d_v_t*d_t_gz*d_gz_x5)),
142             d_f2_x6 = d_f2_u2 * (d_u2_gx*d_gx_x6 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x6 + ...
                    d_v_t*d_t_gz*d_gz_x6)),
143             d_f2_x7 = d_f2_u2 * (d_u2_gx*d_gx_x7 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x7 + ...
                    d_v_t*d_t_gz*d_gz_x7)),
144             d_f2_x8 = d_f2_u2 * (d_u2_gx*d_gx_x8 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x8 + ...
                    d_v_t*d_t_gz*d_gz_x8)),
145             d_f2_x9 = d_f2_u2 * (d_u2_gx*d_gx_x9 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x9 + ...
                    d_v_t*d_t_gz*d_gz_x9)),
146             d_f2_x10 = d_f2_u2 * (d_u2_gx*d_gx_x10 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x10 + ...
                    d_v_t*d_t_gz*d_gz_x10)),
147             d_f2_x11 = d_f2_u2 * (d_u2_gx*d_gx_x11 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x11 + ...
                    d_v_t*d_t_gz*d_gz_x11)),
148             d_f2_x12 = d_f2_u2 * (d_u2_gx*d_gx_x12 + ...
                    d_u2_w*d_w_v*(d_v_s*d_s_gy*d_gy_x12 + ...
                    d_v_t*d_t_gz*d_gz_x12)),
149
150             d_f3_x1 = d_f3_u3 * (d_u3_Mx2*(d_Mx2_f1*d_f1_x1 + ...
                    d_Mx2_f2*d_f2_x1) + d_u3_My2*d_My2_f1*d_f1_x1),
151             d_f3_x2 = d_f3_u3 * (d_u3_Mx2*(d_Mx2_f1*d_f1_x2 + ...
                    d_Mx2_f2*d_f2_x2) + d_u3_My2*d_My2_f1*d_f1_x2),
152             d_f3_x3 = d_f3_u3 * (d_u3_Mx2*(d_Mx2_f1*d_f1_x3 + ...
```

```
                    d_Mx2_f2*d_f2_x3) + d_u3_My2*d_My2_f1*d_f1_x3),
153             d_f3_x4 = d_f3_u3 * (d_u3_Mx2*(d_Mx2_f1*d_f1_x4 + ...
                    d_Mx2_f2*d_f2_x4) + d_u3_My2*d_My2_f1*d_f1_x4),
154             d_f3_x5 = d_f3_u3 * (d_u3_Mx2*(d_Mx2_f1*d_f1_x5 + ...
                    d_Mx2_f2*d_f2_x5) + d_u3_My2*d_My2_f1*d_f1_x5),
155             d_f3_x6 = d_f3_u3 * (d_u3_Mx2*(d_Mx2_f1*d_f1_x6 + ...
                    d_Mx2_f2*d_f2_x6) + d_u3_My2*d_My2_f1*d_f1_x6),
156             d_f3_x7 = d_f3_u3 * (d_u3_Mx2*(d_Mx2_f1*d_f1_x7 + ...
                    d_Mx2_f2*d_f2_x7) + d_u3_My2*d_My2_f1*d_f1_x7),
157             d_f3_x8 = d_f3_u3 * (d_u3_Mx2*(d_Mx2_f1*d_f1_x8 + ...
                    d_Mx2_f2*d_f2_x8) + d_u3_My2*d_My2_f1*d_f1_x8),
158             d_f3_x9 = d_f3_u3 * (d_u3_Mx2*(d_Mx2_f1*d_f1_x9 + ...
                    d_Mx2_f2*d_f2_x9) + d_u3_My2*d_My2_f1*d_f1_x9),
159             d_f3_x10 = d_f3_u3 * d_u3_Mx2*d_Mx2_Mx,
160             d_f3_x11 = d_f3_u3 * (d_u3_Mx2*d_Mx2_My + ...
                    d_u3_My2*d_My2_My),
161             d_f3_x12 = d_f3_u3 * (d_u3_Mx2*d_Mx2_Mz + ...
                    d_u3_My2*d_My2_Mz);
162
163     double _J[3][12] = {{d_f1_x1, d_f1_x2, d_f1_x3, d_f1_x4, ...
            d_f1_x5, d_f1_x6, d_f1_x7, d_f1_x8, d_f1_x9, d_f1_x10, ...
            d_f1_x11, d_f1_x12},
164                         {d_f2_x1, d_f2_x2, d_f2_x3, d_f2_x4, ...
                                d_f2_x5, d_f2_x6, d_f2_x7, d_f2_x8, ...
                                d_f2_x9, d_f2_x10, d_f2_x11, d_f2_x12},
165                         {d_f3_x1, d_f3_x2, d_f3_x3, d_f3_x4, ...
                                d_f3_x5, d_f3_x6, d_f3_x7, d_f3_x8, ...
                                d_f3_x9, d_f3_x10, d_f3_x11, d_f3_x12}};
166     Mat J = Mat(3,12,CV_64FC1,_J),
167         R = J*N*J.t();
168
169     return R.clone();
170 }
```

D.3.2 TRIAD Measurement Noise

The measurement noise covariance matrix of the TRIAD algorithm is calculated in a similar manner to that of the tilt-heading algorithm.

The noise source covariance matrix remains the same. The C++ function to calculate the Jacobian is shown below.

```
1 Mat EKF::triad2Noise (double Vel_current[3], double accelMeas[3], ...
      double magMeas[3], double noise_vel[3], double noise_accel, ...
      double noise_mag)
```

```
2  {
3      double AccelI[3] =
4          {(Vel_current[0]-Vel_prev.at<double>(0))/TimeSinceLastUpdate,
5          (Vel_current[1]-Vel_prev.at<double>(1))/TimeSinceLastUpdate,
6          (Vel_current[2]-Vel_prev.at<double>(2))/TimeSinceLastUpdate};
7      Mat AccelInertial = Mat(3,1,CV_64FC1,AccelI);
8      Mat AccelBody = Mat(3,1,CV_64FC1,accelMeas);
9      Mat eulerEst = (Mat_<double>(3,1) <<
10         X.at<double>(0), X.at<double>(1), X.at<double>(2));
11     double e[3] = {eulerEst.at<double>(0),
12         eulerEst.at<double>(1), eulerEst.at<double>(2)};
13     Mat gravity_ref = (Mat_<double>(3,1) << 0, 0, -g);
14
15     Mat N =  diag ((Mat_<double>(12,1) <<
16         P.at<double>(0,0), P.at<double>(1,1), P.at<double>(2,2),
17         2*noise_vel[0]/TimeSinceLastUpdate,
18         2*noise_vel[1]/TimeSinceLastUpdate,
19         2*noise_vel[2]/TimeSinceLastUpdate,
20         noise_accel, noise_accel, noise_accel,
21         noise_mag, noise_mag, noise_mag), 12);
22
23     Mat B_D = Mat(3,1,CV_64FC1,magMeas),
24         G_D = AccelBody - dcm(eulerEst)*AccelInertial,
25         B_E = Mat(3,1,CV_64FC1,magRef),
26         G_E = gravity_ref;
27
28     Mat i_D = B_D/norm(B_D),
29         j_D = B_D.cross(G_D)/norm(B_D.cross(G_D)),
30         k_D = i_D.cross(j_D),
31         i_E = B_E/norm(B_E),
32         j_E = B_E.cross(G_E)/norm(B_E.cross(G_E)),
33         k_E = i_E.cross(j_E);
34
35     double B_D1 = B_D.at<double>(0),
36         B_D2 = B_D.at<double>(1),
37         B_D3 = B_D.at<double>(2),
38         G_D1 = G_D.at<double>(0),
39         G_D2 = G_D.at<double>(1),
40         G_D3 = G_D.at<double>(2);
41     double i_D1 = i_D.at<double>(0),
42         i_D2 = i_D.at<double>(1),
43         i_D3 = i_D.at<double>(2),
44         j_D1 = j_D.at<double>(0),
45         j_D2 = j_D.at<double>(1),
46         j_D3 = j_D.at<double>(2),
47         k_D1 = k_D.at<double>(0),
48         k_D2 = k_D.at<double>(1),
```

```
49          k_D3 = k_D.at<double>(2);
50      double i_E1 = i_E.at<double>(0),
51          i_E2 = i_E.at<double>(1),
52          i_E3 = i_E.at<double>(2),
53          j_E1 = j_E.at<double>(0),
54          j_E2 = j_E.at<double>(1),
55          j_E3 = j_E.at<double>(2),
56          k_E1 = k_E.at<double>(0),
57          k_E2 = k_E.at<double>(1),
58          k_E3 = k_E.at<double>(2);
59      double a[3] = { B_D2*G_D3 - B_D3*G_D2,
60          B_D3*G_D1 - B_D1*G_D3, B_D1*G_D2 - B_D2*G_D1};
61
62      // partial BD by partial magnetometer
63      Mat d_BD_M = (Mat_<double>(3,3) <<
64          1, 0, 0,
65          0, 1, 0,
66          0, 0, 1);
67
68      // partial GD by partial accelBody,eulerEst,accelInertial
69      Mat d_GD_AEI = (Mat_<double>(3,9) <<
70          1, 0, 0, 0, AccelI[2]*cos(e[1]) + ...
                AccelI[0]*cos(e[2])*sin(e[1]) + ...
                AccelI[1]*sin(e[2])*sin(e[1]), ...
                AccelI[0]*cos(e[1])*sin(e[2]) - ...
                AccelI[1]*cos(e[2])*cos(e[1]), -cos(e[2])*cos(e[1]), ...
                -cos(e[1])*sin(e[2]),  sin(e[1]),
71          0, 1, 0, AccelI[1]*(cos(e[2])*sin(e[0]) - ...
                cos(e[0])*sin(e[2])*sin(e[1])) - ...
                AccelI[0]*(sin(e[0])*sin(e[2]) + ...
                cos(e[0])*cos(e[2])*sin(e[1])) - ...
                AccelI[2]*cos(e[0])*cos(e[1]), ...
                AccelI[2]*sin(e[0])*sin(e[1]) - ...
                AccelI[0]*cos(e[2])*cos(e[1])*sin(e[0]) - ...
                AccelI[1]*cos(e[1])*sin(e[0])*sin(e[2]),   ...
                AccelI[0]*(cos(e[0])*cos(e[2]) + ...
                sin(e[0])*sin(e[2])*sin(e[1])) + ...
                AccelI[1]*(cos(e[0])*sin(e[2]) - ...
                cos(e[2])*sin(e[0])*sin(e[1])),   cos(e[0])*sin(e[2]) ...
                - cos(e[2])*sin(e[0])*sin(e[1]), - cos(e[0])*cos(e[2]) ...
                - sin(e[0])*sin(e[2])*sin(e[1]), -cos(e[1])*sin(e[0]),
72          0, 0, 1, AccelI[1]*(cos(e[0])*cos(e[2]) + ...
                sin(e[0])*sin(e[2])*sin(e[1])) - ...
                AccelI[0]*(cos(e[0])*sin(e[2]) - ...
                cos(e[2])*sin(e[0])*sin(e[1])) + ...
                AccelI[2]*cos(e[1])*sin(e[0]), ...
                AccelI[2]*cos(e[0])*sin(e[1]) - ...
```

```
            AccelI[0]*cos(e[0])*cos(e[2])*cos(e[1]) - ...
            AccelI[1]*cos(e[0])*cos(e[1])*sin(e[2]), - ...
            AccelI[0]*(cos(e[2])*sin(e[0]) - ...
            cos(e[0])*sin(e[2])*sin(e[1])) - ...
            AccelI[1]*(sin(e[0])*sin(e[2]) + ...
            cos(e[0])*cos(e[2])*sin(e[1])), - sin(e[0])*sin(e[2]) ...
            - cos(e[0])*cos(e[2])*sin(e[1]),   cos(e[2])*sin(e[0]) ...
            - cos(e[0])*sin(e[2])*sin(e[1]), -cos(e[0])*cos(e[1]) );
73
74      // partial BDGD by partial ...
            magnetometer,accelBody,eulerEst,accelInertial
75      Mat d_BDGD_MAEI = Mat::zeros(6,12,CV_64FC1);
76      for (int i = 0; i < 3; i++)
77          d_BDGD_MAEI.at<double>(i,i) = d_BD_M.at<double>(i,i);
78      for (int i = 0; i < 3; i++)
79          for (int j = 0; j < 9; j++)
80              d_BDGD_MAEI.at<double>(i,j) = d_GD_AEI.at<double>(i,j);
81
82      // partial i_D by partial BD,GD
83      Mat d_iD_BDGD = (Mat_<double>(3,6) <<
84          1/pow((pow(B_D1,2) + pow(B_D2,2) + pow(B_D3,2)),1./2) - ...
                pow(B_D1,2)/pow((pow(B_D1,2) + pow(B_D2,2) + ...
                pow(B_D3,2)),3./2), -(B_D1*B_D2)/pow(pow(B_D1,2) + ...
                pow(B_D2,2) + pow(B_D3,2),(3./2)), ...
                -(B_D1*B_D3)/pow((pow(B_D1,2) + pow(B_D2,2) + ...
                pow(B_D3,2)),3./2),
85          -(B_D1*B_D2)/pow((pow(B_D1,2) + pow(B_D2,2) + ...
                pow(B_D3,2)),3./2), 1/pow(pow(B_D1,2) + pow(B_D2,2) + ...
                pow(B_D3,2),1./2) - pow(B_D2,2)/pow(pow(B_D1,2) + ...
                pow(B_D2,2) + pow(B_D3,2),3./2), ...
                -(B_D2*B_D3)/pow(pow(B_D1,2) + pow(B_D2,2) + ...
                pow(B_D3,2),3./2),
86          -(B_D1*B_D3)/pow(pow(B_D1,2) + pow(B_D2,2) + ...
                pow(B_D3,2),3./2), -(B_D2*B_D3)/pow(pow(B_D1,2) + ...
                pow(B_D2,2) + pow(B_D3,2),3./2), 1/pow(pow(B_D1,2) + ...
                pow(B_D2,2) + pow(B_D3,2),1./2) - ...
                pow(B_D3,2)/pow(pow(B_D1,2) + pow(B_D2,2) + ...
                pow(B_D3,2),3./2) );
87
88      // partial a by partial BD,GD
89      Mat d_a_BDGD = (Mat_<double>(3,6) <<
90          0, G_D3, -G_D2, 0, -B_D3, B_D2,
91          -G_D3, 0, G_D1, B_D3, 0, -B_D1,
92          G_D2, -G_D1, 0, -B_D2, B_D1, 0 );
93
94      // partial j_D by partial a
95      Mat d_jD_a = (Mat_<double>(3,3) <<
```

```
96          1/pow((pow(a[0],2) + pow(a[1],2) + pow(a[2],2)),1./2) - ...
                pow(a[0],2)/pow((pow(a[0],2) + pow(a[1],2) + ...
                pow(a[2],2)),3./2), -(a[0]*a[1])/pow(pow(a[0],2) + ...
                pow(a[1],2) + pow(a[2],2),(3./2)), ...
                -(a[0]*a[2])/pow((pow(a[0],2) + pow(a[1],2) + ...
                pow(a[2],2)),3./2),
97          -(a[0]*a[1])/pow((pow(a[0],2) + pow(a[1],2) + ...
                pow(a[2],2)),3./2), 1/pow(pow(a[0],2) + pow(a[1],2) + ...
                pow(a[2],2),1./2) - pow(a[1],2)/pow(pow(a[0],2) + ...
                pow(a[1],2) + pow(a[2],2),3./2), ...
                -(a[1]*a[2])/pow(pow(a[0],2) + pow(a[1],2) + ...
                pow(a[2],2),3./2),
98          -(a[0]*a[2])/pow(pow(a[0],2) + pow(a[1],2) + ...
                pow(a[2],2),3./2), -(a[1]*a[2])/pow(pow(a[0],2) + ...
                pow(a[1],2) + pow(a[2],2),3./2), 1/pow(pow(a[0],2) + ...
                pow(a[1],2) + pow(a[2],2),1./2) - ...
                pow(a[2],2)/pow(pow(a[0],2) + pow(a[1],2) + ...
                pow(a[2],2),3./2) );
99
100     // partial k_D by partial i_D
101     Mat d_kD_iD = (Mat_<double>(3,3) <<
102         0, j_D3, -j_D2,
103         -j_D3, 0, j_D1,
104         j_D2, -j_D1, 0 );
105
106     // partial k_D by partial j_D
107     Mat d_kD_jD = (Mat_<double>(3,3) <<
108         0, -i_D3,  i_D2,
109         i_D3, 0, -i_D1,
110         -i_D2, i_D1, 0 );
111
112
113     Mat d_phi_iD = (Mat_<double>(1,3) <<
114         0, i_E3/((pow(i_D2*i_E3 + j_D2*j_E3 + ...
                k_D2*k_E3,2)/pow(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3,2) ...
                + 1)*(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3)), ...
                -(i_E3*(i_D2*i_E3 + j_D2*j_E3 + ...
                k_D2*k_E3))/((pow(i_D2*i_E3 + j_D2*j_E3 + ...
                k_D2*k_E3,2)/pow(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3,2) ...
                + 1)*pow(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3,2)) );
115
116     Mat d_phi_jD = (Mat_<double>(1,3) <<
117         0, j_E3/((pow(i_D2*i_E3 + j_D2*j_E3 + ...
                k_D2*k_E3,2)/pow(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3,2) ...
                + 1)*(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3)), ...
                -(j_E3*(i_D2*i_E3 + j_D2*j_E3 + ...
                k_D2*k_E3))/((pow(i_D2*i_E3 + j_D2*j_E3 + ...
```

```
            k_D2*k_E3,2)/pow(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3,2) ...
            + 1)*pow(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3,2)) );
118
119     Mat d_phi_kD = (Mat_<double>(1,3) <<
120         0, k_E3/((pow(i_D2*i_E3 + j_D2*j_E3 + ...
                k_D2*k_E3,2)/pow(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3,2) ...
                + 1)*(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3)), ...
                -(k_E3*(i_D2*i_E3 + j_D2*j_E3 + ...
                k_D2*k_E3))/((pow(i_D2*i_E3 + j_D2*j_E3 + ...
                k_D2*k_E3,2)/pow(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3,2) ...
                + 1)*pow(i_D3*i_E3 + j_D3*j_E3 + k_D3*k_E3,2)) );
121
122     Mat d_theta_iD = (Mat_<double>(1,3) <<  -i_E3/pow(1 - ...
            pow(i_D1*i_E3 + j_D1*j_E3 + k_D1*k_E3,2),0.5), 0, 0);
123     Mat d_theta_jD = (Mat_<double>(1,3) <<  -j_E3/pow(1 - ...
            pow(i_D1*i_E3 + j_D1*j_E3 + k_D1*k_E3,2),0.5), 0, 0);
124     Mat d_theta_kD = (Mat_<double>(1,3) <<  -k_E3/pow(1 - ...
            pow(i_D1*i_E3 + j_D1*j_E3 + k_D1*k_E3,2),0.5), 0, 0);
125
126     Mat d_psi_iD = (Mat_<double>(1,3) << (i_E2/(i_D1*i_E1 + ...
            j_D1*j_E1 + k_D1*k_E1) - (i_E1*(i_D1*i_E2 + j_D1*j_E2 + ...
            k_D1*k_E2))/pow(i_D1*i_E1 + j_D1*j_E1 + ...
            k_D1*k_E1,2))/(pow(i_D1*i_E2 + j_D1*j_E2 + ...
            k_D1*k_E2,2)/pow(i_D1*i_E1 + j_D1*j_E1 + k_D1*k_E1,2) + ...
            1), 0, 0);
127
128     Mat d_psi_jD = (Mat_<double>(1,3) << (j_E2/(i_D1*i_E1 + ...
            j_D1*j_E1 + k_D1*k_E1) - (j_E1*(i_D1*i_E2 + j_D1*j_E2 + ...
            k_D1*k_E2))/pow(i_D1*i_E1 + j_D1*j_E1 + ...
            k_D1*k_E1,2))/(pow(i_D1*i_E2 + j_D1*j_E2 + ...
            k_D1*k_E2,2)/pow(i_D1*i_E1 + j_D1*j_E1 + k_D1*k_E1,2) + ...
            1), 0, 0);
129
130     Mat d_psi_kD = (Mat_<double>(1,3) << (k_E2/(i_D1*i_E1 + ...
            j_D1*j_E1 + k_D1*k_E1) - (k_E1*(i_D1*i_E2 + j_D1*j_E2 + ...
            k_D1*k_E2))/pow(i_D1*i_E1 + j_D1*j_E1 + ...
            k_D1*k_E1,2))/(pow(i_D1*i_E2 + j_D1*j_E2 + ...
            k_D1*k_E2,2)/pow(i_D1*i_E1 + j_D1*j_E1 + k_D1*k_E1,2) + ...
            1), 0, 0);
131
132     Mat d_jD_BDGD = d_jD_a * d_a_BDGD;
133     Mat d_kD_BDGD = d_kD_iD * d_iD_BDGD + d_kD_jD * d_jD_a * d_a_BDGD;
134
135     Mat d_phi_BDGD = d_phi_iD * d_iD_BDGD
136         + d_phi_jD * d_jD_BDGD
137         + d_phi_kD * d_kD_BDGD;
138
```

```
139     Mat d_theta_BDGD = d_theta_iD * d_iD_BDGD
140         + d_theta_jD * d_jD_BDGD
141         + d_theta_kD * d_kD_BDGD;
142
143     Mat d_psi_BDGD = d_psi_iD * d_iD_BDGD
144         + d_psi_jD * d_jD_BDGD
145         + d_psi_kD * d_kD_BDGD;
146
147     Mat d_phi_MAEI = d_phi_BDGD * d_BDGD_MAEI;
148     Mat d_theta_MAEI = d_theta_BDGD * d_BDGD_MAEI;
149     Mat d_psi_MAEI = d_psi_BDGD * d_BDGD_MAEI;
150
151     Mat J = Mat::zeros(3,12,CV_64FC1);
152
153     for (int j = 0; j < 12; j++)
154     {
155         J.at<double>(0,j) = d_phi_MAEI.at<double>(j);
156         J.at<double>(1,j) = d_theta_MAEI.at<double>(j);
157         J.at<double>(2,j) = d_psi_MAEI.at<double>(j);
158     }
159
160     Mat R = J*N*J.t();
161
162     return R.clone();
163 }
```

D.4 KF Process Noise

The partial derivatives of the Jacobian in (6.4.5) are given as follows:

$$\frac{\partial u_1}{\partial \alpha_x} = \cos\psi\cos\theta$$
$$\frac{\partial u_1}{\partial \alpha_y} = \cos\psi\sin\phi\sin\theta - \cos\phi\sin\psi$$
$$\frac{\partial u_1}{\partial \alpha_z} = \sin\phi\sin\psi + \cos\phi\cos\psi\sin\theta$$
$$\frac{\partial u_1}{\partial \phi_{heli}} = \alpha_2(\sin\phi\sin\psi + \cos\phi\cos\psi\sin\theta) + \alpha_3(\cos\phi\sin\psi - \cos\psi\sin\phi\sin\theta)$$
$$\frac{\partial u_1}{\partial \theta_{heli}} = \alpha_3\cos\phi\cos\psi\cos\theta - \alpha_1\cos\psi\sin\theta + \alpha_2\cos\psi\cos\theta\sin\phi$$
$$\frac{\partial u_1}{\partial \psi_{heli}} = \alpha_3(\cos\psi\sin\phi - \cos\phi\sin\psi\sin\theta) - \alpha_2(\cos\phi\cos\psi + \sin\phi\sin\psi\sin\theta) - \alpha_1\cos\theta\sin\psi$$
$$\frac{\partial u_2}{\partial \alpha_x} = \cos\theta\sin\psi$$
$$\frac{\partial u_2}{\partial \alpha_y} = \cos\phi\cos\psi + \sin\phi\sin\psi\sin\theta$$
$$\frac{\partial u_2}{\partial \alpha_z} = \cos\phi\sin\psi\sin\theta - \cos\psi\sin\phi$$
$$\frac{\partial u_2}{\partial \phi_{heli}} = -\alpha_2(\cos\psi\sin\phi - \cos\phi\sin\psi\sin\theta) - \alpha_3(\cos\phi\cos\psi + \sin\phi\sin\psi\sin\theta)$$
$$\frac{\partial u_2}{\partial \theta_{heli}} = \alpha_3\cos\phi\cos\theta\sin\psi - \alpha_1\sin\psi\sin\theta + \alpha_2\cos\theta\sin\phi\sin\psi$$
$$\frac{\partial u_2}{\partial \psi_{heli}} = \alpha_3(\sin\phi\sin\psi + \cos\phi\cos\psi\sin\theta) - \alpha_2(\cos\phi\sin\psi - \cos\psi\sin\phi\sin\theta) + \alpha_1\cos\psi\cos\theta$$
$$\frac{\partial u_3}{\partial \alpha_x} = -\sin\theta$$
$$\frac{\partial u_3}{\partial \alpha_y} = \cos\theta\sin\phi$$
$$\frac{\partial u_3}{\partial \alpha_z} = \cos\phi\cos\theta$$
$$\frac{\partial u_3}{\partial \phi_{heli}} = \alpha_2\cos\phi\cos\theta - \alpha_3\cos\theta\sin\phi$$
$$\frac{\partial u_3}{\partial \theta_{heli}} = -\alpha_1\cos\theta - \alpha_3\cos\phi\sin\theta - \alpha_2\sin\phi\sin\theta$$
$$\frac{\partial u_3}{\partial \psi_{heli}} = 0$$

D.5 Deck Position Estimate Noise

Equation (5.2.1) shows the transformation used to calculate the deck position estimate. It is repeated here for convenience:

$$\hat{\mathbf{p}}_{deck} = \mathbf{DCM}^{T}_{\hat{\boldsymbol{\theta}}_{ship}} \mathbf{p}_{CG \to deck} + \hat{\mathbf{p}}_{ship}.$$

The noise sources comprise of uncertainty in the initial ship position and attitude estimates. The position offset between the ship center and deck can be measured perfectly, so has no uncertainty. A diagonal covariance matrix can be constructed for the noise sources. Let s_i represent the i_{th} element of the initial ship position estimate:

$$\mathbf{N} = \text{diag}(\sigma^2_{\phi}, \sigma^2_{\theta}, \sigma^2_{\psi}, \sigma^2_{s_1}, \sigma^2_{s_2}, \sigma^2_{s_3}). \tag{D.5.1}$$

The estimated deck position covariance matrix is determined as follows:

$$\mathbf{R}_{\hat{\boldsymbol{\theta}}_{deck}} = \mathbf{J}\ \mathbf{N}\ \mathbf{J}^{T}. \tag{D.5.2}$$

The following script was used to determine the Jacobian:

```
1  %% partial differential equations for estimated deck position
2
3  syms phi theta psi p1 p2 p3 s1 s2 s3
4
5  inv_dcm = [cos(psi)*cos(theta), ...
       cos(psi)*sin(theta)*sin(phi)-sin(psi)*cos(phi), ...
       cos(psi)*sin(theta)*cos(phi)+sin(psi)*sin(phi); ...
6          sin(psi)*cos(theta), ...
               sin(psi)*sin(theta)*sin(phi)+cos(psi)*cos(phi), ...
               sin(psi)*sin(theta)*cos(phi)-cos(psi)*sin(phi); ...
7          -sin(theta), cos(theta)*sin(phi), cos(theta)*cos(phi)];
8
9  shipDeckPosEst = inv_dcm*[p1;p2;p3] + [s1;s2;s3];
10
11 jac = jacobian(shipDeckPosEst, [phi theta psi s1 s2 s3])
```

D.6 Initial Relative Estimate Noise

D.6.1 Position

Initial relative estimates are calculated from the helicopter and deck position estimates. This is shown in Equation (5.2.2).

Let $\mathbf{r} = \mathbf{p}_{rel}$, $\mathbf{d} = \mathbf{p}_{deck}$, $\mathbf{h} = \mathbf{p}_{heli}$ and $\boldsymbol{\theta}_{heli} = [\,\phi \;\; \theta \;\; \psi\,]^T$. Rewriting Equation (5.2.2) we get:

$$\mathbf{r} = \mathbf{DCM}^T_{\boldsymbol{\theta}_{heli}}(\mathbf{d} - \mathbf{h})$$

$$\begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} d_1 - h_1 \\ d_2 - h_2 \\ d_3 - h_3 \end{bmatrix}.$$

The noise covariance matrix of the initial relative position estimate, $\mathbf{R}_{\hat{\mathbf{p}}_{rel}}$, is calculated as follows:

$$\mathbf{R}_{\hat{\mathbf{p}}_{rel}} = \mathbf{J}\ \mathbf{N}\ \mathbf{J}^T. \tag{D.6.1}$$

The noise sources consist of helicopter position and attitude estimates, as well as deck position estimates:

$$\mathbf{N} = \text{diag}(\sigma^2_\phi, \sigma^2_\theta, \sigma^2_\psi, \sigma^2_{h_1}, \sigma^2_{h_2}, \sigma^2_{h_3}, \sigma^2_{d_1}, \sigma^2_{d_2}, \sigma^2_{d_3}). \tag{D.6.2}$$

A first-order linearisation of Equation (5.2.2) is taken:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial r_1}{\partial\phi} & \frac{\partial r_1}{\partial\theta} & \frac{\partial r_1}{\partial\psi} & \frac{\partial r_1}{\partial h_1} & \frac{\partial r_1}{\partial h_2} & \frac{\partial r_1}{\partial h_3} & \frac{\partial r_1}{\partial d_1} & \frac{\partial r_1}{\partial d_2} & \frac{\partial r_1}{\partial d_3} \\ \frac{\partial r_2}{\partial\phi} & \frac{\partial r_2}{\partial\theta} & \frac{\partial r_2}{\partial\psi} & \frac{\partial r_2}{\partial h_1} & \frac{\partial r_2}{\partial h_2} & \frac{\partial r_2}{\partial h_3} & \frac{\partial r_2}{\partial d_1} & \frac{\partial r_2}{\partial d_2} & \frac{\partial r_2}{\partial d_3} \\ \frac{\partial r_3}{\partial\phi} & \frac{\partial r_3}{\partial\theta} & \frac{\partial r_3}{\partial\psi} & \frac{\partial r_3}{\partial h_1} & \frac{\partial r_3}{\partial h_2} & \frac{\partial r_3}{\partial h_3} & \frac{\partial r_3}{\partial d_1} & \frac{\partial r_3}{\partial d_2} & \frac{\partial r_3}{\partial d_3} \end{bmatrix}. \tag{D.6.3}$$

The partial derivatives are determined as follows:

$$\frac{\partial r_1}{\partial \phi} = (d_2 - h_2)(c_\phi\ s_\theta\ c_\psi + s_\phi\ s_\psi) + (d_3 - h_3)(-s_\phi\ s_\theta\ c_\psi + c_\phi\ s_\psi)$$
$$\frac{\partial r_1}{\partial \theta} = (d_1 - h_1)(-s_\theta\ c_\psi) + (d_2 - h_2)(s_\phi\ c_\theta\ s_\psi) + (d_3 - h_3)(c_\phi\ c_\theta\ c_\psi)$$
$$\frac{\partial r_1}{\partial \psi} = (d_1 - h_1)(c_\theta\ s_\psi) + (d_2 - h_2)(-s_\phi\ s_\theta\ s_\psi - c_\phi\ c_\psi) + (d_3 - h_3)(-c_\phi\ s_\theta\ s_\psi + s_\phi\ c_\psi)$$
$$\frac{\partial r_1}{\partial h_1} = -c_\theta\ c_\psi$$
$$\frac{\partial r_1}{\partial h_2} = -(s_\phi\ s_\theta\ c_\psi - c_\phi\ s_\psi)$$
$$\frac{\partial r_1}{\partial h_3} = -(c_\phi\ s_\theta\ c_\psi - s_\phi\ s_\psi)$$
$$\frac{\partial r_1}{\partial d_1} = c_\theta\ c_\psi$$
$$\frac{\partial r_1}{\partial d_2} = s_\phi\ s_\theta\ c_\psi - c_\phi\ s_\psi$$
$$\frac{\partial r_1}{\partial d_3} = c_\phi\ s_\theta\ c_\psi - s_\phi\ s_\psi$$

$$\frac{\partial r_2}{\partial \phi} = (d_2 - h_2)(c_\phi\ s_\theta\ s_\psi - s_\phi\ c_\psi) + (d_3 - h_3)(-s_\phi\ s_\theta\ s_\psi - c_\phi\ c_\psi)$$
$$\frac{\partial r_2}{\partial \theta} = (d_1 - h_1)(-s_\theta\ s_\psi) + (d_2 - h_2)(s_\phi\ c_\theta\ s_\psi) + (d_3 - h_3)(c_\phi\ c_\theta\ s_\psi)$$
$$\frac{\partial r_2}{\partial \psi} = (d_1 - h_1)(c_\theta\ c_\psi) + (d_2 - h_2)(s_\phi\ s_\theta\ c_\psi - c_\phi\ s_\psi) + (d_3 - h_3)(c_\phi\ s_\theta\ c_\psi + s_\phi\ s_\psi)$$
$$\frac{\partial r_2}{\partial h_1} = -c_\theta\ s_\psi$$
$$\frac{\partial r_2}{\partial h_2} = -(s_\phi\ s_\theta\ s_\psi + c_\phi\ c_\psi)$$
$$\frac{\partial r_2}{\partial h_3} = -(c_\phi\ s_\theta\ s_\psi - s_\phi\ c_\psi)$$
$$\frac{\partial r_2}{\partial d_1} = c_\theta\ s_\psi$$
$$\frac{\partial r_2}{\partial d_2} = s_\phi\ s_\theta\ s_\psi + c_\phi\ c_\psi$$
$$\frac{\partial r_2}{\partial d_3} = c_\phi\ s_\theta\ s_\psi - s_\phi\ c_\psi$$

$$\frac{\partial r_3}{\partial \phi} = (d_2 - h_2)(c_\phi\ c_\theta) + (d_3 - h_3)(-s_\phi\ c_\theta)$$
$$\frac{\partial r_3}{\partial \theta} = (d_1 - h_1)(-c_\theta) + (d_2 - h_2)(-s_\phi\ s_\theta) + (d_3 - h_3)(-c_\phi\ s_\theta)$$
$$\frac{\partial r_3}{\partial \psi} = 0$$
$$\frac{\partial r_3}{\partial h_1} = s_\theta$$
$$\frac{\partial r_3}{\partial h_2} = -s_\phi\ c_\theta$$
$$\frac{\partial r_3}{\partial h_3} = -c_\phi\ c_\theta$$
$$\frac{\partial r_3}{\partial d_1} = -s_\theta$$
$$\frac{\partial r_3}{\partial d_2} = s_\phi\ c_\theta$$
$$\frac{\partial r_3}{\partial d_3} = c_\phi\ c_\theta$$

D.6.2 Attitude

Initial relative attitude estimates are calculated using Equation (5.2.2), which is reproduced here for convenience:

$$\mathbf{DCM}_{\boldsymbol{\theta}_{rel}} = \mathbf{DCM}_{\boldsymbol{\theta}_{heli}}^{T} \mathbf{DCM}_{\boldsymbol{\theta}_{deck}}.$$

Let $\mathbf{r}_{jk}$ be the element of $\mathbf{DCM}_{\boldsymbol{\theta}_{rel}}$ at row j, column k. Euler angles are extracted as follows:

$$\boldsymbol{\theta}_{rel} = \begin{bmatrix} \phi_{rel} \\ \theta_{rel} \\ \psi_{rel} \end{bmatrix} = \begin{bmatrix} \arctan\left(r_{23}/r_{33}\right) \\ -\arcsin(r_{13}) \\ \arctan2(r_{12}, r_{11}) \end{bmatrix}. \tag{D.6.4}$$

where

$$\begin{aligned}
r_{23} &= (c_{\theta_h}\ s_{\psi_h})\ (-s_{\theta_d}) + (s_{\phi_h}\ s_{\theta_h}\ s_{\psi_h} + c_{\phi_h}\ c_{\psi_h})\ (s_{\phi_d}\ c_{\theta_d}) \\
&\quad + (c_{\phi_h}\ s_{\theta_h}\ s_{\psi_h} - s_{\phi_h}\ c_{\psi_h})\ (c_{\phi_d}\ c_{\theta_d}) \\
r_{33} &= (-s_{\theta_h})(-s_{\theta_d}) + (s_{\phi_h}c_{\theta_h})(s_{\phi_d}c_{\theta_d}) + (c_{\phi_h}c_{\theta_h})(c_{\phi_d}c_{\theta_d}) \\
r_{13} &= (c_{\theta_h}c_{\psi_h})(-s_{\theta_d}) + (s_{\phi_h}s_{\theta_h}c_{\psi_h} - c_{\phi_h}s_{\psi_h})(s_{\phi_d}c_{\theta_d}) \\
&\quad + (c_{\phi_h}s_{\theta_h}c_{\psi_h} + s_{\phi_h}s_{\psi_h})(c_{\phi_d}c_{\theta_d}) \\
r_{12} &= (c_{\theta_h}c_{\psi_h})(c_{\theta_d}s_{\psi_d}) + (s_{\phi_h}s_{\theta_h}c_{\psi_h} - c_{\phi_h}s_{\psi_h})(s_{\phi_d}s_{\theta_d}s_{\psi_d} + c_{\phi_d}c_{\psi_d}) \\
&\quad + (c_{\phi_h}s_{\theta_h}c_{\psi_h} + s_{\phi_h}s_{\psi_h})(c_{\phi_d}c_{\theta_d}) \\
r_{11} &= (c_{\theta_h}c_{\psi_h})(c_{\theta_d}c_{\psi_d}) + (s_{\phi_h}s_{\theta_h}c_{\psi_h} - c_{\phi_h}s_{\psi_h})(s_{\phi_d}s_{\theta_d}c_{\psi_d} - c_{\phi_d}s_{\psi_d}) \\
&\quad + (c_{\phi_h}s_{\theta_h}c_{\psi_h} + s_{\phi_h}s_{\psi_h})(c_{\phi_d}s_{\theta_d}c_{\psi_d} + s_{\phi_d}s_{\psi_d}).
\end{aligned}$$

The noise covariance matrix of the initial relative attitude estimate is calculated as follows:

$$\mathbf{R}_{\hat{\boldsymbol{\theta}}_{rel}} = \mathbf{J}\ \mathbf{N}\ \mathbf{J}^T. \tag{D.6.5}$$

The noise sources consist of noise in the helicopter and deck attitude estimates:

$$\mathbf{N} = \text{diag}(\sigma^2_{\phi_h}, \sigma^2_{\theta_h}, \sigma^2_{\psi_h}, \sigma^2_{\phi_d}, \sigma^2_{\theta_d}, \sigma^2_{\psi_d}). \tag{D.6.6}$$

The Jacobian is calculated by taking a first-order linear approximation of Equation (5.2.2):

$$\begin{aligned}
\mathbf{J} &= \begin{bmatrix}
\frac{\partial\phi_{rel}}{\partial\phi_h} & \frac{\partial\phi_{rel}}{\partial\theta_h} & \frac{\partial\phi_{rel}}{\partial\psi_h} & \frac{\partial\phi_{rel}}{\partial\phi_d} & \frac{\partial\phi_{rel}}{\partial\theta_d} & \frac{\partial\phi_{rel}}{\partial\psi_d} \\
\frac{\partial\theta_{rel}}{\partial\phi_h} & \frac{\partial\theta_{rel}}{\partial\theta_h} & \frac{\partial\theta_{rel}}{\partial\psi_h} & \frac{\partial\theta_{rel}}{\partial\phi_d} & \frac{\partial\theta_{rel}}{\partial\theta_d} & \frac{\partial\theta_{rel}}{\partial\psi_d} \\
\frac{\partial\psi_{rel}}{\partial\phi_h} & \frac{\partial\psi_{rel}}{\partial\theta_h} & \frac{\partial\psi_{rel}}{\partial\psi_h} & \frac{\partial\psi_{rel}}{\partial\phi_d} & \frac{\partial\psi_{rel}}{\partial\theta_d} & \frac{\partial\psi_{rel}}{\partial\psi_d}
\end{bmatrix} \\
&= \begin{bmatrix}
\frac{\partial\phi_{rel}}{\partial r_{23}}\frac{\partial r_{23}}{\partial\phi_h} + \frac{\partial\phi_{rel}}{\partial r_{33}}\frac{\partial r_{33}}{\partial\phi_h} & \frac{\partial\phi_{rel}}{\partial r_{23}}\frac{\partial r_{23}}{\partial\theta_h} + \frac{\partial\phi_{rel}}{\partial r_{33}}\frac{\partial r_{33}}{\partial\theta_h} & \cdots & \frac{\partial\phi_{rel}}{\partial r_{23}}\frac{\partial r_{23}}{\partial\psi_d} + \frac{\partial\phi_{rel}}{\partial r_{33}}\frac{\partial r_{33}}{\partial\psi_d} \\
\frac{\partial\phi_{rel}}{\partial r_{13}}\frac{\partial r_{13}}{\partial\phi_h} & \frac{\partial\phi_{rel}}{\partial r_{13}}\frac{\partial r_{13}}{\partial\theta_h} & \cdots & \frac{\partial\phi_{rel}}{\partial r_{13}}\frac{\partial r_{13}}{\partial\psi_d} \\
\frac{\partial\phi_{rel}}{\partial r_{12}}\frac{\partial r_{12}}{\partial\phi_h} + \frac{\partial\phi_{rel}}{\partial r_{11}}\frac{\partial r_{11}}{\partial\phi_h} & \frac{\partial\phi_{rel}}{\partial r_{12}}\frac{\partial r_{12}}{\partial\theta_h} + \frac{\partial\phi_{rel}}{\partial r_{11}}\frac{\partial r_{11}}{\partial\theta_h} & \cdots & \frac{\partial\phi_{rel}}{\partial r_{12}}\frac{\partial r_{12}}{\partial\psi_d} + \frac{\partial\phi_{rel}}{\partial r_{11}}\frac{\partial r_{11}}{\partial\psi_d}
\end{bmatrix}.
\end{aligned}$$

The partial differential equations are calculated as follows:

$$\frac{\partial \phi_{rel}}{\partial r_{23}} = \frac{r_{33}}{{r_{23}}^2 + {r_{33}}^2} \qquad \frac{\partial \theta_{rel}}{\partial r_{23}} = 0 \qquad \frac{\partial \psi_{rel}}{\partial r_{23}} = 0$$
$$\frac{\partial \phi_{rel}}{\partial r_{33}} = -\frac{r_{23}}{{r_{23}}^2 + {r_{33}}^2} \qquad \frac{\partial \theta_{rel}}{\partial r_{23}} = 0 \qquad \frac{\partial \psi_{rel}}{\partial r_{23}} = 0$$
$$\frac{\partial \phi_{rel}}{\partial r_{13}} = 0 \qquad \frac{\partial \theta_{rel}}{\partial r_{13}} = -\frac{1}{\sqrt{1 - {r_{13}}^2}} \qquad \frac{\partial \psi_{rel}}{\partial r_{13}} = 0$$
$$\frac{\partial \phi_{rel}}{\partial r_{12}} = 0 \qquad \frac{\partial \theta_{rel}}{\partial r_{12}} = 0 \qquad \frac{\partial \psi_{rel}}{\partial r_{12}} = \frac{r_{11}}{{r_{11}}^2 + r_{12}^2}$$
$$\frac{\partial \phi_{rel}}{\partial r_{11}} = 0 \qquad \frac{\partial \theta_{rel}}{\partial r_{11}} = 0 \qquad \frac{\partial \psi_{rel}}{\partial r_{11}} = -\frac{r_{12}}{{r_{11}}^2 + r_{12}^2}.$$

The following Matlab script was used to calculate the remaining partial derivatives:

```
%% partial derivatives for the Initial Relative Attitude Estimate ...
    Jacobian

syms phi_h theta_h psi_h phi_d theta_d psi_d

r_23 = (cos(theta_h)*sin(psi_h)) * (-sin(theta_d))...
    +(sin(phi_h)*sin(theta_h)*sin(psi_h)+cos(phi_h)*cos(psi_h)) * ...
        (sin(phi_d)*cos(theta_d))...
    +((cos(phi_h)*sin(theta_h)*sin(psi_h)-sin(phi_h)*cos(psi_h)) * ...
        (cos(phi_d)*cos(theta_d)));

r_33 = ((-sin(theta_h)) * (-sin(theta_d)))...
    +((sin(phi_h)*cos(theta_h)) * (sin(phi_d)*cos(theta_d)))...
    +((cos(phi_h)*cos(theta_h)) * (cos(phi_d)*cos(theta_d)));

r_13 = ((cos(theta_h)*cos(psi_h)) * (-sin(theta_d)))...
    +((sin(phi_h)*sin(theta_h)*cos(psi_h)-cos(phi_h)*sin(psi_h)) * ...
        (sin(phi_d)*cos(theta_d)))...
    +((cos(phi_h)*sin(theta_h)*cos(psi_h)+sin(phi_h)*sin(psi_h)) * ...
        (cos(phi_d)*cos(theta_d)));

r_12 = ((cos(theta_h)*cos(psi_h)) * (cos(theta_d)*sin(psi_d)))...
    +((sin(phi_h)*sin(theta_h)*cos(psi_h)-cos(phi_h)*sin(psi_h)) * ...
        (sin(phi_d)*sin(theta_d)*sin(psi_d)+cos(phi_d)*cos(psi_d)))...
    +((cos(phi_h)*sin(theta_h)*cos(psi_h)+sin(phi_h)*sin(psi_h)) * ...
        (cos(phi_d)*cos(theta_d)));

r_11 = ((cos(theta_h)*cos(psi_h)) * (cos(theta_d)*cos(psi_d)))...
    +((sin(phi_h)*sin(theta_h)*cos(psi_h)-cos(phi_h)*sin(psi_h)) * ...
        (sin(phi_d)*sin(theta_d)*cos(psi_d)-cos(phi_d)*sin(psi_d)))...
    +((cos(phi_h)*sin(theta_h)*cos(psi_h)+sin(phi_h)*sin(psi_h)) * ...
        (cos(phi_d)*sin(theta_d)*cos(psi_d)+sin(phi_d)*sin(psi_d)));
```

```
24
25 f = [r_23, r_33, r_13, r_12, r_11];
26 jac = jacobian(f,[phi_h theta_h psi_h phi_d theta_d psi_d])
```

D.7 Ship Deck Measurement Noise

D.7.1 Position

Equation (5.2.9) is used to measure the deck position using final relative position measurements. It is repeated here for convenience:

$$\tilde{\mathbf{p}}_{deck} = \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{heli}} \cdot \bar{\mathbf{p}}_{rel} + \hat{\mathbf{p}}_{heli}.$$

The noise sources for this calculation consist of uncertainty in the helicopter position and attitude estimates, and the final relative position measurement:

$$\mathbf{N} = \text{diag}(\sigma^2_{\phi_h}, \sigma^2_{\theta_h}, \sigma^2_{\psi_h}, \sigma^2_{p_{r_1}}, \sigma^2_{p_{r_2}}, \sigma^2_{p_{r_3}}, \sigma^2_{p_{h_1}}, \sigma^2_{p_{h_2}}, \sigma^2_{p_{h_3}}). \tag{D.7.1}$$

The noise covariance matrix for the deck position measurements is determined using the following:

$$\mathbf{R}_{\tilde{\mathbf{p}}_{deck}} = \mathbf{J}\ \mathbf{N}\ \mathbf{J}^T. \tag{D.7.2}$$

The following script was used to determine the Jacobian:

```
1 %% partial derivatives for measured deck position
2
3 syms phi theta psi r1 r2 r3 h1 h2 h3
4
5 deckPosMeas = [cos(psi)*cos(theta), sin(psi)*cos(theta), ...
      -sin(theta); ...
6      cos(psi)*sin(theta)*sin(phi)-sin(psi)*cos(phi), ...
           sin(psi)*sin(theta)*sin(phi)+cos(psi)*cos(phi), ...
           cos(theta)*sin(phi); ...
7      cos(psi)*sin(theta)*cos(phi)+sin(psi)*sin(phi), ...
           sin(psi)*sin(theta)*cos(phi)-cos(psi)*sin(phi), ...
           cos(theta)*cos(phi)]*[r1;r2;r3]+[h1;h2;h3];
8
9 jac = jacobian(deckPosMeas, [phi theta psi r1 r2 r3 h1 h2 h3])
```

D.7.2 Attitude

Equation (5.2.9) is used to measure the deck attitude using final relative attitude measurements. It is repeated here for convenience:

$$\mathbf{DCM}_{\tilde{\boldsymbol{\theta}}_{deck}} = \mathbf{DCM}_{\hat{\boldsymbol{\theta}}_{heli}} \cdot \mathbf{DCM}_{\bar{\boldsymbol{\theta}}_{rel}}.$$

The noise sources consist of uncertainty in the helicopter and relative attitude measurements:

$$\mathbf{N} = \text{diag}(\sigma^2_{\phi_h}, \sigma^2_{\theta_h}, \sigma^2_{\psi_h}, \sigma^2_{\phi_r}, \sigma^2_{\theta_r}, \sigma^2_{\psi_r}). \tag{D.7.3}$$

The Jacobian is determined similarly to that in Chapter D.6.2. The following script was used to calculate it:

```
%% partial derivatives for measured deck attitude

syms phi_h theta_h psi_h phi_r theta_r psi_r
syms D_23 D_33 D_13 D_12 D_11

deckEulerMeasDCM = [cos(psi_h)*cos(theta_h), ...
    sin(psi_h)*cos(theta_h), -sin(theta_h); ...
    cos(psi_h)*sin(theta_h)*sin(phi_h)-sin(psi_h)*cos(phi_h), ...
        sin(psi_h)*sin(theta_h)*sin(phi_h)+cos(psi_h)*cos(phi_h), ...
        cos(theta_h)*sin(phi_h); ...
    cos(psi_h)*sin(theta_h)*cos(phi_h)+sin(psi_h)*sin(phi_h), ...
        sin(psi_h)*sin(theta_h)*cos(phi_h)-cos(psi_h)*sin(phi_h), ...
        cos(theta_h)*cos(phi_h)] * ...
    [cos(psi_r)*cos(theta_r), sin(psi_r)*cos(theta_r), ...
        -sin(theta_r); ...
    cos(psi_r)*sin(theta_r)*sin(phi_r)-sin(psi_r)*cos(phi_r), ...
        sin(psi_r)*sin(theta_r)*sin(phi_r)+cos(psi_r)*cos(phi_r), ...
        cos(theta_r)*sin(phi_r); ...
    cos(psi_r)*sin(theta_r)*cos(phi_r)+sin(psi_r)*sin(phi_r), ...
        sin(psi_r)*sin(theta_r)*cos(phi_r)-cos(psi_r)*sin(phi_r), ...
        cos(theta_r)*cos(phi_r)];

d_23 = deckEulerMeasDCM(2,3);
d_33 = deckEulerMeasDCM(3,3);
d_13 = deckEulerMeasDCM(1,3);
d_12 = deckEulerMeasDCM(1,2);
d_11 = deckEulerMeasDCM(1,1);

f = [d_23, d_33, d_13, d_12, d_11];
jac5 = jacobian(f,[phi_h theta_h psi_h phi_r theta_r psi_r])

F_1 = atan(D_23/D_33);
F_2 = -asin(D_13);
F_3 = atan(D_12/D_11);  % derivative of atan is same as atan2

F = [F_1, F_2, F_3];
Jac5 = jacobian(F,[D_23 D_33 D_13 D_12 D_11])
```

D.8 Ship Position Measurement

Ship position and attitude measurements are calculated using Equation (5.2.10). This is repeated here for convenience:

$$\tilde{\mathbf{p}}_{ship} = \tilde{\mathbf{p}}_{deck} - \mathbf{DCM}^{T}_{\tilde{\boldsymbol{\theta}}_{deck}} \cdot \mathbf{p}_{ship \to deck}$$

$$\tilde{\boldsymbol{\theta}}_{ship} = \tilde{\boldsymbol{\theta}}_{deck}.$$

Since the ship attitude measurement is equal to the deck attitude measurement, their covariance matrices will be equal too. Therefore, only the ship position covariance matrix needs to be determined here.

The noise sources include deck position and attitude measurement noise:

$$\mathbf{N} = \text{diag}(\sigma^2_{d_1}, \sigma^2_{d_2}, \sigma^2_{d_3}, \sigma^2_{\phi_d}, \sigma^2_{\theta_d}, \sigma^2_{\psi_d}). \tag{D.8.1}$$

The following script was used to calculate the Jacobian:

```
%% partial derivatives for measured ship position at the CG

syms d1 d2 d3 phi theta psi p1 p2 p3

inv_dcm = [cos(psi)*cos(theta), ...
    cos(psi)*sin(theta)*sin(phi)-sin(psi)*cos(phi), ...
    cos(psi)*sin(theta)*cos(phi)+sin(psi)*sin(phi); ...
        sin(psi)*cos(theta), ...
            sin(psi)*sin(theta)*sin(phi)+cos(psi)*cos(phi), ...
            sin(psi)*sin(theta)*cos(phi)-cos(psi)*sin(phi); ...
        -sin(theta), cos(theta)*sin(phi), cos(theta)*cos(phi)];

shipCGPosMeas = [d1;d2;d3] - inv_dcm*[p1;p2;p3];

jac = jacobian(shipCGPosMeas, [d1 d2 d3 phi theta psi])
```

D.9 Vision Sensor Noise

The position and attitude measurement noise covariance matrices are calculated in the same way for the monocular and stereo vision sensors. As a result, $\tilde{\mathbf{p}}$ and $\tilde{\boldsymbol{\theta}}$ will be used here to represent either the monocular or stereo vision position measurement within the sensor reference frame.

Position

Equation (4.2.10) is used to transform monocular and stereo vision position measurements into relative position measurements. It is repeated here for convenience:

$$\tilde{\mathbf{p}}_{rel} = \mathbf{DCM}_{\boldsymbol{\theta}_{offset}}\tilde{\mathbf{p}} + \mathbf{p}_{offset}.$$

The offsets in position and attitude of the sensors can be assumed to be precisely known. As such, they contribute no uncertainty to the relative position measurements. The relative position measurement covariance matrix is therefore composed entirely of vision position measurement noise:

$$\mathbf{N} = \text{diag}(\sigma_{p_1}^2, \sigma_{p_2}^2, \sigma_{p_3}^2). \tag{D.9.1}$$

The Jacobian is therefore simply

$$\begin{aligned}\mathbf{J}_{pos} &= \begin{bmatrix} \frac{\partial \tilde{p}_{rel_1}}{\partial \tilde{p}_1} & \frac{\partial \tilde{p}_{rel_1}}{\partial \tilde{p}_2} & \frac{\partial \tilde{p}_{rel_1}}{\partial \tilde{p}_3} \\ \frac{\partial \tilde{p}_{rel_2}}{\partial \tilde{p}_1} & \frac{\partial \tilde{p}_{rel_2}}{\partial \tilde{p}_2} & \frac{\partial \tilde{p}_{rel_2}}{\partial \tilde{p}_3} \\ \frac{\partial \tilde{p}_{rel_3}}{\partial \tilde{p}_1} & \frac{\partial \tilde{p}_{rel_3}}{\partial \tilde{p}_2} & \frac{\partial \tilde{p}_{rel_3}}{\partial \tilde{p}_3} \end{bmatrix} \\ &= \mathbf{DCM}_{\boldsymbol{\theta}_{offset}}.\end{aligned} \tag{D.9.2}$$

Attitude

Equation (4.2.10) is used to transform monocular and stereo vision attitude measurements into relative position measurements. It is repeated here for convenience:

$$\mathbf{DCM}_{\tilde{\boldsymbol{\theta}}_{rel}} = \mathbf{DCM}_{\boldsymbol{\theta}_{offset}}\mathbf{DCM}_{\tilde{\boldsymbol{\theta}}}.$$

Euler angles are extracted as follows, as shown in (4.2.11):

$$\tilde{\boldsymbol{\theta}}_{rel} = \begin{bmatrix} \arctan2(\tilde{r}_{2,3}, \tilde{r}_{3,3}) \\ -\arcsin(\tilde{r}_{1,3}) \\ \arctan2(\tilde{r}_{1,2}, \tilde{r}_{1,1}) \end{bmatrix}.$$

Only vision attitude measurement noise contributes to the relative attitude estimation uncertainty:

$$\mathbf{N} = \text{diag}(\sigma_{\phi}^2, \sigma_{\theta}^2, \sigma_{\psi}^2). \tag{D.9.3}$$

Equation (4.2.3) shows the calculation of the vision attitude noise covariance matrix:

$$\mathbf{R}_{\tilde{\boldsymbol{\theta}}_{rel}} = \mathbf{J}_{att}\ \mathbf{R}_{\tilde{\boldsymbol{\theta}}}\ \mathbf{J}_{att}^T.$$

The following script was used to calculate the Jacobian:

```
syms r23 r33 r13 r12 r11 phi theta psi a11 a12 a13 a21 a22 a23 a31 ...
    a32 a33

r23 = [a21 a22 a23]*[-sin(theta); cos(theta)*sin(phi); ...
    cos(theta)*cos(phi)];
r33 = [a31 a32 a33]*[-sin(theta); cos(theta)*sin(phi); ...
    cos(theta)*cos(phi)];
r13 = [a11 a12 a13]*[-sin(theta); cos(theta)*sin(phi); ...
    cos(theta)*cos(phi)];
r12 = [a11 a12 a13]*[sin(psi)*cos(theta); ...
    sin(psi)*sin(theta)*sin(phi)+cos(psi)*cos(phi); ...
    sin(psi)*sin(theta)*cos(phi)-cos(psi)*sin(phi)];
r11 = [a11 a12 a13]*[cos(psi)*cos(theta); sin(psi)*cos(theta); ...
    -sin(theta)];

euler_mono = [atan(r23/r33); -asin(r13); atan(r12/r11)];

jac = jacobian(euler_mono,[phi theta psi])
```

Appendix E

Automatic Differentiation

E.1 Overview

There are several methods to compute derivatives of functions. Symbolic differentiation enables analytic derivatives to be calculated. However, it can exhibit exponential complexity. As such, it is not suitable for large, complex functions of many variables.

Numerical differentiation is the second well-known alternative. The standard definition of the derivative is stated as follows:

$$\frac{df(x)}{dx} = \lim_{h \to +0} \frac{f(x+h) - f(x)}{h}. \tag{E.1.1}$$

By taking a small but nonzero value of h, a numerical approximation of the derivative can be calculated. The disadvantage of this technique is that an expression for the derivative is not obtained. More importantly, choosing the value of h is not trivial. If h is too large, the approximation will be inaccurate. This is known as truncation error. If h is too small, round off error results. Floating point numbers have a finite bit size, so have a finite maximum precision [77].

A third method of differentiation exists, known as automatic differentiation (AD). AD involves augmenting every real number x with an abstract number ϵ:

$$x + x'\epsilon, \tag{E.1.2}$$

such that

$$\epsilon^2 = 0. \tag{E.1.3}$$

The product of two augmented variables can be then calculated:

$$\begin{aligned}(x + x'\epsilon) \cdot (y + y'\epsilon) &= xy + xy'\epsilon + x'y\epsilon + x'y'\epsilon \\ &= xy + (xy' + x'y)\epsilon.\end{aligned} \tag{E.1.4}$$

The coefficient of ϵ is the exact derivative according to the product rule.

AD is very useful.It does not return an analytic expression of the derivative. However, it gives the exact derivative to working precision. The algorithmic complexity is only a constant factor of the original expression being derived.

Furthermore, it is perfectly capable of handling any programming statements, such as for loops, conditions, recursion and other constructs. Symbolic differentiation is incapable of this. For this reason, automatic differentiation is sometimes alternatively defined as algorithmic differentiation.

The interested reader should consult [78; 77] for a good introduction to the topic. Several different programming libraries exist to perform AD. Commonly used ones include:

- Ceres Solver (C++)
- Algopy (Python)
- ADOL-C (C++)
- Tapenade

E.2 Verifying the Tilt-Heading Jacobian

Many of the Jacobians calculated in Appendix D happen to be very large and complex. As such it is useful to be able to use an independent method of verifying their accuracy. AD proves very useful for this task. The attitude measurement algorithms are particularly difficult to troubleshoot.

The following Python script can be used to check the tilt-heading Jacobian determined in Appendix D.3.1. The Algopy [79] library is used to compute the derivatives:

```
1  import numpy as np
2  from algopy import *
3
4  def F_tiltheading(x):
5      """ The tilt-heading algorithm.
6          Returns Euler attitude measurements. """
7      y = zeros(3, dtype=x)
8      phi, theta, psi, beta1, beta2, beta3, alpha1, alpha2, alpha3, ...
           Mx, My, Mz = x
9      s_phi, s_theta, s_psi = sin(phi), sin(theta), sin(psi)
10     c_phi, c_theta, c_psi = cos(phi), cos(theta), cos(psi)
11     gx = (cos(psi)*cos(theta))*beta1 + (sin(psi)*cos(theta))*beta2
12         + (-sin(theta))*beta3 - alpha1
```

```
13      gy = (cos(psi)*sin(theta)*sin(phi)-sin(psi)*cos(phi))*beta1
14          + (sin(psi)*sin(theta)*sin(phi)+cos(psi)*cos(phi))*beta2
15          + (cos(theta)*sin(phi))*beta3 - alpha2
16      gz = (cos(psi)*sin(theta)*cos(phi)+sin(psi)*sin(phi))*beta1
17          + (sin(psi)*sin(theta)*cos(phi)-cos(psi)*sin(phi))*beta2
18          + (cos(theta)*cos(phi))*beta3 - alpha3
19
20      phi_meas = arctan(gy/gz)
21      theta_meas = arctan(-gx/sqrt(gy**2 + gz**2))
22      bx = cos(theta_meas)*Mx + sin(phi_meas)*sin(theta_meas)*My
23          + sin(theta_meas)*cos(phi_meas)*Mz
24      by = cos(phi_meas)*My -sin(phi_meas)*Mz
25      BN = 0.093904
26      BE = -0.041366
27
28      y[0] = phi_meas
29      y[1] = theta_meas
30      y[2] = arctan(bx/by) - arctan(BN/BE)
31
32      return y
33
34  def display(mat):
35      print '\n',
36      for i in mat:
37          print '\t',
38          for j in i:
39              print('%.6f,'% j),
40          print '\n',
41
42  def jacobian_autodiff(F, values):
43      """ F = F_tiltheading (for example).
44          values are a list of values.
45
46          Returns the function evaluation of F(values), as well
47          as the Jacobian at (values). """
48      x = UTPM.init_jacobian(values)
49      y = F(x)
50      algopy_jacobian = UTPM.extract_jacobian(y)
51      return y.data[0,0], algopy_jacobian
52
53  if __name__ == '__main__':
54      jacobian_inputs = [0.084248,-0.066721,-0.788913,0.030011,
55                         -0.442617,-0.386065,-0.395853,-0.420594,
56                         -9.698401,0.108862,0.027443,-0.238132]
57      # attitude measurements for current inputs
58      print jacobian_autodiff(F_tiltheading, jacobian_inputs)[0]
59      # Jacobian for current inputs
```

```
60      print display(jacobian_autodiff(F_tiltheading, ...
            jacobian_inputs)[1])
```

Appendix F

Observability of Nonlinear Systems

Consider the following nonlinear system:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) \\ \mathbf{y} &= \mathbf{h}(\mathbf{x}), \end{aligned} \tag{F.0.1}$$

where $\mathbf{x}$ is a column vector of n elements and $\mathbf{h}$ is a column vector of m elements.

Lie derivatives of this system can be calculated as follows for $k = 1, \ldots, n-1$:

$$\begin{aligned} L_f^0\mathbf{h}(\mathbf{x}) &= \mathbf{h}(\mathbf{x}) \\ L_f^k\mathbf{h}(\mathbf{x}) &= \nabla L_f^{k-1}\mathbf{h}(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x}). \end{aligned} \tag{F.0.2}$$

$L_f^1\mathbf{h}(\mathbf{x})$, for example, is calculated as follows:

$$\begin{aligned} L_f^1\mathbf{h}(\mathbf{x}) &= \nabla L_f^0\mathbf{h}(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x}) \\ &= \nabla\mathbf{h}(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x}) \\ &= \sum_{i=1}^{n} \frac{\partial\mathbf{h}(\mathbf{x})}{\partial x_i} \cdot f_i(\mathbf{x}) \\ &= \begin{bmatrix} \frac{\partial h_1(\mathbf{x})}{\partial x_0} & \cdots & \frac{\partial h_1(\mathbf{x})}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_m(\mathbf{x})}{\partial x_0} & \cdots & \frac{\partial h_m(\mathbf{x})}{\partial x_n} \end{bmatrix} \cdot \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{x}) \end{bmatrix}. \end{aligned} \tag{F.0.3}$$

The observability matrix for the nonlinear system is constructed as follows:

$$\mathbf{O}(\mathbf{x}) = \begin{bmatrix} \nabla L_f^0 \mathbf{h}(\mathbf{x}) \\ \nabla L_f^1 \mathbf{h}(\mathbf{x}) \\ \vdots \\ \nabla L_f^{n-1} \mathbf{h}(\mathbf{x}) \end{bmatrix}. \tag{F.0.4}$$

The nonlinear system is locally weakly observable at $\mathbf{x}$ if $\mathbf{O}(\mathbf{x})$ has rank n. The system is locally weakly observable if $\mathbf{O}(\mathbf{x})$ has rank n for all $\mathbf{x}$.

The rank of the observability matrix gives a "yes" or "no" measure of observability of the system. To determine the degree of observability the condition of the observability matrix can be used. The ratio of the maximum and minimum eigenvalues of the observability matrix determine the degree $\delta(\mathbf{x})$ [72; 71]:

$$\delta(\mathbf{x}) = \frac{\left|\lambda_{\max}[\mathbf{O}^T\mathbf{O}, \mathbf{x}]\right|}{|\lambda_{\min}[\mathbf{O}^T\mathbf{O}, \mathbf{x}]|}. \tag{F.0.5}$$

Appendix G

Sensitivity Analysis of Attitude Measurements

Table 7.1 shows the value of perturbations used to simulate the effects of biases. This table is duplicated here for convenience.

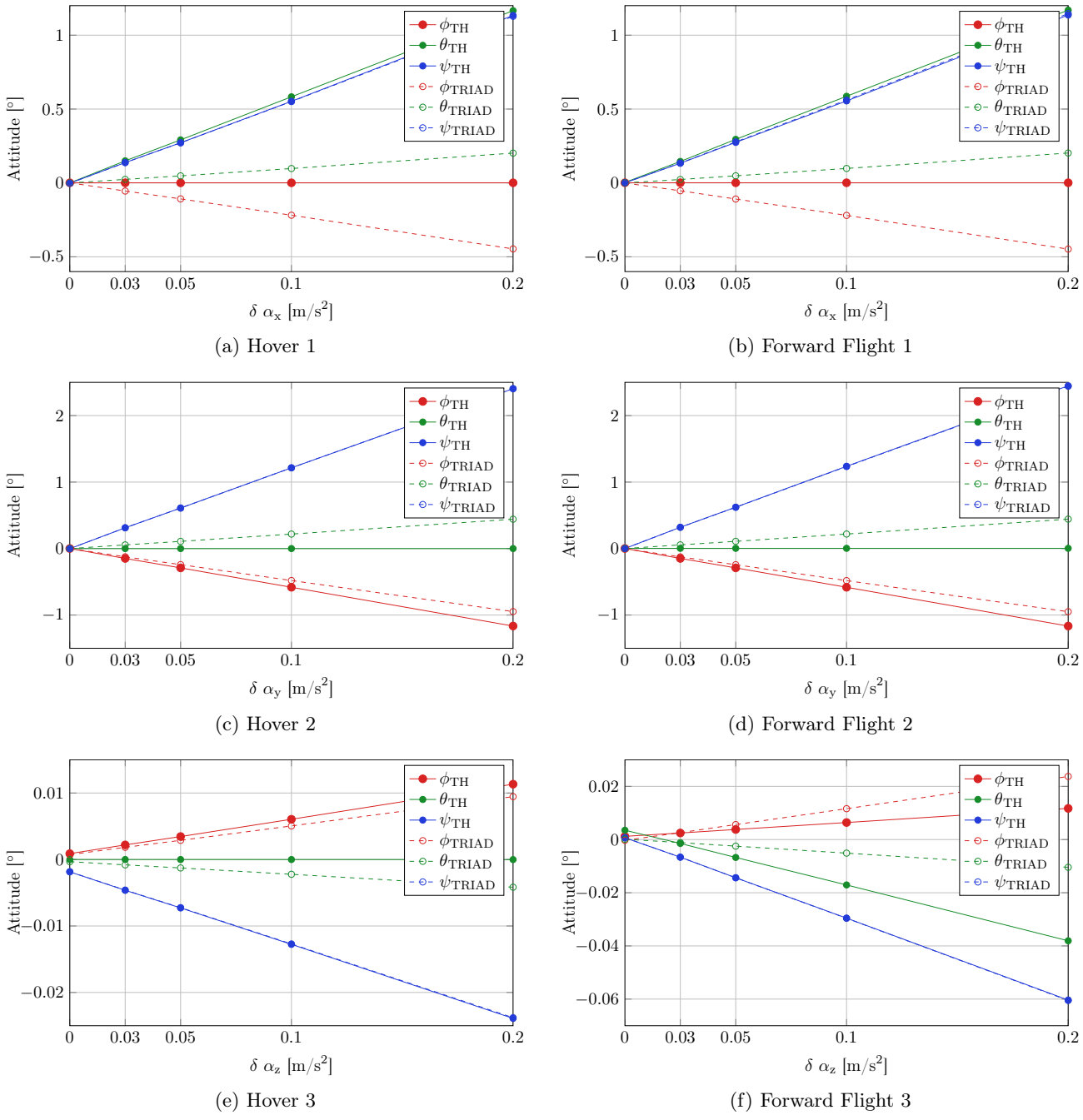
Table G.1: Magnitude of biases tested

Accelerometer	0.1	m/s^2
Magnetometer Rotation	5	$^\circ$
Gravity Reference Vector	0.03	m/s^2
Magnetic Field Reference Vector	7.6×10^{-4}	*Gauss*

The reasons for choosing these values is as follows:

- The accelerometer readings captured over 9 hours exhibit 0.1 m/s^2 drift, as shown in Figure 3.12.
- A nearby magnetic source can easily generate a 5° rotation in the local magnetic field. The magnetic field vector is quite weak, especially in Stellenbosch. Hence it is very susceptible to such disturbances.
- The acceleration due to gravity on the earth's surface ranges from approximately 9.78 m/s^2 at the equator to 9.83 m/s^2 at the poles. Half of this variation is 0.03 m/s^2.
- The gravity bias value is roughly 330 times smaller than the true value of gravity[1]. For comparison of the relative influence of error in the gravitational

[1] $0.03/9.81 \approx 1/330$



(a) Hover 1

(b) Forward Flight 1

(c) Hover 2

(d) Forward Flight 2

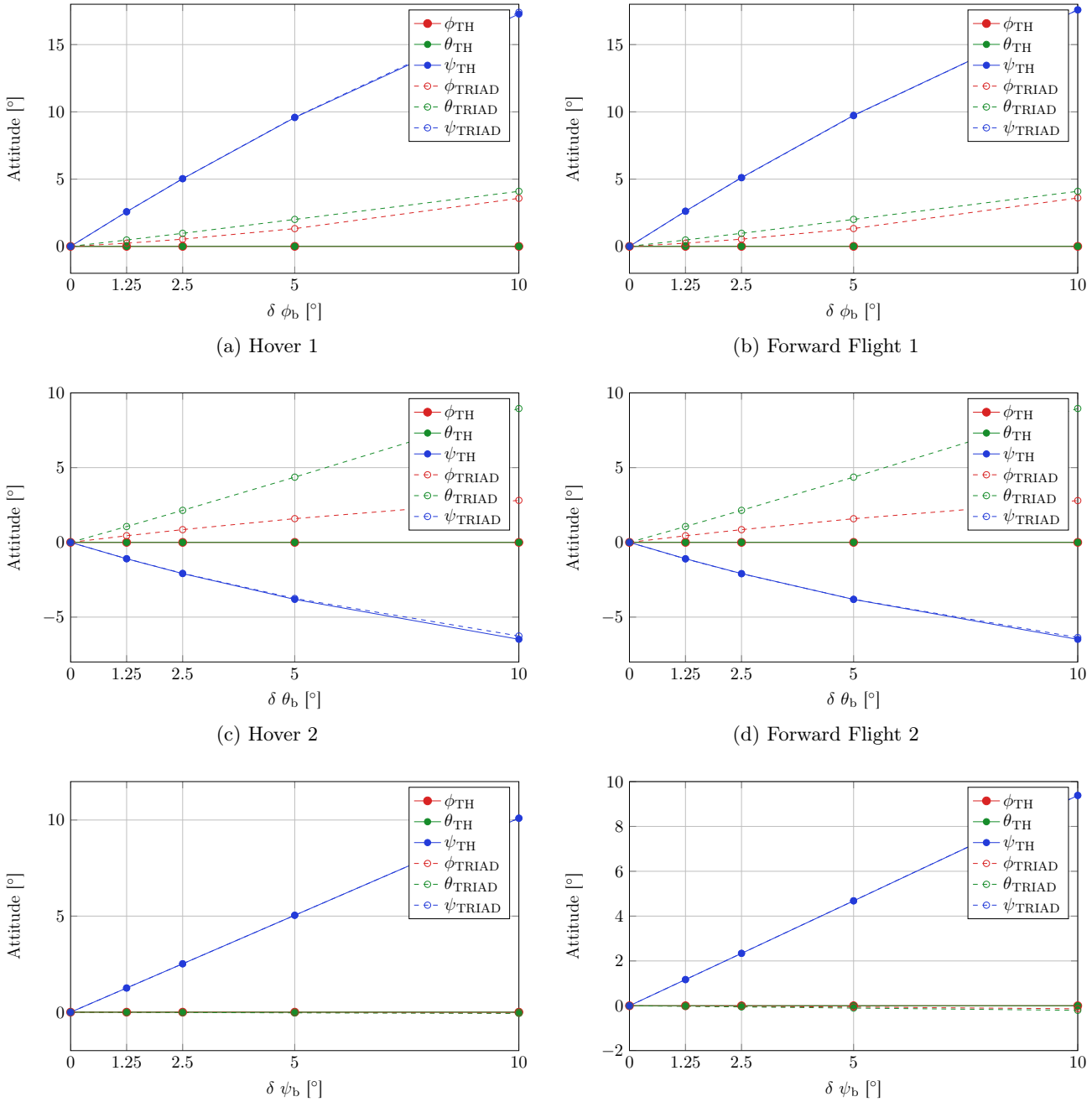
(e) Hover 3

(f) Forward Flight 3

Figure G.1: Effects of bias in accelerometer readings

and magnetic reference vectors, it is useful to ensure both exhibit a similar deviation relative to their respective magnitudes. Therefore $0.25/330 \approx 7.6 \times 10^{-4}$.

The following graphs show the error in attitude estimation as a function of biases.



(a) Hover 1

(b) Forward Flight 1

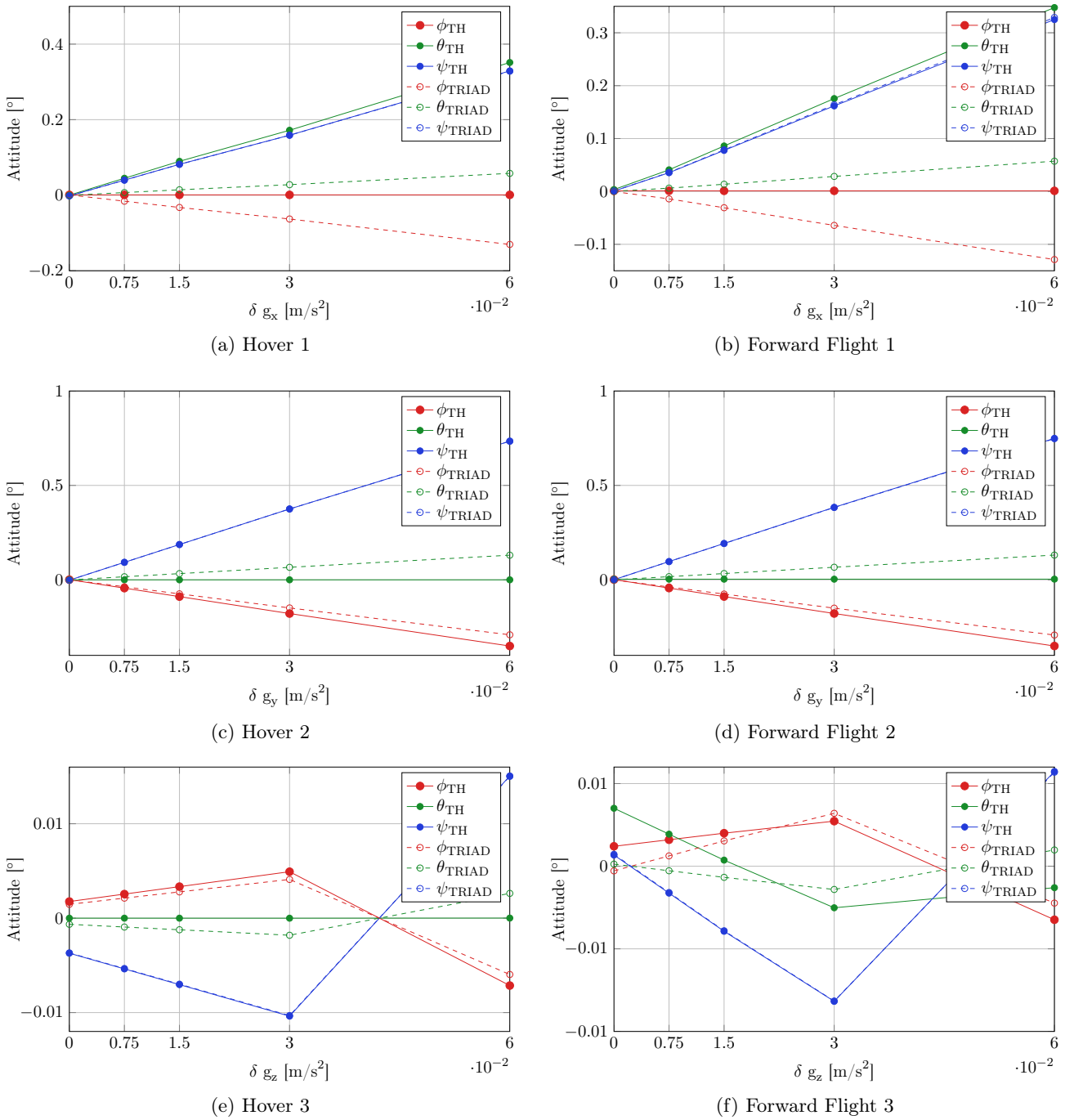
(c) Hover 2

(d) Forward Flight 2

(e) Hover 3

(f) Forward Flight 3

Figure G.2: Effects of bias in magnetic field orientation



(a) Hover 1

(b) Forward Flight 1

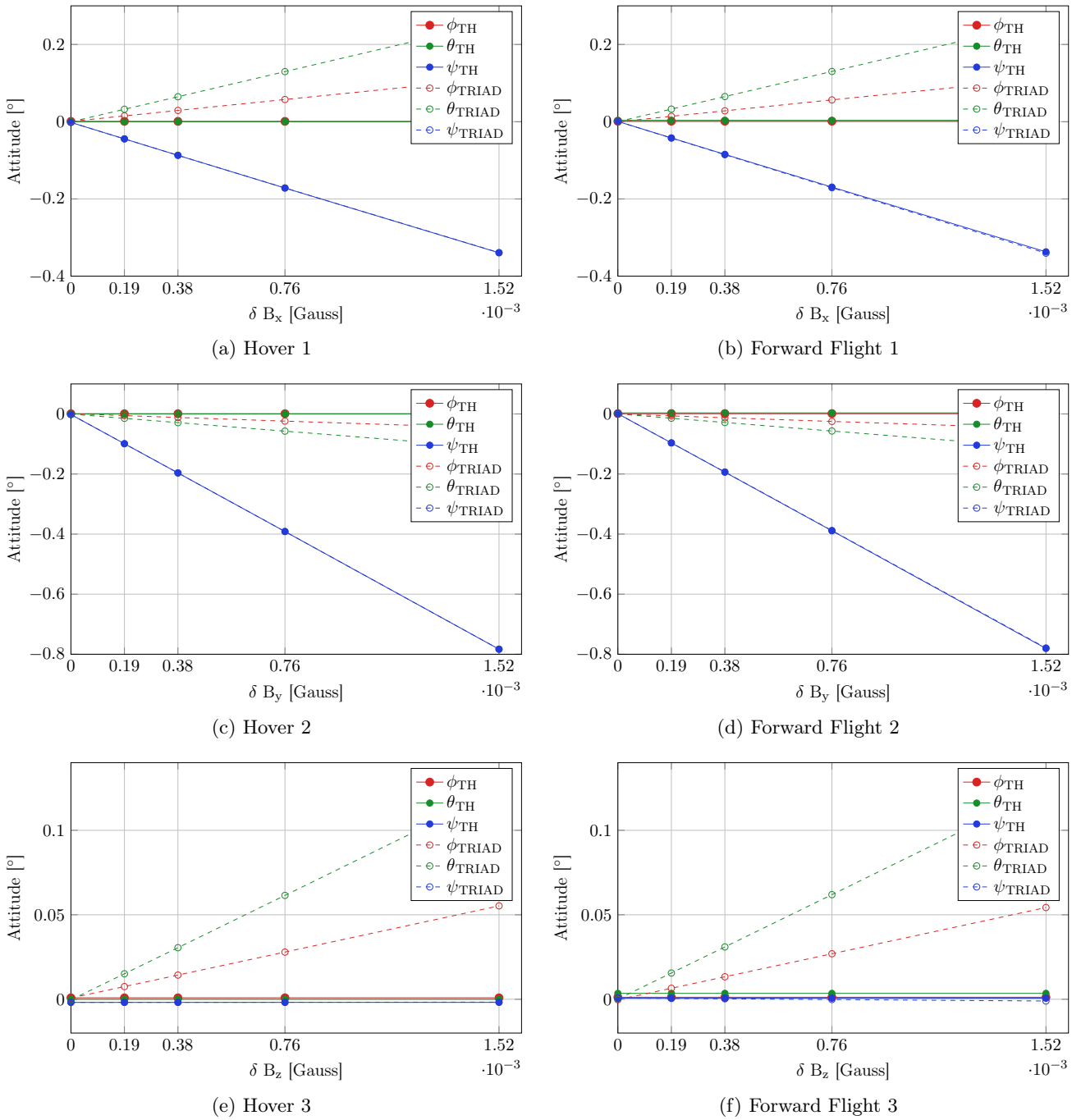
(c) Hover 2

(d) Forward Flight 2

(e) Hover 3

(f) Forward Flight 3

Figure G.3: Effects of bias in magnitude of gravity reference vector



(a) Hover 1

(b) Forward Flight 1

(c) Hover 2

(d) Forward Flight 2

(e) Hover 3

(f) Forward Flight 3

Figure G.4: Effects of bias in magnitude of magnetic field reference vector

Appendix H

Estimation Results

H.1 Helicopter and Ship Trajectories

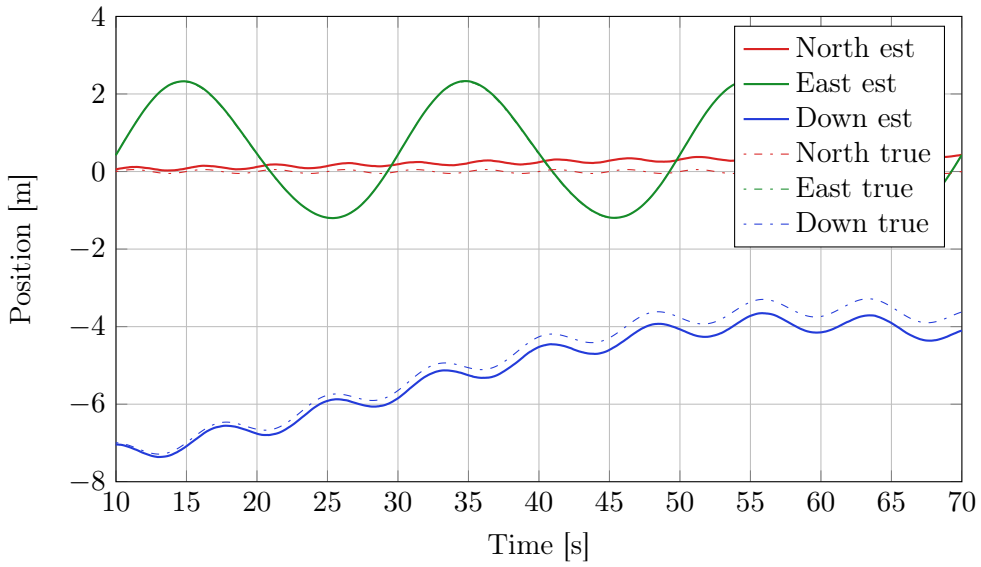


Figure H.1: Helicopter position estimates with Align enabled.

H.2 Performance of Each Sensor Combination

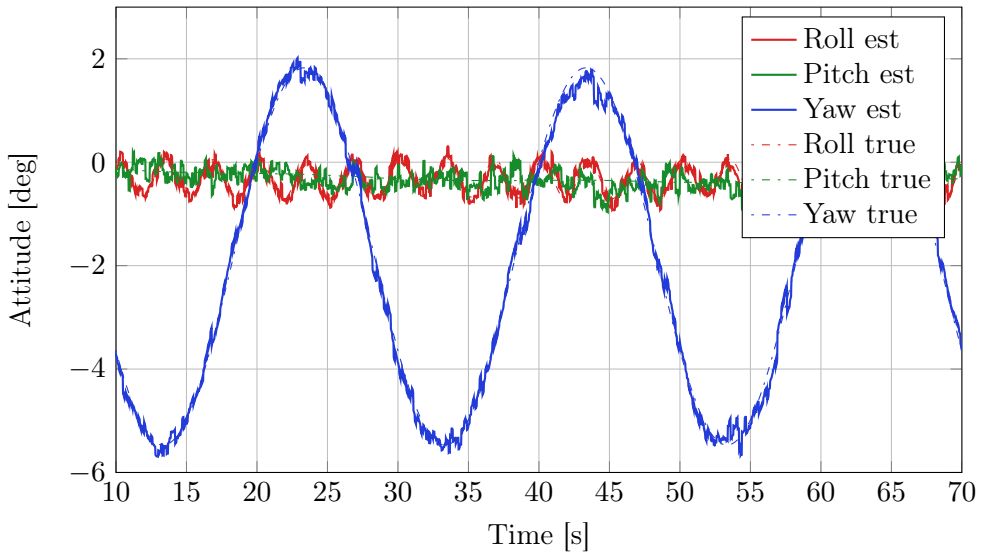


Figure H.2: Helicopter attitude estimates with Align enabled.

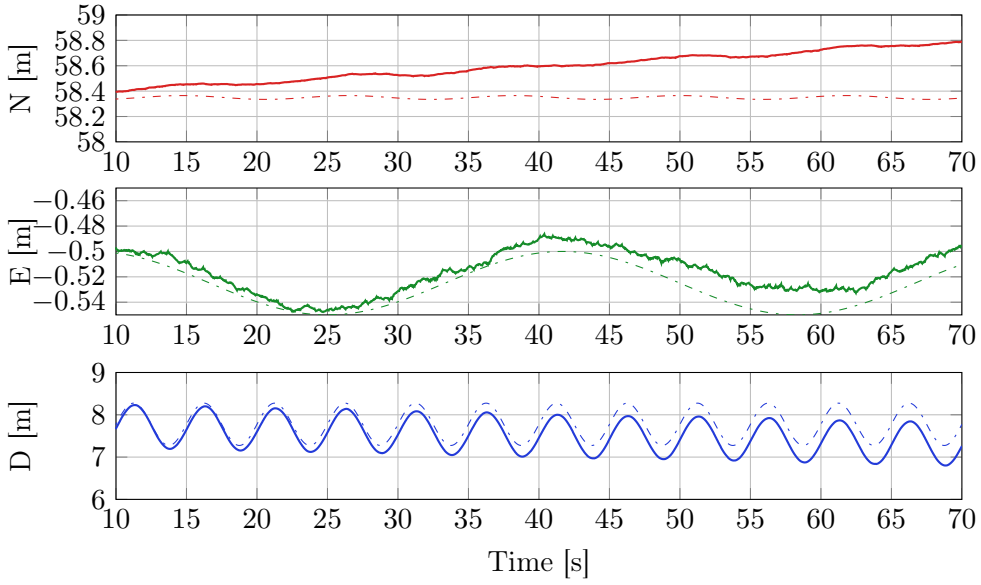


Figure H.3: Ship position estimates with Align enabled.

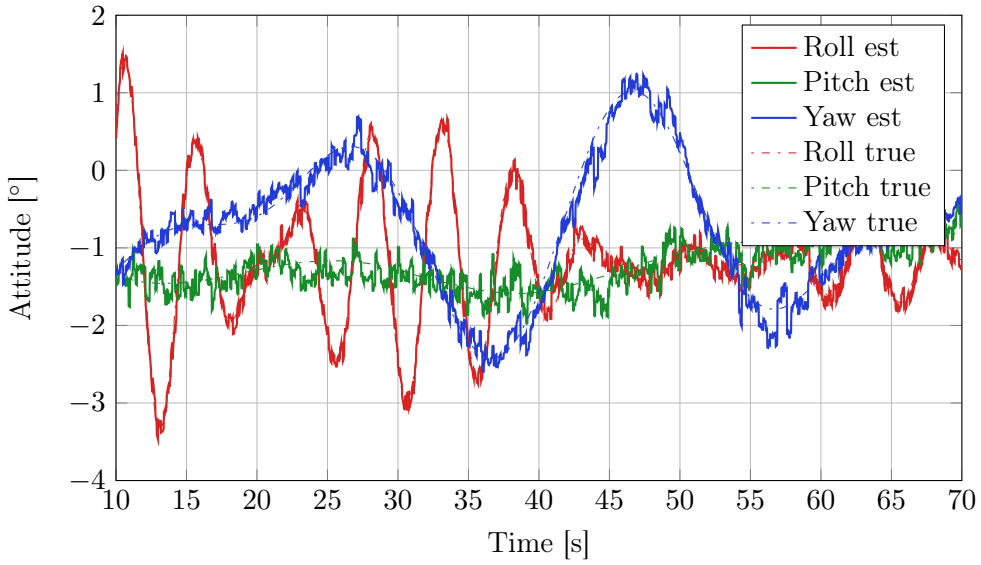


Figure H.4: Ship attitude estimates with Align enabled.

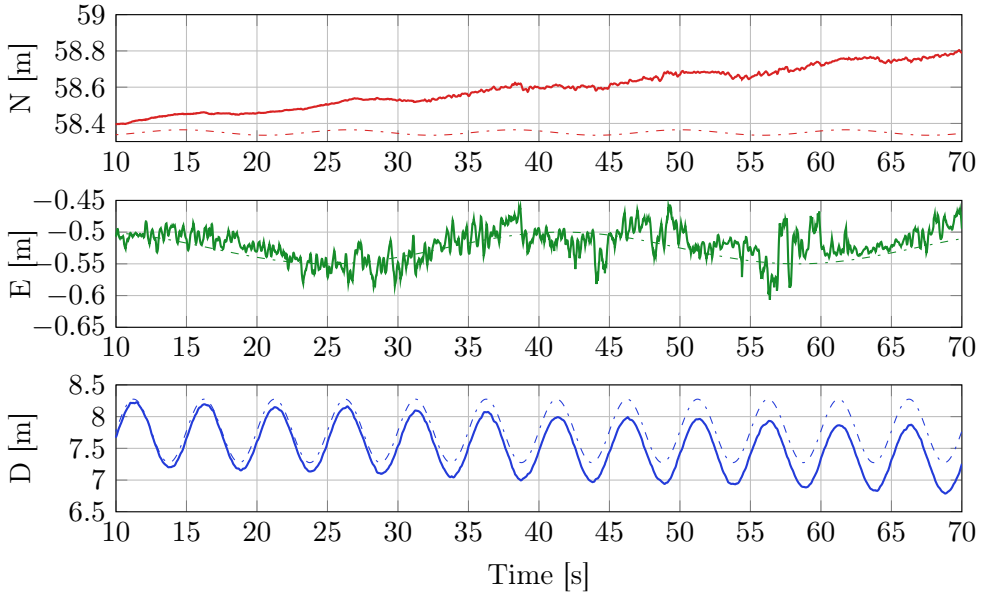


Figure H.5: C1 ship position measurements with Align enabled.

Table H.1: RMS position error for each sensor combination

ID		**M1**	**M2**	**M3**	**M4**	**C1**	**C2**	**C3**	**C4**	**C5**
$\hat{\mathbf{p}}_{rel}$	**N [mm]**	13.4	13.4	-	-	13.4	-	-	-	-
	E [mm]	23.4	23.4	-	-	23.4	-	-	-	-
	D [mm]	33.6	33.6	-	-	33.6	-	-	-	-
$\bar{\mathbf{p}}_{rel}$	**N [mm]**	6.0	13.4	6.0	5.6	6.0	6.0	2.8	5.6	5.6
	E [mm]	17.1	23.4	17.1	17.4	17.1	17.1	21.2	17.4	17.4
	D [mm]	25.8	33.6	25.8	23.6	18.6	20.3	19.6	19.5	19.6
$\bar{\mathbf{p}}_{ship}$	**N [mm]**	7.2	7.5	60.3	60.3	7.5	21.2	19.4	19.0	60.3
	E [mm]	8.6	4.4	211.5	211.5	8.7	204.5	204.0	203.9	211.5
	D [mm]	25.2	25.3	406.1	406.0	25.1	179.1	218.8	176.1	406.3

Table H.2: RMS attitude error for each sensor combination

ID		**M1**	**M2**	**M3**	**M4**	**C1**	**C2**	**C3**	**C4**	**C5**
$\hat{\boldsymbol{\theta}}_{rel}$	$\boldsymbol{\phi}$ **[deg]**	0.047	0.047	-	-	0.047	-	-	-	-
	$\boldsymbol{\theta}$ **[deg]**	0.049	0.049	-	-	0.049	-	-	-	-
	$\boldsymbol{\psi}$ **[deg]**	0.068	0.068	-	-	0.068	-	-	-	-
$\bar{\boldsymbol{\theta}}_{rel}$	$\boldsymbol{\phi}$ **[deg]**	0.049	0.047	0.481	0.481	0.046	0.089	0.038	0.046	0.481
	$\boldsymbol{\theta}$ **[deg]**	0.061	0.049	0.400	0.400	0.046	0.080	0.037	0.047	0.400
	$\boldsymbol{\psi}$ **[deg]**	0.083	0.068	0.108	0.108	0.083	0.108	0.104	0.108	0.108
$\bar{\boldsymbol{\theta}}_{ship}$	$\boldsymbol{\phi}$ **[deg]**	0.117	0.117	0.403	0.403	0.117	0.149	0.259	0.134	0.403
	$\boldsymbol{\theta}$ **[deg]**	0.181	0.181	0.373	0.373	0.181	0.172	0.185	0.169	0.373
	$\boldsymbol{\psi}$ **[deg]**	0.171	0.171	0.199	0.199	0.171	0.199	0.266	0.199	0.199