NTNU
Norwegian University of
Science and Technology

Fault-tolerant Observer Design

Børge Mokleiv

Marine Technology
Submission date: September 2017
Supervisor: Roger Skjetne, IMT

Norwegian University of Science and Technology
Department of Marine Technology

NTNU Trondheim
Norwegian University of Science and Technology
Department of Marine Technology

MSC THESIS DESCRIPTION SHEET

Name of the candidate: Mokleiv, Børge
Field of study: Marine control engineering
Thesis title (Norwegian): Feil-tolerant observer design for lavkost ROV
Thesis title (English): Fault-tolerant observer design for low-cost ROV

Background

For low-cost consumer-grade Remotely Operated Vehicles (ROVs) there is an increased requirement to autonomy and robustness. While professional ROV systems are typically operated by skilled teams and technicians, whom are able to handle faults and unforeseen events, the general buyer of a low-cost ROV does not necessarily possess such skills. Hence, there is an increased demand for reliability and availability in such consumer-grade products, where fault-tolerant algorithms must automatically detect and handle relevant failure modes. Fault diagnosis and fault-handling functions make out a higher level of autonomy, by making the ROV more aware of its internal condition. Hence, this project will investigate how to increase the fault tolerance in the ROV by statistical signal fault-detection and diagnosis by the observer algorithms for state estimation. The aim is to study best practice algorithms for fault diagnosis and how to implement these in the ROV control system.

The candidate will in this master thesis work with BluEye Robotics on one of their new ROV prototypes, to identify the model and its parameters, and use this to develop relevant state observer(s) with fault-tolerance to sensor disturbances and failures.

Scope of Work

1) Perform a background and literature review to provide information and relevant references on:
   a) Lowcost underwater drones.
   b) Sensor suites and fault-tolerance:
      - Relevant sensor suite for the ROVs of BluEye Robotics.
      - Typical failure modes (FMs) on sensor measurements; in particular, wildpoints/outliers, bias, high noise/variance, loss of signal/signal outage, signal freeze, and signal drift.
      - How to model the relevant FMs, either as stochastic signal sequence or as additive faults on the dynamic design estimation model.
      - Relevant fault diagnosis methods for detection of FMs on ROV sensor signals.
   c) Navigation filter/observer and sensor fusion for ROVs:
      - Kalman filter and Extended Kalman filter – incl. how to design the covariance matrices.
      - Sensor fusion between absolute hydro acoustic position data, relative position data from e.g. a camera observer system (typically using feature tracking), and IMU, gyro and pressure/depth measurements.

   Write a list with abbreviations and definitions of terms, explaining relevant concepts related to the literature study and project assignment.

2) Decide on a BluEye ROV platform as design case, and provide a simulator platform with ROS in the loop to simulate the relevant sensor suite, sensor disturbances, and sensor failure modes.

3) Estimate the model parameters of the Blueye ROV to use for a model-based observer module:
   - Damping parameters from towing test in MC-lab.
   - Mass and added mass.
   - Thrust forces and allocation module.

4) Design an EKF model-based observer module for the BluEye ROV with robustness to disturbances and fault-tolerance to:
   - Wildpoint/outlier.
   - High noise/variance.
   - Loss of signal/signal outage and signal freeze.
   - Sudden bias or signal drift.

5) Sensor fusion between HPR (or posref system), IMU, depth/pressure, and compass. Map the increased robustness by sensor fusion and potential common errors that might happen at the same time if any.

6) Simulate the observer for all failure modes and quantify robustness.

7) Test observer in MC-lab with the Blueye ROV for all failure modes and assess performance and robustness.

Tentatively:

8) With a camera-based position reference system, e.g. by monocular SLAM, include this as part of the sensor fusion in the observer design.

Specifications

The scope of work may prove to be larger than initially anticipated. By the approval from the supervisor, described topics that may be deleted or reduced in extent without consequences with regard to grading.

The candidate shall present personal contribution to the resolution of problems within the scope of work. Theories and conclusions should be based on mathematical derivations and logic reasoning identifying the various steps in the deduction.

The report shall be organized in a logical structure to give a clear exposition of background, results, assessments, and conclusions. The text should be brief and to the point, with a clear language. Rigorous mathematical deductions and illustrating figures are preferred over lengthy textual descriptions. The report shall have font size 11 pts. It shall be written in English (preferably US) and contain the following elements: Title page, abstract, acknowledgements, thesis specification, list of symbols and acronyms, table of contents, introduction with objective, background, and scope and delimitations, main body with problem formulations, derivations/developments and results, conclusions with recommendations for further work, references, and optional appendices. All figures, tables, and equations shall be numerated. The original contribution of the candidate and material taken from other sources shall be clearly identified. Work from other sources shall be properly acknowledged using quotations and a Harvard citation style (e.g. *natbib* Latex package). The work is expected to be conducted in an honest and ethical manner, without any sort of plagiarism and misconduct. Such practice is taken very seriously by the university and will have consequences. NTNU can use the results freely in research and teaching by proper referencing, unless otherwise agreed upon.

The thesis shall be submitted with a printed and electronic copy to the main supervisor, with the printed copy signed by the candidate. The final revised version of this thesis description must be included. The report must be submitted according to NTNU procedures. Computer code, pictures, videos, data series, and a PDF version of the report shall be included electronically with all submitted versions.

Start date: 15 January, 2017 **Due date:** As specified by the administration.
Supervisor: Roger Skjetne
Co-advisor(s): Petter Norgren, Andreas V. Henriksen, Borja Serra.

Trondheim,

Roger Skjetne
Supervisor

Preface

This report concludes the Master's Thesis in Marine Technology at the University of Science and Technology carried out during the spring of 2017. The thesis is a collaboration with BluEye Robotics, where I have burrowed and used equipment provided by them. The reader of this report is assumed to have a background in control engineering and have an understanding of statistics.

I would like to thank my supervisor Roger Skjetne for help with this project. Further I would like to thank BluEye Robotics for the experience of working with a newly developed ROV, and my contact person there Andreas Viggen for the help he has granted during the process. PhD candidate Stian Sandøy has also been of great help towards some parts of my project. I would also like to thank Kristian Minde for the help in the construction of mounting equipment used in MC-lab, as well as Carina Norvik who let me use a rig that was constructed previously for her experiments. Lastly I would like to thank my fellow students Tore Mo-Bjørkelund, Erlend Harbitz, Mats Håkon Follestad and Mats Skinderhaug for good discussions and problem solving in the MC-lab.

....

Børge Mokleiv Date

Sammendrag

Denne masteroppgaven omhandlet utviklingen av et feiltolerant modellbasert navigasjonssystem for en lavpris ROV utviklet av BluEye Robotitcs, P2-Alpha. Navigasjonssystemet ble utviklet for fire frihetsgrader ved hjelp de tilgjengelige sensorene på P2-Alpha. Disse bestod av en treghetsmåler (IMU), en trykksensor og et magnetisk kompass. I tillegg til dette ble et hydroakustisk posisjoneringssystem (HPR) simulert som en tiltenkt ekstra sensor. Navigasjonssytemet var basert på et Extended Kalman-filter som tok i bruk sensofusjon og dead-reckoning ved hjelp av de medfølgte egenskapene i Kalman Filteret. Modellparametrene som ble brukt i navigasjonssystemet, ble bestemt ved eksperimentell testing i Marine Cybernetics Lab (MC-lab), samt thrusteregenskapene. Statistiske metoder for modellering av sensorfeil, samt deteksjon og diagnose av feilene ble undersøkt.En signalbehandlingsmodul ble utviklet for forhåndsbehandlingen av sensormålingene som kom inn i Kalman-filteret, og komplementære Kalman-filtre ble designet for feildeteksjon og isolasjon ved bruk av kumulativ sum testing (CUSUM-test). Navigasjonssystemet ble testet og verifisert gjennom simuleringer, og viser god ytelse og feiltoleranse mot outliers, signalfrys og høy støy. Sensor bias deteksjon ved bruk av CUSUM-testing ble konseptuelt verifisert gjennom simuleringer. Til slutt ble navigasjonssystemet testet eksperimentelt i MC-laboratoriet, og viser tilfredsstillende resultater for estimering av de fire frihetsgradene og en grad av feiltoleranse mot outliers, signalfrys og høy støy på alle sensormålinger.

Abstract

This master thesis dealt with the development of a fault-tolerant model-based observer for a low-cost ROV developed by BluEye Robotitcs, P2-Alpha. The observer was developed for 4 degrees of freedom (DOFs) using the installed sensor suite on P2-Alpha consisting of an Inertial Measurement Unit (IMU), a pressure sensor and a compass. In addition to this a hydro-acoustic positioning system (HPR) was added to the sensor suite by simulation. With this available sensor suite, a model-based Extended Kalman filter was used for sensor fusion towards fault-tolerance. The model parameters used in the observer was determined through experimental testing in the Marine Cybernetics Lab (MC-lab), as well as the thruster characteristics. Statistical methods for signal fault modelling with detection and diagnosis by the observer algorithms for state estimation was investigated. A signal processing module was developed for preprocessing the measurements entering the Kalman Filter, and additional Kalman Filters were designed for fault-detection and isolation in conjunction with the cumulative sum test(CUSUM). The observer was tested and verified through simulations, showing good performance and fault-tolerance towards outliers, signal freeze and high variance. Sensor bias detection using the CUSUM test was conceptually verified. Lastly the observer was tested experimentally in the MC-lab, showing satisfying results for state-estimation and a degree of fault-tolerance towards outliers, signal freeze and high variance on all sensor measurements.

Contents

List of Figures

List of Tables

Acronyms

ARL - Average run length
CUSUM - Cumulative sum
CAD - Computer aided design
CFD - Computational Fluid Dynamics
DOF - Degree of freedom
IMU - Inertial measurement unit
HPR - Hydro-acoustic positioning reference
GNSS - Global Navigation Satellite System
ROV - Remotely operated vehicle
ROS - The Robot Operating System
GRL - Generalized likelihood ratio
pdf - probability density function
MC-lab - Marine Cybernetics laboratory
MEMS - Microelectromechanical systems
MMAE - Multiple Model Adaptive Estimation
INS - Inertial Navigation System

Chapter 1

Introduction

1.1 Background and motivation

As the market for low-cost consumer-grade Remotely Operated Vehicles (ROVs) is growing, there is an increased requirement to autonomy and robustness. There are major differences between professional ROV systems and consumer-grade systems, both in equipment, costs and the user. A professional ROV system is typically operated by skilled teams and technicians, whom are able to handle faults and unforeseen events. Furthermore a professional system has general demands to robustness depending on the operational tasks and environment, requiring installation of high-grade sensors and redundancy. On the other hand, a consumer-grade ROV has a higher focus on user experience and cost. Although sensor technology has improved a lot, and decent ROV sensors are now available for low prices, the difference in quality is still very large. Additionally for low-cost consumer-grade ROVs the concept of sensor redundancy just for the sake of fault-tolerance is a low priority. This thesis aims to develop an observer module for such a low-cost ROV, and achieve fault-tolerance for sensor failures without or with restrictive sensor redundancy. This thesis was done in cooperation with BluEye Robotics, who provided a newly developed ROV to use in the development and testing of the observer module.

1.2 Objective

We consider a low-cost ROV equipped with the following sensors

- 9 DOF Inertial Measurement Unit
- Magnetic compass
- Depth sensor
- Hydro-acoustic relative positioning

where all sensors can have the possible sensor faults to different degrees

- Wildpoints/Outliers
- Signal Freeze
- Drop out
- High noise
- Sensor bias and drift

The objective is to make a navigation filter/observer module with fault tolerance towards these sensor faults. To this end, a model-based observer is to be developed.

1.3 Scope and Delimitations

The scope of work for this thesis is detailed in the thesis description sheet in the front of the report. Towards the end of a model-based observer, a ROV vessel model has to be determined. The model parameters will be decided through experiments using the available equipment in the Marine Cybernetics Lab (MC-lab). After the ROV model has been determined, this will be used in an observer implementing the Extended Kalman Filter in 4 degrees of freedom (DOF). The observer will be developed and tested through simulations, making it necessary to provide or obtain a suitable simulation platform. The simulation platform must be able to simulate the ROV and its sensor suite. In addition to this it must be able to simulate the relevant sensor faults. Lastly the observer should be tested experimentally in MC-lab, subjected to some or all the sensor faults either through simulations of said faults on top of real data or real sensor faults motivated through manipulation of known pit-falls.

1.4 Organization of thesis

The thesis starts with preliminaries covering relevant theory used in this thesis. Then in Chapter 3 the experimental platform is introduced, and Chapter 4 introduces the simulator platform. The next chapter covers parameter estimation of the ROV vessel parameters to be used in the observer, as well as the thrust allocation. After this, the sensor suite and their variances are determined. After the parameters for sensors and ROV model are determined, the observer design is discussed. Then the observer is tested for 3 cases in a simulation study, and finally 3 cases with experimental testing results.

1.5 Literature study

Important literature used in this thesis concerning the modelling of underwater vehicles and marine control systems are found in Fossen (2011) and Sørensen (2013). The ROV model for all degrees of freedom, including explanations and simplifications of the relevant hydrodynamic parameters and thrust allocation modules in relation to marine vessels. Several observer designs are shown, with implementation aspects and discussions around tuning parameters. Experimental parameter estimation for the hydrodynamic parameters and thruster characteristics for the ROV models are discussed and implemented with useful comments and experiences from MC-lab in Sandøy (2016) and Eidsvik (2015), whose methods was used in this thesis for parameter estimation. Further documentation more specific and in depth concerning statistical sensor fusion and Kalman Filter sensor fusion is covered in Gustafsson (2012) with many relevant fusion schemes either through decentralized filter banks or through each filter. Implementation examples following in Gustafsson and Gustafsson (2000),Gustafsson (2001), Gustafsson (2003). A specific focus on signal modelling and applied Kalman Filtering is discussed in Brown and Hwang (2012), where much of the basic statistical theory talked about in this thesis is found, as well as discussions around the Kalman Tuning parameters. Blanke et al. (2006) covers much around the subject of fault-diagnosis and fault-tolerant control, including stochastic modelling of sensor faults and detection algorithms concerning this, including the CUSUM algorithm as implemented in this thesis, and residual generation for fault-detection from Kalman Filters. Hassani et al. (2010) covers an implementation of multiple model adaptive estima-

tion using Kalman Filter banks (MMAE), a concept revolving decentralized filtering with weighing of state estimates based on the error covariance matrix calculated in the Kalman Filter, which was originally looked into. For HPR and DVL modelling with implementation for fault-tolerant state estimation using particle filtering is covered in Zhao et al. (2012), as well as a unified framework for fault-detection and diagnosis using the particle filter where in particular the sensor fault modelling used is essential for this thesis covered in Zhao and Skjetne (2014). Blain et al. (2003) also concerns HPR modelling and DVL measurements for fault-tolerance, but now implemented in a Kalman Filter. Candeloro et al. (2012) proposes a set of model-based observers for the ROV Minerva, which are good guidelines showing different strengths of the different observers including the Extended Kalman Filter, Sectorial Kalman Filter, Adaptive Kalman Filter and a Nonlinear Passive Observer. Another similar example that was followed is Dukan et al. (2011) for a sectorial Kalman Filter. Mahony et al. (2008) covers IMU modelling, while Bryne (2013) covers fault-tolerant observer design between IMU and GNSS which can be compared to the methodology of IMU and HPR used in this thesis.

Chapter 2

Preliminaries

This chapter contains the theoretical basis of the thesis. It will introduce the vessel motion dynamics and notation used. Kalman filtering will be summarized with its use for fault-detection together with fault-detection algorithms. In context with fault-tolerance and the modelling of uncertainties the most relevant statistical theory will be introduced, together with modelling of relevant sensor faults.

2.1 Stochastic processes

All sensors are affected by sensor noise to some degree. To model noiselike behavior or random signals, mathematical models of stochastic processes are needed. It is assumed the reader knows basic statistics, but the most important elements that are to be used in this thesis will be covered here. Relevant sources for the modelling of sensor faults are Blanke et al. (2006), Zhao et al. (2014), Thrun (2005), Brown and Hwang (1997), Zhao and Skjetne (2014)

2.1.1 Probability basics

A probability density function f_X describes the relative likelihood for a random variable to take on a given value. The function itself is abstract, but the integral of the function gives the probability of a random variable

to take on a value between the integrated limits. Such that the probability that a random variable X takes on the a value between θ_1 and θ_2 is

$$p(\theta_1 < X < \theta_2) = \int_{\theta_1}^{\theta_2} f_X(\theta) d\theta. \tag{2.1}$$

The expected value $E(X)$, or the expectation of X is defined as

$$E(X) = \int_{-\infty}^{\infty} x f_X(x). \tag{2.2}$$

The variance σ_X^2, is a measure of dispersion of a random variable about its expected value given as

$$\sigma_X^2 = E[(X - E(X))^2] = E(X^2) - (E(X))^2. \tag{2.3}$$

While the covariance σ_{XY}^2 is a measure of the degree of correlation between two random variables X and Y,

$$\sigma_{XY}^2 = E[(X - E(X))(Y - E(Y))]. \tag{2.4}$$

The correlation coefficient σ_{XY} is a normalized measure of the degree of correlation given as

$$\sigma_{XY} = \frac{\sigma_{XY}}{\sqrt{\sigma_X}\sqrt{\sigma_Y}}. \tag{2.5}$$

By using the above definition, we can define a covariance matrix $\mathbf{Q}$ as

$$\mathbf{Q} = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \sigma_{13}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \sigma_{23}^2 \\ \sigma_{31}^2 & \sigma_{32}^2 & \sigma_{33}^2 \end{bmatrix} \tag{2.6}$$

2.1.2 Gaussian white noise

White noise is defined as a stationary random process having a constant spectral density function. In reality this is impossible. The central limit theorem of statistics states that a superposition of independent random variables always tend towards a normal distribution Brown and Hwang (1997). Under the assumption that that the noisy part of a signal can be described as independent random variables, we can then model sensor noise or process noise as random variables picked from a normal distribution. For practicality, white noise can then be considered as random variables picked from a normal distribution with zero mean. This approximation will be used equivalently with white noise in this thesis. The normal distribution, also called *Gaussian* distribution has a specific probability density function on the form

$$f_X = \frac{1}{\sqrt{2\pi}\sigma} exp\left[-\frac{1}{2\sigma^2}(x-\mu)^2\right] \tag{2.7}$$

where σ^2 is the variance and μ is the mean. As the distribution is only dependent on μ and σ^2 the following notation

$$x \sim \mathcal{N}(\mu, \sigma^2) \tag{2.8}$$

is used for a random variable x picked from a normal distribution $\mathcal{N}(\mu, \sigma^2)$. By choosing the mean μ to be zero, we have *Gaussian white noise.*

2.1.3 Wiener or Brownian-motion process

A Wiener or Brownian-motion process is a random walk process, where each step taken is random. The continuous analog of a random-walk process is the output of an integrator driven with random inputs. If the random input for the integrator is *Gaussian white noise* the resulting process is a Wiener or Brownian-motion process. This can be defined as the differential equation

$$\dot{b} = w \tag{2.9}$$

where w is Gaussian white noise. As the input is a Gaussian process and integration is a linear operation this means that the process itself will stay Gaussian. So the mean of the process remains zero over long periods of time.

2.1.4 Gauss-Markov process

A Markov process is a statistical process that only depends on its previous state. All information before the previous state is irrelevant. By adding Gaussian white noise as an additional input you have a Gauss-Markov process. A first order Gauss-Markov process can be defined as the differential equation

$$\dot{b} = -\frac{1}{T}b + w \tag{2.10}$$

where T is a time constant, and w is Gaussian white noise.

2.2 Sensor fault modelling

Given the stochastic processes talked about in the previous section, sensor noise can be modelled as Gaussian white noise. The measurement from a sensor can be considered consisting of a signal part y_{real} and an additive noise part v

$$y_{nominal} = y_{real} + v, \qquad v \sim \mathcal{N}(0, \sigma^2). \tag{2.11}$$

By superposition another way to define this is

$$y_{nominal} \sim \mathcal{N}(y_{real}, \sigma_0^2). \tag{2.12}$$

This will be used as the baseline for a nominal fault-free measurement. If a signal fault occurs, we will examine how the distribution of the measurement will change based on the following sensor failures:

- Wildpoints/Outliers
- Signal Freeze
- Drop out
- High noise
- Sensor sudden bias
- Sensor drift

2.2.1 Wildpoints/Outliers

An outlier is a measurement that deviates far from the expected value, such that the signal to noise ratio is very low and the measurement is considered to give little useful information. Equivalently this means that the variance of a measurement outlier is a lot higher than the variance of a nominal fault free measurement. As the name states, outliers generally come for a single or few measurements at a time before the noise levels are returned to a nominal level. An outlier can be modelled as

$$y_{outlier} = y_{nominal} + v_{outlier}. \tag{2.13}$$

Assuming $v_{outlier}$ to be independent random variables this can also be modelled as variables picked from a normal distribution. Where

$$y_{outlier} \sim \mathcal{N}(y_{real}, \sigma^2_{outlier}) \qquad \sigma^2_{outlier} \gg \sigma^2_0 \tag{2.14}$$

(Sørensen 2013) states that a measurement is normally considered an outlier if the standard deviation σ of the signal is 3-9 times higher than the nominal noise level.

2.2.2 Signal freeze

A signal freeze is a measurement that is equal to the previous measurement, this means the variance of the measurement compared to the previous measurement is zero

$$y_{sigfreeze} \sim \mathcal{N}(y_{k-1}, 0). \tag{2.15}$$

2.2.3 Drop out

How a drop out is manifested as a signal depends on the sensor and the software running it. In this thesis all sensors are run through the ROS software, which only posts new measurements. So for a dropout this will manifest itself the same way as a signal freeze.

2.2.4 High noise

High noise is a fault mode where the noise level is increased over a period of time. As opposed to outliers where the variance is increased for a single or few measurements, and then returned to nominal values for the next set of measurements. Depending on the increase of variance for the high noise signal, this is possible to adapt for

$$y_{highnoise} \sim \mathcal{N}(y_{nominal}, \sigma^2_{highvar}) \tag{2.16}$$

2.2.5 Sudden sensor bias

A sudden sensor bias is a change in the signal, but the noise level remains the same such that a biased measurement will be

$$y_{bias} \sim \mathcal{N}(y_{nominal} + y_{bias}, \sigma_0^2) \tag{2.17}$$

where y_{bias} is a constant.

2.2.6 Sensor drift

A sensor drift is also a change in the signal where the noise level remains the same

$$y_{drift} \sim \mathcal{N}(y_{nominal} + y_{drift}, \sigma_0^2). \tag{2.18}$$

But opposed to a constant bias the sensor drift varies with time. Because of this y_{drift} is often modelled as a first order Gauss-Markov process or a Wiener-motion process.

2.3 Fault tolerance

Fault tolerance is a systems capability of performing its function satisfactory in the presence of a system fault. Alternatively it is a systems capability of avoiding system failure in the presence of a fault. **?** defines a fault in a dynamical system as a deviation of the system structure or the system parameters from the nominal situation. The first task of a fault-tolerant system concerns the detection and identification of existing faults, known as fault diagnosis. Fault diagnosis can further be distinguished into diagnostic steps according to their depth as follows

- **Fault detection** Decide whether or not a fault has occurred, and at which time the fault occurred.
- **Fault isolation** Find the location of the fault.
- **Fault identification and estimation** Identify the fault and estimate its magnitude.

The ultimate goal is fault-tolerance, so the requirement for fault diagnosis depends on the nature of the fault. Blanke et al. (2006) further divides fault tolerance into two subcategories

- **Passive fault tolerance.** The system is designed so it does not have to adapt in the presence of a fault to avoid system failure or degradation of its function. This approach is very similar to the robust approach, where a fault can be considered as uncertainties affecting the system. As the systems inspected in this thesis are all prone to uncertainties in the form of sensor and process noise this, it is natural that faults that cause similar noise should be dealt with naturally.
- **Active fault tolerance.** In the presence of a fault, the system structure or parameters must be changed to maintain achieve its objective satisfactory (i.e estimate the state). Active fault tolerance often has the strongest requirements for fault diagnosis, requiring at least the location of the fault, but also in many cases requiring identifying the fault and its magnitude.

Mode	Stochastic model	Note
Nominal	$y_{nominal} \sim \mathcal{N}(y_{real}, \sigma_0^2)$	
Outlier	$y_{outlier} \sim \mathcal{N}(y_{real}, \sigma_{outlier}^2)$	$\sigma_{outlier}^2 \gg \sigma_0^2$
Signal freeze	$y_{freeze} \sim \mathcal{N}(y_{k-1}, 0)$	
High variance	$y_{highvar} \sim \mathcal{N}(y_{nominal}, \sigma_{highvar}^2)$	
Sensor bias	$y_{bias} \sim \mathcal{N}(y_{real} + y_{bias}, \sigma_0^2)$	$y_{bias} = \text{const}$
Sensor drift	$y_{drift} \sim \mathcal{N}(y_{real} + y_{drift}, \sigma_0^2)$	$y_{drift} = \text{varying}$

Table 2.1: *Stochastic models of sensor failure modes*

2.3.1 Fault detection

By modelling the different sensor failures as described in Section 2.2, the problem of fault detection can be reduced to deciding which distribution it most likely belongs to. As can be seen from the summarized stochastic models in Table 2.3.1, this amounts to either deciding a change in mean or a change in variance from the nominal values y_{real} and σ_0^2. The failure modes that just changes variance can be detected without any estimation of y_{real}, which makes these modes possible to detect using a signal processing module.

Signal freeze can be checked by checking the new measurement versus the previous measurement, while outliers and high variance both can be checked by requiring

$$y_k \in [\bar{y}_k - a\sigma_0, \bar{y}_k + a\sigma_0], \tag{2.19}$$

where σ_0 is the expected standard deviation in the fault-free case, and $\bar{y}$ is the mean of the signal for a time-window. The constant a is typically set in the interval 3-9 for outlier detection Sørensen (2013). For high variance detection a would be set to a smaller threshold. For signal drift or signal bias however, fault detection requires an estimation of y_{real} which must be dealt with either by redundant sensor information or a model.

2.4 Cumulative sum test

The CUSUM (Culumative sum) test is used for detection of a known change by hypothesis testing between the fault-free condition H_0 and the condition with faults H_1. It was used and tested in the pre-project for this thesis, and most of this section is from the pre-project. The CUSUM test compares two probability density functions and their means, and which probability density function a random variable is most likely to belong. This requires that you know or can estimate the probability density function of your process when the process is fault-free, but also that you know or can estimate the probability density function of the process when it has a fault. This test will be used in conjecture with the Kalman Filter for detection of signal bias or drift as explained in Section 2.7.7.

A likelihood function of an observation z is by definition equal to the probability density $p(z)$. So given the log-likelihood function

$$s(z) = ln\frac{p_{\theta_1}(z)}{p_{\theta_0}(z)} \tag{2.20}$$

Where θ is the mean of the distribution. This has the fundamental statistical property (Blanke et al. 2006)

$$E_{\theta_0}(s) = \int_{-\infty}^{\infty} s(z)p_{\theta_0}(z)dz < 0 \tag{2.21}$$

$$E_{\theta_1}(s) = \int_{-\infty}^{\infty} s(z)p_{\theta_1}(z)dz > 0 \tag{2.22}$$

If you then consider the cumulative sum of the log-likelihood function in (2.20)

$$S(k) = \sum_{i=1}^{k} s(z(i)) = \sum_{i=1}^{k} ln\frac{p_{\theta_1}(z)}{p_{\theta_0}(z)}. \tag{2.23}$$

Because of the fundamental statistical properties in (2.21)-(2.22), the cumulative sum will exhibit a negative drift as long as the likelihood of the random variable z is highest for belonging to the probability density function p_{θ_0}. If the random variable is most likely to belong to the pdf p_{θ_1} the cumulative sum will get a positive drift. The CUSUM test decision is expressed as

$$g(k) = S(k) - m(k) = S(k) - \min_{1 \leq j \leq k} S(j). \tag{2.24}$$

As long as the cumulative sum is drifting negatively, $g(k)$ will take on the value of 0. This is described very intuitively in figure 7.3 page 279 in Blanke et al. (2006). When the cumulative sum gets a positive drift, $g(k)$ will have a positive value. It can be shown that (2.24) is identical to

$$g(k) = \max_{1 \leq j \leq k} \sum_{i=j}^{k} ln \frac{p_{\theta_1}(z)}{p_{\theta_0}(z)} = \max_{1 \leq j \leq k} s(z(i)) \tag{2.25}$$

Even though there is a higher likelihood that a random variable belongs to a different pdf, it is not certain. The criteria for detecting a known change is therefore expressed as

$$\begin{array}{llll} \textbf{if} & g(k) \leq h & \textbf{accept} & H_0 \\ \textbf{if} & g(k) > h & \textbf{accept} & H_1 \end{array} \tag{2.26}$$

h is the parameter that decides how far you allow the cumulative sum to drift positively, thus how far you allow the random variable z to most likely belong in the pdf of hypothesis H_1 before you accept that H_1 better represents your system. For implementation of the CUSUM algorithm for detection of a known change, it can be efficient to use its recursive form. The CUSUM decision function can be expressed on recursive form as

$$g(k) = max(0, g(k-1) + s(z(k))) \tag{2.27}$$

2.4.1 Detecting a change in the mean of a Gaussian sequence with CUSUM

This detection is useful as both the signal drift and signal bias failures can be modelled as changes in the mean to the normal distribution of the measurement as suggested in section 2.2. If a data sequence is normally distributed there is a special relationship between the design parameter h

used in the decision logic in (2.26). Based on the mean delay for detection τ_{acc} and the mean time between false alarms T_{acc} this relationship can be found with an approximation of the average run length (ARL) function (Blanke et al. 2006)

$$L(\mu) = \left(exp[-2(\frac{\mu_s h}{\sigma^2} + 1.166\frac{\mu_s}{\sigma})] - 1 + 2(\frac{\mu_s h}{\sigma^2} + 1.166\frac{\mu_s}{\sigma}) \right) \left(\frac{\sigma^2}{2\mu_s^2} \right), \tag{2.28}$$

where

$$\mu_s = \frac{\mu_1 - \mu_0}{2\sigma^2}. \tag{2.29}$$

μ_0 belongs to hypothesis H_0 and μ_1 belongs to hypothesis H_1. The mean time between false alarms and mean delay for detection can be calculated as

$$T_{acc} = L(\mu_s) \tag{2.30}$$

$$\tau_{acc} = L(-\mu_s) \tag{2.31}$$

T_{acc} and τ_{acc} can then be plotted as functions of the design parameter h to find a suitable value for h. In the pre-project of this thesis, a MATLAB script was made for finding h for the detection of a known change in the mean of a normally distributed sequence. The script is found in Appendix C.1.

DOF	Forces/Moments	Linear/Angular velocities	Positions/Angles(euler)
Surge	X	u	x
Sway	Y	v	y
Heave	Z	w	z
Roll	K	p	ϕ
Pitch	M	q	θ
Yaw	N	r	ψ

Table 2.2: *SNAME notation for motion of marine vessels*

DOF	Added mass	Linear damping	Quadric damping	Cubic damping
Surge	$X_{\dot{u}}$	X_u	$X_{\|u\|u}$	X_{uuu}
Sway	$Y_{\dot{v}}$	Y_v	$Y_{\|v\|v}$	Y_{vvv}
Heave	$Z_{\dot{w}}$	Z_w	$Z_{\|w\|w}$	Z_{www}
Roll	$K_{\dot{p}}$	K_p	$K_{\|p\|p}$	K_{ppp}
Pitch	$M_{\dot{q}}$	M_q	$M_{\|q\|q}$	M_{qqq}
Yaw	$N_{\dot{r}}$	N_r	$N_{\|r\|r}$	N_{rrr}

Table 2.3: *SNAME notation for hydrodynamic parameters*

2.5 SNAME notation for marine crafts

The notation introduced in the SNAME (1950) convention is used in this thesis. The notation for motion of marine vessels and used hydrodynamic parameters are summarized for all 6 degrees of freedom (DOF) in tables 2.2 and 2.3.

For a more compact form the vector notations in table 2.4 below are introduced.

Parameter	Total	Linear	Angular
Position and orientation in base-frame	$\boldsymbol{\eta} = \begin{bmatrix} \mathbf{p} \\ \boldsymbol{\Theta} \end{bmatrix}$	$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$	$\boldsymbol{\Theta} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}$
Velocities in body-frame	$\boldsymbol{\nu} = \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix}$	$\mathbf{v} = \begin{bmatrix} u \\ v \\ w \end{bmatrix}$	$\boldsymbol{\omega} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$

Table 2.4: *Vector notation for motion of marine vessels*

2.6 Dynamic equation of motion for ROV

The 6 DOF non-linear dynamic equation of motion for an underwater vehicle is expressed in Fossen (2011) with the following kinematic and kinetic equations, using the SNAME notation.

$$\dot{\boldsymbol{\eta}} = \mathbf{J}(\boldsymbol{\eta})\boldsymbol{\nu} \qquad \textbf{Kinematic equation} \tag{2.32}$$

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} \qquad \textbf{Kinetic equation} \tag{2.33}$$

(2.32) represents the relationship between the vessels velocities in the body-frame and the vessels position in the desired coordinate frame of the position vector $\boldsymbol{\eta}$. The matrix $\mathbf{J}(\boldsymbol{\eta})$ is the transformation matrix between the body frame and the desired base-frame, for instance the NED frame. The kinetic equation in (2.33) is in body-frame and can further be divided into rigid body and hydrodynamic forces

$$\underbrace{\mathbf{M}_{RB}\dot{\boldsymbol{\nu}} + \mathbf{C}_{RB}(\boldsymbol{\nu})\boldsymbol{\nu}}_{\text{Rigid body forces}} + \underbrace{\mathbf{M}_{A}\dot{\boldsymbol{\nu}} + \mathbf{C}_{A}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}_{NL}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}_{L}\boldsymbol{\nu}}_{\text{Hydrodynamic forces}} + \mathbf{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} \tag{2.34}$$

where

- $\mathbf{M} = \mathbf{M}_{RB} + \mathbf{M}_{A}$ is the mass matrix consisting of rigid body mass and added mass.
- $\mathbf{C} = \mathbf{C}_{RB} + \mathbf{C}_{A}$ is the Coriolis forces due to rigid body and added mass effects.
- $\mathbf{D} = \mathbf{D}_{L} + \mathbf{D}_{NL}$ is the linear and non-linear damping forces.
- $\mathbf{g}$ represents the forces due to buoyancy and gravity affecting the system.
- $\boldsymbol{\tau}$ is the remaining outside forces and moments acting on the system including control forces, currents e.g.

2.7 The Extended Kalman Filter

The Kalman Filter is a recursive filter for estimation of a linear or nonlinear dynamic system from a series of noisy measurements. The linear version of the filter gives the optimal solution based on minimizing the least-squares error. The central problem is separating signal from noise, and it can be used for both coloured and white noise. Many different variations of the filter has been developed for non-linear systems such as the *Adaptive Kalman Filter* or the *Sectorial Kalman Filter*. This thesis will focus on *The Extended Kalman Filter*, which is another extension of the Kalman Filter that deal with non-linear systems. One key assumption when designing a Kalman Filter is *observability* which is defined in section 2.7.2. This chapter will focus on the algorithms and implementation aspects of the filter, while not going into detail for the derivations of these algorithms. The reader is referred to Gustafsson (2012) or Brown and Hwang (2012) for detailed derivations of the filter.

2.7.1 The Kalman Filter Algorithm

By transforming a given model, such as the the dynamics for an ROV in (2.32)-(2.33), to state-space system on the form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w} \tag{2.35}$$

where $\mathbf{u}$ is the driving force (gains) of the system. The Kalman Filter can be used to estimate the states $\mathbf{x}$ by comparing the states to a measurement vector $\mathbf{y}$ with the relation

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v}. \tag{2.36}$$

The vectors $\mathbf{w}$ and $\mathbf{v}$ are assumed Gaussian white noise that represent process noise and measurement noise respectively. Given the system (2.35)-(2.36) the Kalman Filter algorithm for state estimation is divided into a corrector step and a predictor step (Fossen 2011) shown in Algorithm 1

In the corrector step, the algorithm updates the current state estimate $\bar{\mathbf{x}}_k$ based on the new measurement data from $\mathbf{y}_k$ and the Kalman gain $\mathbf{K}_k$. The

Algorithm 1 Discrete Kalman filter algorithm

Design matrices

Process noise covariance matrix $$\mathbf{Q}_k = \mathbf{Q}_k^T > 0 \quad (2.37)$$

Sensor noise covariance matrix $$\mathbf{R}_k = \mathbf{R}_k^T > 0 \quad (2.38)$$

Initial conditions $$\bar{\mathbf{P}}_k = \mathbf{P}_0, \quad \bar{\mathbf{x}}_k = \mathbf{x}_0 \quad (2.39)$$

Corrector step

Kalman gain matrix $$\mathbf{K}_k = \bar{\mathbf{P}}_k \mathbf{H}^T (\mathbf{H}\bar{\mathbf{P}}_k \mathbf{H}^T + \mathbf{R})^{-1} \quad (2.40)$$

State estimate update $$\hat{\mathbf{x}}_k = \bar{\mathbf{x}}_k + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}\bar{\mathbf{x}}_k) \quad (2.41)$$

Error covariance update $$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\bar{\mathbf{P}}_k \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}\right)^T + \mathbf{K}_k\mathbf{R}\mathbf{K}_k^T \quad (2.42)$$

Predictor step

State estimate propagation $$\bar{\mathbf{x}}_{k+1} = \mathbf{A}_k\hat{\mathbf{x}}_k + \mathbf{B}\mathbf{u}_k \quad (2.43)$$

Error covariance propagation $$\bar{\mathbf{P}}_{k+1} = \mathbf{A}_k\mathbf{P}_k\mathbf{A}_k^T + \mathbf{E}_k\mathbf{Q}\mathbf{E}_k^T \quad (2.44)$$

Kalman gain $\mathbf{K}_k$ is calculated by minimizing the least-squares error of the state-estimate. Lastly it updates the error covariance matrix $\mathbf{P}_k$ which is a reflection of the confidence the filter has in its estimated states. In the predictor step the algorithm simply propagates the states to the next step, based on the updated states from the corrector step and the gains of the system such as thrusters. The design initial conditions are defined as

$$E(\mathbf{x}_0) = \mathbf{x}_0 \quad (2.45)$$

$$Cov(\mathbf{x}_0) = \mathbf{P}_0. \quad (2.46)$$

If the initial conditions $\mathbf{x}_0$ are unknown or poorly estimated, this can be remedied by setting $\mathbf{P}_0$ to high values so that the filter will converge to the first measurements quickly. This means that the filter will have a transient start-up period before being able to reconstruct unmeasured states.

2.7.2 Observability criterion

Given observability the filter can reconstruct unmeasured states of the state-space model, and it is an underlying assumption when designing a Kalman Filter. Gustafsson (2012) states that the classical definition of observability can be checked by inspecting the rank of the observability matrix

$$\mathcal{O}_n = \begin{bmatrix} H \\ HA \\ HA^2 \\ \vdots \\ HA^{n-1} \end{bmatrix} \tag{2.47}$$

If the observability matrix has full column rank, the system is fully observable.

2.7.3 Design matrices

The design matrices

$$Cov(\mathbf{w}_k) = \mathbf{Q}_k \tag{2.48}$$
$$Cov(\mathbf{v}_k) = \mathbf{R}_k \tag{2.49}$$
$$\tag{2.50}$$

represent how much you trust the model and how much you trust the measurement respectively. The calculation of the Kalman Gain $\mathbf{K}$ is highly dependant on the relationship between these. If $\mathbf{Q} \gg \mathbf{R}$, then $\mathbf{K}$ will be small and vice versa.

$$\mathbf{Q} \gg \mathbf{R} \qquad \rightarrow \mathbf{K} = \mathbf{I} \tag{2.51}$$
$$\mathbf{Q} \ll \mathbf{R} \qquad \rightarrow \mathbf{K} = \mathbf{0} \tag{2.52}$$

The uncertainty in the measurements comes from noise that is assumed to be white Gaussian noise and independent of each other such that $\mathbf{R}_k$ is a diagonal matrix and

$$\mathbf{v}_k \in \mathcal{N}(0, \mathbf{R}_k). \tag{2.53}$$

The values in $\mathbf{R}_k$ can then be decided by generating a dataset of the measurements and calculating the variance. The $\mathbf{Q}_k$ matrix however can be harder to decide as it depends on how uncertain the model is. According to Fossen (2011) it is often chosen to be a diagonal matrix and is tuned ad hoc , generally as a fraction of the set fixed values in $\mathbf{R}_k$. A numerical method for estimation of $\mathbf{Q}$ has been developed in Loan (1978), but it is not utilized in this thesis as it requires estimates of the process noise spectral density.

2.7.4 Handling multiple measurement rates and dead-reckoning

Dead-reckoning is the term used for the Kalman Filter when it updates its estimate without using any information from a sensor. Thus it estimates its states based purely off the observer model. The Kalman Filter has multiple ways to handle different measurement rates and signal-loss. Fossen (2011) suggests modifying the Kalman gain $\mathbf{K}$ or the measurement vector $\mathbf{y}$. The measurements including no new or faulty data can then be ignored in the corrector step by setting the corresponding values in $\mathbf{K}$ to zero. Alternatively for full dead-reckoning the corrector step can be skipped entirely. In this thesis the method of modifying the Kalman gain is used, and the implementation shown in 7.6.

2.7.5 Sensor fusion in the Kalman Filter

Sensor fusion can also be done directly in the Kalman Filter by altering the matrix $\mathbf{H}$. There is no limit to how many redudant sensors that can be fused in this manner, and each sensor will be weighted depending on the values in $\mathbf{R}$. By minimizing the covariance of the error, this also means that a faulty measurement from one of the sensors will be naturally less weighted if the faulty measurement has a higher variance than expected.

2.7.6 Linearisation and discretization

In the case of a non-linear system you have to use the extended kalman filter. The dynamic equations in (2.32)-(2.33) contain nonlinear damping terms

and nonlinear Coriolis forces that depend on the ROV velocity components. In addition to this the rotational matrix in the kinematic equation depend on the attitude. As the matrix $\mathbf{A}_k$ is non-linear , it must be linearised for each loop of the filter. The steps as described in equations 2.40-2.44 are the same, just now with a with the system linearized about the eastimated state for each time step. These can be linearlized by forward euler integration with step h as suggested in Fossen (2011).

$$\bar{\boldsymbol{x}}_{\boldsymbol{k+1}} \approx \hat{\boldsymbol{x}}_{\boldsymbol{k}} + h[f(\hat{\boldsymbol{x}}(k)) + \boldsymbol{B}\boldsymbol{u}(k)] \tag{2.54}$$

$$f(\hat{\boldsymbol{x}}(k)) \approx \boldsymbol{A}_k(\psi_k)\hat{\boldsymbol{x}}_k \tag{2.55}$$

$$\boldsymbol{A}_k \approx \boldsymbol{I} + h\frac{\partial f((\boldsymbol{x}_k), \boldsymbol{u}_k)}{\partial \boldsymbol{x}_k}\bigg|_{\mathbf{x}_k=\hat{\mathbf{x}}_k} \tag{2.56}$$

It should be noted that there are no proofs of global asymptotic stability for the filter when the system is linearised (Sørensen 2013).

2.7.7 Residual generation for fault detection

One quantity of note that can be generated from the Kalman Filter is the residual, also known as the innovation. In it's simplest form it is

$$\mathbf{r}_k = \mathbf{y}_k - \mathbf{H}\hat{\mathbf{x}}_\mathbf{k} = \mathbf{y}_k - \hat{\mathbf{y}}_\mathbf{k}. \tag{2.57}$$

For the generated residuals to work for fault detection, it has two requirements (Blanke et al. 2006):

- **1.** The sequence of output values $\mathbf{r}_k$, $k = 1, 2...$ is a zero mean white noise vector sequence which is not affected by $\mathbf{u}$ or disturbances, once the transient due to initial conditions has vanished.
- **2.** In the presence of a fault ($\mathbf{f} \neq 0 \quad \forall \quad k \geq k_0$) the mean of $\mathbf{r}_k$ is different from zero for at least some $k \geq k_0$.

Remembering the stochastic models of signal failure modes in Table 2.3.1,

$$y_{bias} \sim \mathcal{N}(y_{real} + y_{bias}, \sigma_0^2) \tag{2.58}$$

$$y_{drift} \sim \mathcal{N}(y_{real} + y_{drift}, \sigma_0^2). \tag{2.59}$$

Signal bias and signal drift required an estimate of $\mathbf{y}_{real}$ for fault detection as the variance of the faulty signal did not change. Depending on how good the Kalman filter estimate approximates

$$\hat{\mathbf{y}}_k \approx \mathbf{y}_{real} \tag{2.60}$$

requirement **2** can be met when

$$\mathbf{y}_{drift} \neq 0, \qquad \text{or} \qquad \mathbf{y}_{bias} \neq 0. \tag{2.61}$$

As described in Section 2.4 the CUSUM test can be used to detect a change in mean between two known probability density functions. The residual can be normalized by (Hassani et al. 2010)

$$\mathbf{r}_k^{norm} = \frac{\mathbf{r}_k}{\sqrt{\mathbf{S}_k}} \tag{2.62}$$

$$\tag{2.63}$$

where $\mathbf{S}_k$ is the covariance matrix of the innovation

$$\mathbf{S}_k = \mathbf{H}\bar{\mathbf{P}}_k\mathbf{H}^T + \mathbf{R} \tag{2.64}$$

The normalized residual has the property that its variance is 1 (Gustafsson 2012). If requirement **1** and **2** are met the normalized residual can then be used for detecting signal bias or drift with hypothesis testing using the CUSUM test between

$$H_0 : \qquad r_k^{norm} \sim \mathcal{N}(0, 1) \tag{2.65}$$
$$H_1 : \qquad r_k^{norm} \sim \mathcal{N}(\mu_1, 1) \tag{2.66}$$

where H_0 is the fault-free state, and H_1 is the faulty state state with sensor bias or drift. μ_1 will be sensitivity parameter set to a threshold that depends on how good the Kalman Filter estimates are, thus depending on how good requirement **1** is met.

Chapter 3

Experimental platform

This chapter will give a summary and specifications for the ROV used for experiments, the test basin and specific equipment used.

3.1 BluEye P2-Alpha

The drone that is used in this thesis is the BluEye P2-Alpha. It is an alpha version used in the development of BluEye drones. The drone can be seen attached to its rig in Figures 3.1 and 3.2.

3.1.1 Geometrical properties

The drone's principal geometrical properties are needed for the observer design and found using the CAD software Inventor with the CAD-file for P2-Alpha provided by BluEye Robotics.

3.1.2 Thrusters

BluEye P2-Alpha is equipped with 3 thrusters from BlueRobotics (*BlueRobotics* 2017), 2 surge thrusters and one vertical thruster for heave thrust.

Length	440 mm
Height	327 mm
Width	140 mm
Projected area front	45427 mm^2
Projected area side	128582 mm^2
Projected area top	62300 mm^2
Mass m	7.15 kg
Moment of inertia in yaw I_z	0.06235 kgm^2

Table 3.1: *Geometric properties of P2-Alpha*

	x [cm]	y [cm]	z [cm]
Surge thruster starboard	-8	35	-3
Surge thruster port	-8	-35	-3
Heave thruster	0	0	0

Table 3.2: *Thruster position relative to center of gravity*

The surge thrusters are equipped with one clock-wise and one counter-clockwise propeller to counter torque. The position of the thrusters relative to center of gravity is shown in table 3.2.

3.1.3 P2-Alpha Sensor suite

The available sensors that are installed on P2 Alpha are

- *SparkFun 9DoF IMU Breakout - LSM9DS1* (2017). A 9 DOF Inertial Measurement Unit, with a 3-axis accelerometer, 3-axis gyroscope, and 3-axis magnetometer
- Depth sensor measuring pressure
- Magnetic compass
- Camera

Of these, the IMU, pressure sensor and compass will be used for the observer. They will be further discussed in Chapter 6.

3.1.3.1 Hardware and software

To power the sensors and thrusters the BluEye P2 uses an Arduino Mega. A Raspberry Pi 3 is used for communication through an ethernet cable to a topside transmitter, this transmitter can then be used to communicate with a topside computer or phone through wifi. The open software ROS is used for communication. Although the student was given full access to the ROS code running the drone, the code was treated as a black box in that no code was changed during the thesis, only using the existing outputs.

3.2 Test basin

The test basin in this thesis is the Marin Cybernetics Lab. It consists of a small wave basin, used by students for experimental testing. It also has a towing carriage capable of precise movement in six degrees of freedom, and a special motion capture system for the lab which will be discussed further in section 3.3. This thesis has taken use of the towing carriage for towing tests and the Qualisys motion capture system as a sensor for the ROV. Further information on MC-lab can be found in Appendix E.

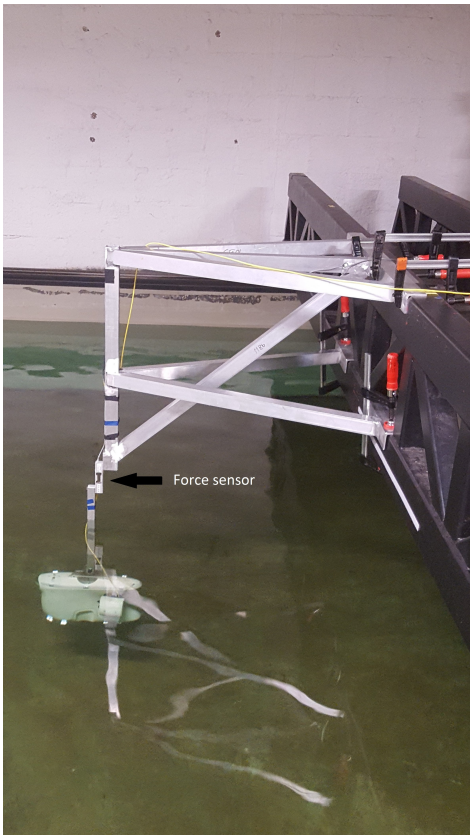
3.3 Qualisys motion capture system

The Qualisys motion capture system is based on several cameras that can identify a vessels position based on reflective balls that are mounted on the vessels. This system is not representative of any reliable system used in real world applications of marine vessels, but due to its high accuracy it is a good system for verifying control systems as it can be used as a baseline for the true position and orientation of the vessel. In this thesis the Qualisys will be used to simulate a hydro-acoustic positioning system. Further information on Qualisys can be found in Appendix E.0.1.

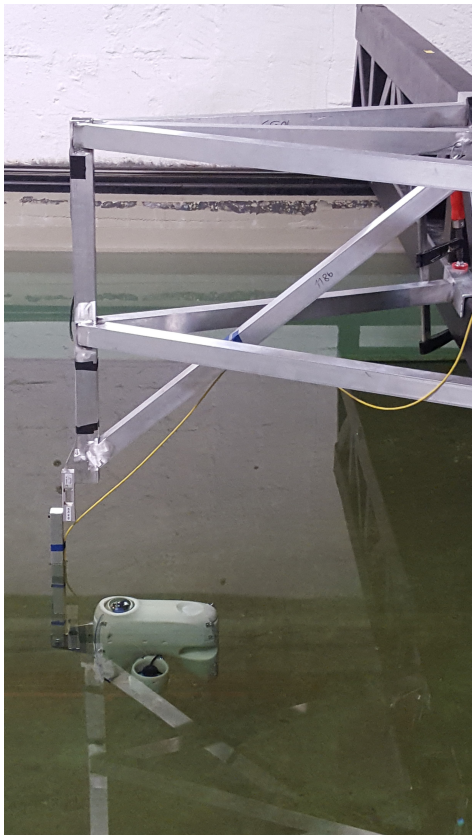
3.3.1 Experimental setup for towing tests

The damping forces and thrusters parameters were identified by using the towing carriage in MC-lab. The methods used are based on previous work done in Eidsvik (2015) and Sandøy (2016). The method is explained in detail in Chapter 5 with the results. The method required a way of attaching the

ROV to the towing carriage. For this a rig was used, with the setup and attachment of the ROV as shown in Figures 3.1 and 3.2. Figure 3.1a shows the force sensor used to measure lateral forces during tests.



(a) *Surge test configuration showing force sensor*



(b) *Heave test configuration*

Figure 3.1: *Rig for surge and heave tests*

(a) *Close up on ROV attachment* ***(b)*** *Surge test with just rig*

Figure 3.2: *Rig description*

Chapter 4

Simulation platform

The purpose of the simulator was to be able to test code and implementation without having to connect to the drone itself and running it in the lab for each test. The simulator must be able to simulate the vessel motions with control forces as input, but also the sensor measurements with and without faults. Additionally to analyse and test for fault-tolerance it is a big advantage being able to accurately dictate the nature of the sensor faults, so sensor faults must be simulated on top of real data during real testing in the lab. The simulation aspect is divided into two parts:

- **Full simulation**, where the vessel motion, sensor measurements of the motion and sensor faults are all simulated.
- **Part real, part simulation**, where the vessel motion with available sensors are real from experiments, but additional sensors and sensor faults are simulated on top of the real values.

4.1 Full simulation

Figure 4.1 shows the concept drawing of the full simulation. The input for the simulator was a power reference signal, as this was the available information from P2-Alpha. This makes it possible to record typical power reference signals by driving the ROV around, and then do simulations of your run later. The blocks on the drawing will be explained in this section, except for the thrust allocation block that relates the power reference signal to simulated thrust.

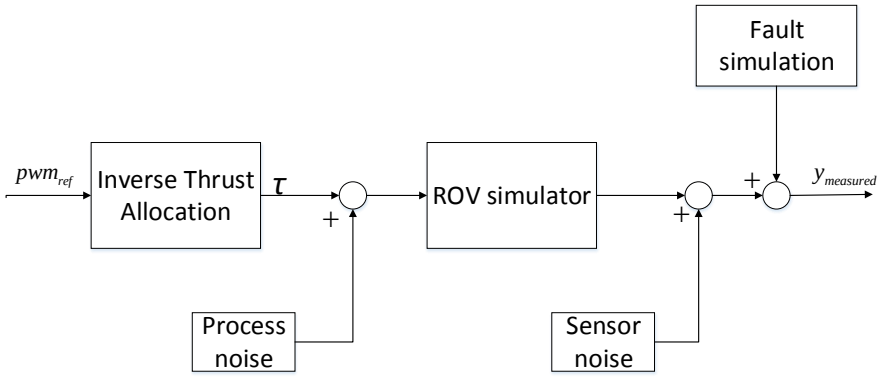


Figure 4.1: *Simulation concept drawing*

4.1.1 ROV simulator

The ROV simulator uses the dynamic equation of motion for the ROV as defined in (2.33)-(2.32) in Section 2.6. It first solves the kinetic equation to get the accelerations, then integrates this to get the velocities which it uses to solve the kinematic equation. Lastly it integrates up to positions. It then simulates all required states used by the sensors.

4.1.2 Process noise

Process noise is simulated in two steps. Firstly white noise is integrated and added independently to the thruster forces affecting the ROV, so that

a random walk force bias is introduced as talked about in Section 2.1.3. White noise was also added directly to the thruster forces, which gives

$$\mathbf{F}_{noise} = \int \mathbf{w}_{\tau_{wiener}} dt + \mathbf{w}_{\tau_{noise}} \tag{4.1}$$

where

$$\mathbf{w}_{\tau_{wiener}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}^2_{\tau_{wiener}}), \qquad \mathbf{w}_{\tau_{noise}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}^2_{\tau_{noise}}) \tag{4.2}$$

where $\boldsymbol{\sigma}^2_{\tau_{wiener}}$ and $\boldsymbol{\sigma}^2_{\tau_{noise}}$ are matrices for the multivariate distributions. The process noise was chosen to be independent of each other, making these matrices diagonal. In addition to this the thruster forces used as gains the ROV simulator was increased by 10% in relation to the control forces entering the observer.

4.1.3 Sensor noise

Sensor noise was simulated by adding white noise to the states related to the different sensors, also independently. The variance used for the sensor noise simulation is found in Section 6.4.

4.1.4 Fault simulation

All the failure modes is modelled as explained in Section 2.2. Sensor bias is added to the measurement as a sudden constant bias. Sensor drift is implemented as a constant drift added to the measurement. The high noise failure mode is modelled by increasing the sensor noise variance. Outliers are generated by utilizing the random generated variables function in Simulink and adding them to the signal. And signal freeze is simulated by a function outputting only the value at the start of the signal freeze period. All of the simulated fault parameters can activated and the parameters defined in the matlab-script *faultsim_param* that is ran before each simulation.

4.2 Part real, part simulation

MATLAB Simulink was used for all the simulations, as well as the experimental results. It was chosen due to the authors previous experience with the software and that Simulink also supports ROS implementation which is the software used for communication on P2-Alpha. As the drone is equipped with the sensors in 3.1.3, the measurements from these sensors are used directly directly. However, the QUALISYS motion capture system need additive measurement noise to simulated HPR measurements, which is done in Simulink in real time during experimental tests. During experimental tests, matlab is interfaced with an external machine running ROS, for instance P2-Alpha. The ROS system has the possibility to record real runs by logging all the measurements used, and later play them back in real time. This means that any machine running ROS can simulate runs of P2-Alpha in real time. As this thesis does not use any control systems, and just needs the sensor measurements and thruster inputs to be tested it was possible to do several runs in MC-lab, log them and then test the observer for the experiments at a later time. In this thesis a virtual machine by *VMware* was used as a ROSMASTER for the logged experiments for playback. The interfacing between Simulink and ROS in real time requires a real-time synchroniser to slow down Simulink. More information on interfacing Matlab with ROS and the virtual machine is found in Appendix D

4.3 Implementation

The implementation for the simulator and the part real, part simulation is found in Appendix B.1 and B.1.

Chapter 5

Vessel Model parameter estimation and tuning

This chapter presents the model parameter estimation of the vessel model in 4 degrees of freedom. The model used is the dynamic model given in equations (2.32)-(2.33), which is rewritten here for readability

$$\dot{\boldsymbol{\eta}} = \mathbf{J}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{5.1}$$

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} \tag{5.2}$$

with the 4 degrees of freedom defined in the SNAME convention as

$$\boldsymbol{\eta} = \begin{bmatrix} x \\ y \\ z \\ \psi \end{bmatrix}, \qquad \boldsymbol{\nu} = \begin{bmatrix} u \\ v \\ w \\ r \end{bmatrix}. \tag{5.3}$$

Each parameter and method of estimation said parameter will be discussed here.

5.1 Control forces

To find control forces acting on the ROV, namely $\boldsymbol{\tau}$, it is neccessary to find a relationship between the thruster forces and the resulting forces on the

Thruster	Thrust force	Power input to thrust
Surge starboard	f_{t_1}	u_1
Surge port	f_{t_2}	u_2
Heave	f_{t_3}	u_3

Table 5.1: *Thruster indexes*

ROV. The input to the thruster is a power signal, which for the P2-Alpha is generated by an x-box controller for steering. The drone previously had a thrust allocation, but the generated forces was calculated based on open-water tests of the thrusters done by the producer. As the hull of drone affects the hydrodynamic forces produced by thrusters, new tests were performed to more accurately find the relationship between power signal and thrust. The 4 DOF control forces for surge, sway, heave and yaw are defined in body-coordinates as

$$\boldsymbol{\tau} = \begin{bmatrix} F_x \\ F_y \\ F_z \\ M_z \end{bmatrix}. \tag{5.4}$$

The P2-Alpha has 3 thrusters as defined in section 3.1.2, two for surge and one for heave, with positions defined in table 3.2. The information available from the code already installed on P2-Alpha is the power input signal to the thrusters. In (Fossen 2011) the thruster forces and moments relate to the control forces by

$$\boldsymbol{\tau} = \mathbf{T}\mathbf{f}_t \tag{5.5}$$

$$\tag{5.6}$$

The thrust configuration matrix $\mathbf{T}$ is decided based on the positions of the thruster relative to the center of gravity as defined in Table 3.2 with the thruster indexes defined in Table 5.1 resulting in

$$\mathbf{T} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0.35 & -0.35 & 0 \end{bmatrix}. \tag{5.7}$$

It should be noted that the surge thrusters will produce a pitch motion as they are located 3 cm above the center of gravity, but pitch motion is not an included degree of freedom and thus neglected. The input to the model is $\mathbf{u}$. What remains then is finding the relationship between control signal $\mathbf{u}$ and generated thrust $\mathbf{f}_t$. This was done based on the thrust allocation done in Sandøy (2016), as this was done on the same thrusters from Blueye Robotics, but equipped on a different ROV. In Sandøy (2016) the relationship between power signal and produced thrust is decided by the partitioned function

$$f_{t_i} = \begin{cases} K_L^-(u+25) + K_{NL}^-(u+25)|u+25| & , \quad u < 0 \\ K_L^+(u-25) + K_{NL}^+(u-25)|u-25| & , \quad u > 0 \\ 0 & , \quad u = 0 \end{cases} \tag{5.8}$$

where 25 is the dead zone for the power signal sent to the thrusters. The $\mathbf{K}$ matrix is split into a linear and a non-linear part. The function is chosen this way because of expected non-linearity of the thrust curve between power signal and thrust. It is partitioned into two functions because the thrust curve is not expected to be symmetric since the hull of the ROV is not symmetric in surge or heave direction.

5.1.1 Method for deciding thrust characteristics

To decide the thrust characteristics the rig in Section 3.3.1 was used. The drone was mounted in surge or heave configurations, and power signals were sent to the thrusters while the force sensor measured the exerted forces from the thrusters. The data from the tests was curve fitted using least-squared curve fitting to the function in (5.8) using the *lsqcurvefit* function in Matlab.

The data from the tests are gathered in an excel sheet with MATLAB code for post processing and plotting the thrust characteristics in Appendix A.2.

Backward thrust	$K_L^- = 0.1395$ $K_{NL}^- = 1.2786e-04$
Forward thrust	$K_L^+ = 0.1936$ $K_{NL}^+ = 8.1218e-05$

Table 5.2: *Thrust parameters of f_t for surge thrusters*

5.1.2 Surge thrust characteristics

The force was logged for power signals between ±40 to ±300, where negative signal is backwards thrust and positive signal is forward thrust . This translates to 2.5-50% of the thruster capacities. For the surge thrusters both thrusters were run at the same time, assuming they produce the same power. So the resulting force is divided by 2. The resulting thruster characteristics are shown in Figure 5.1, together with the least-squared curve fit.

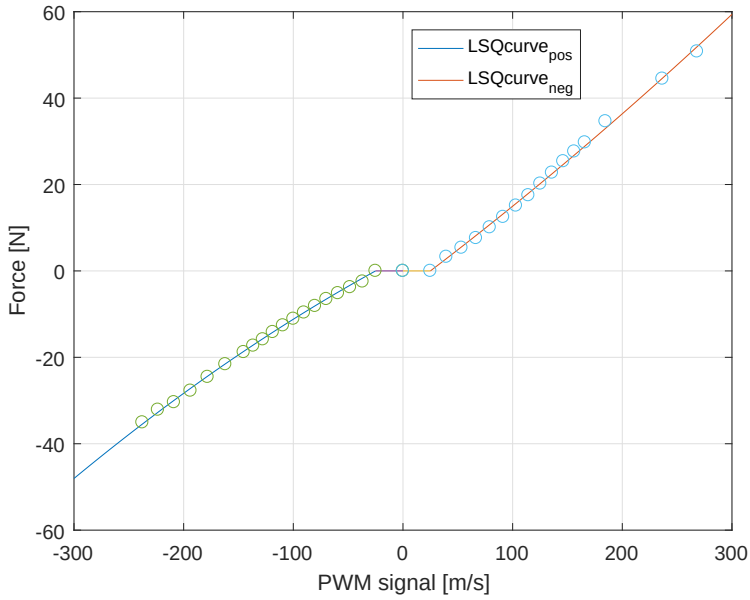


Figure 5.1: *Thrust characteristics for surge thrusters*

The curve fit shown in figure 5.1 corresponds to constants of f_t listed in table 5.2

5.1.2.1 Heave thrust characteristics

The force was logged for power signals between -600 to 600. However the curve fit was only done for the values between -400 to 400. As can be seen from the thruster characteristics in Figure 5.2 the force caps when the power is around ±400. This corresponds to 70% of the thruster capacity. Using more power on the heave thruster will thus just drain the battery without producing more thrust force. Curve fitting to include all the logged data would also decrease the accuracy as the chosen fitting function f_t is a second degree polynomial, and to fit all the data the shape would require a third degree polynomial. It should be noted that the test is done when the drone is not moving, so it is possible there is merit to using more than 70% of the power when the drone is moving and has incoming water velocity to the propeller.

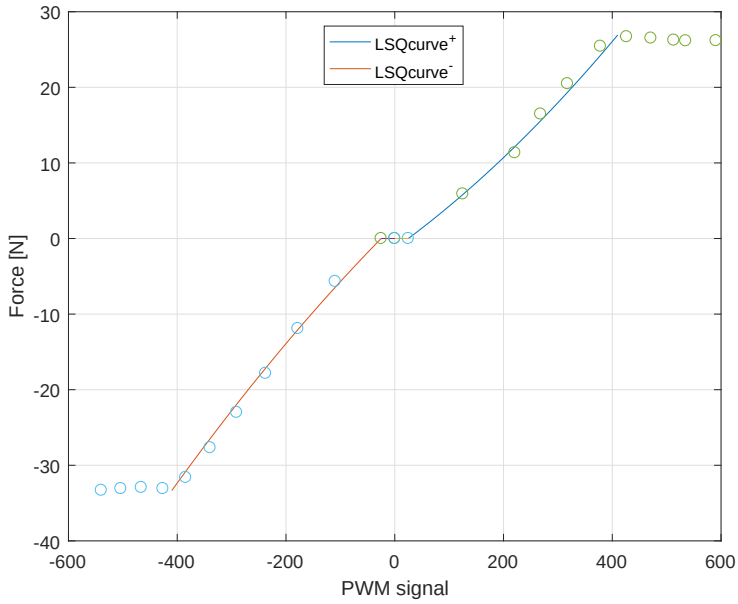


Figure 5.2: *Thrust characteristics for heave thruster*

The curve fit shown in Figure 5.2 corresponds to the parameters in Table 5.3.

Upward thrust	$K_L^- = 0.0735$ $K_{NL}^- = 3.3980e-05$
Downward thrust	$K_L^+ = 0.0539$ $K_{NL}^+ = 4.1697e-05$

Table 5.3: *Thrust parameters for heave thruster, where negative is upwards thrust and positive is downwards thrust*

5.2 Mass and added mass

The mass matrix in (5.2) consists of a rigid body mass and added mass

$$\mathbf{M} = \mathbf{M}_{RB} + \mathbf{M}_A \tag{5.9}$$

where for the 4 DOF model

$$\mathbf{M}_{RB} = \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & m & 0 & 0 \\ 0 & 0 & m & 0 \\ 0 & & 0 & I_z \end{bmatrix} \tag{5.10}$$

$$\mathbf{M}_A = \begin{bmatrix} -X_{\dot{u}} & 0 & -X_{\dot{w}} & 0 \\ 0 & -Y_{\dot{v}} & 0 & mx_g - Y_{\dot{r}} \\ X_{\dot{w}} & 0 & -Z_{\dot{w}} & 0 \\ 0 & mx_g - Y_{\dot{r}} & 0 & -N_{\dot{r}} \end{bmatrix}. \tag{5.11}$$

The parameters in (5.10) are all found from the geometrical properties in Section 3.1.

5.2.1 Empirical method of estimation for added mass

Fossen (2011) defines added mass as a result of hydrodynamic forces that can be seen as virtual mass added to a system because an accelerating or decelerating body must move some volume of the surround fluid as it moves through it. For underwater vehicles the added mass is independent of wave excitation frequency, such that the added mass can be considered

Projected area front	$45427.5mm^2$
Projected area side	$128582mm^2$
Projected area top	$62300mm^2$
Length	$440mm$
Height	$327mm$
Width	$140mm$
Density of water	$1000kg/m^3$

Table 5.4: *Parameters needed for Eidsvik method of added mass estimation*

a constant. To find the added mass of the ROV an empirical procedure was used. The procedure was presented in Eidsvik (2015), later dubbed Eidsviks method and used and validated in Sandøy (2016). Eidsvik (2015) also used Wadam simulations to verify the method, which is a numerical computational software for hydrodynamics. The procedure estimates the added mass based off analytical data, with the DNV standard for reference, with the assumption that the ROV shape can be estimated as a rectangular prism where 2 of 3 sides are equal. The result will be used directly in this thesis, and the reader is referred to Eidsvik (2015) for the detailed procedure. The algorithm used is found in Appendix A.4, taken from Eidsvik (2015) yielding the following result

$$\mathbf{M}_A = \begin{bmatrix} 8.1577 & 0 & 0 & 0 \\ 0 & 51.6391 & 0 & 0 \\ 0 & 0 & 25.0200 & 0 \\ 0 & 0 & 0 & 0.3210 \end{bmatrix} \tag{5.12}$$

5.2.2 Comment on added mass estimation

The assumption of prisme 2/3 equal is not entirely valid for BluEye P2-alpha as it does not have any strict edges for hydrodynamical purposes as well as thrusters sticking out from the main body at each side. However, in his thesis Eidsvik used his method on the ROV Neptunus which has a similar shape to the P2-Alpha. By comparing the results from the empirical method to CFD calculations done in Wadam, he found almost no difference in the surge direction. The added mass in sway and pitch was also satisfactory. The biggest offset was in the heave direction, where wadam calculated less than half the added mass in heave. In the roll and yaw directions wadam

calculated 70-90 percent more added mass than the empirical method. P2-alpha has a more box-like shape on the topsides than the ROV Neptunus, so the assumption should be valid for the top and thus not have as much offset as ROV Neptunus did for heave. Roll is passively stable for P2, and not included for the 4 DOF observer. However, the big difference for added mass in yaw could be a concern.

5.3 Damping forces

The damping forces in can be divided into linear and non-linear forces

$$\mathbf{D} = \mathbf{D_L} + \mathbf{D_{NL}} \tag{5.13}$$

where the non-linear forces are divided into a quadric and cubic term. For the 4 DOF model we then have

$$\mathbf{D_L} = diag\begin{bmatrix} X_u & Y_v & Z_w & N_r \end{bmatrix} \tag{5.14}$$

$$\mathbf{D_{NL}} = \begin{bmatrix} X_{|u|u}|u| + X_{uuu}u^2 & 0 & 0 & 0 \\ 0 & Y_{|v|v}|v| + Y_{vvv}v^2 & 0 & 0 \\ 0 & 0 & Z_{|w|w}|w| + Z_{www}w^2 & 0 \\ 0 & 0 & 0 & N_{|r|r}|r| + N_{rrr}r^2 \end{bmatrix} \tag{5.15}$$

These forces were found in an experimental procedure, similar to the ones done in Sandøy (2016) and Eidsvik (2015). The lateral damping forces were found using the towing carriage as will be explained in Section 5.3.1. The rig used did not support rotation, so the yaw damping term used a different method which is covered in Section 5.3.3

5.3.1 Experimental setup for identification of lateral damping forces

The three lateral damping forces were identified by using the towing carriage in MC-lab together with the rig described in Section 3.3.1. A rig was constructed to attach the ROV to the towing carriage. A force sensor was

used as the connection between the rig and the towing carriage, measuring the lateral force in the forward direction asserted from the rig to the carriage as the carriage is moving forward. The figures 3.1 explain the setup, with different configurations for determining surge, sway and heave. By measuring the force asserted on the carriage from the rig when the carriage is moving the drag forces can be measured. After the transient accelerations of carriage on the rig, the remaining constant force is only due to drag forces on the ROV and rig which is the damping term in the dynamic equation of motion. By doing multiple runs with different velocities a plot is made for the relationship between drag forces and velocities to determine the values for the linear and non-linear terms of the damping force. In order to identify the drag forces due to the ROV, runs where made with just the rig and then subtracted from the total forces

$$F_{ROV} = F_{RIG+ROV} - F_{RIG}. \tag{5.16}$$

To find the linear and non-linear terms a third degree polynomial on the form

$$Ax^3 + B|x|x + Cx \tag{5.17}$$

is fitted to the plot by least-squared estimation. Where A is the linear damping term, B is the quadratic term and C is the cubic term of (5.14)-(5.15). The script doing the curve-fitting and plotting can be found in Appendix A.1.

$$\begin{aligned} X_u &= 1.843 \\ X_{|u|u} &= 9.4873 \\ X_{uuu} &= -0.7339 \end{aligned}$$

Table 5.5: *Surge damping parameters*

$$\begin{aligned} Y_v &= 3.1802 \\ Y_{|v|v} &= 45.7127 \\ Y_{vvv} &= 6.2125 \end{aligned}$$

Table 5.6: *Sway damping parameters*

5.3.2 Results from towing test damping forces

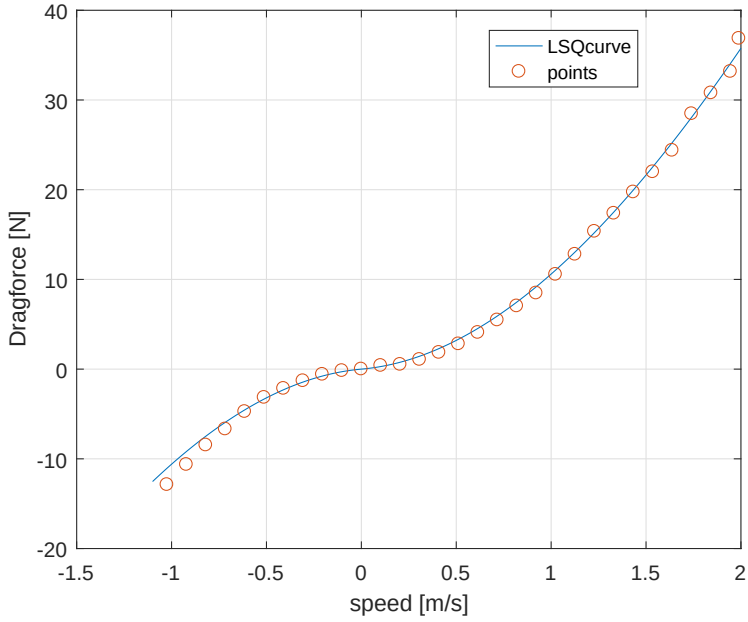


Figure 5.3: *Surge drag force and curve fit*

$$\begin{aligned} Z_w &= -0.3778 \\ Z_{|w|w} &= 25.4825 \\ Z_{www} &= -1.2523 \end{aligned}$$

Table 5.7: *Heave damping parameters*

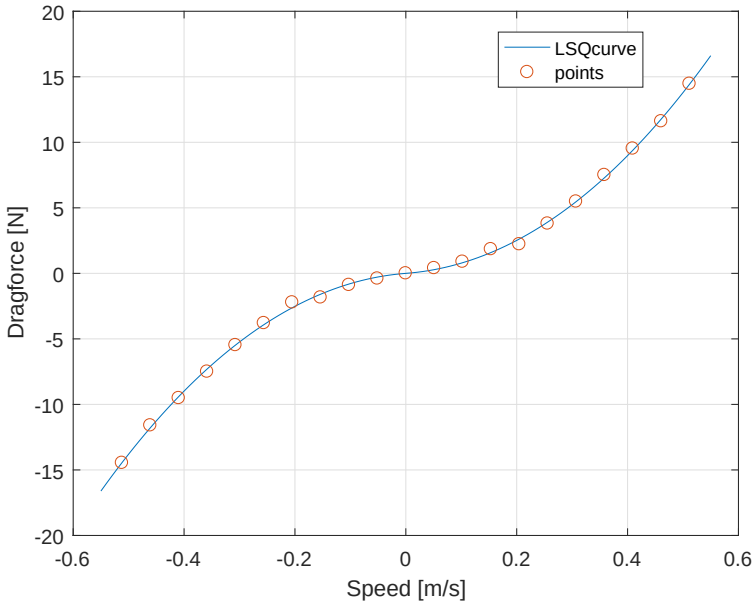


Figure 5.4: *Sway drag force and curve fit*

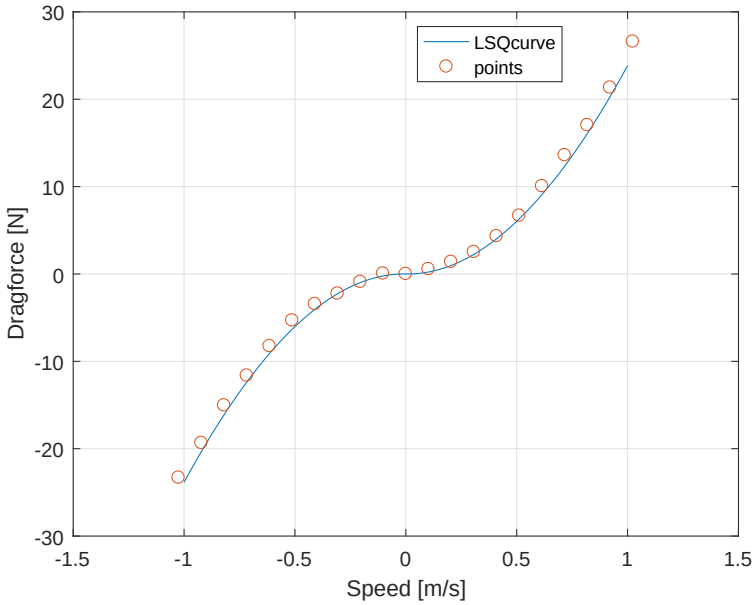


Figure 5.5: *Heave drag force and curve fit*

5.3.3 Experimental setup for deciding damping forces in yaw direction

As the rig did not support measurements of rotational forces it could not be used for estimation of yaw damping forces. Instead the ROV used its own thrusters to rotate the ROV around its own axis. The ROV was weighed down to get a slight negative buoyancy, while a rope was tied from the top of the ROV to an anchor point to keep it more stable while rotating. The power signals sent to the thrusters were then mapped to moment using the results from Section 5.1. The gyroscope equipped on the ROV was used to measure angular velocities. Runs were then logged for constant moment exerted on the ROV by the thrusters. After the transient acceleration period the angular velocities would become constant, and plotted versus the moment. The data was then curve fitted as for the lateral damping forces. The script doing the curve fitting and plotting can be found in Appendix A.1

$$\boxed{\begin{array}{c} N_r = -0.0868 \\ N_{|r|r} = 0.5344 \\ N_{rrr} = -0.0523 \end{array}}$$

5.3.4 Results of yaw damping forces

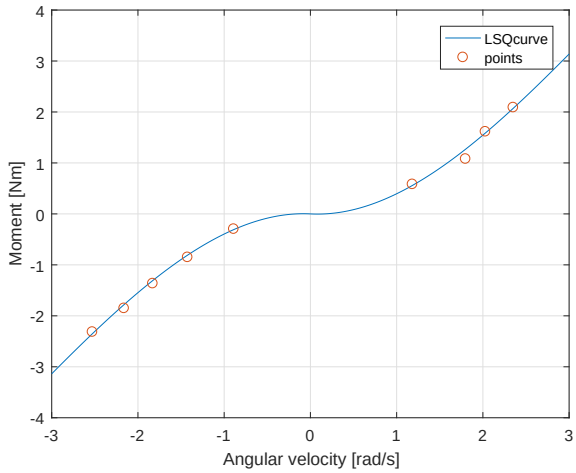


Figure 5.6: *Yaw damping and curve fit*

Chapter 6

Sensor suite statistical parameters

Given the sensor suite equipped on P2-Alpha as mentioned in Section 3.1.3, with the addition of the QUALISYS pos-ref system to simulate HPR we have the following sensor suite to be used in the 4 DOF observer

- Accelerometer measuring $\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$
- Gyroscope measuring r
- Magnetic compass measuring ψ
- Pressure sensor measuring z_{pres}
- Simulated HPR measuring $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$

6.1 Inertial Measurement Unit

The IMU installed on P2-Alpha is a 9-DOF MEMS based unit from Sparkfun. This includes a 3-axis accelerometer, 3-axis gyroscope, and 3-axis magnetometer. The IMU then measures three orthogonal accelerations, three orthogonal angular velocities and three magnetic readings that can be used

as a magnetic compass. The measurements from an IMU including bias and noise can then be described as (Mahony et al. 2008)

$$\mathbf{a}^b_{imu} = \dot{\mathbf{v}}^b_{real} - \mathbf{J}^b_n(\boldsymbol{\Theta})\mathbf{g}^n + \mathbf{b}^b_{acc} + \boldsymbol{\epsilon}^b_{acc} \tag{6.1}$$

$$\boldsymbol{\omega}^b_{imu} = \boldsymbol{\omega}_{real} + \mathbf{b}_{gyro} + \boldsymbol{\epsilon}^b_{gyro} \tag{6.2}$$

$$\mathbf{m}^b_{imu} = \mathbf{R}^b_n(\boldsymbol{\Theta})\mathbf{m}^n + \mathbf{b}^b_{mag} + \boldsymbol{\epsilon}^b_{mag}. \tag{6.3}$$

where

- $\mathbf{b}$ is the sensor bias for the respective sensor.
- $\boldsymbol{\epsilon}$ is the sensor noise.
- $\mathbf{m}^b_{imu} = \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix}$ is the magnetic force measured in the orthogonal xyz-space.
- $\mathbf{J}^b_n$ is the transformation between the frame of the gravitational vector {n} and the IMU body-frame {b}.
- $\mathbf{g}^n = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}$ is the gravity of earth in {n}.

If the magnetometer is sitting or the measurements calibrated to sit in a horizontal plane levelled to the surface of the earth such that $\theta = \phi = 0$ the heading angle ψ_m can be calculated from the magnetic readings as (Fossen 2011)

$$tan(\psi_m) = \frac{m_y}{m_x}. \tag{6.4}$$

In this thesis the heading angle will be used directly from the online calculations of the IMU. The measurements simplified by ignoring bias used in the observer from the IMU are then

$$\mathbf{a}^b_{acc} = \dot{\mathbf{v}}^b_{real} - \mathbf{J}^b_n(\boldsymbol{\Theta})\mathbf{g}^n + \boldsymbol{\epsilon}^b_{acc} \tag{6.5}$$

$$r_{gyro} = r + \epsilon_r \tag{6.6}$$

$$\psi_{comp} = \psi + \epsilon_\psi. \tag{6.7}$$

As can be seen from (6.5) the gravitational force must be subtracted from the accelerometer by $\mathbf{J}(\mathbf{\Theta})$ which is dependent on the roll and pitch. As these are outside the observers 4 DOF, this is done before the measurements are entering the observer by estimates of roll and pitch that are estimated by the IMU online. This however does mean that the noise affecting the accelerometer will be affected by the noise created by roll and pitch angles. When finding the sensor noise parameters this must be accounted for.

6.2 HPR

When properly calibrated the Qualisys motion capture system that will be used for the HPR measurements has a variance in the order of 10^{-8}, as documented in Sandøy (2016). The update frequency of the motion capture system is also very fast, so to emulate a HPR sensor system a higher variance and slower update frequency will be added to the Qualisys motion capture system before it the measurement enters the observer. The chosen values for the HPR system is from (Blain et al. 2003), and is also close to the accuracy of several hydro-acoustic single systems from Kongsberg with an accuracy at 0.1-0.15m. It should be mentioned that HPR measurements are normally done in spherical coordinates(Zhao et al. 2012), and then transformed to Cartesian coordinates by a build-in algorithm for the HPR system. Hence, there will be a correlation between measurements in each direction of the Cartesian coordinates. This is not simulated however, and each measurement is considered independent.

6.3 Sensor offsets

The magnetic compass measures the magnetic north, while the frame used by the Qualisys-motion capture system is tilted in comparison. The offset was measured by placing the drone along the ”north” line of the calibrated Qualisys-frame, which in reality is north-west. This resulted in an offset by 63 degrees, which must be included in the rotational matrix for the kinematic equations for the ROV observer model. In addition to this the pressure sensor and the Qualisys depth measurements had an offset as well.

Measurement	Variance	unit	Mean update frequency
a_x	$2.1166e-04$	$[m/s]^2$	23.08 ms
a_y	$1.3312e-04$	$[m/s]^2$	23.08 ms
a_z	$7.8301e-04$	$[m/s]^2$	23.08 ms)
r	$9.1092e-07$	$[rad/s]^2$	23.08ms
ψ	$5.0539e-04$	rad^2	23.08 ms
z_{pres}	$3.422e-05$	m^2	49.7816ms
xyz_{hpr}	0.02777	m^2	300ms

Table 6.1: *Sensor measurement variance*

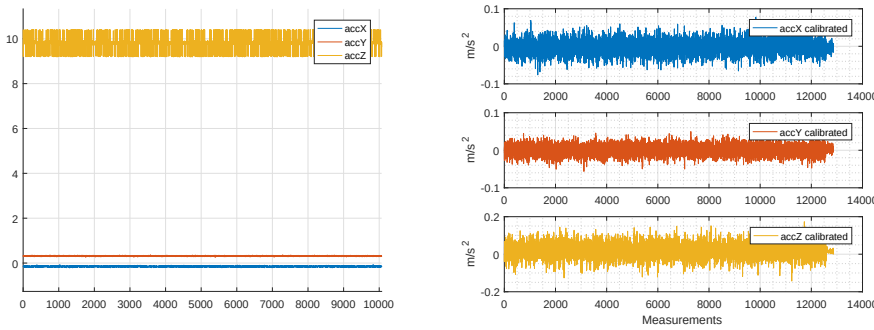
$$z_{pres} = z_{qual} - 1.2106 \tag{6.8}$$
$$\psi_n = \psi qual - 63 \tag{6.9}$$

6.4 Sensor measurement noise

The measurement variances was found by logging measurements over 300 seconds for all sensors. The P2-Alpha was attached to the towing rig during the logging, and the amount of measurements per sensor vary depending on the measurement update frequencey of the sensor. The nominal values for variance are summarized in table **??**. Although be mindful that the measurement noise might be higher during movement due to vibrations from thrusters. The stand-still accelerometer variance was calculated to be

$$\sigma^2_{a_x} = 2.1333e-06 \tag{6.10}$$
$$\sigma^2_{a_y} = 1.38e-06 \tag{6.11}$$
$$\sigma^2_{a_z} = 3.8395e-04 \tag{6.12}$$

by removing the gravity from the measurements by (6.5), and removing the outliers from the a_z measurements as shown in Figure **??** the new variance was calculated. The sensor measurement variance for all the measurements used in the observer are summarized in Table 6.4. The code and timeseries used are found in Appendix A.3.



(a) Stand still accelerometer measurements

(b) Calibrated accelerometer measurements

Figure 6.1: *Accelerometer standstill measurements*

Chapter 7

Fault-tolerant Observer design

This chapter concerns the fault-tolerant model-based observer design and its implementation, the main objective of the thesis. The chapter will start with introducing the overall observer design, then go into further detail for each of its respective blocks. The observer can be divided into a signal processing module and an Extended Kalman Filter module. The algorithms used are covered in the preliminaries. The observer uses the inherit properties of the Kalman Filter for sensor fusion and handling of asynchronous measurements.

7.1 Observer design

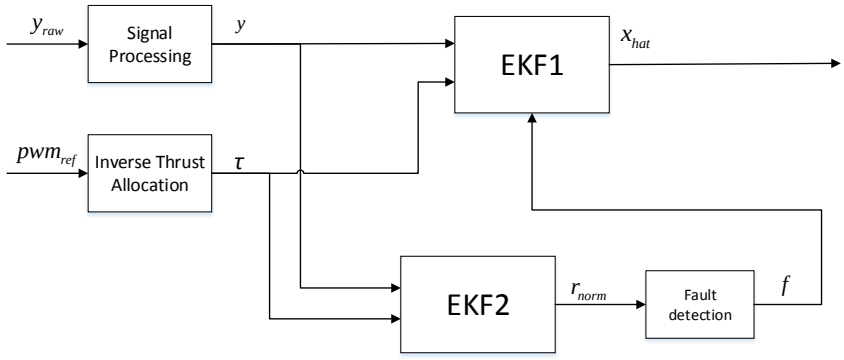


Figure 7.1: *Observer concept drawing. EKF1 is the main filter that uses all available sensor measurements. EKF2 represents additional filters that are used for fault detection for specific sensors, and reports to the main filter so that the main filter can adjust in case of a faulty sensor.*

The concept of the observer design is shown in Figure 7.1. The observer takes in the raw sensor measurements and control forces in body-coordinates from the thrust allocation module. The measurements first go through a signal processing module before the signal enters the filters. One main filter is used for state-estimation, which uses all healthy sensor measurements for state-estimation. Additional filters are created in a bank, where it does not use redundant sensor information for state-estimation. In the case of all sensors being in a fault-free state, these filters should provide a suboptimal estimate as they do not use all the information. But by isolating the sensors, you deal with the problem of fault isolation. If one of the non-redundant sensor filters detects a fault, this will be reported to the main filter and the corresponding faulty sensor can be deactivated. The main filter can be used for fault-detection as well, but it can not properly decide which sensor is faulty unless it has a redundancy of 3. (Bryne) The additional Kalman Filters uses

7.2 Thrust allocation

The thrust allocation module is called the inverse thrust allocation module in the drawing because generally when talking about a thrust allocation module you transform the wanted control forces to thruster forces depending on the thruster configuration. Here we transform the *PWM_ref* signal to control forces in body-coordinates by using Equations 5.6, (5.7) and (**??**) with the determined constants from Section 5.1.

7.3 Observer model state-space equations

All the Kalman Filters use the same observer model. Using the dynamic equation of motion for the ROV in (2.32)-(2.33) adding a bias for unmodelled forces in 4 DOF we get the full 12 state observer model with

$$\boldsymbol{\eta} = \begin{bmatrix} x \\ y \\ z \\ \psi \end{bmatrix}, \qquad \boldsymbol{\nu} = \begin{bmatrix} u \\ v \\ w \\ r \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} b_x \\ b_y \\ b_z \\ b_r \end{bmatrix}. \tag{7.1}$$

$$\dot{\boldsymbol{\eta}} = \mathbf{J}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{7.2}$$

$$\mathbf{M}\dot{\boldsymbol{\nu}} = -\mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{b} + \boldsymbol{\tau} \tag{7.3}$$

$$\dot{\mathbf{b}} = \mathbf{w} \tag{7.4}$$

where

- $\mathbf{M} = \mathbf{M_{RB}} + \mathbf{M_A}$ is the mass matrix consisting of rigid body mass and added mass.
- $\mathbf{C} = \mathbf{C_{RB}} + \mathbf{C_A}$ is the Coriolis forces due to rigid body and added mass effects.
- $\mathbf{D} = \mathbf{D_L} + \mathbf{D_{NL}}$ is the linear and non-linear damping forces.
- $\boldsymbol{\tau}$ is the control forces from the thrusters.
- $\mathbf{J}(\boldsymbol{\eta})$ is the rotational matrix from body frame b and the inertial frame denoted as n, which would be the frame defined by the Qualisys calibration.

The bias is modelled as a wiener-process as talked about in Section 2.1.3. All the matrices and estimated parameters for said matrices are shown in Chapter 5 with the exception of $\mathbf{C}$ which is defined for 4 DOF as

$$\mathbf{C}_A = \begin{bmatrix} 0 & 0 & 0 & Y_{\dot{v}}v \\ 0 & 0 & 0 & -X_{\dot{u}}u \\ 0 & 0 & 0 & 0 \\ -Y_{\dot{v}}v & X_{\dot{u}}u & 0 & 0 \end{bmatrix} \tag{7.5}$$

where

$$\mathbf{C}_{RB} = \begin{bmatrix} 0 & 0 & 0 & -m(v + rx_g) \\ 0 & 0 & 0 & m(u - ry_g) \\ 0 & 0 & 0 & 0 \\ m(v + rx_g) & -m(u - ry_g) & 0 & 0 \end{bmatrix} \tag{7.6}$$

where

$$\mathbf{r}_g^b = \begin{bmatrix} x_g & y_g & z_g \end{bmatrix} \tag{7.7}$$

is the vector from the vessel origin (VO) to the center of gravity expressed in, which is chosen to be [0 0 0]. The state-space system used in the explanation of the Kalman Filter in Section 2.7 was defined as

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{w} \tag{7.8}$$

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v}. \tag{7.9}$$

By transforming the observer model to this state-space form with

$$\mathbf{x} = \begin{bmatrix} \boldsymbol{\eta} \\ \boldsymbol{\nu} \\ \mathbf{b} \end{bmatrix} \tag{7.10}$$

$$\mathbf{A}_{12x12} = \begin{bmatrix} \mathbf{0} & \mathbf{J}(\boldsymbol{\eta}) & \mathbf{0} \\ \mathbf{0} & -\mathbf{M}^{-1}(\mathbf{C}(\boldsymbol{\nu}) + \mathbf{D}(\boldsymbol{\nu})) & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{7.11}$$

$$\mathbf{B}_{12x4} = \begin{bmatrix} \mathbf{0} \\ \mathbf{M}^{-1} \\ \mathbf{0} \end{bmatrix} \tag{7.12}$$

for 4 DOF

$$\mathbf{J}(\boldsymbol{\eta}) = \mathbf{J}(\psi) = \begin{bmatrix} c\psi_{qual} & -s\psi_{qual} & 0 & 0 \\ s\psi_{qual} & c\psi_{qual} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \boldsymbol{\nu} \tag{7.13}$$

As mentioned in Section 6.3 there is a constant offset in the angles between the compass heading ψ_{comp} and the calibrated Qualisys-frame for the experimental results. For the simulated case this offset is set to zero, but for the experimental case we get

$$\dot{\boldsymbol{\eta}}_{\boldsymbol{qual}} = \mathbf{J}(\psi_{\mathbf{comp}})\mathbf{J}(\psi_{\mathbf{offset}})\boldsymbol{\nu} \tag{7.14}$$

7.4 Measurement vector and H

The available sensor measurements for P2-Alpha are

$$\mathbf{y}_{hpr} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \qquad y_{comp} = \psi, \qquad y_{pres} = z, \qquad \mathbf{y}_{imu} = \begin{bmatrix} a_x \\ a_y \\ a_z \\ r \end{bmatrix}. \tag{7.15}$$

The state space-equations in 7.3 does not model the jerk, i.e the derivative of accelerations. This means the Kalman Filter can not use the acceleration measurements directly in it's corrector step, as it only creates PV-estimates. In order to utilize information from the acceleration measurements, these are integrated from the previous velocity estimates. In other words the accelerometer is used together with the state-estimate from the Kalman Filter to produce a pseudo velocity measurement. This gives

$$\mathbf{y}_{imu} = \begin{bmatrix} \hat{u}_{k-1} + a_x \Delta t \\ \hat{v}_{k-1} + a_y \Delta t \\ \hat{w}_{k-1}\ a_z \Delta t \\ r \end{bmatrix} = \begin{bmatrix} u \\ v \\ w \\ r \end{bmatrix} \tag{7.16}$$

The full input measurement vector for the main Kalman Filter at step k is defined as

$$\mathbf{y}_k = \begin{bmatrix} x_{hpr} \\ y_{hpr} \\ z_{hpr} \\ z_{pres} \\ \psi_{comp} \\ \hat{u}_{k-1} + a_x \Delta t \\ \hat{v}_{k-1} + a_y \Delta t \\ \hat{w}_{k-1} + a_z \Delta t \\ r \end{bmatrix} \tag{7.17}$$

If all sensor measurements are used in the Kalman Filter for the model in **??** the measurement mapping matrix **H** becomes the following 9x12 matrix

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0}_{1x4} \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0}_{1x4} \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \mathbf{0}_{1x4} \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \mathbf{0}_{1x4} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \mathbf{0}_{1x4} \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \mathbf{0}_{1x4} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \mathbf{0}_{1x4} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \mathbf{0}_{1x4} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \mathbf{0}_{1x4} \end{bmatrix}. \tag{7.18}$$

From (7.18) it can be seen that the only redundant measurement available is for z, for which the Kalman Filter will do sensor fusion with this **H**. To change which measurements are used in the filter, it is possible to just alter **H**. For a complementary filter designed for fault detection that only uses the following measurements

$$\begin{bmatrix} x^{hpr} \\ y^{hpr} \\ z^{depth} \\ \psi_{comp} \end{bmatrix} \tag{7.19}$$

the same measurement vector as defined in (7.17) can be used with the following alteration to **H**

$$\mathbf{H}* = \begin{bmatrix} 1 & 0 & 0 & 0 & \mathbf{0}_{1x8} \\ 0 & 1 & 0 & 0 & \mathbf{0}_{1x8} \\ 0 & 0 & 0 & 0 & \mathbf{0}_{1x8} \\ 0 & 0 & 1 & 0 & \mathbf{0}_{1x8} \\ 0 & 0 & 0 & 1 & \mathbf{0}_{1x8} \\ \mathbf{0}_{4x1} & \mathbf{0}_{4x1} & \mathbf{0}_{4x1} & \mathbf{0}_{4x1} & \mathbf{0}_{4x8} \end{bmatrix} \tag{7.20}$$

7.5 Signal processing module

The signal processing module has three jobs. The first job is to sample the sensor measurements to the same size as the Kalman Filter step size. The step size is fixed at $\Delta t = 0.025$, which is equal to the fastest sensor update rate. By doing this, the Kalman Filter receives all sensor measurements at the same frequency. This is the first step in the asynchronous measurement handling method used in the observer. The second step will be covered in 7.6. The second job of the signal processing module is outlier detection and removal. It does this by variance testing. The mean of a signal is calculated for the last 10 measurements. If a new measurement is outside of the interval

$$y_k \in [\bar{y}_k - a\sigma_0, \bar{y}_k + a\sigma_0] \tag{7.21}$$

the measurement is refused, where a is set to a value 3-9 (Sørensen 2013). In place of the outlier measurement, the previous accepted measurement is sent to the Kalman Filter. This means that in the case of an outlier, this will affect the Kalman Filter in the same way as a signal freeze. The third job of the signal processing module is to remove the gravity from the

accelerometer measurements. It does this by Equation 6.5 from Section 6.1 rewritten here for readability

$$\mathbf{a}^b_{imu} = \dot{\mathbf{v}}^b_{real} - \mathbf{J}^b_e(\mathbf{\Theta})\mathbf{g}^e + \mathbf{b}^b_{acc} \tag{7.22}$$

$\mathbf{J}^b_e$ is the rotational matrix from the gravity vector coordinate system to the accelerometer body-frame dependent on roll and pitch. The subtraction of the gravity vector is done by using the roll and pitch estimates from the IMU directly in the signal processing. The accelerometer body-frame is calibrated to be the same as the ROV body-frame.

7.6 Asynchroneus measurement and signal freeze handling

All the Kalman Filters run on a $\Delta t = 0.025$ which is close to the sampling frequency of the fastest sensor. As the signal processing module is sampling every sensor measurement to the same frequency, the filters get an incoming measurement from each sensor at every time step. However, as for instance the HPR sensor update rate is 0.3 sec, only every 12th measurement from the HPR is a new measurement. Likewise only every second measurement from the pressure sensor is new. This means that for the observer the asynchronous sensor updates are seen as a signal freeze. The HPR "freezes" for 11 time steps consistently. As outliers are sent to the filter as a signal freeze from the signal processing module, the filter deals with asynchronous measurements, outliers and real signal freezes in the same way. It does this by modifying the Kalman Gain $\mathbf{K}_k$ before the corrector step shown in the Kalman Filter Algorithm in Section 2.7.1. The Kalman gain is modifying as according to Algorithm 2 below in case of a signal freeze.

By modifying the Kalman-gain before the corrector step, the freezed sensor will not be used in the corrector step in the Kalman Filter Algorithm. In case of no sensor redundancy this will result in dead-reckoning for the corresponding DOF. The Kalman Filter algorithm is calculated in a fixed step regardless, so the increase in $\mathbf{P}_k$ will then be $\mathbf{Q}_k * \Delta t$ for Δt steps.

Algorithm 2 Kalman Gain-modify algorithm

Measurement vector $\mathbf{y}_k$
Kalman Gain $\mathbf{K}_k$

for $i = 1 : \text{length}(\mathbf{y}_k)$
  if $\mathbf{y}_k(i) = \mathbf{y}_{k-1}(i)$
    $\mathbf{K}_k(:, i) = 0$
end

7.7 Kalman Design Matrices Q and R

Q is found by tuning. As mentioned in Section 2.7 the values were set to a fraction of the values in **R** as a starting point. This assumes a good ROV model of the states. Other things to take into account is the value of the stepsize for the Kalman Filter, the higher frequency on the Kalman Filter the lower the values in **Q** should be. The **R** matrix is obtained from the variance measurements in Section 6.4. The variance for the accelerometer can not be taken directly in the **R** matrix because the observer integrates its measurements for each step instead of using it directly. The variance elements for the pseudo velocity measurements are set based on the standard deviation for each time-step where

$$\mathbf{R}_{acc} = (\boldsymbol{\sigma_a cc}\Delta t)^2. \tag{7.23}$$

This gives

$$\mathbf{R} = diag\{\mathbf{R}_{HPR}, \sigma^2_{z_{pres}}, \mathbf{R}_{comp}, \mathbf{R}_{acc}, \sigma^2_{r_{gyro}}\} \tag{7.24}$$

$$\mathbf{R}_{HPR} = \begin{bmatrix} \sigma_x^2 \\ \sigma_y^2 \\ \sigma_z^2 \end{bmatrix}, \qquad \mathbf{R}_{IMU} = \begin{bmatrix} (\sigma_{a_x}\Delta t)^2 \\ (\sigma_{a_y}\Delta t)^2 \\ (\sigma_{a_z}\Delta t)^2 \end{bmatrix} \tag{7.25}$$

7.8 Fault detection module

The fault detection module uses the residuals generated from the secondary Kalman Filters by following the steps as described in Section 2.7.7. The problem for implementation comes down to the designing the parameters of $\mu_{\mathbf{1}}$, and then deciding on a cusum threshold value h which is done by the code found in Appendix C.1

$$H_0 : \quad r_k^{norm} \sim \mathcal{N}(0,1) \tag{7.26}$$
$$H_1 : \quad r_k^{norm} \sim \mathcal{N}(\mu_1,1). \tag{7.27}$$

Deciding on a $\mu_{\mathbf{1}}$ for the given sensor is something that can be done iteratively, by doing multiple runs to see how sensitive the normalized residuals are.

Chapter 8

Simulation Results

The following chapter will present the results from the simulations. The simulated thrust on the model is taken from power reference signals from real runs in MC-lab. The ROV was steered around using an x-box controller, and the power reference created by the controller was saved to be used for simulations. All simulations were done with a Kalman Filter step at 0.025ms, equal to the set IMU sampling time. The simulink model with initialization scripts are found in Appendix B.1. Two filters were used in all the simulations, the main filter using all sensor measurements which be referred to as *ekf1*, while the additional filter is referred to as *ekf2*. The two filters uses the following measurements with reference to Section 7.4

$$\mathbf{y}_k^{ekf1} = \begin{bmatrix} x_{hpr} \\ y_{hpr} \\ z_{hpr} \\ z_{pres} \\ \psi_{comp} \\ \hat{u}_{k-1} + a_x \Delta t \\ \hat{v}_{k-1} + a_y \Delta t \\ \hat{w}_{k-1} + a_z \Delta t \\ r \end{bmatrix}, \qquad \mathbf{y}_k^{ekf2} = \begin{bmatrix} x_{hpr} \\ y_{hpr} \\ z_{pres} \\ \psi_{comp} \end{bmatrix} \tag{8.1}$$

The process noise used in the simulations follows equations 4.1-4.2 in Section 4.1.2 with

$$\boldsymbol{\sigma}^2_{\tau_{wiener}} = \begin{bmatrix} 0.1^2 & 0 & 0 & 0 \\ 0 & 0.001^2 & 0 & 0 \\ 0 & 0 & 0.05^2 & 0 \\ 0 & 0 & 0 & 0.01^2 \end{bmatrix} \tag{8.2}$$

$$\boldsymbol{\sigma}^2_{\tau_{noise}} = \begin{bmatrix} 0.01^2 & 0 & 0 & 0 \\ 0 & 0.001^2 & 0 & 0 \\ 0 & 0 & 0.01^2 & 0 \\ 0 & 0 & 0 & 0.001^2 \end{bmatrix} \tag{8.3}$$

8.1 Design matrices

The Q matrix was kept a diagonal matrix for the simulations and decided by tuning with the initial guess as discussed in **??**.

$$\mathbf{Q}^{ekf1}_{12x12} = diag\{\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3\} \tag{8.4}$$

$$\mathbf{Q}^{ekf2} = 0.2\mathbf{Q}^{ekf1} \tag{8.5}$$

$$\mathbf{Q}_1 = \begin{bmatrix} 1.56e-8 & 0 & 0 & 0 \\ 0 & 1.56e-8 & 0 & 0 \\ 0 & 0 & 1.95e-9 & 0 \\ 0 & 0 & 0 & 3.91e-8 \end{bmatrix} \tag{8.6}$$

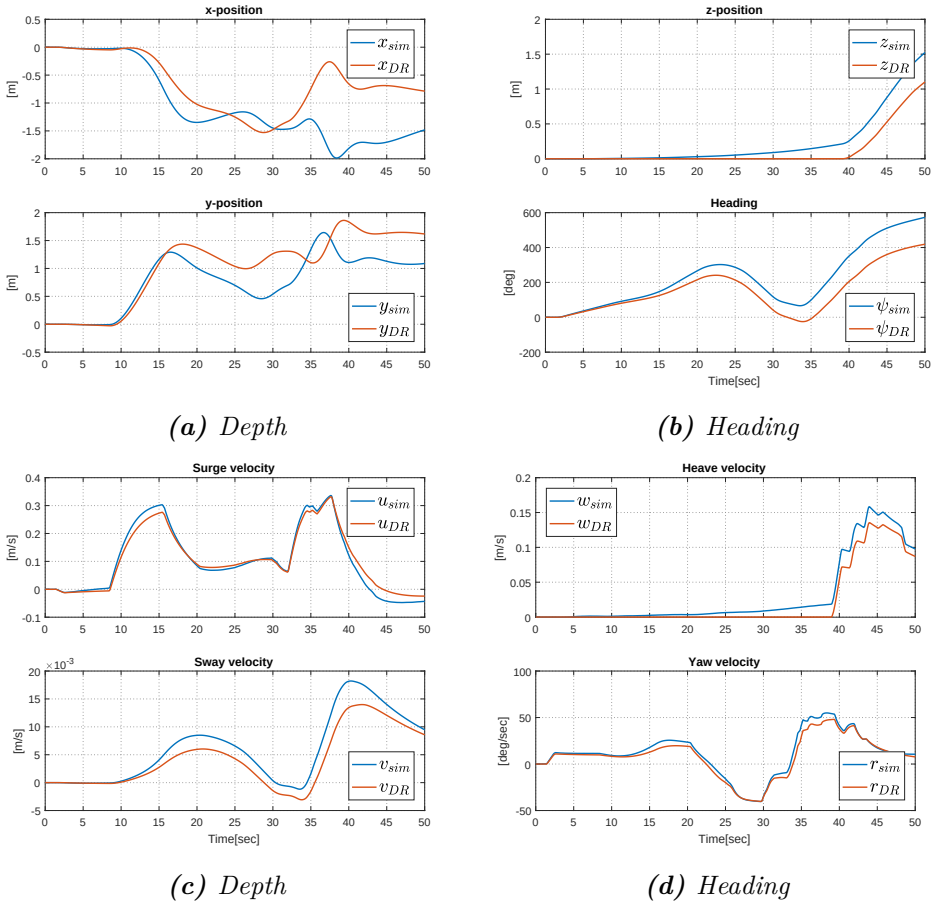
$$\mathbf{Q}_2 = \begin{bmatrix} 1.875e-5 & 0 & 0 & 0 \\ 0 & 6.25e-9 & 0 & 0 \\ 0 & 0 & 1.5625e-6 & 0 \\ 0 & 0 & 0 & 6.25e-8 \end{bmatrix} \tag{8.7}$$

$$\mathbf{Q}_3 = \begin{bmatrix} 0.0013 & 0 & 0 & 0 \\ 0 & 2.5e-7 & 0 & 0 \\ 0 & 0 & 1.875e-4 & 0 \\ 0 & 0 & 0 & 5e-6 \end{bmatrix} \tag{8.8}$$

$$\mathbf{R}_{9x9} = \begin{bmatrix} \mathbf{R}_{5x5}^{pos} & \mathbf{0}_{5x4} \\ \mathbf{0}_{4x5} & \mathbf{R}_{4x4}^{imu} \end{bmatrix} \tag{8.9}$$

$$\mathbf{R}_{5x5}^{pos} = \begin{bmatrix} 0.027 & 0 & 0 & 0 & 0 \\ 0 & 0.027 & 0 & 0 & 0 \\ 0 & 0 & 0.027 & 0 & 0 \\ 0 & 0 & 0 & 3.42e-5 & 0 \\ 0 & 0 & 0 & 0 & 5.05e-4 \end{bmatrix} \tag{8.10}$$

$$\mathbf{R}_{4x4}^{imu} = \begin{bmatrix} 5.29e-6 & 0 & 0 & 0 \\ 0 & 3.32e-6 & 0 & 0 \\ 0 & 0 & 1.95e-5 & 0 \\ 0 & 0 & 0 & 9.11e-7 \end{bmatrix} \tag{8.11}$$



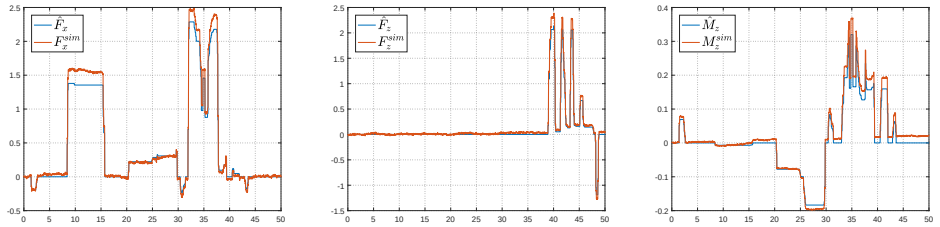
(a) *Depth* ***(b)*** *Heading*

(c) *Depth* ***(d)*** *Heading*

Figure 8.1: *Simulated true movement with process noise versus ROV model dead reckoning*

8.2 Dead-reckoning performance

To demonstrate the effects of the simulated process noise that represents modelled uncertainties, Figure **??** shows the difference in position in respect to a simulation without process noise which is the model used in the Kalman Filter and would thus be the dead-reckoning. Figure 8.2 also shows the difference in the simulated thrust with and without process noise. The simulation used for dead-reckoning uses the same power reference signals as Case 1 and Case 2.



(a) F_x *(b)* F_z *(c)* M_z

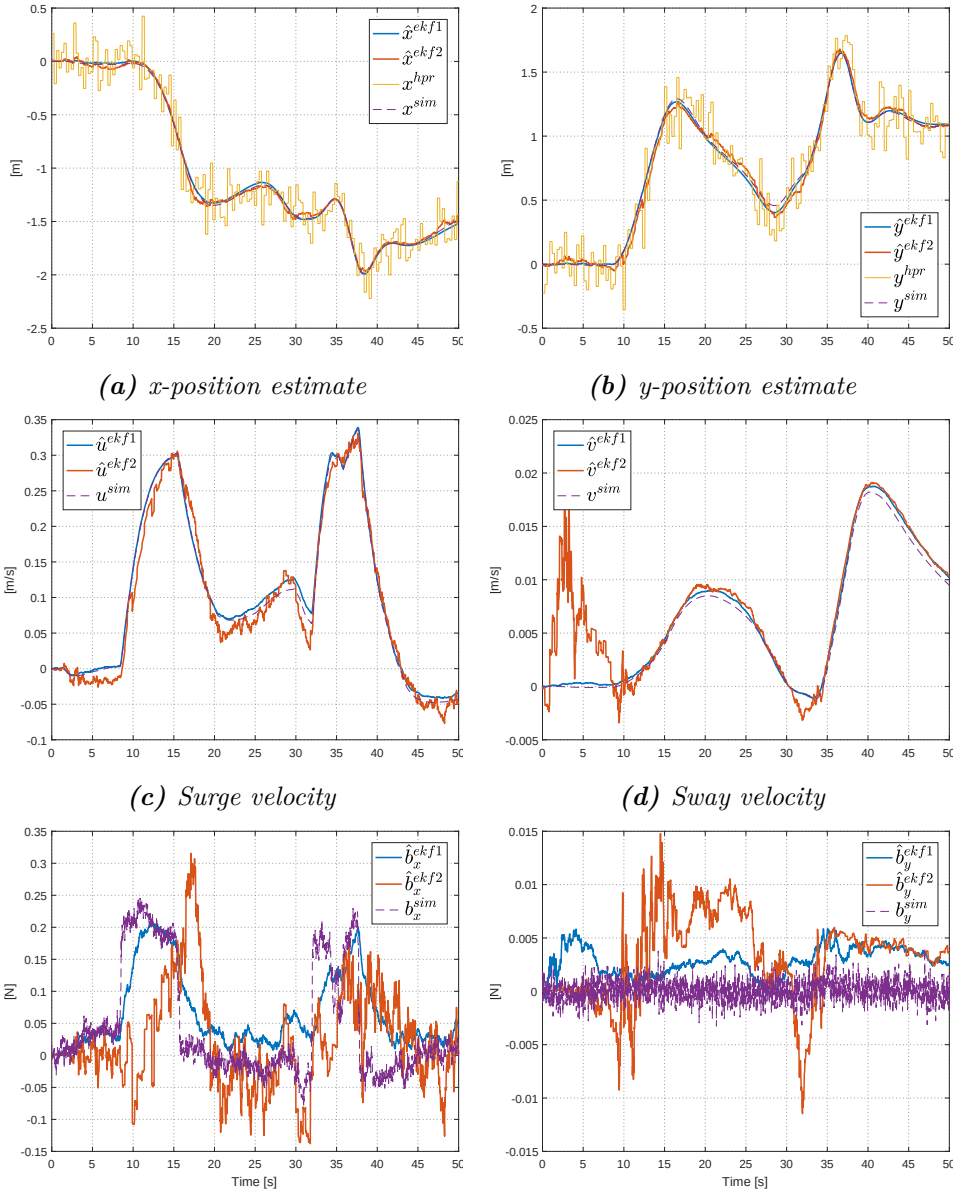
Figure 8.2: *Simulated thrust acting on the ROV in body coordinates with process noise versus thrust allocation estimate*

8.2.1 Discussion of dead-reckoning

As can be seen from Figure **??** there is quite a big difference between the model dead-reckoning and the true simulated movement of the ROV with the simulated process noise as shown in Figures 8.2. Most affected are the movements in the x-y directions as seen in Figure 8.1a. The difference in velocities are not as different as can be seen in Figures 8.1c-8.1d. This shows that the impact a wrong heading-estimate has on the estimates for x-y positions. This also demonstrates potential importance of covariance elements in the design matrix **Q**.

8.3 Case 1: Fault-free observer performance

The simulation is initialized by running *main_SIMU.m*, and activating *fault-sim_param.m* which takes input from the power signals in *PWM_ref_case5.mat*.



(a) *x-position estimate*

(b) *y-position estimate*

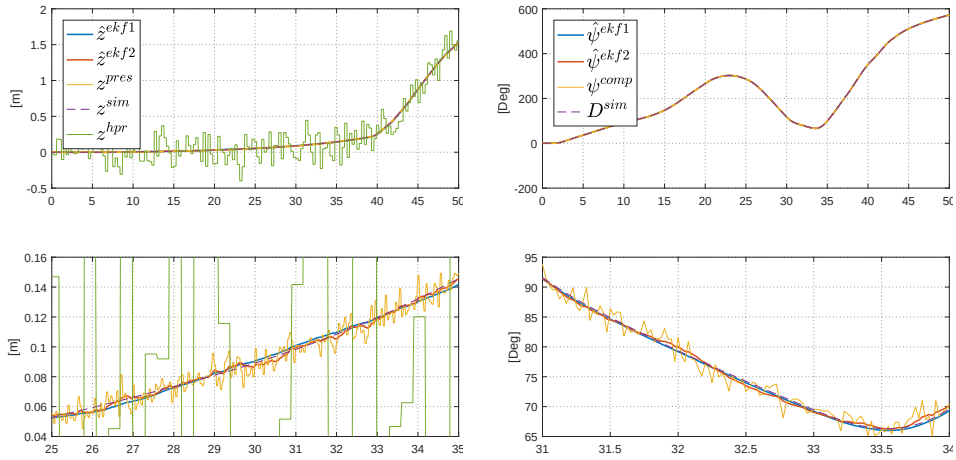
(c) *Surge velocity*

(d) *Sway velocity*

(e) *Surge bias*

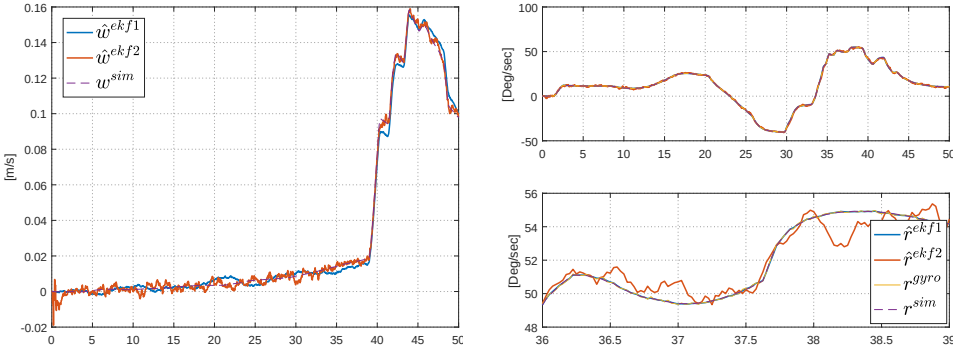
(f) *Sway bias*

Figure 8.3: *Case 1: Surge and sway state estimates*



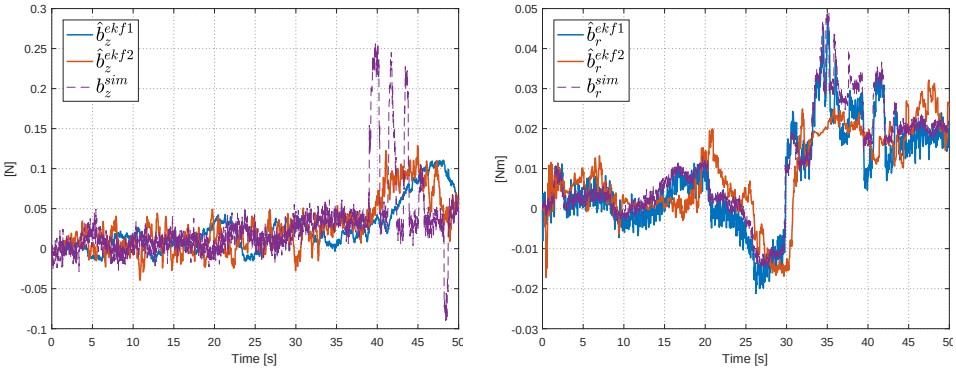
(a) *z-position with zoom*

(b) *Heading estimate with zoom*



(c) *Heave velocity*

(d) *Yaw velocity*



(e) *Heave bias*

(f) *Yaw bias*

Figure 8.4: *Case 1: Heave and Yaw estimates*

8.3.1 Discussion of fault-free performance

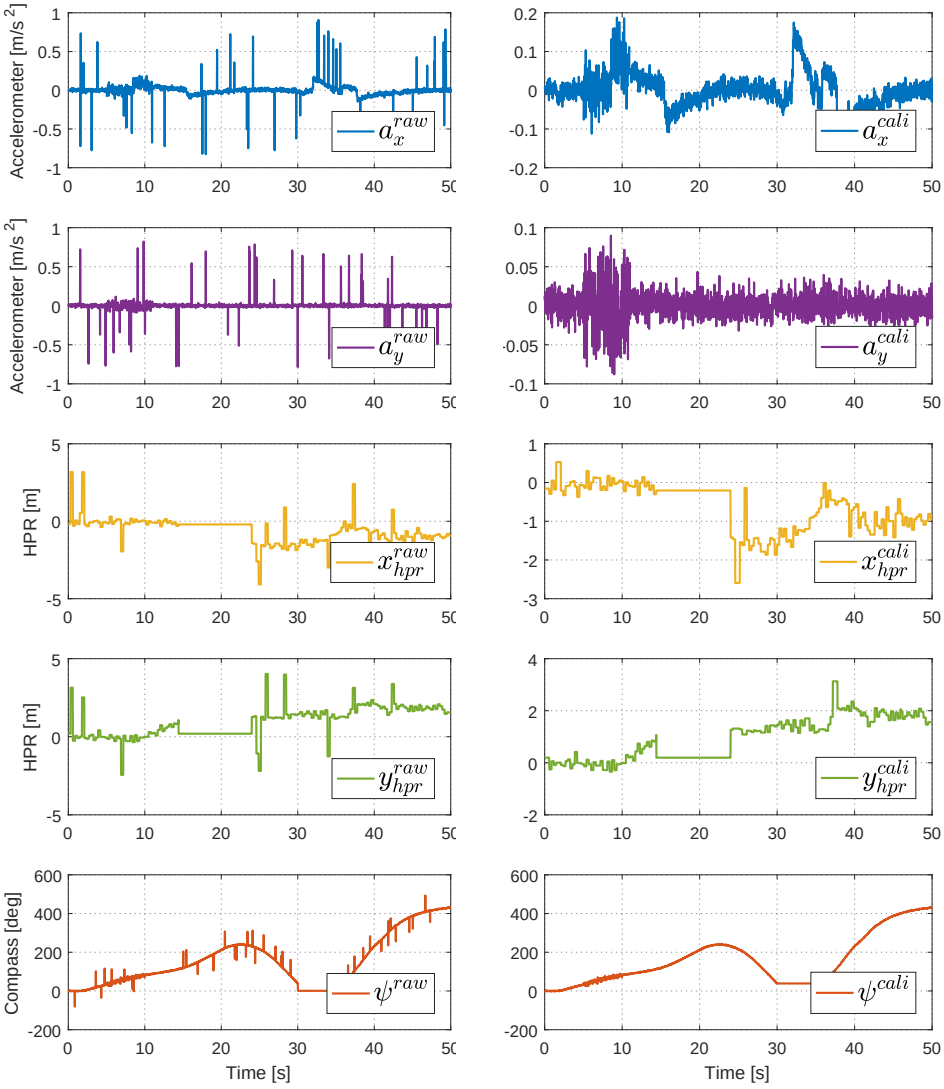
As can be seen from the plots in Figures 8.3-8.4 the general performance of the observer is good. The main filter which includes all the sensors is outperforming the filter using just position measurements in all cases, which verifies that the sensor fusion by including the IMU measurements improve the performance. The only exception is the heave velocity seen in Figure 8.4c. Even though the heave velocity is smoother, it also deviates more than *ekf2* does in several places. It is possible this could be improved by further tuning of the $\mathbf{Q}$ matrix. The estimated biases are also very close to the simulated values for the main filter, in particular for yaw bias as seen in Figure 8.4f. As the gyro rate measurement can be used directly in filter instead of being transformed to a pseudo velocity measurement by integration, this makes sense. In addition to this the general noise of the compass is very low, so the estimation of the yaw bias is good for both filters.

8.4 Case 2: Outliers on all measurements, signal freeze on HPR

The same simulation as in Case 1 is used. So the simulation has the same model performance and thrust as in Figures **??** and 8.2. The faults used in this simulation are defined by the *faultsim_param_case2* script, and the simulation is started by running *main_sim_case2*. The observer was subjected to the following sensor faults:

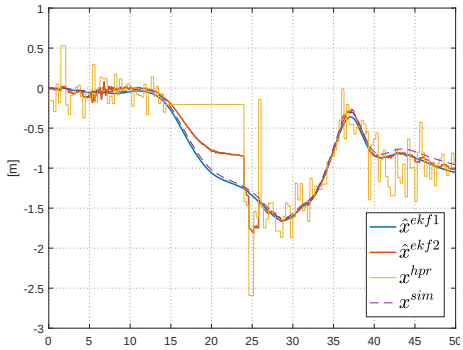
- Outliers on HPR, compass and accelerometer
- HPR signal freeze in time interval $[14.4, 24]$
- Compass freeze in time interval $[30, 35]$
- HPR high variance in time interval $[39, 46]$ with $\sigma^2_{highvar} = 0.06$
- Compass high noise in time interval $[5, 10]$ with $\sigma^2_{highvar} = 0.01$
- Accelerometer high variance in the time interval $[5, 11]$ with $\sigma^2_{highvar} = 0.0016$

The results will not be shown for the heave estimates, as the sensors affecting heave are not subjected to faults. Many of the simulated faults are dealt with by the signal processing module, so this will be shown as well.

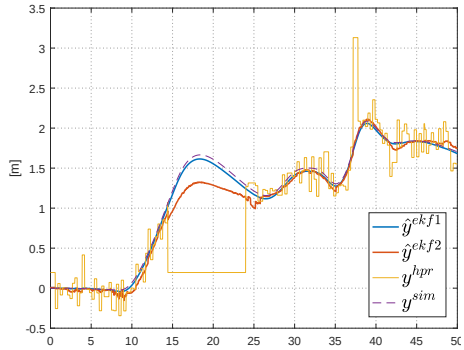


(a) Raw measurements *(b) Signal processed measurements*

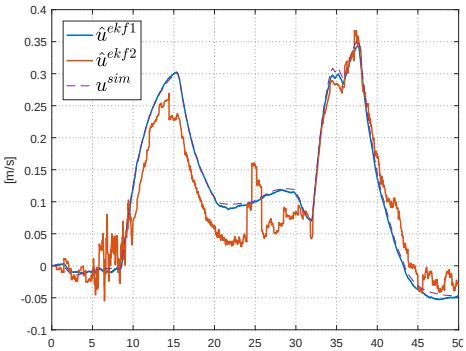
Figure 8.5: *Case 2: Faulty measurements versus signal processed measurements*



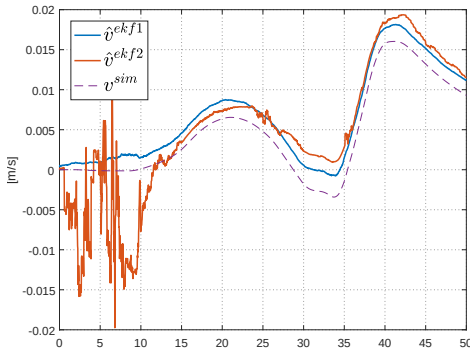
(a) x-position estimate



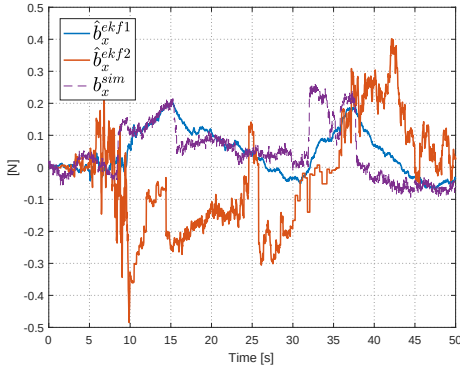
(b) y-position estimate



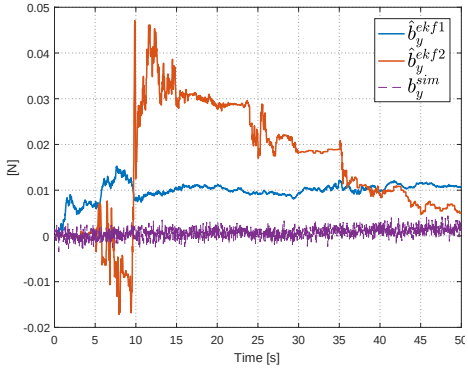
(c) Surge velocity



(d) Sway velocity

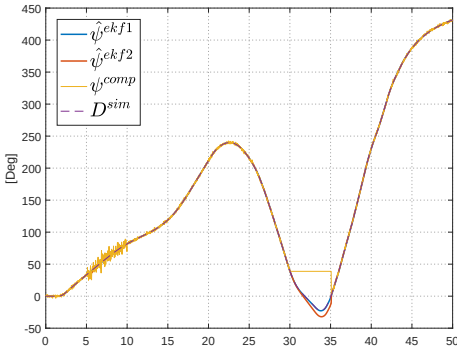


(e) Surge bias

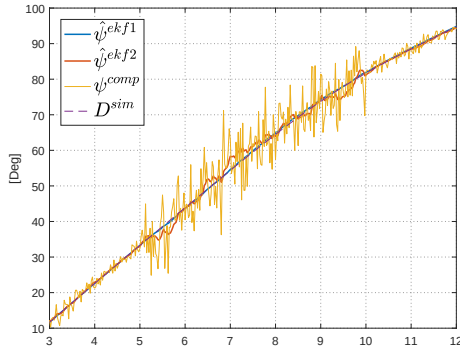


(f) Sway bias

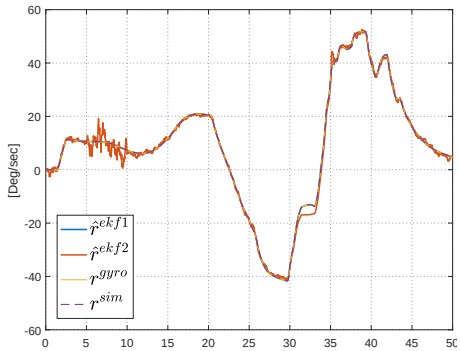
Figure 8.6: *Case 2: Surge and sway*



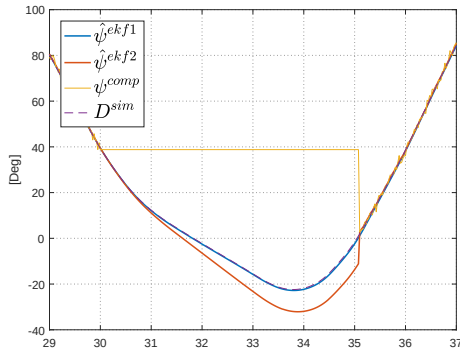
(a) Heading estimate



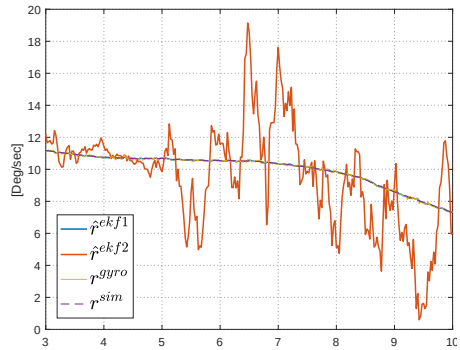
(b) Heading estimate zoom



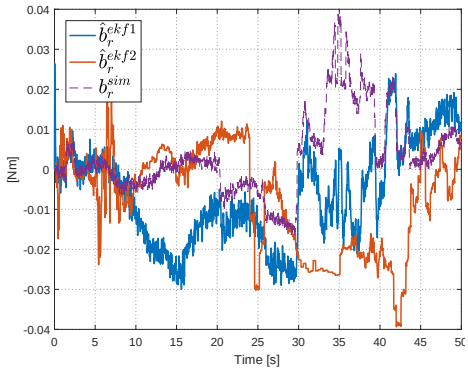
(c) Yaw velocity



(d) Heading estimate zoom2

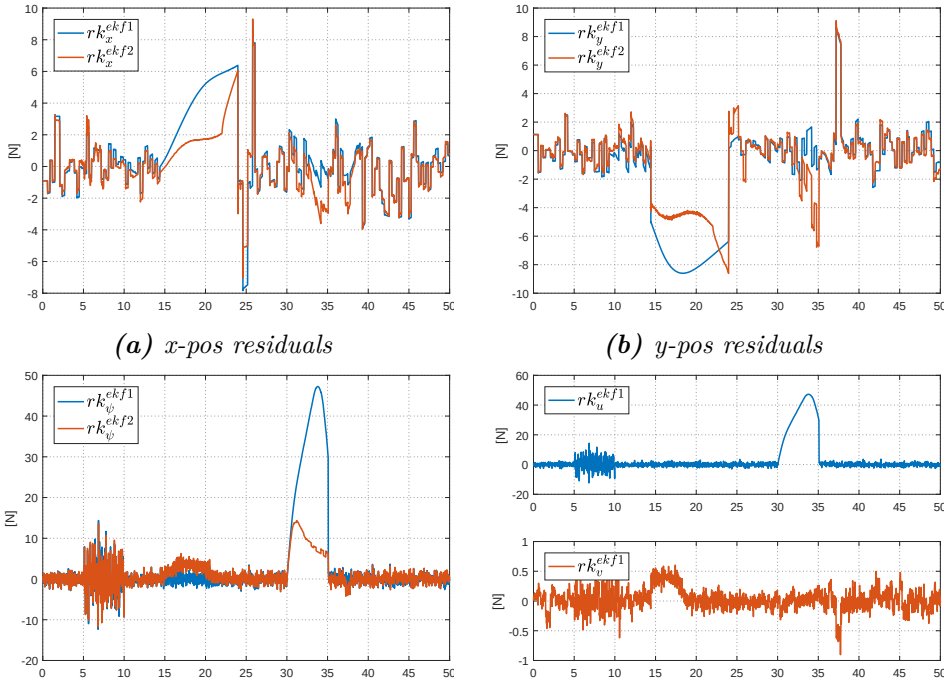


(e) Yaw velocity zoom



(f) Yaw bias

Figure 8.7: *Case 2: Heading*



(a) *x-pos residuals*

(b) *y-pos residuals*

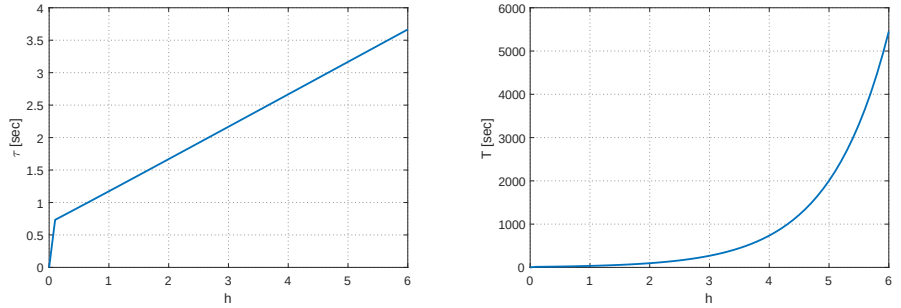
(c) *Heading residuals*

(d) *u and v residuals for ekf1*

Figure 8.8: *Normalized Residuals*

8.4.1 Case 2: Discussion

As can be seen from the signal processing in Figure **??** the signal processing module is performing well by removing most outliers. Some of the outliers on the HPR measurements are sent through to the filter, which suggests to decrease the a parameter for the HPR outlier detection slightly. As for the state estimates, again the main filter using all sensor measurements is outperforming the one using just position estimates. Both filters enters dead-reckoning mode when the signal freezes occurs for both HPR and compass, but as seen from Figures 8.6a-8.6b *ekf1* outperforms *ekf2* during dead-reckoning. *ekf2* is affected more strongly by the outliers that got through signal processing as well. Also during high variance for the compass the main filter is unaffected due to its gyroscope measurements while *ekf2* becomes more noisy as can be seen from Figure 8.7b. This is also reflected with a more noisy estimate for the yaw rate in Figure 8.7e. By comparing these results to the fault-free ones in Case 1, one can see that the main filter using all sensors have a similar performance. This shows that the main filter is robust towards the simulated sensor faults. The filter using just position measurements is less robust, but still capable. Figure 8.8 shows how the normalized residuals are affected by the sensor faults. The residuals are affected by all the sensor faults, becoming more noisy in the high noise time intervals. During signal freeze the corresponding residuals deviate very far from their mean at 0. While during fault-free periods the normalized residuals show themselves as normally distributed variables with a variance of about 1. This shows that the method of detecting failure modes with the CUSUM-test on normalized residuals would trigger during signal freeze and possibly during high variance as well. The CUSUM-test alone is thus not sufficient to determine if a fault is a signal bias/drift or a signal freeze.



(a) *Mean detection time* ***(b)*** *Mean time between false alarms*

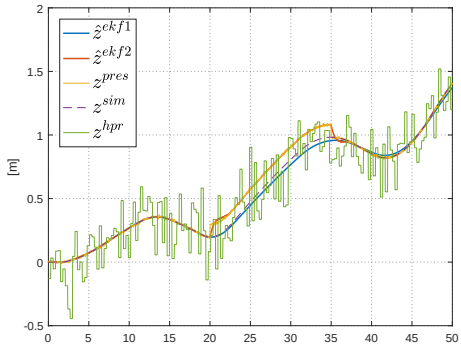
Figure 8.9: *Cusum threshold variable*

8.5 Case 3: Sudden bias on depth measurement

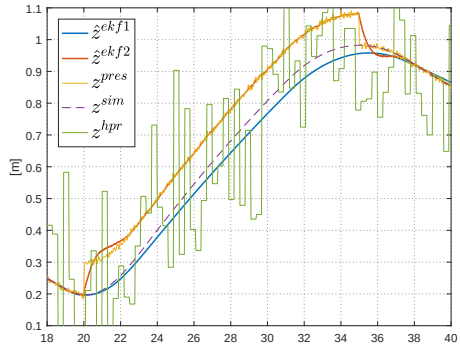
Case 3 is testing the sudden bias detection implemented by using the CUSUM-test. The simulation is run through *main_SIMUcase3.m* which runs a different simulink model called *Observer_simulated_cusum.slx* where the CUSUM-fault detection block was implemented. Case also loads a different power reference signal saved as *PWM_ref_heave2.mat*. The sudden bias is defined by the script *faultsim_param_case3.m*. The sudden bias is introduced in the time interval $[20, 35]$ and has an amplitude of $0.1m$. The CUSUM test design variables for the pressure sensor was set to

$$H_0 : \quad r_k^{pres} \sim \mathcal{N}(0, 1) \tag{8.12}$$
$$H_1 : \quad r_k^{pres} \sim \mathcal{N}(2, 1). \tag{8.13}$$

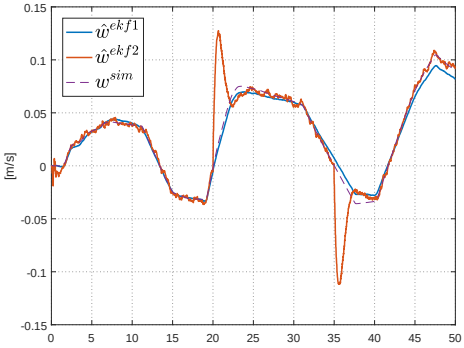
The threshold variable h was set to 4, based on mean detection time and mean time between false alarm shown in Figure **??**



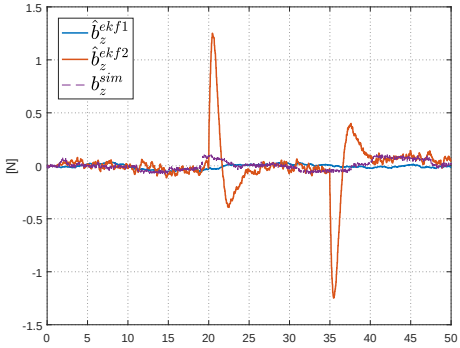
(a) z-position



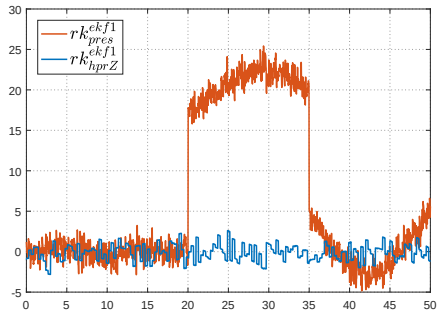
(b) z-position zoom



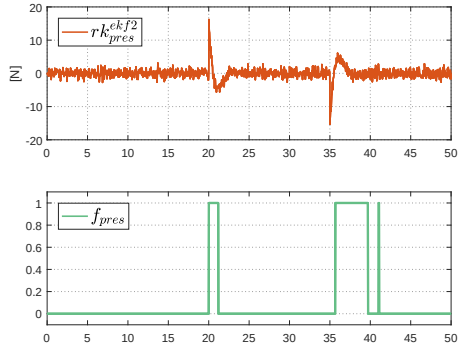
(c) Heave velocity



(d) Heave bias



(e) Normalized residual and CUSUM decision function



(f) Sway bias

Figure 8.10: *Case 2: Surge and sway*

8.5.1 Case 3: Discussion

As can be seen from Figure 8.10f the CUSUM-test detects the sudden bias instantly. It is detected so fast that the main filter is not affected by the sudden bias at all, as the measurements from the pressure sensor are ignored when the bias hits. There is a slight decrease in performance for the main filter as you can see it deviates slightly from the simulated real depth. It is still very smooth, but it does drift slightly away from the real value. This is because the **Q** matrix is tuned for using the pressure sensor. **Q** is tuned in relation to **R**, and the variance of the pressure sensor is a lot lower than the variance of the HPR. The drift could possibly be avoided by increasing the values in **Q** to better fit the variance of the HPR depth measurement, but this could also create a less smooth estimate. When the sudden bias is gone at the 35 second mark, the CUSUM-test triggers again. This could possibly be used for reactivating the pressure sensor measurements in the main kalman filter, but additional tests would have to be done to confirm that the bias is gone. As can be seen Figure 8.10f the CUSUM test also does a false trigger once around the 42 second mark. The designed H_1 hypothesis used in the CUSUM-test has a mean of 2, which is very low as can be seen for the jump in the normalized residuals for *ekf2* which jumps to values of 15 when the bias is introduced to the pressure sensor. By choosing new CUSUM-test parameters, the false alarm could be avoided. But the results shows that the fault detection module of the observer works for detecting a sudden bias jump. The false alarm however shows that the CUSUM-test can be designed to be very sensitive, making it possible to use for detection of small signal drifts as well. However, this does cause a potential problem with false alarms.

8.6 Overall discussion of simulation results

Case 1 shows that the observer performance in a fault-free environment is good for both filters for 4 DOF. The main filter using all sensor measurements have better estimates overall, particularly for the velocity estimates. Case 2 shows the fault-tolerance of the observer subjected to high variance, signal freeze and outliers, which is also good. Here you can see how the main filter has improved fault-tolerance in comparison to the filter using just position measurements. Case 2 also demonstrates how the normalized residuals are affected by high variance, signal freeze and outliers. This shows that using the CUSUM-test for detection of signal bias or drift will

potentially trigger for the other failure modes as well, making it necessary to develop additional algorithms to decide the nature of the fault as the main filter handles sensor bias and drift differently than . Case 3 shows that the CUSUM-test can be used for detection of signal bias, and that the decision made by the fault detection module lets the main filter remain unaffected by the sudden bias. It also shows the sensitivity of the normalized residual generation for the current tuning.

Chapter 9

Experimental Results

This chapter covers the experimental results and discussion of these results. The first section covers the design parameters of the Extended Kalman Filter. The results are then divided into cases, and will be presented in each of the following. The same setup as in the simulation result is used, with the same two filters with measurement vectors according to (8.1) in the previous section. In the experimental case, no reference signal is available for the velocities and biases as in the simulated case. The reference used as the true reference state for x-y positions is the QUALISYS system, while the compass and pressure sensor are used as references by themselves as they have low. The implementation in Matlab can be found in Appendix B.2

9.1 Design matrices

The design matrix $\mathbf{Q}$ was retuned for the experimental results, but was still chosen to be a diagonal matrix. In addition to this, the noise levels of the IMU was higher for the experimental tests than the calculated values from Chapter 6. In particular the a_y measurement was deteriorating for the observer, and the accelerations in this DOF is expected to be close to zero as P2-Alpha has no sway thrusters and the test basin does not have a current. This could be because of bad calibration, or that the roll and pitch measurements used directly in the signal processing module for removing the gravity are too inaccurate. As the accelerations from the gravity is a lot higher than the actual accelerations from movement, small errors in the

pitch and roll angle can have a strong effect on the accelerometer measurements. This was solved by increasing the values in $\mathbf{R}$ for the accelerometer pseudo measurements by a factor of 10 for σ_{ax}^2 and σ_{az}^2, and a factor of 100 for σ_{ay}^2.

$$\mathbf{Q}_{12x12}^{ekf1} = 10^{-4} diag\{\mathbf{Q}_1^{ekf1}, \mathbf{Q}_2^{ekf1}, \mathbf{Q}_3^{ekf1}\} \tag{9.1}$$

$$\mathbf{Q}_1^{ekf1} = \begin{bmatrix} 0.675 & 0 & 0 & 0 \\ 0 & 0.675 & 0 & 0 \\ 0 & 0 & 0.0017 & 0 \\ 0 & 0 & 0 & 0.2527 \end{bmatrix} \tag{9.2}$$

$$\mathbf{Q}_2^{ekf1} = \begin{bmatrix} 0.0265 & 0 & 0 & 0 \\ 0 & 0.1664 & 0 & 0 \\ 0 & 0 & 0.0979 & 0 \\ 0 & 0 & 0 & 0.0046 \end{bmatrix} \tag{9.3}$$

$$\mathbf{Q}_3^{ekf2} = \begin{bmatrix} 0.005 & 0 & 0 & 0 \\ 0 & 0.005 & 0 & 0 \\ 0 & 0 & 0.005 & 0 \\ 0 & 0 & 0 & 0.005 \end{bmatrix} \tag{9.4}$$

$$\mathbf{Q}_{12x12}^{ekf2} = diag\{\mathbf{Q}_1^{ekf2}, 10^{-3}\mathbf{Q}_2^{ekf2}, 10^{-3}\mathbf{Q}_3^{ekf2}\} \tag{9.5}$$

$$\mathbf{Q}_1^{ekf2} = 0.4\mathbf{Q}_1^{ekf1} \tag{9.6}$$

$$\mathbf{Q}_2^{ekf1} = \begin{bmatrix} 0.2 & 0 & 0 & 0 \\ 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0.2 \end{bmatrix} \tag{9.7}$$

$$\mathbf{Q}_3^{ekf2} = \begin{bmatrix} 0.002 & 0 & 0 & 0 \\ 0 & 0.002 & 0 & 0 \\ 0 & 0 & 0.002 & 0 \\ 0 & 0 & 0 & 0.0002 \end{bmatrix} \tag{9.8}$$

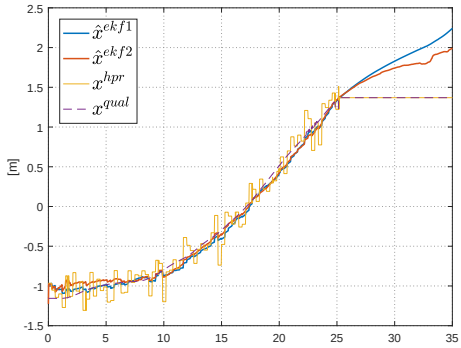
$$\mathbf{R}_{9x9} = \begin{bmatrix} \mathbf{R}_{5x5}^{pos} & \mathbf{0}_{5x4} \\ \mathbf{0}_{4x5} & \mathbf{R}_{4x4}^{imu} \end{bmatrix} \tag{9.9}$$

$$\mathbf{R}_{5x5}^{pos} = \begin{bmatrix} 0.027 & 0 & 0 & 0 & 0 \\ 0 & 0.027 & 0 & 0 & 0 \\ 0 & 0 & 0.027 & 0 & 0 \\ 0 & 0 & 0 & 3.42e-5 & 0 \\ 0 & 0 & 0 & 0 & 5.05e-4 \end{bmatrix} \tag{9.10}$$

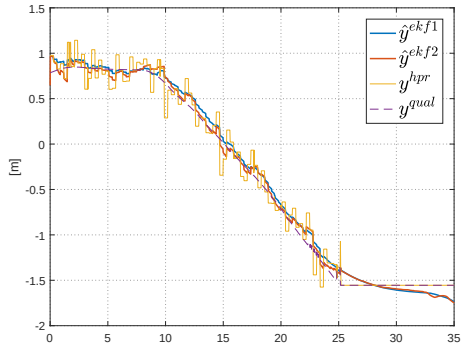
$$\mathbf{R}_{4x4}^{imu} = \begin{bmatrix} 10 * 5.29e-6 & 0 & 0 & 0 \\ 0 & 100 * 3.32e-6 & 0 & 0 \\ 0 & 0 & 10 * 1.95e-5 & 0 \\ 0 & 0 & 0 & 9.11e-7 \end{bmatrix} \tag{9.11}$$

9.2 Case 1: Slow-driving

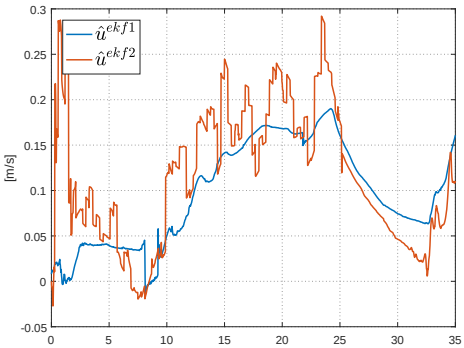
Case 1 shows the filter performance for a run where the ROV started at the bottom and drove to the edge of the test basin for pickup. It did not do much advanced movement, as to show the filter performance for a case where the non-linear terms of the ROV model are small.



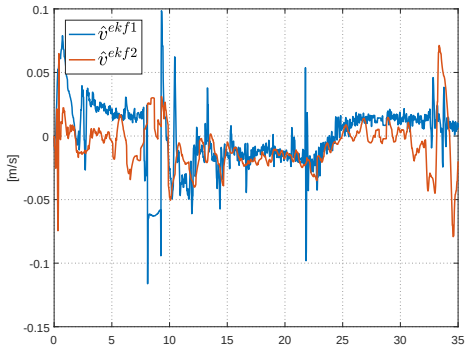
(a) x-position estimate



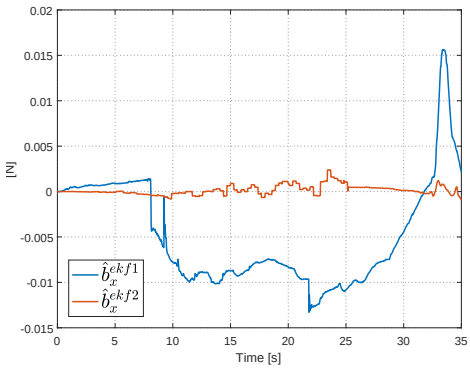
(b) y-position estimate



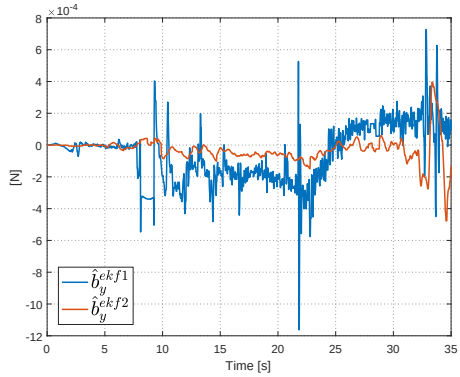
(c) Surge velocity



(d) Sway velocity

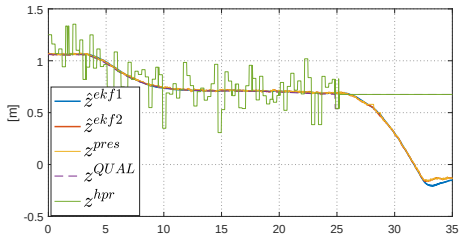


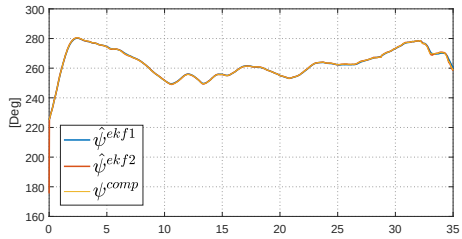
(e) Surge bias

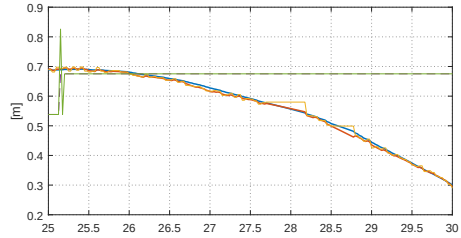


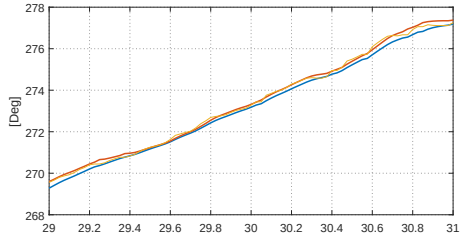
(f) Sway bias

Figure 9.1: *Case 1: Surge and sway*

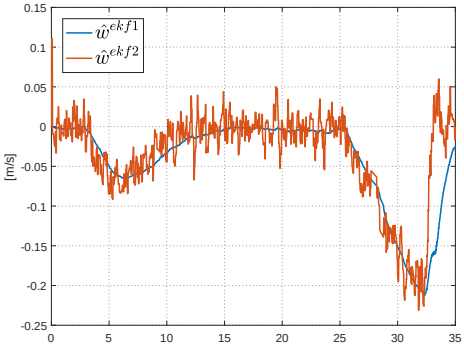


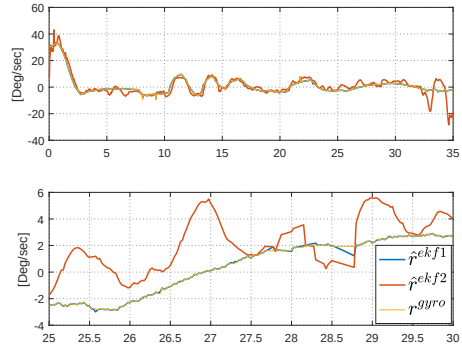




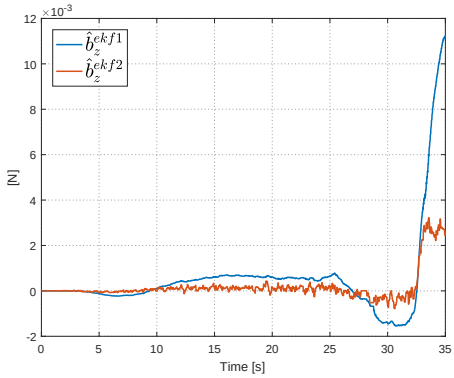


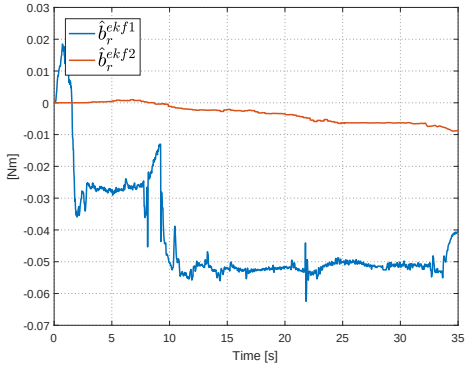
(a) *z-position* ***(b)*** *Heading*





(c) *Heave velocity* ***(d)*** *Yaw velocity*



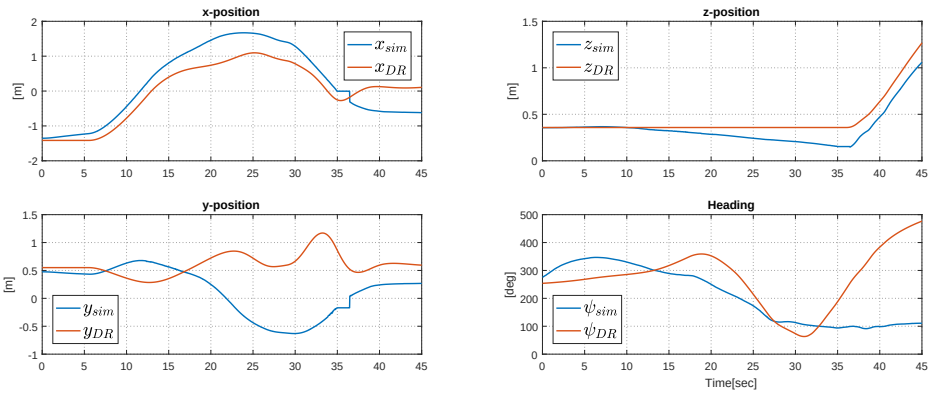


(e) *Heave bias* ***(f)*** *Yaw bias*

Figure 9.2: *Case 1: Heave and yaw*

9.2.1 Case 1: Discussion

The observer performance is quite good for this case. As can be seen from the compass estimates in Figure 9.2b, the heading angle is kept within the interval of $[240, 280]$. It shows that when the heading angle is kept within a sector, the performance is good. The x-y position estimates in 9.1 are good improvements upon the unfiltered HPR measurements and keeps true to the Qualisys reference. The same can be said about the z-position and heading estimate in 9.2. By inspecting the zoomed plot in Figure 9.2a you can also see that there is a small signal freeze in the pressure sensor at time 27.5, and that both filters are unaffected by this. Further by inspecting the slopes of the position measurements and the estimates velocities in heave and surge, it shows that the velocity estimates are very good. It also shows how the main filter using all sensor information has much smoother velocity measurements. A bit after the 30 second mark the ROV hits the surface and the heave velocity plummets towards zero. Before reaching the surface the ROV also went out of the Qualisys area and there was a signal freeze for the HPR measurements, which both filters dealt with by entering dead-reckoning mode. But there is no reference to see how good this dead-reckoning is.



(a) x-y position *(b) Heading and z-position*

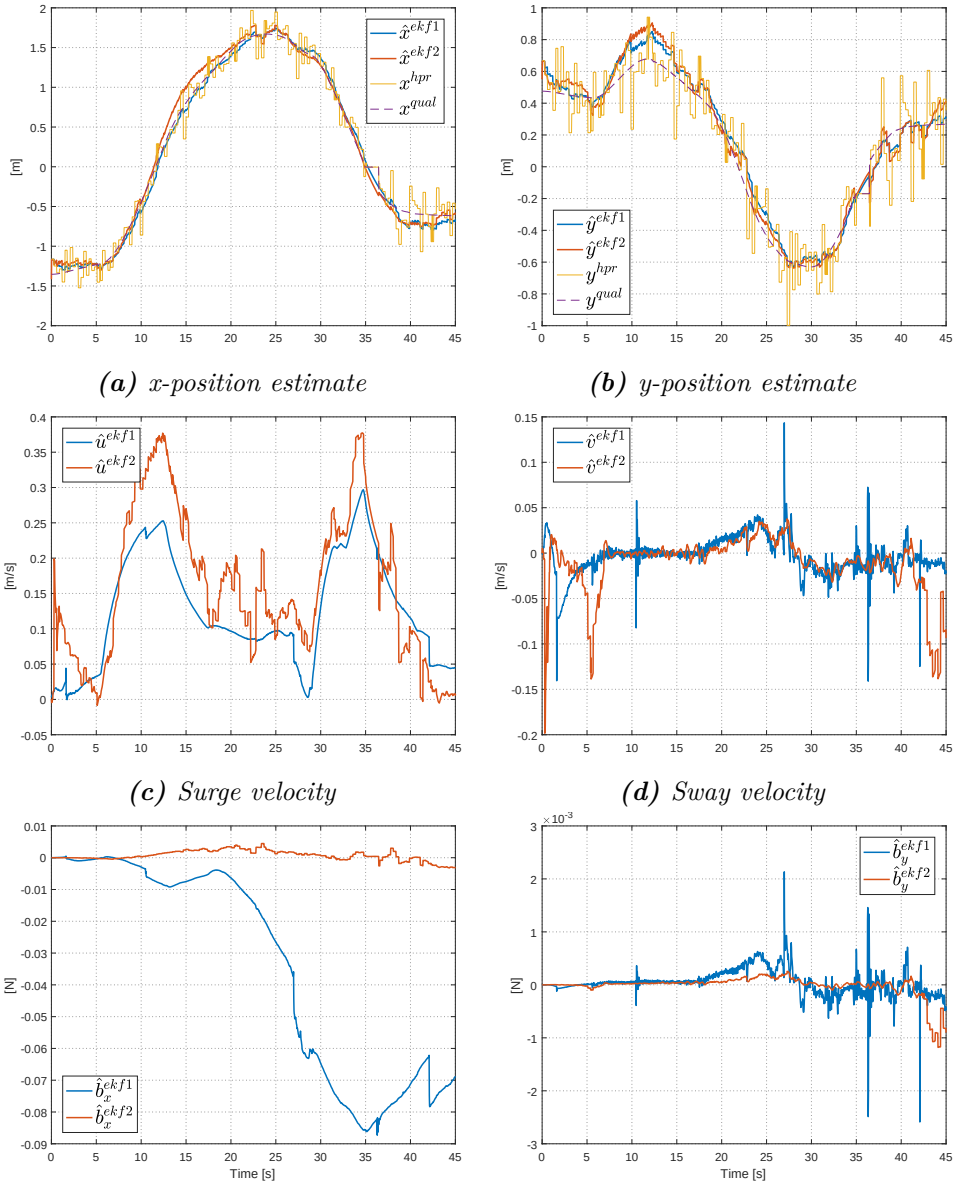
Figure 9.3: *Model dead-reckoning performance*

9.3 Case 2: Nominal performance

In this case the ROV was driven around more roughly than in Case 1. It is the same run that was simulated for Case 1 and 2 in the simulation results, so it has the same thruster input as seen in Figures 8.2 from Case 1 in the simulation results. The ROSBAG file that was running is called *case5* and can be found in Appendix [x]. In this case we also inspect a full dead-reckoning run, where the filter was initialized for some seconds before entering dead-reckoning mode to get an estimate of the initial conditions. This is shown in Figure 9.3

9.3.1 Case 2: Discussion

From the dead-reckoning in Figure 9.3b we can see that the heading estimates during dead-reckoning are quite off, which ruins the x-y position estimates quite a lot. As the yaw damping parameters used a different method for determination, without using the rig or a force sensor, this could be one of the reasons. The z-position dead-reckoning however is not too bad. It is also worth noting that it is quite similar to the motion done in Case 1 for the Simulations, indicating that the simulated process noise was fitting for the heave estimate. The heave thrusters produce zero thrust the first 35 seconds or so, so the heave motion should have a negative force bias due to the buoyancy not being exactly 0. By looking at the heave bias in Fig-



(a) *x-position estimate*

(b) *y-position estimate*

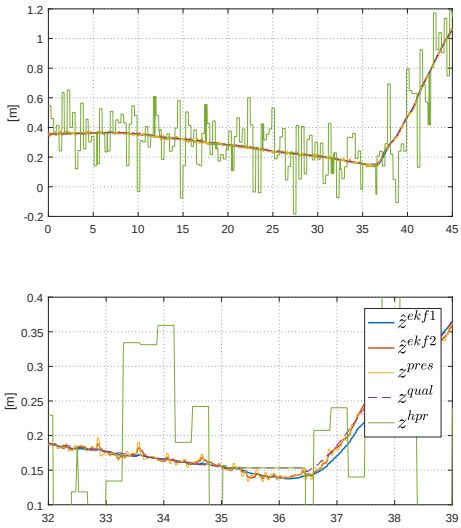
(c) *Surge velocity*

(d) *Sway velocity*

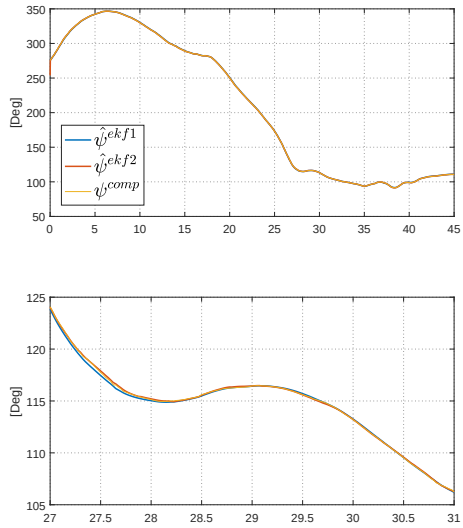
(e) *Surge bias*

(f) *Sway bias*

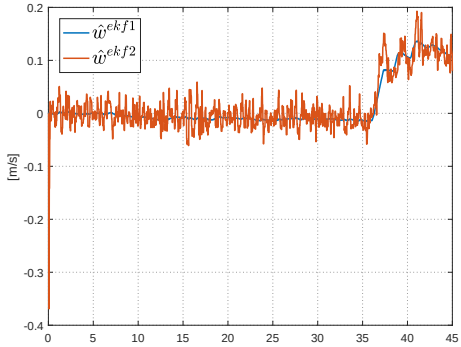
Figure 9.4: *Case 2: Surge and sway*



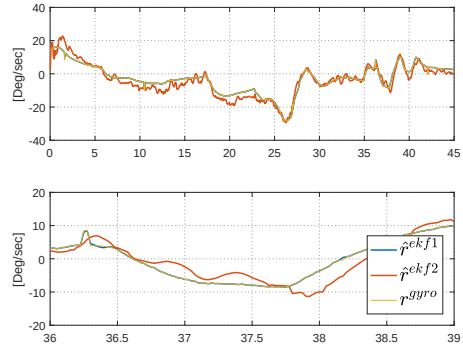
(a) *z-position*



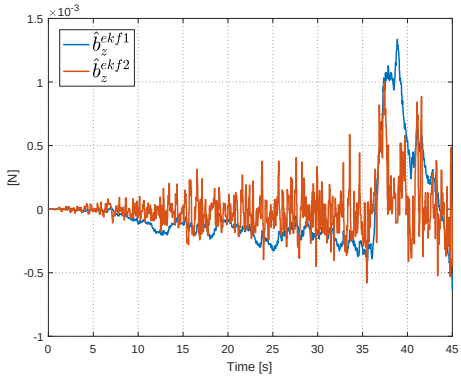
(b) *Heading*



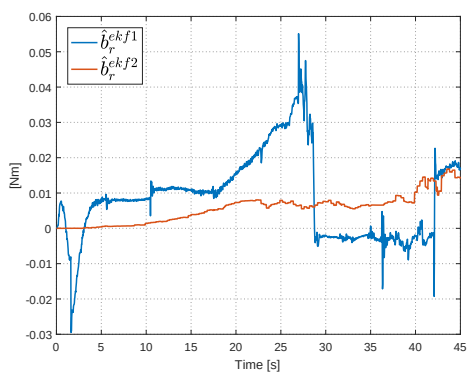
(c) *Heave velocity*



(d) *Yaw velocity*



(e) *Heave bias*



(f) *Yaw bias*

Figure 9.5: *Case 2: Heave and yaw*

ure 9.5e we can see that the main filter estimates a small negative drifting force that corresponds to an unmodelled buoyancy force, which is good. As can be seen from 9.4 and **??** the general observer performance is good. The state-estimates also show that the main filter has smoother estimates in all cases, in particular for velocities as was the case for the simulation study. In surge velocity the main filter has a lower estimate than the filter using just position measurements, which appears to be correct as ekf2 overshoots the x-position estimate at the 10-15 second mark. Both filters overshoot the estimates in y-position during this period as well, but the main filter is still better. By looking back at the surge thruster forces in Figure 8.2 which is the simulated thrust for this case, we can see that there is a sudden jump in the surge thruster forces at the 10 second mark. This indicates that the thruster forces calculated by the thrust allocation module are too high or changes too rapidly.

9.4 Case 3: Subjected to sensor faults

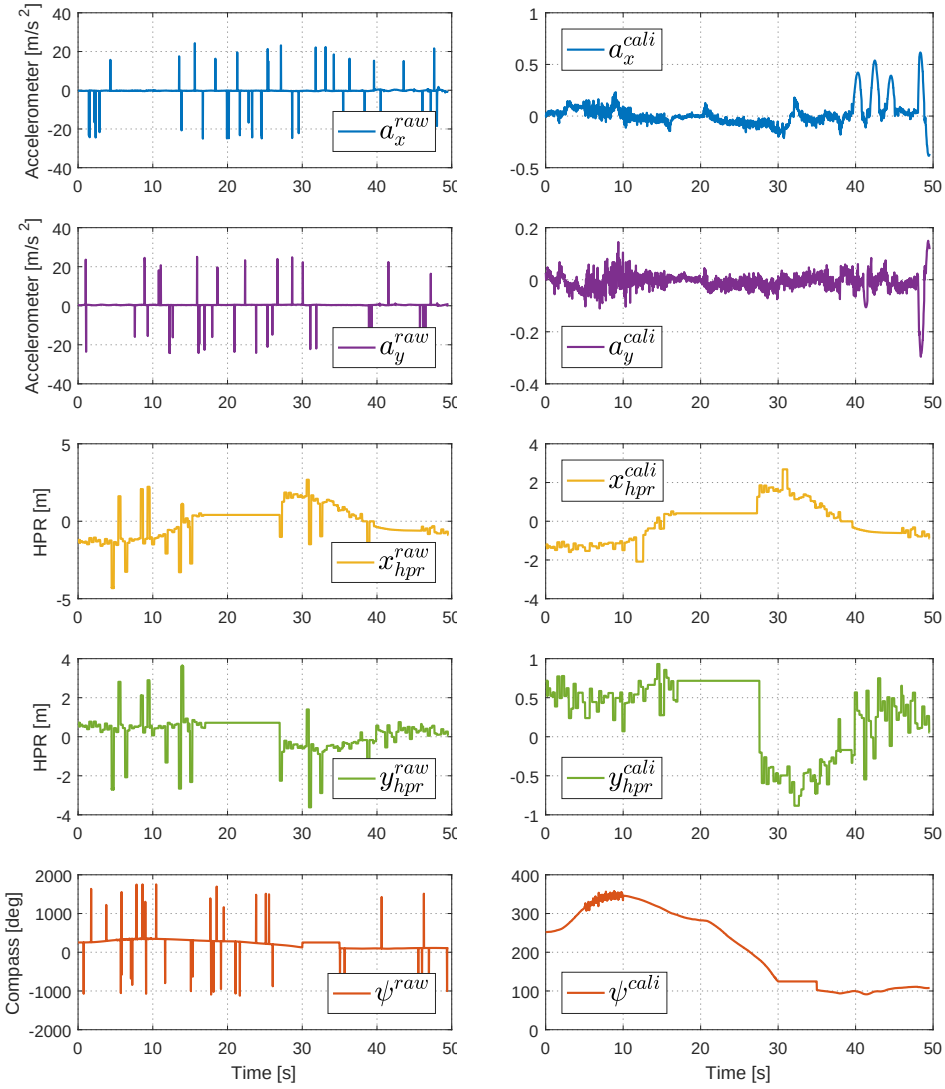
Case 3 ran the same ROSBAG file as Case 2, *Case5* located in Appendix [x]. Thus the same calculated control forces are entering the observer as shown in 8.2 for the simulated case. But in this case the following faults are simulated on top of the real data

- Outliers on HPR, compass and accelerometer
- HPR signal freeze in time interval $[17, 27]$
- Compass freeze in time interval $[30, 35]$
- HPR high variance in time interval $[39, 46]$ with $\sigma^2_{highvar} = 0.06$
- Compass high noise in time interval $[5, 10]$ with $\sigma^2_{highvar} = 0.01$
- Accelerometer high variance in the time interval $[5, 11]$ with $\sigma^2_{highvar} = 0.0016$

The simulated faults are defined in *faultsim_param_case2.m*. First the signal processing module is shown in Figure 9.6, then the state estimates in Figures 9.7 and 9.8.

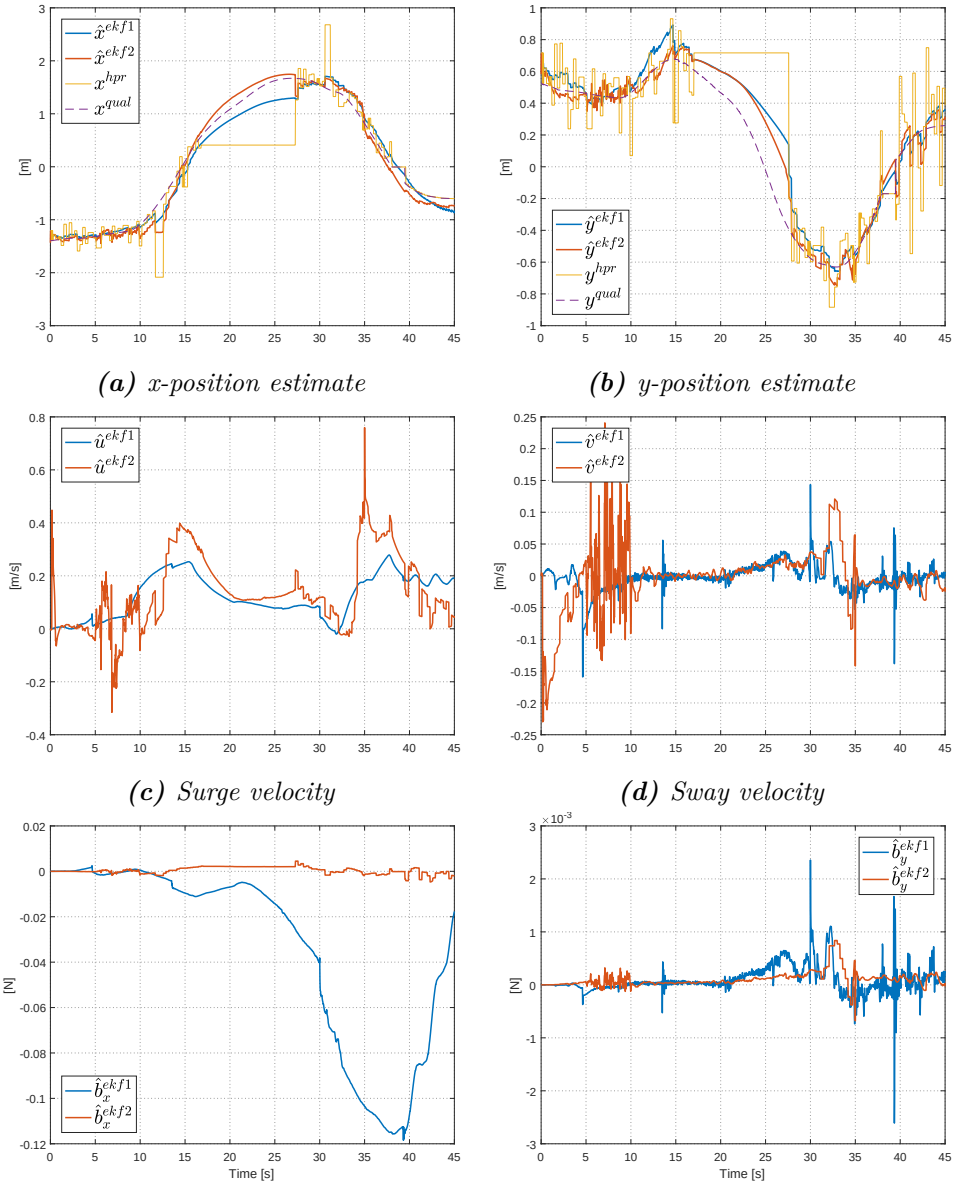
9.4.1 Case 3: Discussion

As can be seen from the signal processing module in 9.6 the outliers are removed from all the measurements, with the exception of two outliers on the HPR measurement for x-direction. From Figure 9.7 we can see that the filter performance for the main filter is reduced quite a lot when introduced to the failure modes. By looking at the surge velocity measurement in Figure 9.7c and comparing it to the fault-free run in Case 2 shown in Figure 9.4c the surge velocity estimates are quite similar. But when both the filters enter dead-reckoning mode for x and y-position, $ekf2$ outperforms the main filter for both state estimates. One reason for this could be the outlier that appeared in the x-measurement from HPR at time 12 that can be seen. This outlier causes both filters to jump their estimate towards the outlier in the negative direction. And during dead-reckoning this time, $ekf2$ hits the Qualisys reference very good during dead-reckoning. While for the fault-free case it overshot the estimate during the same time-period. In Case 2 without the outlier $ekf1$ hit the Qualisys reference better than $ekf2$, but



(a) *Raw measurements* ***(b)*** *Signal processed measurements*

Figure 9.6: *Case 3: Faulty measurements versus signal processed measurements*



(a) *x-position estimate*

(b) *y-position estimate*

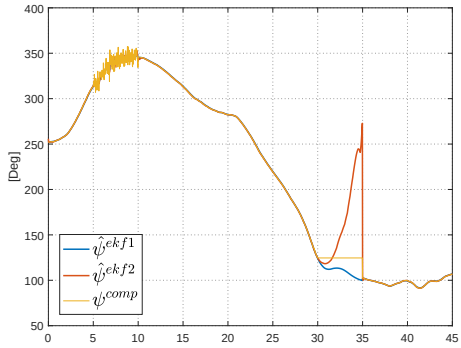
(c) *Surge velocity*

(d) *Sway velocity*

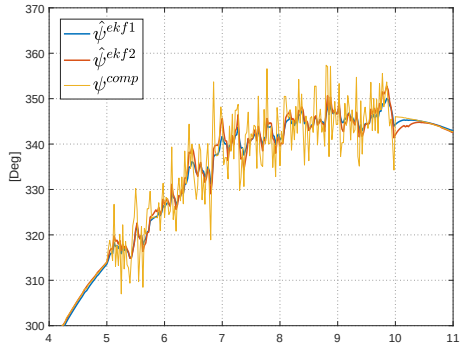
(e) *Surge bias*

(f) *Sway bias*

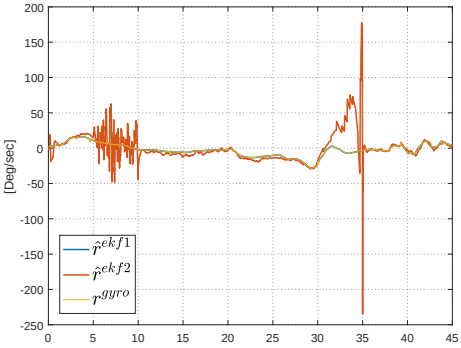
Figure 9.7: *Case 3: Surge and sway*



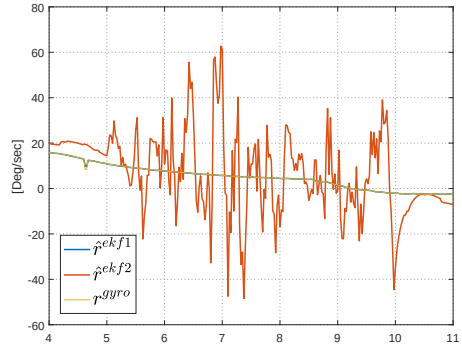
(a) Heading estimate



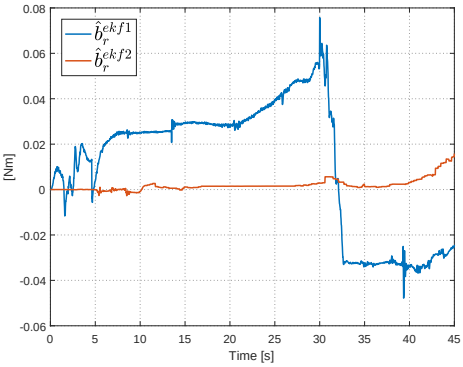
(b) Heading zoom



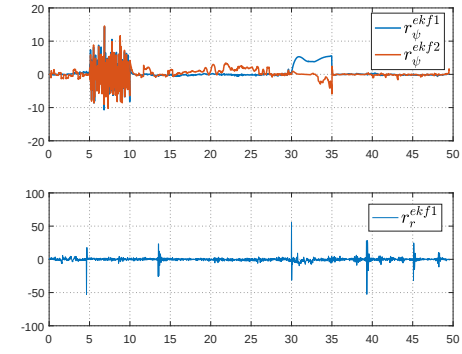
(c) Yaw velocity



(d) Yaw velocity zoom



(e) Heading bias and yaw bias



(f) Residuals

Figure 9.8: *Case 3: Surge and sway*

now undershoots it during dead-reckoning. So it is possible that the outlier at time 12 is the reason for this. For the velocity measurements the $ekf1$ still outperforms $ekf2$. And when it comes to the heading estimates in Figure 9.8 the main filter outperforms $ekf2$ for all states. In particular during the compass signal freeze at time 30, $ekf2$ deviates very far from the actual heading during dead-reckoning. By looking back at Figure 9.3 for Case 2, we can see that the dead-reckoning performance at this time is bad for the heading estimate. As $ekf2$ has no redundant sensor for the heading during a signal freeze, this results in the jump seen in 9.8a. The main filter however can rely on the gyroscope measurements to aid it during dead-reckoning, and thus avoids shooting away the way $ekf2$ does. During the compass high variance interval from time 5 to 10, we can also see how $ekf2$ has reduced performance in all position and velocity estimates, while $ekf1$ is not as badly affected because of the additional information gained from the gyroscope.

9.5 Overall discussion

Case 1 shows that the observer performance is good for both filters for a heading angle that doesn't change too fast and keeps within an interval of 40 degrees. The main filter using all sensor measurements has a better overall performance during Case1, in particular for the velocities. Case 2 shows the filter performance during a more advanced run. The observer performance is still decent, but not as good as for Case 1. This indicates that the observer could be improved by using methods such as the Sectorial Kalman Filter, by dividing the sectors into intervals of 30-40 degrees to gain the performance of Case 1. But the performance of the main filter is still better for Case 2. In Case 3 sensor more sensor faults are introduced. It shows that the fault-tolerance algorithms implemented works, as both filters still function. However, both have a reduced performance. Case 3 also shows that the sensor processing module can do its job, but it also demonstrates how one single outlier can affect the performance of the observer, thus emphasising the importance of good outlier rejection. The main filter has a very good fault-tolerance for heading estimates due to the addition of a gyroscope in Case 3, while $ekf2$ deviates substantially reference values during compass signal freeze. This also indicates that by using the accelerometer as pseudo velocity measurements is a suboptimal approach and that better performance could be achieved by using the accelerometer measurements directly as done by the gyroscope. But this requires another observer model. It is

also worth mentioning that the gyroscope has a lower measurement noise than the accelerometer.

Chapter 10

Conclusion and Further Work

10.1 Conclusion

The main objective of this thesis was to design and implement a 4 DOF fault-tolerant model-based observer for BluEye P2-Alpha. The observer model was based off the non-linear dynamic equation of motion for underwater vehicles, together with thruster characteristics for modelling the control forces acting on the ROV. The thruster characteristics and ROV model parameters were decided through towing tests performed in MC-lab. To this end a rig was used to attach the ROV to the towing carriage in MC-lab, and a force sensor was used to determine the damping terms and thruster characteristics of the drone. As the rig was not designed to test for yaw forces, this was done using the developed thruster characteristics. The added mass parameters were found using a script developed in (Eidsvik 2015) using the geometrical properties of P2-Alpha. A simulation platform that was able to simulate the relevant sensor failures was developed to aid in designing and testing the observer. The Extended Kalman Filter was developed from the observer model, using its properties for sensor fusion by its natural weighing algorithm and dead-reckoning by modifying the Kalman Gain to handle sensor faults and asynchronous sensor measurements. To augment the observer a signal processing module was developed to remove outliers by variance testing. Additional Kalman Filters were designed to aid in fault detection using residual generation together with the CUSUM-test for detection a change in mean. The implementation was done in Simulink for the simulations and Simulink interfaced with ROS for the experimental

tests. The design parameters for the Kalman Filters and CUSUM-test was tuned by trial and error. Through simulations it was shown that the observer had good fault-tolerance towards outliers, high variance and signal freeze. It was shown that the observer was able to handle multiple asynchronous measurements. Simulations was also done to test for signal bias detection by the CUSUM-algorithm, and shown to work conceptually. However, in the presence of other sensor faults the residual generation properties was not sufficient to determine the nature of a fault by the CUSUM-test as all sensor faults affected the residuals. But the general performance of the observer was good for the simulations. Through experimental testing the observer performance was also found to be satisfying. The observer was tested for two fault-free conditions where the performance was good. The advantages of sensor fusion was shown by testing the comparative performance of a Kalman filter without sensor fusion. Fault-tolerance against outliers, signal freeze and high variance was tested, and the performance was decreased, but the observer design still showed a good degree of fault-tolerance. One of the main problems in the design was the tuning of the Kalman Design parameters, as this was done by trial and error. Detection of signal bias and drift was not tested in an experimental setup.

10.2 Further work

There are several ways to improve the observer. One natural next step would be to develop a control system to work in conjunction with the observer. Besides that there is much room for improvement for the observer itself. A loose integration between IMU and HPR was started during this thesis, using the IMU strapdown equations as seen in Bryne (2013). In the current observer IMU bias is not accounted for, so the thought was to estimate the IMU bias by the loosely coupled integration between HPR and IMU and feedforward this to the main Kalman filter. There are several other improvements that could done, such as creating a sectorial kalman filter instead of the Extended Kalman filter, where you linearise the filter around set heading angles. This allows for better tuning of the $\mathbf{Q}$ matrix, as the covariance terms can easier be determined. The IMU accelerations could also be sent to the filter using the direct measurements, by modelling the jerk instead of the accelerations in the ROV vessel model. Not much literature was found for this unfortunately.

Bibliography

Blain, M., Lemieux, S. & Houde, R. (2003), Implementation of a rov navigation system using acoustic/doppler sensors and kalman filtering, *in* 'Oceans 2003. Celebrating the Past ... Teaming Toward the Future (IEEE Cat. No.03CH37492)', Vol. 3, pp. 1255–1260 Vol.3.

Blanke, M., Kinnaert, M., Lunze, J., Staroswiecki, M. & Schröder, J. (2006), *Diagnosis and fault-tolerant control*, Vol. 2, Springer.

BlueRobotics (2017), `http://bluerobotics.com/product-category/thrusters/t100-t200-thrusters`. Accessed: 2017-05-30.

Brown, R. G. & Hwang, P. Y. (1997), 'Introduction to random signals and applied kalman filtering: with matlab exercises and solutions', *Introduction to random signals and applied Kalman filtering: with MATLAB exercises and solutions, by Brown, Robert Grover.; Hwang, Patrick YC New York: Wiley, c1997.* **1**.

Brown, R. G. & Hwang, P. Y. (2012), *Introduction to random signals and applied Kalman filtering: with MATLAB exercises and solutions*, Vol. 4.

Bryne, T. H. (2013), 'Fault-tolerant sensor fusion based on inertial measurements and gnss', *Thesis* .
URL: *http://hdl.handle.net/11250/260845*

Candeloro, M., Sørensen, A. J., Longhi, S. & Dukan, F. (2012), 'Obsevers for dynamic positioning of rovs with experimental results', *IFAC Proceedings Volumes* **45**(27), 85–90.
URL: *http://www.sciencedirect.com/science/article/pii/S1474667016312095*

Dukan, F., Ludvigsen, M. & Sørensen, A. J. (2011), Dynamic positioning system for a small size rov with experimental results, *in* 'OCEANS 2011 IEEE - Spain', pp. 1–10.

Eidsvik, O. A. (2015), 'Identification of hydrodynamic parameters for remotely operated vehicles', *Thesis* .
URL: *http://hdl.handle.net/11250/2350869*

Fossen, T. (2011), *Handbook of Marine Craft Hydrodynamics and Motion Control*, John Wiley & Sons.

Gustafsson, F. (2001), Stochastic observability and fault diagnosis of additive changes in state space models, *in* '2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)', Vol. 5, pp. 2833–2836 vol.5.

Gustafsson, F. (2003), Implementation of a rov navigation system using acoustic/doppler sensors and kalman filtering, *in* 'Oceans 2003. Celebrating the Past ... Teaming Toward the Future (IEEE Cat. No.03CH37492)', Vol. 3, pp. 1255–1260 Vol.3.

Gustafsson, F. (2012), *Statistical Sensor Fusion*, Studentlitteratur.

Gustafsson, F. & Gustafsson, F. (2000), *Adaptive filtering and change detection*, Vol. 1, Wiley New York.

Hassani, V., Pascoal, A. M., Aguiar, A. P. & Athans, M. (2010), 'A multiple model adaptive wave filter for dynamic ship positioning', {*IFAC*} *Proceedings Volumes* **43**(20), 120 – 125. 8th {IFAC} Conference on Control Applications in Marine Systems.
URL: *http://www.sciencedirect.com/science/article/pii/S1474667016334486*

Loan, C. V. (1978), 'Computing integrals involving the matrix exponential', *IEEE Transactions on Automatic Control* **23**(3), 395–404.

Mahony, R., Hamel, T. & Pflimlin, J. M. (2008), 'Nonlinear complementary filters on the special orthogonal group', *IEEE Transactions on Automatic Control* **53**(5), 1203–1218.

Sandøy, S. S. (2016), 'System identification and state estimation for rov udrone', *Thesis* .
URL: *http://hdl.handle.net/11250/2409503*

SNAME (1950), *The society of Naval Architects and Marine Engineers. Nomenclature for Treating the Motion of a Submerged Body Through a Fluid*, Vol. 1-5, Technical and Research Bulletin.

Sørensen, A. J. (2013), *TMR4240 Marine Control Systems. Propultion and-Motion Control of Ships and Ocean Structures*, Akademika forlag.

SparkFun 9DoF IMU Breakout - LSM9DS1 (2017), `https://www.sparkfun.com/products/13284`. Accessed: 2017-05-30.

Thrun, S. (2005), *Probabilistic robotics*, MIT Press.

Zhao, B., Blanke, M. & Skjetne, R. (2012), 'Fault tolerant rov navigation system based on particle filter using hydro-acoustic position and doppler velocity measurements', *IFAC Proceedings Volumes* **45**(27), 280 – 286. **URL:** *http://www.sciencedirect.com/science/article/pii/S1474667016312423*

Zhao, B. & Skjetne, R. (2014), 'A Unified Framework for Fault Detection and Diagnosis Using Particle Filter', *Modeling, Identification and Control* **35**(4), 303–315.

Zhao, B., Skjetne, R., Blanke, M. & Dukan, F. (2014), 'Particle filter for fault diagnosis and robust navigation of underwater robot', *IEEE Transactions on Control Systems Technology* **22**(6), 2399–2407.

Appendices

Appendix A

Model parameter estimation

A.1 Damping forces

The results from the towing tests concerning surge, sway, heave and yaw damping forces together with timeseries of the results and scripts for reading and plotting the timeseries is found in the attachments at

- .../Appendix/Appendix_A/1_Damping_forces

A.2 Thrust Allocation

The results for the thruster characteristics together with the timeseries and plotting functions are found in the attachments at

- .../Appendix/Appendix_A/2_Thrust_allocation.

As well as the matlab function *PWM2Thrust* that transforms the PWN_ref signals from the controller to the thruster forces acting on the ROV.

A.3 Sensor variance

The timeseries and matlab scripts used to determine the sensor measurement noise from logging data is found in the attachments at

- .../Appendix/Appendix_A/3_Sensor_variance

The ones used in the thesis are the stand still tests, however some tests were done with added mechanical vibrations but not utilized.

A.4 Added mass estimation

The script developed by Eidsvik (2015) is copied directly, with constants replaced to fit the geometrical properties of P2-Alpha to calculate Added Mass using Eidsviks method is found in

- .../Appendix/Appendix_A/4_Added_mass_Eidsviks_method

Appendix B

Matlab and Simulink implementation of Observer

B.1 Observer simulated implementation in Matlab and Simulink

The Observer simulation studies are found in the attachments at

- .../Appendix/Appendix_B/1_Observer_Simulated

The matlab workspaces for the results for Case1, 2 and 3 are located in the folder and can be loaded up and plots can be generated by the plotting functions in each folder. The Simulink model contains more than the 2 Observers that were used in the report. The Observer named $ekf1$ is marked as blue and is the nominal Kalman Filter using all sensor measurements, while $ekf2$ is the one using just positions. The cases used in the report can be resimulated by running main_SIMU for case 1 and 2 by either activating faultsim_param or faultsim_param_case2. Case 3 is initated with main_SIMUcase3.

B.2 Observer experimental implementation in Matlab and Simulink

The experimental implementation is found in

- .../Appendix/Appendix_B/2_Observer_Simulated

It is started by running *main_ROS*. The simulink model runs in real time, and requires connection to a ROSMASTER sending the appropriate rosmessages, as can be done by interfacing matlab and simulink as per Appendix **??**. P2-Alpha uses custom messages that are included in the file as well, that must be built by matlab. In addition to this the simulink model uses the *msfun_realtime_pacer* simulink block found in Appendix **??**. The workspaces from Case 1, 2 and 3 are in the folder and be loaded however, and the plotting functions in the respective folders can be used to examine to plots in matlab.

Appendix C

Statistical tests

C.1 CUSUM-test

The CUSUM-test script developed in the pre-project for this thesis can be found in the Attachments at

- .../Appendix/Appendix_C/1_CUSUMtest

The function used in this thesis is the *cusumthreshold.m* for determining the design parameter h.

Appendix D

External Libraries and open-software

D.1 Qualisys ROS implementation

A library by KumarRobotics for Qualisys to interface with ROS is used and can be found at:

- ”https://github.com/KumarRobotics/qualisys”

D.2 ROS-Matlab interface

To use Simulink and matlab together with the ROS, the Robotics System Toolbox is necessary. The system toolbox can be found with many guides and tutorials at

- ”https://se.mathworks.com/hardware-support/robot-operating-system.html”

D.3 Real-time pacer

For running SIMULINK in real time a real-time pacer, found at

- https://se.mathworks.com/matlabcentral/fileexchange/29107-real-time-pacer-for-simulink

D.4 VMware

The virtual machine used to run ROS was *VMware player*, and can be found for free download at their sites

- "https://www.vmware.com/products/player/playerpro-evaluation.html"

Appendix E

Marine Cybernetics Lab

This Appendix is copied directly from: "https://www.ntnu.edu/imt/lab/cybernetics". The marine cybernetics laboratory is a small wave basin, located in what was originally a storage tank for ship models made of paraffin wax.

The facility is especially suited for tests of motion control systems for marine vessels, due to the relatively small size and advanced instrumentation package. It is also suitable for more specialized hydrodynamic tests, mainly due to the advanced towing carriage, which has capability for precise movement of models in six degrees of freedom.

The MC-lab is operated by the Department of Marine Technology. It is mainly used by Master students and PhD-candidates, but it is also available for external users.

The software in use was developed using rapid prototyping techniques and automatic code generation under Matlab/Simulink and Opal. The target PC onboard the vessel runs the QNX real-time operating system while experimental results are presented in real-time on a host PC using Labview.

E.0.1 Real-time positioning system

Real-time positioning system

Qualisys supplies a range of hardware and software products for motion capture and analysis of movement data. The key components of the system

are the Oqus cameras and the Qualisys Track Manager (QTM) software. For advanced analysis of the movement data Qualisys supplies third party software products such as Visual3D from C-Motion, Inc.

Measurement data can be exported in different standard formats for use in customer-developed and other third party software. The Qualisys products are designed to meet the highest demands for quality, simplicity, speed and precision. Systems, built from the Qualisys products, are flexible, mobile and expandable and are therefore easy to adapt to varying needs in industry, research and clinical use.

Tracking a model vessel's motions under different wave, current or wind conditions is one of the fundamental tasks at a hydrodynamics lab or a naval test site. Traditionally, this has been accomplished with potentiometer systems attached to a model or with bulky and expensive gyroscopes and accelerometers. Qualisys AB offers a easy, quick and functional way to obtain accurate 3D and 6 DOFs. With optical capture technology, your models remain completely unburdened by heavy sensors. Thanks to the low-mass optical targets, even very small and light models can be used.

Towing tank - In a towing tank, as few as 2 or 4 cameras can suffice, depending on the configuration. The cameras can be put on the towing carriage, which puts the model and the markers in the line of sight.