NTNU
Norwegian University of
Science and Technology

System Identification and State Estimation for ROV uDrone

Stian Skaalvik Sandøy

Master of Science in Engineering and ICT

Submission date:	June 2016
Supervisor:	Roger Skjetne, IMT
Co-supervisor:	Hans-Martin Heyn, IMT
	Mauro Candeloro, 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:	Stian Skaalvik Sandøy
Field of study:	Marine control engineering
Thesis title (Norwegian):	Systemidentifikasjon og tilstandsestimering for ROV uDrone
Thesis title (English):	System identification and state estimation for the ROV uDrone

Background
There has been an increased interest in research and development of technical solutions for underwater vehicles. We now possess the necessary knowledge and technology to perform complex underwater operations with high precision, such as seabed mapping, online underwater monitoring, subsea installations, and maintenance on pipes. In the project thesis of Andreas Viggen Henriksen and Stian S. Sandøy in autumn 2015, a mini-ROV was designed using low cost of-the-shelf parts along with open source software (Linux and Robot Operating System). It was named uDrone and developed to test control algorithms in the MC-Lab.
To develop model-based control algorithms for uDrone, it is also necessary to have a proper control model and thrust allocation model. Also to use onboard sensors, an observer is needed to lower measurement noise. In this project the objective is to obtain a satisfying control model and thrust allocation model along with the necessary parameters for uDrone. The former will be used to develop a model-based observer and an adaptive heave controller. All experiential testing will be perform in the MC-Lab at NTNU.

Work description
1. Perform a background and literature review to provide information and relevant references on:
    - Mathematical dynamic modelling and thrust allocation for ROVs.
    - System identification methods relevant for ROVs.
    - Observer designs relevant for ROVs.

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

2. Provide relevant control design model(s) and observer design model(s) to be used in model-based control and observer algorithms.

3. Provide the thruster configuration for the ROV uDrone, including the thruster mappings from thruster command signals to achieved thrust from each thruster. Derive a thrust allocation algorithm to be used for low-speed positioning control of the ROV.

4. Provide a low-speed observer design along with following considerations:
    - Performance with respect to state estimation subject to realistic measurement noise.
    - Sensor fusion.
    - Dead reckoning when loss of sensor signal.

   Perform simulations to verify the design, both based on the “perfect observer design model” and a more realistic simulation model.

5. Perform software implementation of the designed observer algorithm(s), including necessary HMI functionality to specify the model parameters, tune the observer injection gains, and visualize the estimated states.

6. Perform experimental testing and analysis for:
    - Thruster characteristics to parametrize the provided thrust allocation model.
    - Towing test for uDrone to find damping terms for the design and simulation models.
    - Validation of the ROV observer design.

Tentatively:

7. Provide a Model-Reference Adaptive Control (MRAC) algorithm to be used for heave control of the uDrone:
   - Simulate to verify the correctness of the algorithm and the achieved performance.
   - Perform software implementation of the MRAC algorithm.
   - Plan and perform experimental testing of the MRAC algorithm.

Guidelines

The scope of work may prove to be larger than initially anticipated. By the approval from the supervisor, described topics 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. The report 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 and background, problem formulations, scope, and delimitations, main body with 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, each 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, 2016	**Due date:**	As specified by the administration.

Supervisor:	Roger Skjetne
Co-advisor(s):	Mauro Candeloro

Trondheim, 09.06.2016

Roger Skjetne
Supervisor

Preface

This report is the result of the Master's Thesis in Marine Technology at the Norwegian University of Science and Technology carried out during the spring of 2016. The motivation for the thesis was the development of the uDrone test bed for the marine cybernetics lab. The models developed can also be utilized in other model-based control projects in the future. The idea came up after using the project report of building uDrone with a partly complete control system which needed improvement.

The assumed background of the reader should be an education in control engineering with some knowledge within hydrodynamics.

Trondheim, 2016-06-10

Stian Skaalvik Sandøy

Acknowledgment

I want to thank my supervisor Professor Roger Skjetne and co-supervisor Phd. Candidate Mauro Candeloro for the support in granting easy access of equipment in addition to interesting discussions and help with typographical errors.
Further, I would like to thank Torgeir Wahl for an introduction to all new equipment in the MC-Lab. I am also thankful to BluEye Robotics which has been supporting during the project and given me a great experience by working on the newly developed ROV. For helping with the construction of the bracket for the towing rig, I am in gratitude to Oddvar Paulsen.
I also need to thank my friends and co-students Andreas Viggen Henriksen and Torkild Eide for the cooperation with the uDrone and all the great discussions along the way.
I further would like to thank my girlfriend Inês Lourenço for all the support along the way, and for being a great help with the correction of typographical error.

S.S.S.

Acronyms

DOF Degree Of Freedom

CG Center of Gravity

CB Center of Buoyancy

VO Vehicle Origin

TBF Test Basin Frame

TBO Test Basin Origin

ROS Robotic Operating System

CAD Computer Aided Design

DVL Doppler Velocity Log

KF Kalman Filter

EKF Extended Kalman Filter

GNSS Global Navigation Satellite System

IMU Inertial Measurement Unit

INS Inertial Navigation System

MIMO Multiple Input Multiple Output

RPM Rotations Per Minute

PWM	Pulse Width Modulation
PSD	Power Spectrum Density
LQR	Linear Quadratic Control
LQG	Linear Quadratic Gaussian
HMI	Human Machine Interface
CFD	Computational Fluid Dynamic
DP	Dynamic Positioning
MC-Lab	Marine Cybernetics Laboratory
NED	North-East-Down
QMT	Qualisys Motion Tracking

Summary

This master thesis makes an introduction to all aspects of the design of an advanced model-based control system for the Remotely Operated Vehicle (ROV) uDrone. The design of thrust allocation along with a known mathematical model and the estimation of its parameters will be the basis for an implementation of a state estimation algorithm. The contribution can also be used in control systems implemented in the future on ROV uDrone. Further, the thesis discusses in detail the design of the worldwide known estimator called Kalman Filter along with a Linear-Quadratic Controller used in heave and yaw motion. Together they are known as LQG-control which is the optimal control for a given model together with the assumption that all noises are Gaussian distributed. Implementation in Simulink and interfaced with ROS proves uDrone as an excellent test bed. The results of the simulation and experimental results validate the design of thrust allocation, mathematical model, and estimator.

Sammendrag

Denne master oppgaven gir en introduksjon til alle aspekter av å designe et avansert model-basert kontroll system for den fjernstyrte undervannsdronen uDrone. Design av thruster allokering sammen med en matematisk modell og estimering av dens parametere vil gi grunnlaget for implementasjon av en tilstandsestimator. Bidraget kan også brukes i design av modelbaserte design i fremtiden. Videre diskutere oppgaven design av den verdenberømte estimatoren Kalman Filter. Den blir testet sammen med en såkalt (LQR) Lineær Kvadratisk Regulator som er den optimale kontrolleren. Sammen med Kalman filteret gir dette en Lineær Kvadratisk Gaussian Kontroller eller forkortet som LQG-kontroller. Den gir det best kontrolsystemet under antagelsen Gaussisk hvit støy og en god model. Implentasjonen av systemet er gjort i Simulink integrert sammen med Robot Operativ System (ROS) som er et rammeverk for implentasjon av programmvare for roboter. Sammen med Simulink viser det seg at det er en meget god platform å drive utvikling av kontroll algoritmer.

Contents

Chapter 1

Introduction

1.1 Background and Motivation

There has been an increased interest in research and development of technical solutions for underwater vehicles. We now possess the necessary knowledge and technology to perform complex underwater operations with high precision, such as seabed mapping, online underwater monitoring, subsea installations, and maintenance on pipes. In the project thesis of Andreas Viggen and Stian Sandøy in autumn 2015, a mini-ROV was designed using low-cost of-the-shelf parts along with open source software (Linux and Robot Operating System). It was named uDrone and developed to test control algorithms in the MC-lab. To develop model-based control algorithms for uDrone, it is also necessary to have a proper control model and thrust allocation model. Also to use onboard sensors, an observer is needed to lower measurement noise, estimate unmeasured states and enable dead reckoning mode during sensor dropout. In this project, the objective is to obtain a satisfying control model and thrust allocation model along with the necessary parameters for uDrone. The former will be used to develop a model-based observer and an adaptive heave controller. All experimental testing is performed in the MC-lab at NTNU.

1.2 Literature Survey

Essential literature used in this thesis for mathematical models for ROV is stated in Fossen (2011). It discusses the full process model with all degrees of freedom, and it can be simplified to be used in underwater vehicles. Further, thrust and thrust allocation models are discussed in detail in Fossen (2011), Sorensen (2013) and Dukan et al. (2011). Fossen (2011) discusses a thrust allocation model described by sets of forces and lever arms. Furthermore, unconstrained and constrained allocation are elaborated by setting up minimization problems and solving them with known algorithms like quadratic programming. Sorensen (2013) discusses more the propulsion control and what the thruster losses are in different scenarios. Dukan et al. (2011) did an experimental implementation of theory from both Fossen (2011) and Sorensen (2013) with more on NTNU ROV Minerva. It is a great example to follow for implementation on another ROV.

Parameter estimation is discussed in Eidsvik (2015), Hegrenas et al. and Aras et al.. In Eidsvik (2015), there are experimental methods that simplify the work for finding hydrodynamical parameters. Using computational software to obtain added mass is also discussed. There are also other approaches to investigate step responses by using the MATLAB toolbox called "System Identification toolbox" documented in Ljung (1999). This is done in Aras et al. and partly in Hegrenas et al. where a similar solver as in Ljung (1999) is used.

There exists a lot of observers, and each has different advantages and disadvantages. Firstly, the type designed in this thesis is a model-based observer which depends on a simplified linearized process model. They are usually used when an operation is around a working point as for instance in low-speed maneuvering for ROV's. There exists a lot of variations as deterministic, and Luenberger observer discussed in Chen (2013). The former rely on a perfect modeling and has an open loop process, while the latter uses measurement updates to update the in-perfect model with fixed gain. Both use linearized equations. A case of an observer that has variable gain is the Kalman Filter (KF), which is developed in Kalman (1960). It uses an updated covariance matrix for each iteration to determine the gain of error in the model on the measurement. The KF is optimal for Gaussian white noise disturbances.The disadvantage of the KF is the lack of a stabilization proof when the covariances have not converged.

ROV's usual operate in a wide range of velocities which means that non-linear observers or a bank of linear observers are needed as discussed in Magill (1965). The former can be an Extended Kalman Filter and Sectoral Kalman Filter that is similar to Magill (1965) implemented in Dukan et al. (2011). The EKF linearizes the process model around a working point for each iteration, while the sectoral KF switches between linearized "sectors" of the process model. Another option is the non-linear passive observer developed in Fossen (2011) which can use the non-linear differential equations directly. As mentioned in Fossen (1987), ROV's operate in different domains of velocity and it is, therefore, difficult to use simple linear or model-based observers based on kinetics due to erroneousness modeling. But if only the kinematics are used and all accelerations are modeled as process noise this will be avoided. The complementary filter developed in Mahony et al. (2008) is applied on INS setups. Further, in Kjerstad (2016) filter method based on combining IMU's at different locations can give good acceleration estimates even for rotations which are a great contribution for estimates. Recently, testing of the particle filter has also been tested in Zhao et al. (2014) which was developed in par (2004). The particle filter as the KF does not base its assumptions on a Gaussian distribution, which is a very strong claim. Relaxing this shows clearly better performance in the Particle filter in comparison with the KF using velocity updates and hydro-acoustic positioning. But the computational power needed to using a KF as compared with the Particle Filter is large due to the estimation of each particle instead of using the recursive procedure of the KF. In this case, for a simple and low-cost ROV as uDrone, the KF will be used using pressure and Qualisys sensor updates. Experimental testing is done in combination with an LQR controller's to demonstrate the performance of the observers.

1.3 Objectives

The main objectives of this Master's project are

1. Design a fitting thrust allocation model for uDrone.
2. Design an appropriate mathematical model for the motion of the uDrone.

   Process and Control/Observer model.

3. Design a state estimation algorithm for uDrone and assess the following.

Sensor fusion.

Dead Reckoning.

4. Find the thruster characteristics.
5. Find the necessary parameters for a mathematical model of motion.
6. Implementation and experimental testing of state estimation algorithms.

1.4 Experimental Platform

In this thesis, there are multiple platforms to perform experiments. This chapter will introduce the systems along with its use. An introduction to the uDrone with its hardware, thrusters and software interfaces along with the test basin and Qualisys motion capture is done.

1.4.1 Summary of uDrone

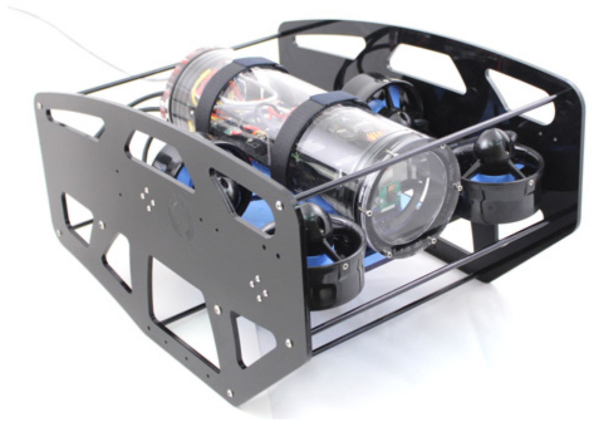


Figure 1.1: The BlueROV kit.

uDrone, shown in Figure 1.1, is designed for development projects related to testing new algorithms for navigation and motion control of an ROV. It was developed in a project thesis in autumn 2015 Sandøy (2015). It is built with an ROV kit from BlueRobotics BlueRobotics (2015a). The package includes six BlueRobotics thrusters of type T200 thrusters and each one has an open water thrust measured to be 40-50 N BlueRobotics (2015b). The on-board circuit boards are an Arduino Mega and a Raspberry Pi 2. For I/O purposes like sending a signal to the motor controllers or collect sensor data, the Arduino was used. The Raspberry Pi is the onboard processing unit. It is running the operation system, Ubuntu 14.04 with the popular framework called Robot Operating System, or simply ROS. On the topside, there is a computer also running Ubuntu 14.04 and ROS. Communication to topside is made through a 40 m long ethernet cable which is also umbilical. The hardware is shown in Figure 1.2. It is self-powered meaning it has on-board batteries that are distributed with two distribution boards as demonstrated by the power map in Figure 1.3.

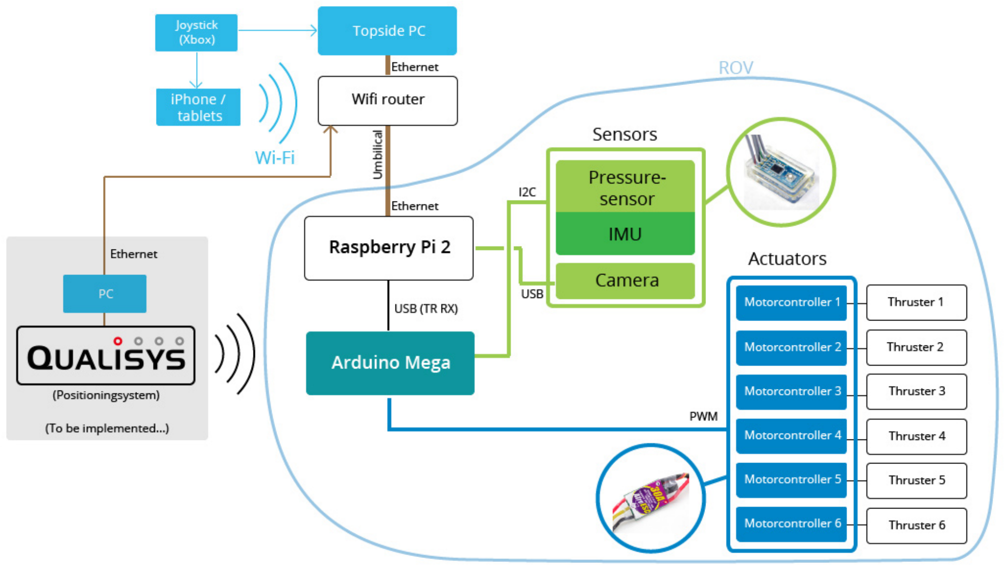


Figure 1.2: The Communication line diagram shows how the different components in the uDrone system communicate. Communication between topside computer, Qulisys, and the uDrone, is made through Ethernet signals, while communication for sensors and camera is with I2C/USB to the Arduino and RPI2, respectively. Thrusters are controlled by motor controllers who get the desired thrust from the Arduino through PWM signals.

The T200 Thruster

For a control system is a good performance of the thruster allocation vital. The uDrone have six T200 thrusters BlueRobotics (2016) 1.4. They are controlled by PWM signals from the Arduino to motor controllers. The signal ranges from a max negative thrust signal at 1100 to a max at 1900. 1500 is the zero thrust signal with a dead zone of ±25, which means that there is no thrust on the range 1475 to 1525. The only part of the whole signal range will be utilized here due to low

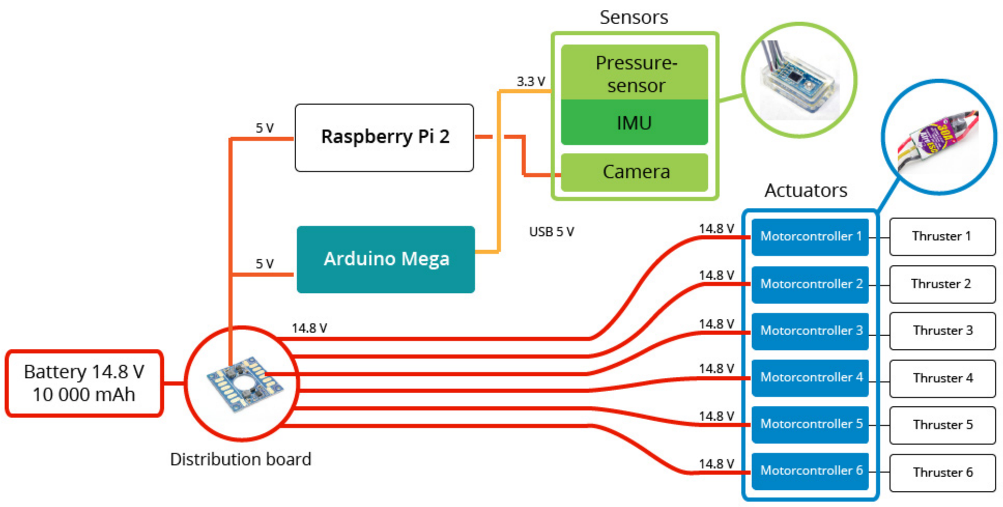


Figure 1.3: The Power map display the voltage of each component and how it is distributed. In this system, a distribution board is used with a DC/DC converter to power low voltage component. The sensors and camera are power through the Arduino and RPI2, respectively. There is also implemented a voltage sensor from the battery to the Arduino to allow live monitoring.

velocities. There is not possible to take any RPM measurements of the propellers, so we have to assume that the signal is proportional to the RPM.

Figure 1.4: The T200 thruster.

Sensors

Available sensor suite for uDrone is an Inertial Measuring Unit (IMU) with pressure and a compass measurement. The sensor from OpenROV consists of a sensor suite of 4 low-cost units which are an accelerometer, gyro, compass and a pressure sensor. The IMU documented in Bosch (2014), consists of both a gyro and an accelerometer. The pressure sensor is of type

MS5837 (2015). There are also more space in the electronic housing for additional sensors.

Software

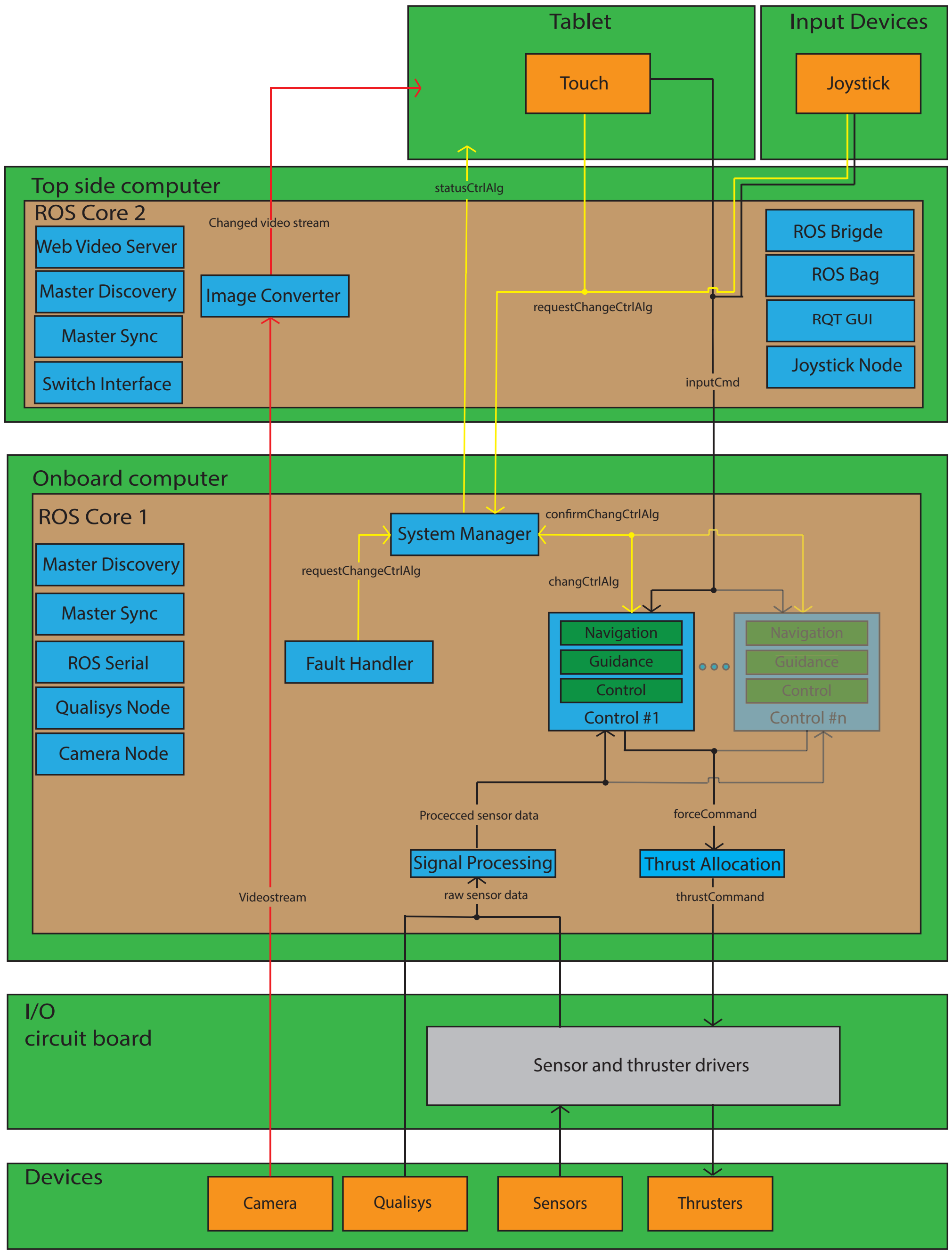


Figure 1.5: Software map.

The software design of uDrone is presented in Figure 1.5. The desing was developed in Sandøy (2015) and restated here. It consist of four layers. The bottom layer in bright green describes the hardware components. Each component is written in an orange colored block. The brown layer illustrates two separate ROS cores; the reason is that partitioning provide better data abstraction and a more restricted traffic through the network cable to top side. The blue blocks represent the ROS nodes in the system, where each green block within each node elaborates the process. Since the software running on the Arduino is separated from ROS, it is illustrated in grey. In reality, ROS has a peer to peer communication topology. The communication lines have three different colors. Red is assigned to the video stream topic, yellow denotes system massages and black is related to control system messages. The nodes that are not connected to any lines are interfaces for various use.

ROS - Simulink Interface

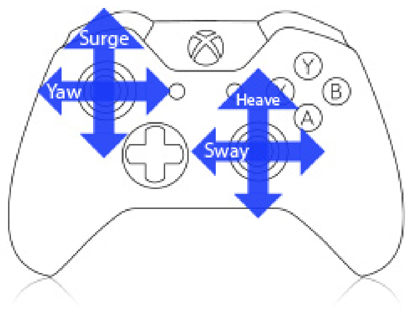
To implement a full control system, easy simulation, testing, debugging and implementation is important. Simulink solves this excellent when it is interfaced with ROS. Using input and output messages from and to ROS in real time is made possible with the Robotics Toolbox for MATLAB. Simulink also makes it possible to change tunable and proportional parameters live in the simulation, making it an excellent HMI for a testbed. A manual of how the system is interfaced can be seen in the Appendix E.1.

Joystick Configuration

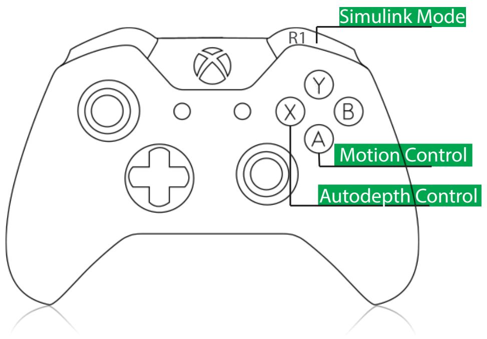
The ROV can be manually controlled by a x-box 360 controller through the topside computer. The controller manages also the different control modes in the system. Control mode switching is illustrated in in Figure 1.6b. The specified thrust in Direct motion control is seen in Figure 1.6a. The same controls are used when controlling the system in Simulink Mode.

1.4.2 Test Basin and Qualisys Motion Capture

The test basin at the Marine Cybernetics Lab can be viewed in the appendix at Figure D.1. It was frequently used for testing all features of the uDrone. The use was nearly weekly to get a satisfying performance for thrust allocation, observers, and controllers. More information is given in Appendix D.



(a) Direct motion control.



(b) Switching of control modes.

Figure 1.6: Manual joystick control and control mode switching.

1.5 Approach

- Finding thrust allocation, mathematical models, and state estimation algorithms by customizing known approaches.
- Finding drag forces by towing test.
- Finding added mass based on estimation method from Eidsvik (2015).
- Finding other kinematic and kinetic properties by use of CAD.
- Verify mathematical model by testing model-based state estimation algorithms against a motion capturing system. They will be tested on uDrone in the MC-lab at NTNU using Qualisys as motion capturing system.

1.6 Structure of the Report

Chapter 2 goes through the design of thrust allocation along with experimental procedure and results of thrust characteristics. **Chapter 3** will give the development of the process model going into kinematics, kinetics and stochastic processes needed to model the system with acceptable fidelity. Following this is the parameter identification which is **Chapter 4**. As the name suggests, it corresponds to the parameters needed for the process model found here. This means the methods to find them and the final results along with discussions. **Chapter 5** is the last "topic chapter", it renders the design of the state estimator. Simulations and experimental results related to design are also discussed. Last, **Chapter 6** presents the conclusion with further work.

Chapter 2

Thrust Allocation

The thrust allocation is necessary to produce clean motions in the 4 DOF, surge, sway, heave, and yaw. Without a proper design, it is difficult or even impossible to control the ROV. This chapter contains the modeling and development of the thrust allocation. It starts with a presentation of the thrusters and thruster configuration for uDrone along with a discussion for the current setup. A proposed thrust model is developed using theory from Fossen (2011) and Dukan et al. (2011). Further, the experimental setup along with the results is presented.

2.1 Thruster Configuration of uDrone

The thruster configuration is shown in Figure 2.1. There are six thrusters in total, three in the heave, two in the surge and one in the sway direction. Table 2.1 presents the position of the thrusters about the center of gravity (CG) along with the defined indexes. The CAD software Solid Works seen in SOLIDWOKRS (2016), was used to obtain the distances.

T_i	$l_{x_i}[mm]$	$l_{y_i}[mm]$	$l_{z_i}[mm]$	Orientation
T_1	153	-111	-4	Heave Front Port
T_2	153	111	-4	Heave Front Starboard
T_3	-199	0	85	Heave Rear
T_4	-16	111	-14	Surge Starboard
T_5	-16	-111	-14	Surge Port
T_6	8	0	95	Sway

Table 2.1: Thruster configuration. Respectively, the thruster name and position along the x, y and z axis and orientation.

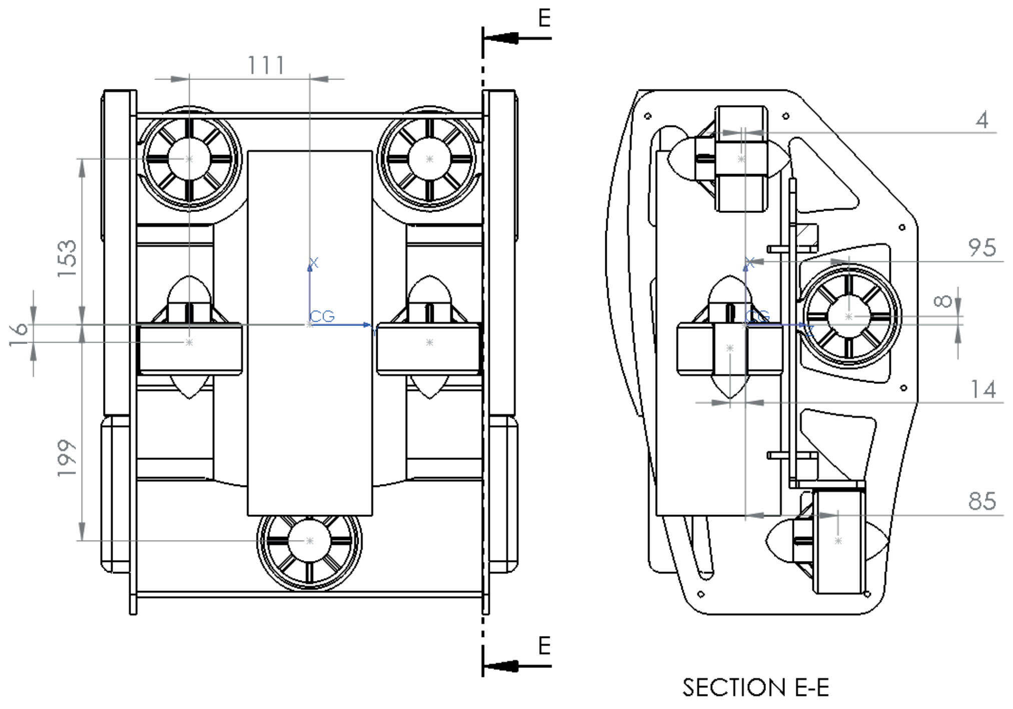


Figure 2.1: Topview and section view of thruster configuration for uDrone. All lever arms are illustrated from Center of Gravity (CG).

For a satisfactory thrust allocation, it is necessary to get the mapping between input signal and exercised thrust force. This is already done by BlueRobotics, as shown in Figure 2.5. However, this is an open water thrust test, which means that losses due to thruster configuration are not considered. Those losses are mainly caused by blocking the thrust beam or Coanda Effect as discussed in Sorensen (2013). In Figure 2.1 there are three noticeable possible thruster loss effects:

- Coanda effect (thrusters are too close to the frame of the ROV)
- Heave thrusters are blocking the thrust beam from the surge thrusters. This is seen to the right in Figure 2.1.
- The hole in the side elements of the frame is too small for the sway thrusters beam. This can downgrade the directions due to thrust hindering.

2.2 Thrust Allocation Model

In Fossen (2011) a thruster configuration is defined as a set of produced force and lever arms. Both are defined about the body frame. Equation (2.1) shows the notation.

$$f = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix}, \qquad l_{t_i} = \begin{bmatrix} l_x \\ l_y \\ l_z \end{bmatrix} \tag{2.1}$$

where $f \in \mathbb{R}^{3x1}$ is the thrust and $l \in \mathbb{R}^{3x1}$ gives the lever arms. The thrust and moments will therefore be:

$$\tau_{t_i} = \begin{bmatrix} f \\ l_{t_i} \times f \end{bmatrix} = \begin{bmatrix} f_x & f_y & f_z & l_y f_z - l_z f_y & l_z f_x - l_x f_z & l_x f_y - l_y f_x \end{bmatrix}^T \tag{2.2}$$

The total thrust and moments from the six thrusters of uDrone will then be the sum of τ for each thruster. The thrust configuration for uDrone is shown in Figure 2.1 and all lever arms is stated in Table 2.1. So if thruster T_1 produces 1 Newton thrust, the resulting forces and moments will be the following:

$$\tau_{t_1} = \begin{bmatrix} 0 & 0 & 1 & -0.11 \times 1 & 0.15 \times 1 & 0 \end{bmatrix}^T.$$

The definition of forces of each thruster to the force vector τ are as the following:

$$\tau = \sum_{i=1}^{6} \begin{bmatrix} f_{t_i} \\ l_{t_i} \times f_{t_i} \end{bmatrix} = B f_t \tag{2.3}$$

$$f_t = B^{-1} \tau \tag{2.4}$$

Where $f_t \in \mathbb{R}^{6\times1}$ is the force vector function of all thrusters, which depends on the direction of the thrust due to the difference in performance. This can be seen in Figure 2.5, which shows the mapping from BlueRobotics. As seen in the Figure, the thrust curve is not symmetrical. Therefore, it is necessary to partition it into two function. The mapping function depend on the input control signal. f_T is chosen to make an optimal approximation of the curve fit. The experimental thruster testing given in Section 2.3. The chosen function can be seen in Equation 2.5.

$$f_{t_i} = \begin{cases} K_L^-(u+25) + K_{NL}^-(u+25)|u+25| & , \quad u<0 \\ K_L^+(u-25) + K_{NL}^+(u-25)|u-25| & , \quad u>0 \\ 0 & , \quad u=0 \end{cases} \tag{2.5}$$

To find the control signal u_i from a desired thruster force f_{t_i}, the inverse of Equation 2.5 is found. Solving the second order equation, it is possible to obtain the following result:

$$u = \begin{cases} -\sqrt{\frac{-f_{t_i}}{K_{NL}} + \frac{K_L}{2K_{NL}}^2} + \frac{K_L}{2K_{NL}} + 25 + 1500 & , \quad u<0 \\ \sqrt{\frac{f_{t_i}}{K_{NL}} + \frac{K_L}{2K_{NL}}^2} - \frac{K_L}{2K_{NL}} - 25 + 1500 & , \quad u>0 \\ 0 & , \quad u=0 \end{cases} \tag{2.6}$$

All thruster characteristics are given in Table 2.2 in the next section.
Implementation was done both in ROS and Simulink. The reason for including Simulink, was for easier live monitoring of parameters. The implementation can be seen in the Appendix A.6.

2.3 Experiment - Thrust Characteristics

This section presents the thrust characteristics experiment setup along with the result. The setup is similar to the one presented in Eidsvik (2015). However, some modifications to the testing bracket were done.

2.3.1 Motivation and Objective

The thrust characteristics represents the mapping between signal and force. For the T200 thrusters, thrust mappings data is ready to use as given in BlueRobotics (2015b), the graph can be seen in Figure 2.5. However, due to the thrust configuration shown in Figure 2.1, there can be a considerable force loss capable of decreasing the performance.
Reduced thrust can lead to unwanted pitch or roll motions due to erroneous thrust mappings, which also leads to decreased performance in the observer and control algorithms. This is why it is important to find the thruster characteristics.
The uDrone have no RPM feedback which means that it is necessary to assume that the difference between the desired and real RPM produced by the thrusters is negligible.

2.3.2 Setup

The experiments are performed in the NTNU Marin Cybernetics Lab using a 4 DOF towing rig which is explained in more detail in Appendix D.

Load Bracket

The bracket used to mount uDrone properly to the load sensor was almost the same as used in Eidsvik (2015) (Chapter 3). The only difference is the bracket attaching the ROV to the rig, as seen to the right in Figure 2.2. It consists of a 10 mm aluminum plate with holes for angular brackets. Their design is illustrated in Figures C.1 and C.2 in the appendix. The reason for the re-design is that the open frame of the uDrone is different from the ones previously tested using the same rig. An example of the final configuration setup for surge is illustrated in Figure 2.3.

Thrust Signal Script

A script for running sequences of thruster signal is programmed to make testing quicker, since a manual testing would have been time-consuming. First, the lowest thrust signal is set at 1300 and then increased by 10 step by step for every 30 second until it reaches the maximum tested signal at 1700. A low pass filter is implemented to create a small force gradient on the loaded rig.

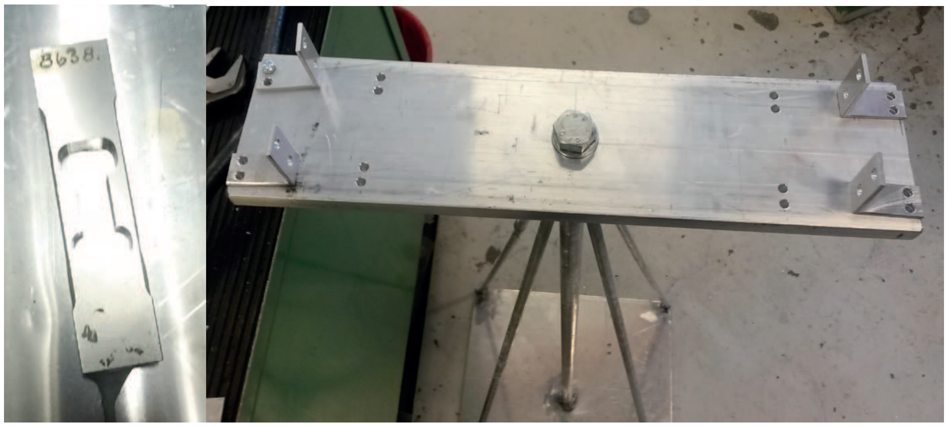


Figure 2.2: 18 kg load cell for measuring forces and brackets made in the workshop by the author.

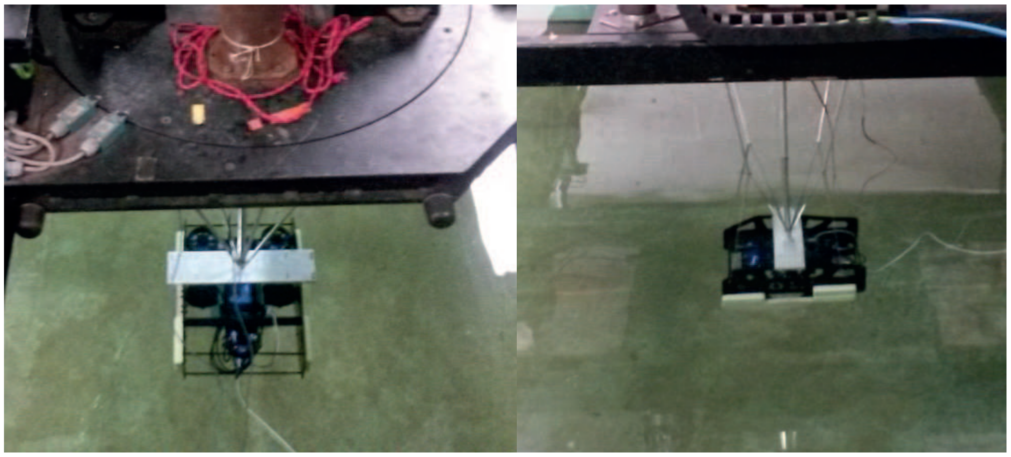


Figure 2.3: Testing of the surge thrust and drag force. The ROV was upside down, in order to have the bracket mounting closer to the CG.

The script is given in Appendix B.1.1. Symmetrical thrusters as surge and heave in front were tested simultaneously and then the resulting force was divided by two.

Reading the Results

Scripting is also implemented to read the results. It is also shown in Appendix B.1.1. By filtering all data with a second order Butterworth filter and then obtaining mean of the last five seconds for every 30 seconds, the thrust forces are retrieved. Thruster characteristics are then obtained by using a least square fit algorithm in MATLAB on Equation (2.5).

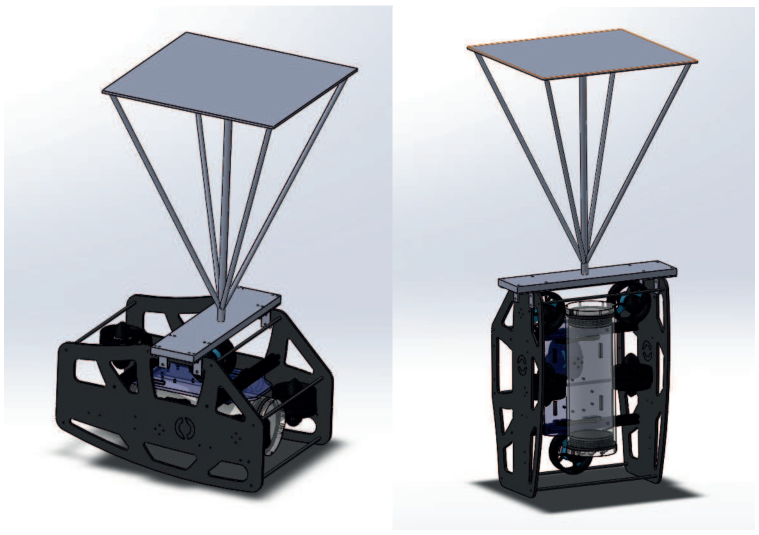


Figure 2.4: ROV configurations used in towing tests to obtain thrust coefficients and damping parameters. The left image shows the configuration used to find surge and sway thrust forces, while the right image shows the setup used to find them in heave.

2.3.3 Procedure

The procedure of conducting the thruster configuration is as following

1. Ensure that the battery is over 15 V.
2. Mount the ROV in the desired configuration.
3. Run the thrust signal script for thrusters in correct direction related to the setup.
4. Obtain force data in ".acm" and analyze the resulting file for obtaining thrust coefficients.

2.3.4 Results

The results are presented indicating both the numerical value and, successively, a graphical representation.

Coefficients Representation of Thrust Characteristics

T_i	K_L^-	K_{NL}^-	K_L^+	K_{NL}^+	Position
T_1	3.28e-2	3.68e-4	6.07e-2	3.33e-4	Heave Front Port
T_2	3.28e-2	3.68e-4	6.07e-2	3.33e-4	Heave Front Starboard
T_3	3.46e-2	2.27e-4	8.24e-2	2.24e-4	Heave Rear
T_4	3.57e-2	1.75e-4	7.22e-2	2.41e-4	Surge Starboard
T_5	3.57e-2	1.75e-4	7.22e-2	2.41e-4	Surge Port
T_6	6.00e-2	2.29e-4	8.63e-2	2.43e-4	Sway
-	4.69e-2	2.67e-4	6.07e-2	2.84e-4	Open water test

Table 2.2: Thrust coefficients. Negative and positive thrust is indicated with a superscript of "-" or "+", respectively. Linear terms have a subscript L while the quadratic term are indicated with a NL.

Graphical Representation of Thrust Characteristics

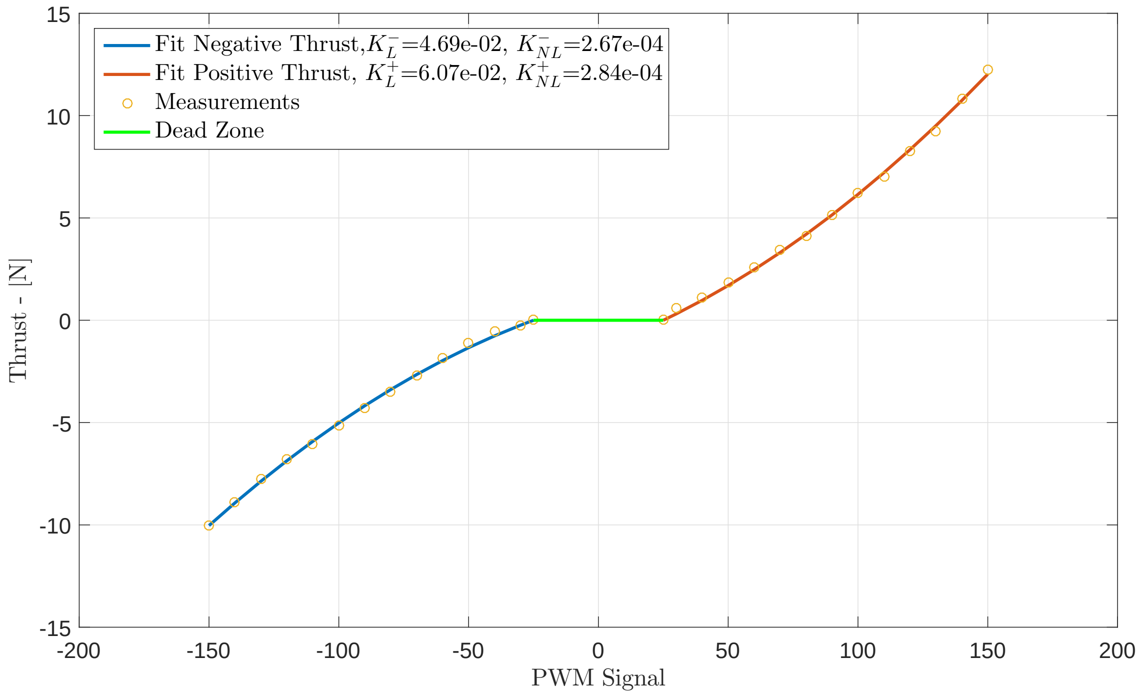
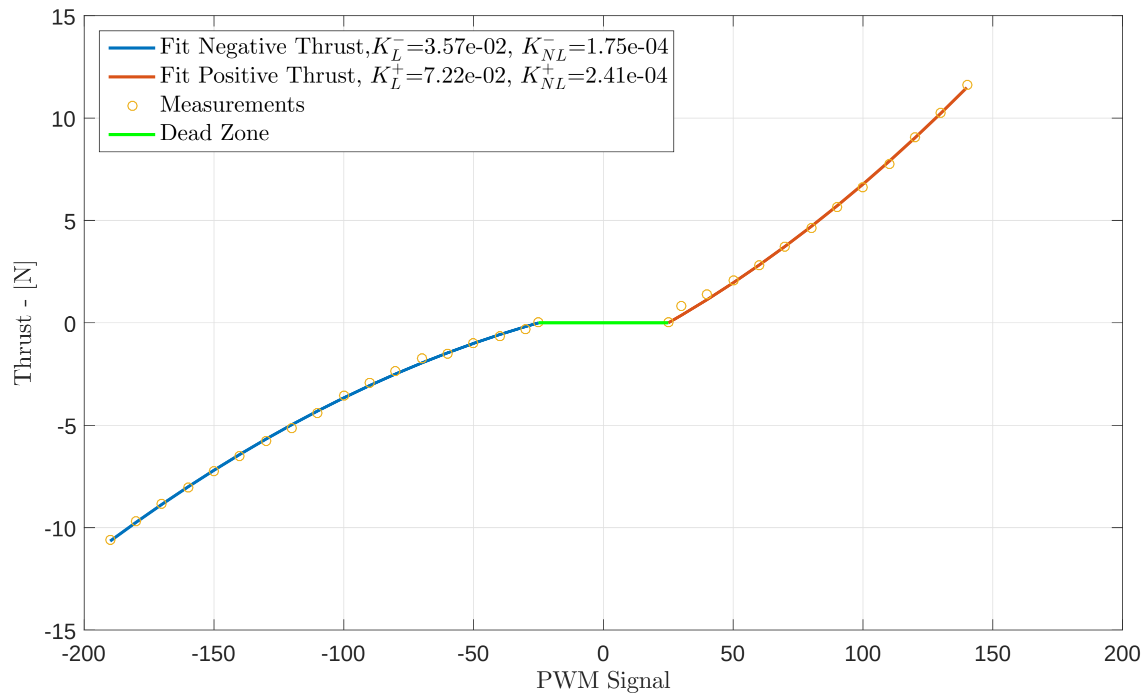
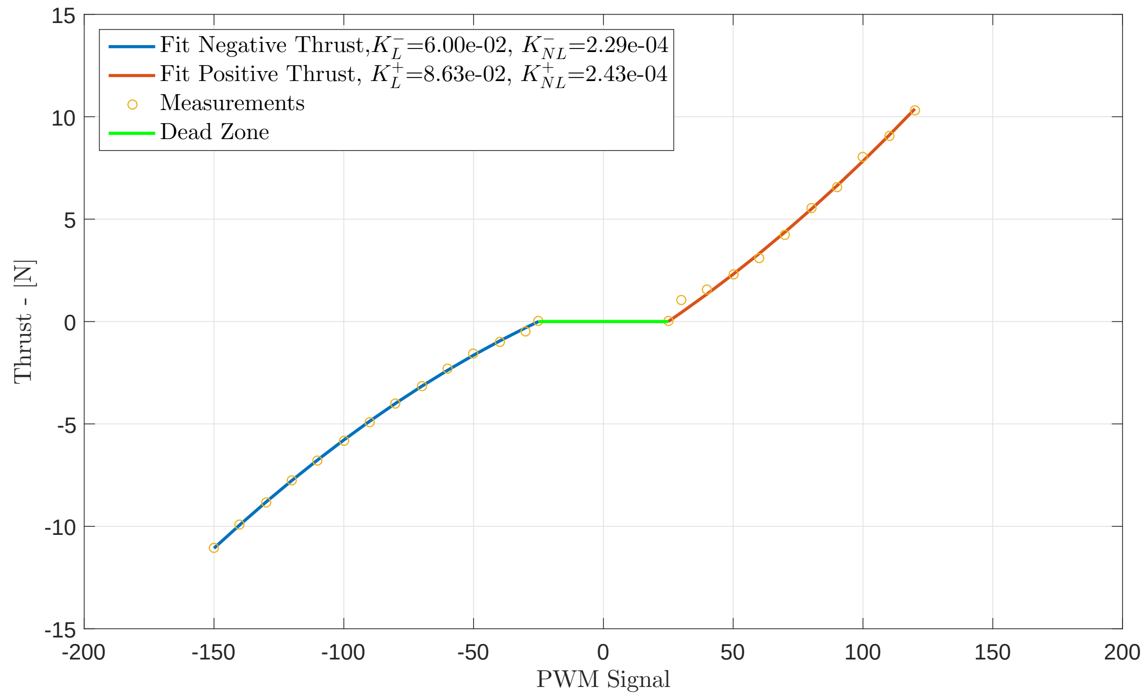


Figure 2.5: Curve fit for BlueRobotics's thrust measurement

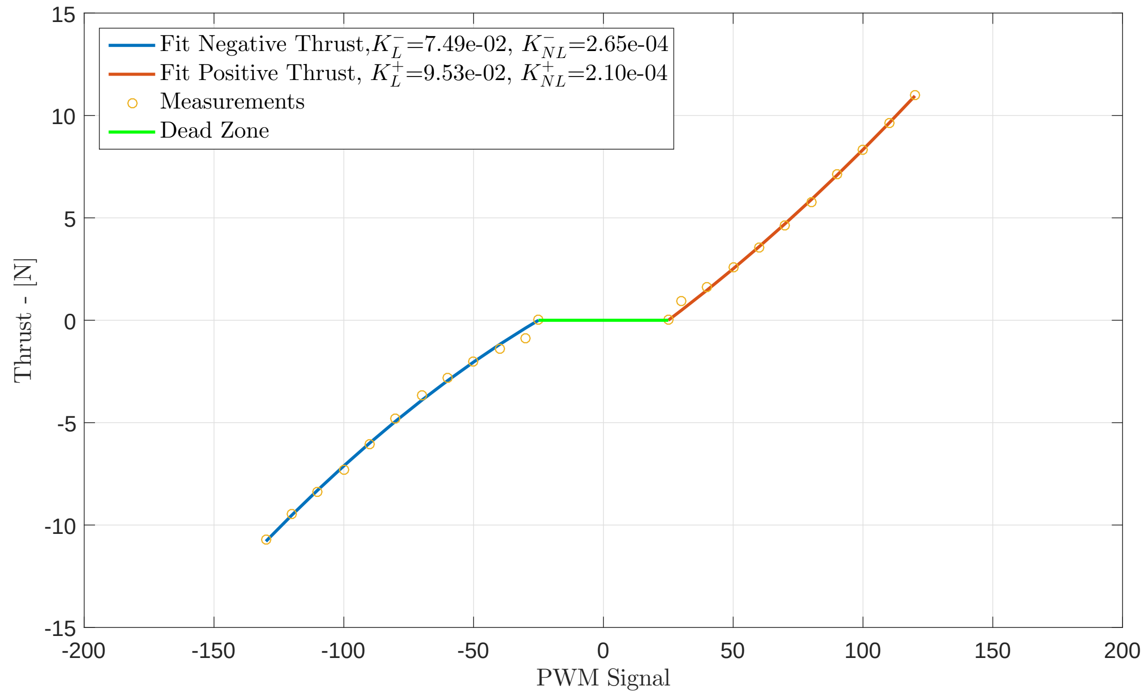


(a) Thrust curve fit surge thrusters (T_4 and T_5).

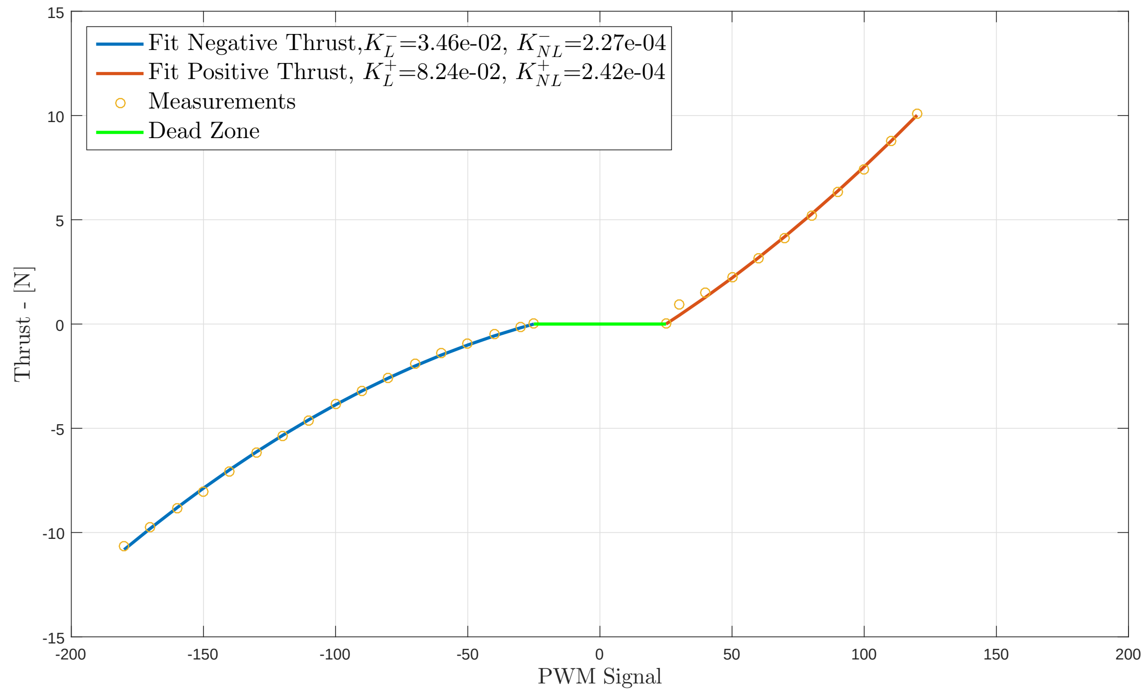


(b) Thrust curve fit sway thruster (T_6)

Figure 2.6: Damping curve fit for surge and sway thrusters.



(a) Thrust curve fit for front heave thruster (T_1 and T_2)



(b) Thrust curve fit for rear heave thruster (T_3).

Figure 2.7: Damping curve fit for heave thrusters.

2.3.5 Discussion

In this section the results of Section 2.3.4 are presented. The experiment of obtaining the characteristic was proceeded as planed. Only one test was run for each thruster, although running multiple test could increase the accuracy. In the measurement of low thrust forces a considerable amount of noise. Furthermore, a more rigid test rig would be advisable to employ for future tests. Forces over 10 N were too high for the rig structure, this could be noticed by evident deformation of the structure itself.

The characteristic for surge thrusters shows better performance than the open water test in Figure 2.5 for forward thruster. Consequently, we can conclude that there is no loss due to Coanda effect here as discussed earlier. The reasons for better performance compared with open water test is unknown. It can be due to the conditions of obtaining the coefficients. The reversed thrust is considerable lower, which is probably due to the blocking of the thrust beam by the heave thruster duct as discussed previously. Coefficients in sway and front heave thrusters are shown in Figure 2.6b and 2.7a and have no considerable differences from the open water tests. Finally, the rear heave thruster has considerable loss in the reversed thrust. This is probably due to the blocking of the thrust beam of the cables from the electronics housing.

In general all the backward thrust is naturally less than the forward one, which happens due to the design of the thruster. Another similarity of all the graphs is the non-linear jump in the the inital PWM signal. A better fit could be done by introducing the phase shift of ± 25 by an optimization parameter in Equation (2.5).

Testing the thrust allocation showed that there where some considerable pitching. The causes for this were multiple and listed as following:

- Tether Cable not in CG.
- Not instantly.
- Drag in different parts of the ROV.
- Encountering speed of the water into the propeller is not considered in the thrust mapping.

The tether of uDrone is mounted in the rear end of the electronic tube. As discussed in Christ and Wernli (2011), the cable should be placed in the CG to avoid rotation due to the cable's weight and drag.

The conclusions are that feedback loops are necessary for a better performing thrust allocation, and further investigation on encountering the speed of the water should be done. However, the precession is good enough to test control algorithms, as it will be shown in later sections.

Chapter 3

Process Model

The mathematical modeling of a process to be controlled is of significant importance to understand the problem at hand. This chapter will present the mathematical model for uDrone along with its system parameters. A setup of the differents parts of the process model is given in Figure 3.1. Relevant references in this chapter will be Fossen (2011), Sorensen (2013), Chen (2013), Brown and Hwang (2012) and references therein.

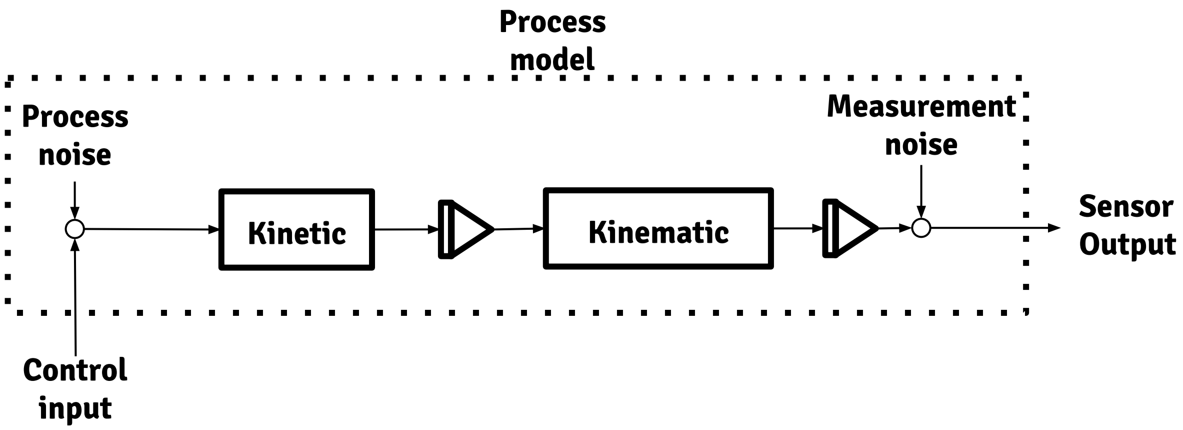


Figure 3.1: The process model consists of four parts. Firstly, the kinetic uses Newtons second law including hydrodynamics to determine the acceleration in the body frame for a given force. Secondly, the kinematics get body relative velocity from an integrator and translate it into another reference frame. In the next integrator, the position is found. Process and measurement noise are random processes which are external disturbances in to the system.

3.1 Dynamic Equations of Motion

First of, notational definitions are needed. SNAME (1950) is the most commonly and that also be the case here. See Table 3.3.

DOF		Forces and moments	Linear and angular velocities	Positions and Euler angles
Surge	motions in the x-direction	X	u	x
Sway	motions in the y-direction	Y	v	y
Heave	motions in the z-direction	Z	w	z
Roll	rotation about the x-axis	K	p	ϕ
Pitch	rotation about the y-axis	M	q	θ
Yaw	rotation about the z-axis	N	r	ψ

Table 3.1: Notation of SNAME (1950) for marine vessels

In this report, there will be two reference outlines. First, the body frame, illustrated in Figure 3.2 as the brown coordinate system which has an origin in VO as drawn in the figure. The ROV's control design will be developed relative to this point. Positions in body frame will be denoted with an lowercase b. The second is the test basin frame, illustrated as the green coordinate system in Figure 3.2. The test basin origin will be shortened to TBO. It will be defined by the Qualisys Motion Capture system which will be as a NED system when using GPS in the "real world".

For convince vectorial notation will be used. It will be based on the SNAME notation also, which is shown in Table 3.2. It should be noted that η is in TBF. For ROVs, it is convenient to describe the orientation in quaternions due to singularities in 90 degrees of pitch. But due to the 4 DOF modeling here, pitch and roll are assumed to be zero. Thereby we will use Euler angles for simplicity.
Further we have hydrodynamical coefficients that are defined as the following.

Parameter	Total	Linear	Angular
TBF position	$\eta = \left[p^T \quad \psi\right]^T \in \mathbb{R}^4$	$p = \left[x \quad y \quad z\right]^T \in \mathbb{R}^3$	$\Theta = \left[\phi \quad \theta \quad \psi\right]^T \in \mathbb{R}^3$
BODY velocity	$\nu = \left[v^T \quad r\right]^T \in \mathbb{R}^4$	$v = \left[u \quad v \quad w\right]^T \in \mathbb{R}^3$	$\omega = \left[p \quad q \quad r\right]^T \in \mathbb{R}^3$
BODY force/moment	$\tau = \left[\mathscr{F}^T \quad N\right]^T \in \mathbb{R}^4$	$\mathscr{F} = \left[X \quad Y \quad Z\right]^T \in \mathbb{R}^3$	$\mathscr{M} = \left[K \quad M \quad N\right]^T \in \mathbb{R}^3$

Table 3.2: Vectorial notation SNAME (1950) in 4 DOF with angular vectors, Θ, ω and $\mathscr{M}$ for convenience.

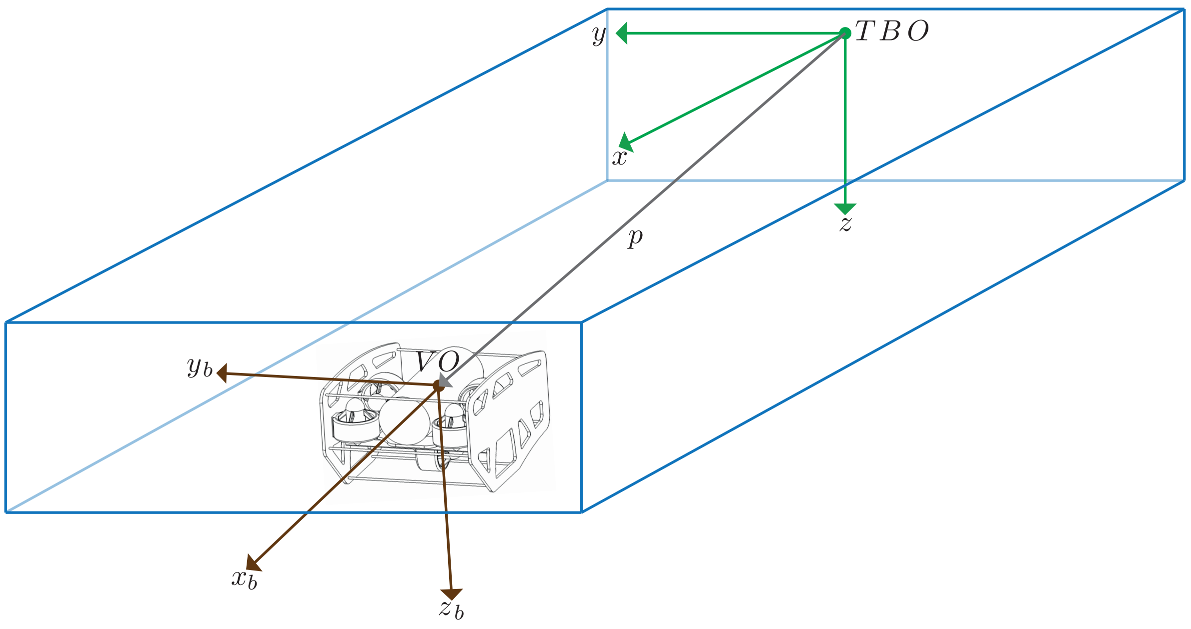


Figure 3.2: uDrone's BODY reference frame with the Test Basin Frame(TBF). The position of the uDrone is decided from vector p in TBF.

DOF	Added mass	Linear damping	Quadratic damping	Cubic damping
Surge	$X_{\dot{u}}$	X_u	$X_{\lvert u \rvert u}$	X_{uuu}
Sway	$Y_{\dot{v}}$	Y_v	$Y_{\lvert v \rvert v}$	Y_{vvv}
Heave	$Z_{\dot{w}}$	Z_w	$Z_{\lvert w \rvert w}$	Z_{www}
Yaw	$N_{\dot{r}}$	N_r	$N_{\lvert r \rvert r}$	N_{rrr}

Table 3.3: Hydrodynamical coeffients in SNAME (1950) notation.

3.1.1 Kinematics Model

In Fossen (2011) the 4 DOF kinematics model for marine vessels are stated as

$$\dot{\eta} = J(\psi)v = \begin{bmatrix} R(\psi) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ r \end{bmatrix} = \begin{bmatrix} \dot{p} \\ \dot{\psi} \end{bmatrix} \tag{3.1}$$

where

Properties	Notation
ROV mass	m
Gravity	g
Water density	ρ_w
Displacement volume	Δ
Yaw moment of inertia	I_{zz}
Boyancy force	$B = \rho g \Delta$
Weight force	$W = mg$
Distance VO to CG	$r_g = \begin{bmatrix} x_g & y_g & z_g \end{bmatrix}$
Distance VO to CB	$r_b = \begin{bmatrix} x_b & y_b & z_b \end{bmatrix}$

Table 3.4: Rigid Body parameters for mathematical modeling.

$$R(\Theta) = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

In this thesis $J_\Theta(\eta) \in \mathbb{R}^{4x4}$ will always mean rotation from BODY frame to TB frame. This means that $J_\Theta(\eta)^T$ is the inverse rotation. Furthermore, this is the reduced order model, saying only surge, sway, heave, and yaw are considered. This assumes $\theta = 0$ and $\phi = 0$ and is a fairly good assumption for a passive stable vessel as the uDrone.

3.1.2 Kinetics Model

According to Fossen (2011) the process model for a ROV is given as

$$\underbrace{M_{RB}\dot{\nu} + C_{RB}(\nu)\nu}_{Rigid\ body\ forces} + \underbrace{M_A\dot{\nu} + C_A(\nu)\nu + D_{NL}(\nu)\nu + D_L\nu}_{Hydrdynamical\ Forces} = \underbrace{\tau}_{Thrust\ Forces} \tag{3.3}$$

where

- $M_A \in \mathbb{R}^{4x4}$ is added mass which is forces from pressure induced by acceleration.
- M_RB is rigid body mass and moment of inertia.
- $D_L \in \mathbb{R}^{4x4}$ is linear damping.

- $D_{NL}(v) \in \mathbb{R}^{4x4}$ is nonlinear damping.
- $C_R B(v) \in \mathbb{R}^{4x4}$ is Coriolis forces for rigid body.
- $C_A(v) \in \mathbb{R}^{4x4}$ is Coriolis forces for added mass.

The complete matrices for the 4 DOF dynamics can be seen in Appendix B.2.

3.2 Stochastic Processes

In control engineering the signal noise is a major challenge. It can be due to external forces acting on the system, or noise due to disturbances in the sensors. The correct terms are process and measurement noise, respectively. Random processes are needed to model the nature of the disorders. These processes are a vast field of study, and there are difficult to include all of the theory, therefore, should the reader look more into for instance Brown and Hwang (2012) for a complete review. But a summary will be given here due to its extensive use in simulations and design of the Kalman Filter. Notation used here will be

Notation	Explanation
$X(t)$	Random Process
σ^2	Variance
$E[\cdot]$	Expectation
$R_x(\tau)$	Autocorrelation
$S_X(jw)$	Power Spectral Density (PSD)

Table 3.5: Statistical notation.

The most fundamental concept in statistics is the random variable with an expectation and variance. A random value can be described by a probability density function which can have an expectation which is mean and a variance which is how significant the change in the variable is. The most known probability function is Gaussian or normal distribution.
Random processes unfold along time and therefore they are useful for describing random signals. They also have an expectation value and variance. One way of describing them are whit an autocorrelation function which is as it suggests, the correlation with itself. This means how fast the process changes with the parameter τ. The definition is in Equation 3.4.

$$R_x(\tau) = E[X(t)X(t+\tau)] \tag{3.4}$$

For further obtaining the frequency content of the process, which is contained in the auto-correlation function we can take the Fourier transform as in Equation 3.5. Which is the

$$S_X(jw) = \mathscr{F}[R_x(\tau)] = \int_{-\infty}^{\infty} R_x(\tau)e^{jw\tau}d\tau \tag{3.5}$$

Some important random processes used in the mathematical model will be discussed in the next sections.

3.2.1 Gaussian White Noise

Gaussian white noise is a normally distributed random process containing all frequencies. Which usually are denoted as $w(t)$. It will be used here to describe sensor noise. A sensor measurement will be described as suggested in Fossen (2011) by Equation 3.6

$$y = Hx + w(t) \tag{3.6}$$

where $y \in \mathbb{R}^m$ is the sensors measurements. $x \in \mathbb{R}^n$ is the state vector and $H \in \mathbb{R}^{m\times n}$ gets the states corresponding to the measurement. The pure white noise has an infinite variance with a given amplitude. This is not possible in practice, but it is a nice abstraction. In the real world, it has a finite variance which is what will be used for simulations. The spectral density function is given as

$$S_w(jw) = A \tag{3.7}$$

Where A is the spectral amplitude. The autocorrelation function is given as

$$R_w(\tau) = A\delta(t) \tag{3.8}$$

where δ is the Dirac impulse function.

3.2.2 First Order Gauss-Markov Process

A Gauss-Markov Process $X(t)$ have an exponential Autocorrelation and spectral density function

$$R_X(\tau) = \sigma^2 e^{-\beta|\tau|} \tag{3.9}$$

$$S_x(j\omega) = \frac{2\sigma^2\beta}{\omega^2 + \beta^2} \tag{3.10}$$

The time constant and variance of the process is described as $\frac{1}{\beta}$ and σ^2, respectively. The mean of the function is zero. Fossen (2011) suggest to use this process to model slowly varying bias as current or the wind for marine surface vessels. For ROV's the current is most relevant, but since there are no current in the test basin other more rapid changes can be modeled from this. For instance, the boyancy is a bias that is constant but can vary from run to run. Another possible un-modelled effect can be the tether cable which pulls on the ROV depending on the drag, weight and so forth. The force from the tether is a rapid changing force, but it can still be modeled as a bias. The differential equation for the random process can be written as

$$\dot{b} = -\beta b + \sqrt{2\sigma^2\beta}\, w(t) \tag{3.11}$$

It should be mentioned that Fossen (2011) also suggest to model it as a random walk or Wiener-Motion Process which is white noise integrated. This is not chosen here due to its infinite variance.

3.2.3 Second Order Gauss-Markov Process

The second order Gauss-Markov process is very popular for modeling the velocity position relationship. This is also done in Fossen (2011) be adding a white noise to Equation 3.3. The differential can the written on state space form as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -\beta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \sqrt{2\sigma^2\beta} \end{bmatrix} w(t) \tag{3.12}$$

This means that for linearized equation and solve for the last matrix to find the desired spectrum of the Gaussian white noise.

3.2.4 Monte Carlo Simulation

To simulate the proper covariance matrices as design, it is necessary to run simulations on the stochastic processes in Simulink. This will be done by using Monte Carlo Simulations as described in Brown and Hwang (2012). The procedure is as following:

We want to simulate a Gaussian white noise with a covariance matrix Q. To do this a vector u which consist of independent Gaussian white noise with standard distribution $\mathcal{N}(0,1)$ and a given matrix C to give the Equation

$$w = Cu \tag{3.13}$$

Further, it is demanded that

$$E[(Cu)(Cu)^T] = CC^T E[(ww^T)] = Q \tag{3.14}$$

and since $E[uu^T]$is the unit the matrix in Equation 3.15.

$$CC^T = Q \tag{3.15}$$

This means that C can be found by simply performing Cholesky factorization that provides a lower and upper matrix-vector Q. The "chol" function in MATLAB can be used to simplify that process.

3.3 Final Process Model

The final process model is obtained by combining the kinematic, kinetic and stochastic processes into one model. The Gauss-Markov process for the bias will be denoted as $b \in \mathbb{R}^4$. Its time constant is written as $T_b \in \mathbb{R}^{4\times 4}$ with terms at the diagonal, strictly. The second order will be modeled by adding a white noise term to the kinetic equation. The final process model can now be described as in Equation (3.16), (3.17) and (3.18).

$$\dot{\eta} = J(\psi)\nu \tag{3.16}$$

$$(M_{RB} + M_A)\dot{\nu} = -C_A(\nu)\nu - D_{NL}(\nu)\nu - D_L\nu - C_{RB}(\nu)\nu + \tau + w_\nu \tag{3.17}$$

$$\dot{b} = -T_b^{-1}b + w_b \tag{3.18}$$

The implementation are done in Simulink and are shown in Appendix A.2.Next, Chapter 4 will deal with finding the necessary parameters to implement the model.

Chapter 4

Estimation of Parameters

4.1 Finding Rigid Body Parameters

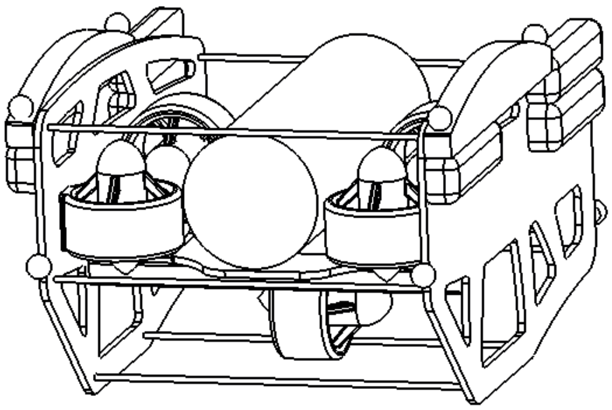


Figure 4.1: CAD model of uDrone

The rigid body properties as stated in Table 3.4, are found by analyzing the ROV with CAD (Computer Aided Design) software. A model of uDrone is already available on the BlueRobotics website BlueRobotics (2015a). However, this model is not accurate enough and therefore a new design is obtained as shown in Figure 4.1. The geometry is fitted and weight distributed to create an a representative model. The ROV chassis, thrusters, electronic housing, markers, and floaters are the main parts of the model. The mass is calculated for each one by dividing their measured weight on the estimated volume. Their density is the set in the CAD model. From this, estimates of the moment of inertia, CG and CB are obtained, being that the last two measures are the distances relative to VO. CG is chosen as the VO point, which also can be seen in Figure 2.1. Table 4.1 shows all the values. It should be noted that these values strongly depend on the placement of parts inside the electronic housing and new gear mounted to the ROV. So it is essential that the battery and electronic housing are correspondingly located as in the model for the estimation to be accurate.

Parameter	Value	Unit
m	7.31	$[kg]$
ρ_w	1000	$[kg/m^3]$
g	9.81	$[m/s^2]$
I_{zz}	0.16	$[kg/m^2]$
Δ	$7.99e-3$	$[m^3]$
r_g	$\begin{bmatrix}0 & 0 & 0\end{bmatrix}^T$	$[m]$
r_b	$\begin{bmatrix}0 & 0 & 0.00019\end{bmatrix}^T$	$[m]$

Table 4.1: Mass and Moment of Inertia Properties. All geometry properties are taken from the CAD model. CG and CB distances with respect to VO. r_g is CG distance and r_b is CB distance.

4.2 Added Mass Parameters Estimation

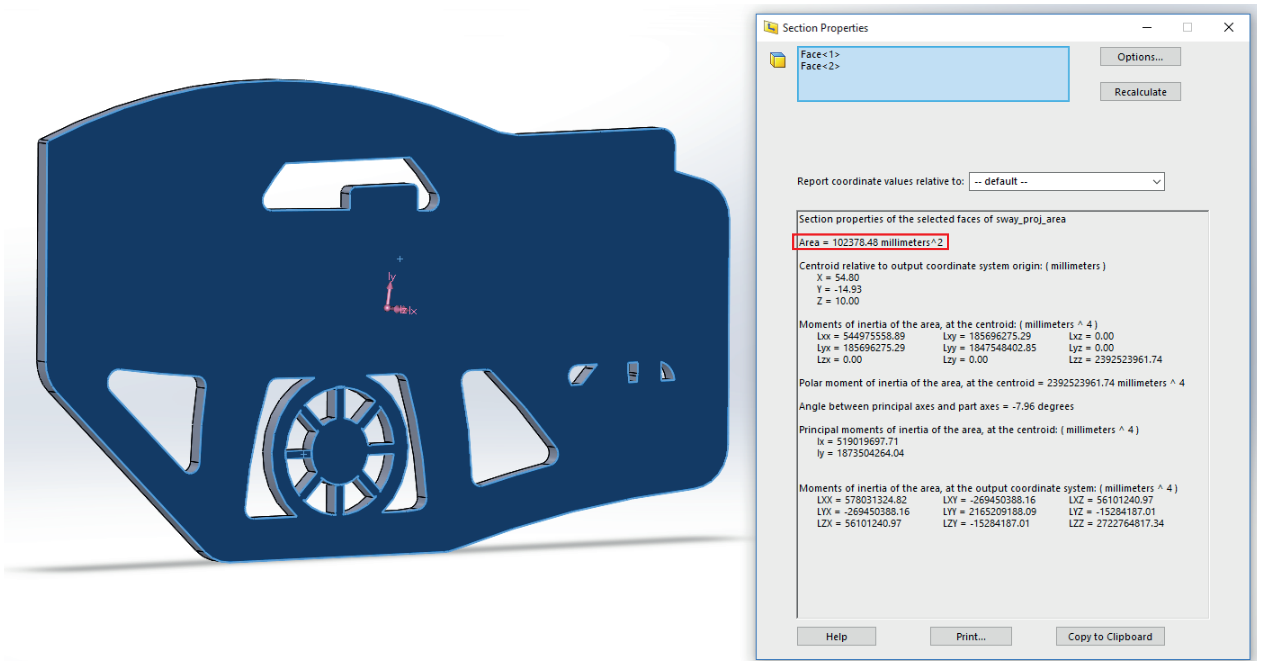
Added mass is the pressure that is induced on the body during acceleration. The parameter is used in M_A and C_A in equation 3.3. Eidsvik (2015) discusses a simple method for its estimation. It uses experimental data of box-shaped volumes for estimating added mass for similar shaped ROVs. Wadam simulations, which is numerical computational software for hydrodynamics, verified its accuracy. The uDrone is boxed-shape, and this method is, therefore, adequate. Another possibility is the use of software like Wadam directly, but that requires a significant workload to obtain a satisfying design model of proper fidelity. Another option is to use the step response. On Aras et al. MATLAB's system identification toolbox were used to analyze the response from a control input, as discussed in Ljung (1999). The added mass found by Eidsvik's method will be tested by using stepwise input and observing the response. Then the system identification toolbox in MATLAB will be used to identify the parameters.

4.2.1 Setup of Eidsvik's Method

The parameters required are shown in Table 4.2.

Properties	Notation	Value	Unit
Width	W	333	mm
Height	H	287	mm
Length	L	482	mm
Projected area front	A_p^{XY}	55915	mm^2
Projected area side	A_p^{YZ}	102378	mm^2
Projected area top	A_p^{XZ}	117375	mm^2
Density of water	ρ	1000	kg/m^3

Table 4.2: All lengths and projected areas required for using Eidsvik's method. All were obtained from the CAD model presented in Appendix C.



Report coordinate values relative to: -- default --

```
Section properties of the selected faces of sway_proj_area

Area = 102378.48 millimeters^2

Centroid relative to output coordinate system origin: ( millimeters )
    X = 54.80
    Y = -14.93
    Z = 10.00

Moments of inertia of the area, at the centroid: ( millimeters ^ 4 )
    Lxx = 544975558.89        Lxy = 185696275.29        Lxz = 0.00
    Lyx = 185696275.29        Lyy = 1847548402.85       Lyz = 0.00
    Lzx = 0.00                Lzy = 0.00                Lzz = 2392523961.74

Polar moment of inertia of the area, at the centroid = 2392523961.74 millimeters ^ 4

Angle between principal axes and part axes = -7.96 degrees

Principal moments of inertia of the area, at the centroid: ( millimeters ^ 4 )
    Ix = 519019697.71
    Iy = 1873504264.04


Moments of inertia of the area, at the output coordinate system: ( millimeters ^ 4 )
    LXX = 578031324.82        LXY = -269450388.16       LXZ = 56101240.97
    LYX = -269450388.16       LYY = 2165209188.09       LYZ = -15284187.01
    LZX = 56101240.97         LZY = -15284187.01        LZZ = 2722764817.34
```

Figure 4.2: The obtained projected area in sway in SolidWorks. The area is indicated by a red rectangle. All CAD models are included in Appendix C.

The parameters were obtained from the complete CAD model of uDrone. Projected areas required some extra drawings using the section properties function as seen in Figure 4.2. Estimating the added mass using experimental data from DNV is based on the use of rectangular prisms where two of three sides are equal in size DNV-RP-H103 (2010). This is a fair assumption for uDrone, where the width is 14% smaller than the height. This is close to the assumption in

Eidsvik (2015), that two of three sides have a relative side of no more than ±10%.

The method uses also the projected area as a scaling factor, since the whole volume is usually not covered covered. This is also the case for the uDrone. The coefficients are defined as:

$$C_p^{XY} = \frac{A_p^{XY}}{LW} \tag{4.1}$$

$$C_p^{YZ} = \frac{A_p^{YZ}}{HW} \tag{4.2}$$

$$C_p^{XZ} = \frac{A_p^{XZ}}{LH} \tag{4.3}$$

Subscripts definitions for the formulas are defined in Table 4.3.

DOF	m	n	o
Surge	X	Y	Z
Sway	Y	Z	X
Heave and Yaw	Z	X	Y

Table 4.3: Notion of subscript used in the experimental data. The notation is the same as used in Eidsvik (2015)

Firstly, the method computes the added mass coefficient in surge by using 3D experimental data by Equation 4.4.

$$X_{\dot{u}} = C_a V_R \rho_{water} (C_p^{YZ})^2 (C_p^{XZ})(C_p^{XY}) \tag{4.4}$$

Where $V_R = aab$ is the reference volume of the experimental data along to equal length sides a and a length b. C_A is the interpolated added mass coefficient calculated from the experimental data. To find the difference between strip-theory and experimental 3D data, added mass in surge is also calculated by strip-theory. For this the reason a reference area is needed instead of a reference volume, and is given by:

$$A_R = \pi a^2$$

The 2D added mass coefficient is then computed by Equation (4.5) for surge and integrated over the length of the ROV to obtain the computed 3D coefficient in Equation (4.6). A_{ii} is the notation for an added mass in a degree of freedom, and are written this way for generality.

$$A_{ii}^{2D} = \rho C_a A_R (C_p^{no})^2 (C_p^{mo})(C_p^{mn}) \tag{4.5}$$

$$A_{ii} = \int_{-L/2}^{L/2} A_{ii}^{2D} dx \tag{4.6}$$

A scaling factor between experimental and strip-theory is found to be:

$$\lambda = \frac{X_{\dot{u}}^{empirical-3D}}{X_{\dot{u}}^{strip-theory}} \tag{4.7}$$

For calculating the remaining added mass in sway, heave and yaw, are strip-theory along with the scaling factor λ applied. Sway and Heave are computed by using Equation 4.5, 4.6 and the scaling factor λ.

The added mass in yaw is also found from strip-theory using the 2D added mass coefficient from DNV-RP-H103 (2010). The expression for the computations are given in Equation (4.8) and (4.9).

$$N_{\dot{r}}^{2D} = \rho C_a \pi a^4 (C_p^{XY} C_p^{ZY} C_p^{ZX}) \tag{4.8}$$

$$N_{\dot{r}}' = \int_{-L/2}^{L/2} N_{\dot{r}}^{2D} dx \tag{4.9}$$

$$\tag{4.10}$$

To find the final result, strip theory multiplied with the scaling factor λ is used. This is given by

$$N_{\dot{r}} = N_{\dot{r}}' \lambda$$

The computations done are given in Appendix A.5. The script used were taken form Eidsvik (2015).

4.2.2 Setup of System Identification for Added Mass in Surge

In order to verify the added mass approximations, an identification test was performed in surge direction. Decoupling and linearization of Equation (3.3) gives the following equation for surge dynamics

$$(m + X_{\dot{u}})\dot{u} + X_u u = \tau \tag{4.11}$$

By using Laplace transformation we obtain the following results

$$\begin{aligned}(m + X_{\dot{u}})us - X_u u = \tau \\ u = \frac{1}{(m - X_{\dot{u}})s + X_u}\tau_u\end{aligned} \tag{4.12}$$

For estimating the results of the coefficients, Equation (4.13) will be the one used. By subtracting the mass m, the added mass is obtained along with the linear damping. This damping parameter can also be compared with the ones obtained in the towing tests.

$$h(s) = \frac{\frac{1}{(m - X_{\dot{u}})}}{s + \frac{X_u}{(m - X_{\dot{u}})}} \tag{4.13}$$

The experimental setup of the system identification will be as follows:

1. Use the designed thrust allocation in Chapter 2 to get a clean surge motion.
2. Oscillate the ROV back and forth in surge by using relay for switching the constant force.
3. Log force inputs and position outputs with Qualisys.
4. Find velocities by derivation of position and run the logged force inputs with the velocity outputs in the system identification toolbox. The model used is a transfer function with one pole.
5. Obtain the added mass coefficient with Equation. (4.13).

The method was only applied in surge direction due to problems with clean motion in the other degrees of freedom.

4.2.3 Results of Eidsvik's Method

The added mass parameters using Eidsvik's method were found to be

Added mass term	Value
$X_{\dot{u}}$	-5.50
$Y_{\dot{v}}$	-12.70
$Z_{\dot{w}}$	-14.57
$N_{\dot{r}}$	-0.12

Table 4.4: Added Mass found by the method developed in Eidsvik (2015)

4.2.4 Results of the System Identification Toolbox

The results from the system identification toolbox include the transfer function of $h(s) = \frac{0.07769}{s+1.197}$ by applying Equation (4.13). From this, the added mass was found to be as in Equation (4.14).

$$-X_{\dot{u}} = \frac{1}{0.07769} - m = 5.517 \tag{4.14}$$

Simulations of the estimated added mass and real output from the added mass are shown in Figure 4.3.

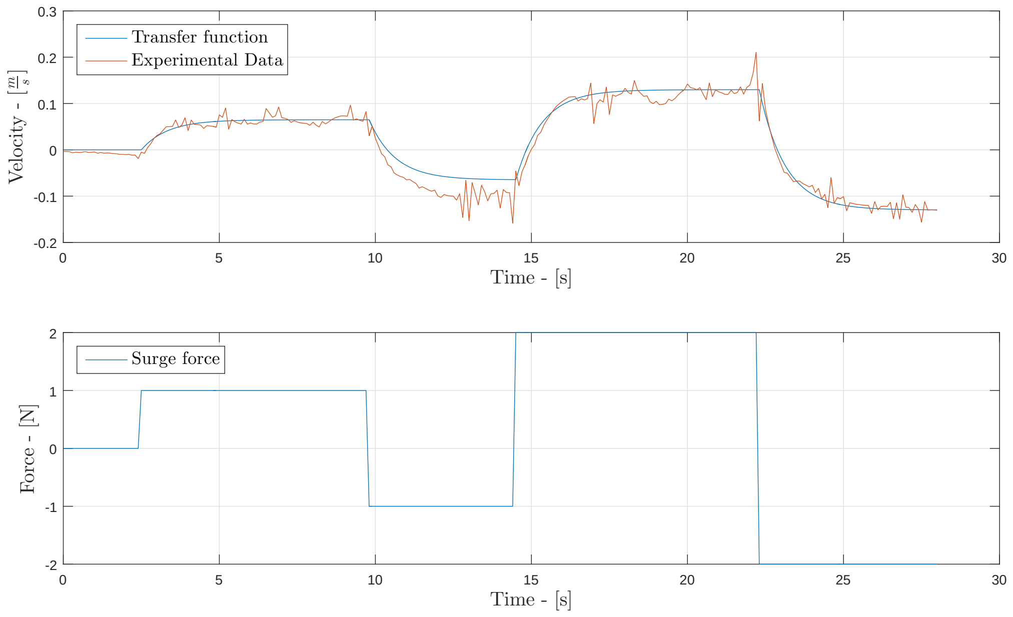


Figure 4.3: Simulated versus real output

Added mass term	Value
$X_{\dot{u}}$	-5.52

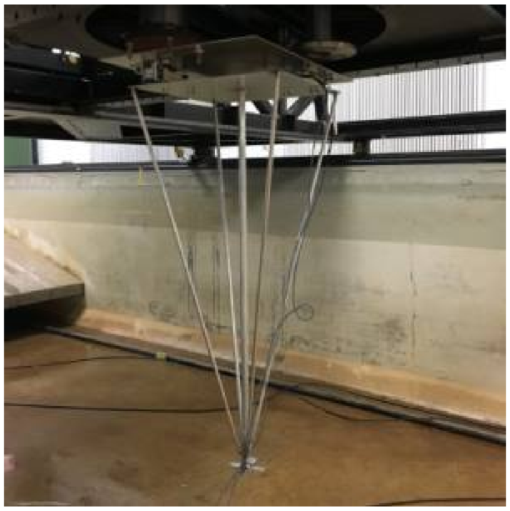
Table 4.5: Added Mass found by using estimations from the System Identification Toolbox developed in Ljung (1999)

4.3 Damping Parameters from Towing Tests

This section goes through the experiments for getting the damping parameters for the uDrone. As shown in Equation 3.3, damping consists of both a linear and nonlinear terms. In Eidsvik (2015) an experimental towing test was performed which is an effective way of finding the translator forces.

4.3.1 Procedure for obtaining Surge, Sway and Heave Damping Parameters

The procedure is very similar to the one done for obtaining thrust, the difference consist in driving of the towing bridge instead of the thrusters. Each degree of freedom is tested in five positive and five negative velocities ranging from -0.5 to 0.5 m/s. The force is measured using the same principle and rig as in the thrust allocation section. The setup of the rig can be seen in Figure 4.4a and Figure 4.4b. Configurations are shown in Figure 2.4.


(a) Setup of test rig on bridge.

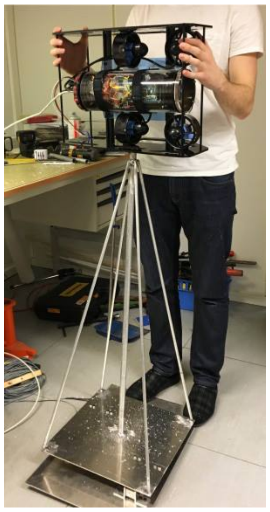

(b) ROV setup on rig for testing heave forces.

Figure 4.4: Towing rig setup.

- Mount the rig in the desierd configuration as shown in Figure 2.4.
- Drive the towing rig in the velocites -0.5 to 0.5 with 10 speeds.
- Obtain the results and read them in the scrips.
- Subtract the rig forces.
- Curve fit the resulting system with a first, second and third order terms.

4.3.2 Procedure for Obtaining Yaw Damping Parameters

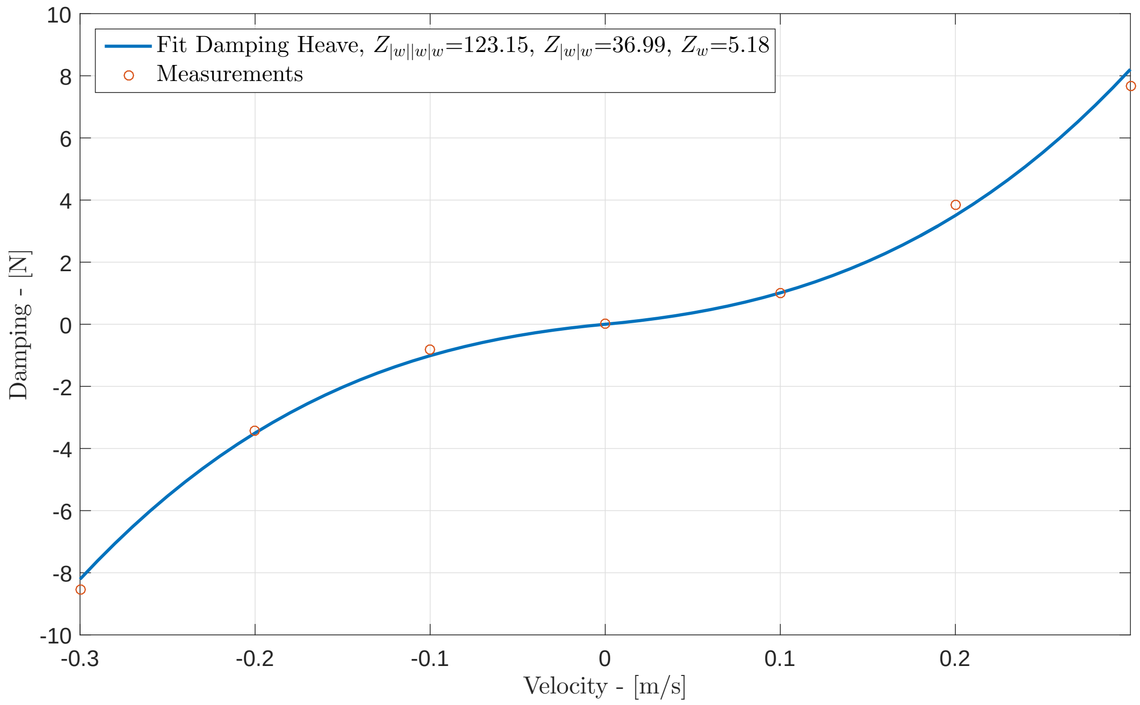
The damping coefficients in yaw were found by using the thrust allocation exerting a constant moment on the ROV. The five moments were 0.25, 0.5, 0.75, 1 and 2. The velocities were then found by taking the derivative of the heading angle. By mapping the moment against the angular velocity and doing a curve fit, the damping coefficients were obtained. Scripts and time series can be found in Appendix B.1.2.

4.3.3 Results for Towing Tests

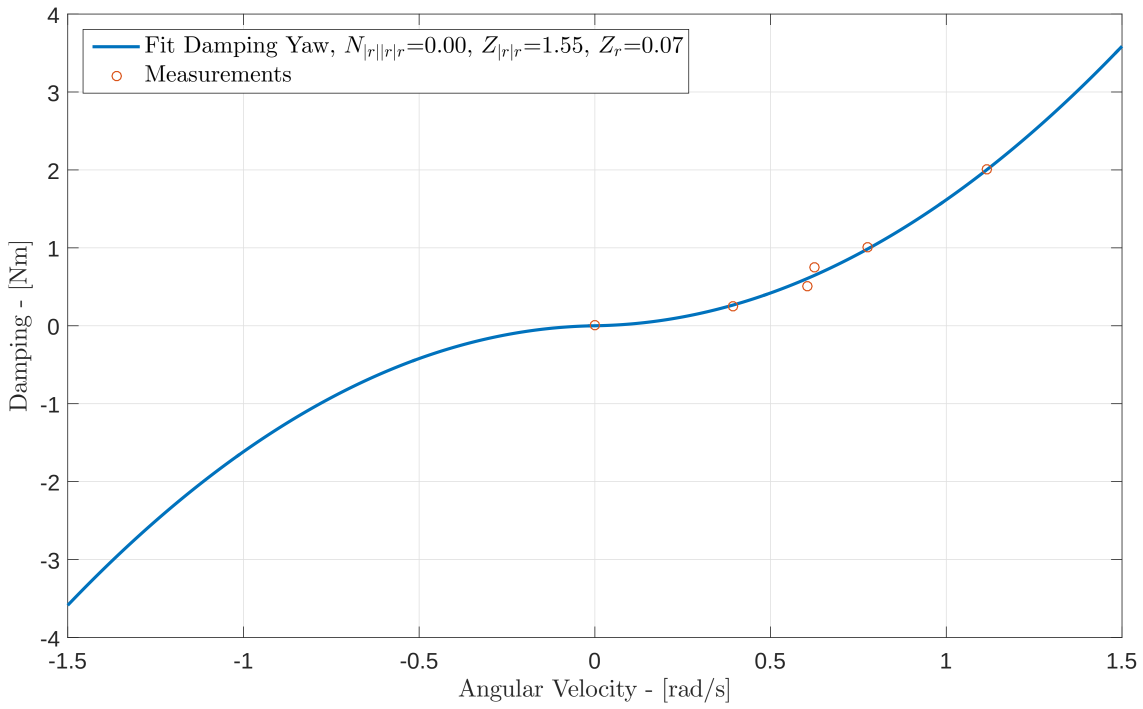
The results of the towing test and driving tests can be viewed in Table 4.6. The result is also presented graphically in Figures 4.6 and 4.5.

Linear term	Value	Quadratic term	Value	Cubic term	Value
X_u	-4.03	$X_{\|u\|u}$	-18.18	$X_{\|u\|\|u\|u}$	-24.24
Y_v	-6.22	$Y_{\|v\|v}$	-21.66	$Y_{\|v\|\|v\|v}$	-128.52
Z_w	-5.18	$Z_{\|w\|w}$	-36.99	$Z_{\|w\|\|w\|w}$	-123.15
N_r	-0.07	$N_{\|r\|r}$	-1.55	$N_{\|r\|\|r\|r}$	0

Table 4.6: Damping coefficients in four DOF.

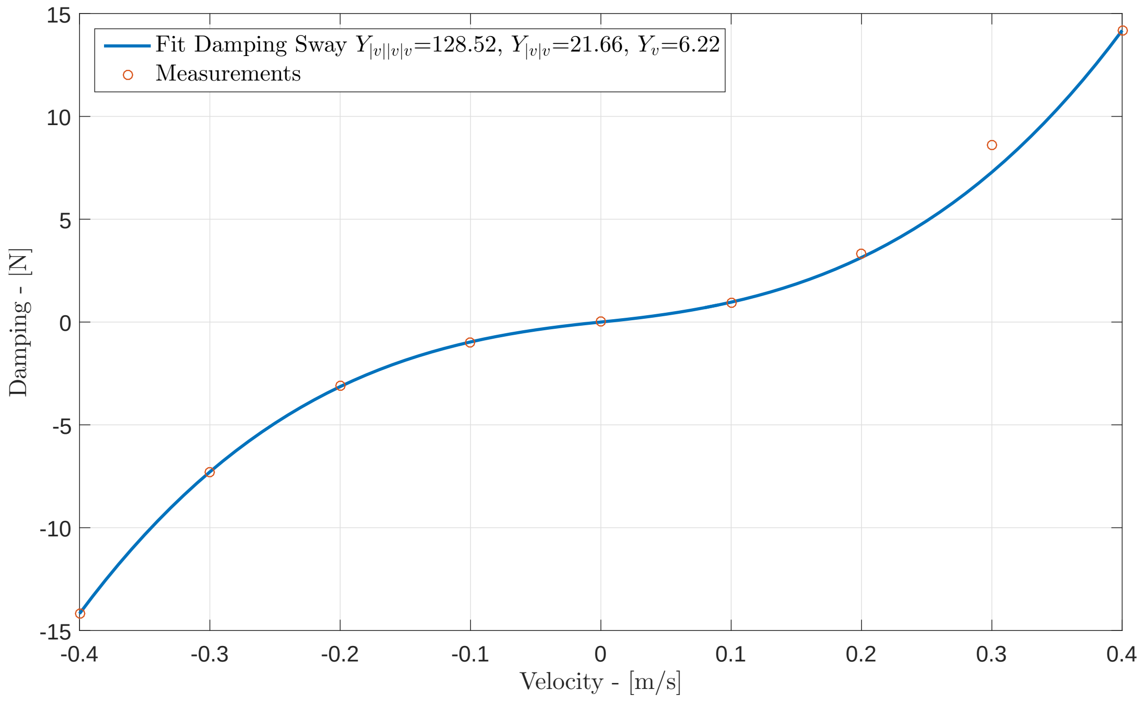


(a) Damping in heave

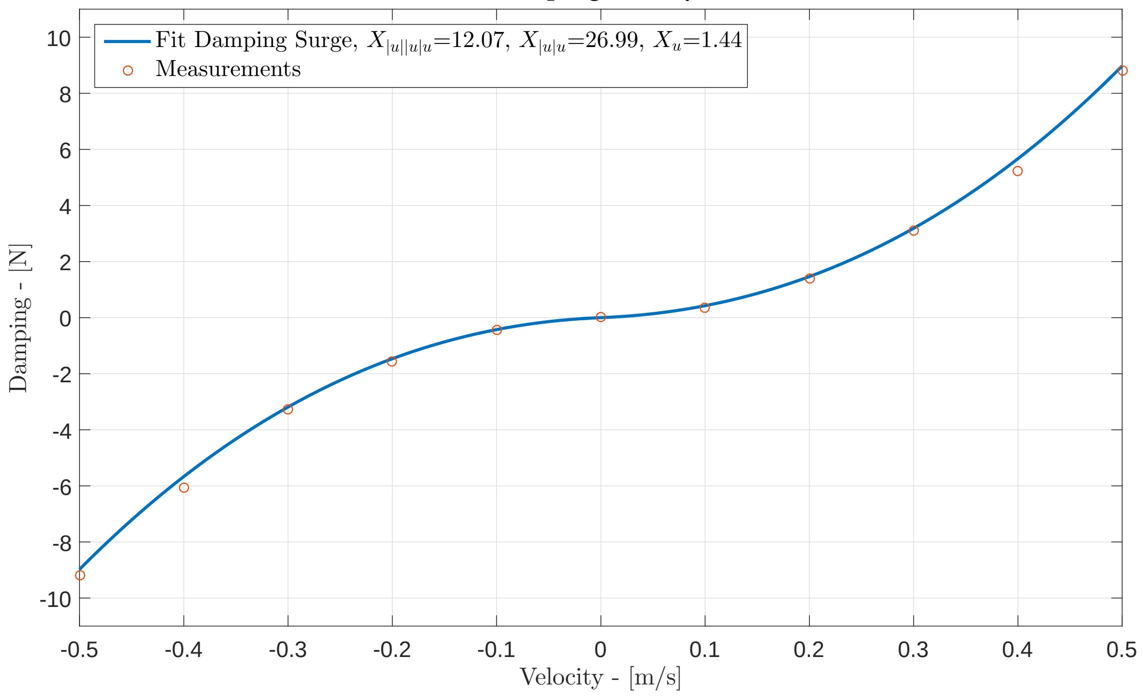


(b) Damping in Yaw

Figure 4.5: Damping Curve Fit



(a) Damping in sway



(b) Damping in surge

Figure 4.6: Damping Curve Fit

4.4 Sensor Suite and Measurement Noise

The sensor suite is stated in Subsection 1.4.1. The measurements used here will be from Qualisys and the pressure sensor for depth. Equation 4.15 states all measurements in the observers.

$$y_1 = \begin{bmatrix} x_{Qual} & y_{Qual} & z_{Qual} & z_{Pressure} & \psi_{Qual} \end{bmatrix} \tag{4.15}$$

The orientation of the ROV will be used even though the model will only be 4 DOF. From this we have a second measurement vector in Equation 4.16

$$y_2 = \begin{bmatrix} \phi_{Qual} & \theta_{Qual} & \phi_{IMU} & \theta_{IMU} \end{bmatrix} \tag{4.16}$$

To find the correct measurement, it is important to consider the position of the sensor relative to the VO (Vessel Origin). The distances from the sensor to VO which are also defined as CG are shown in Table 4.7, and were found by the design model of the ROV in SolidWorks.

Sensor	Distance from p_m to p_b	Unit
IMU sensor suite	$\begin{bmatrix} 0.12315 & 0 & -0.0465 \end{bmatrix}^T$	$[m]$
Qualisys	$\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$	$[m]$

Table 4.7: Distances from sensor measurement to VO for x,y and z, in meters. The Qualisys measurements can be adjusted in the system, but for convenience, it is better to take the geometrical center and translate it into VO. This way, it is also simpler to change VO.

4.4.1 Sensor Noise

Furthermore, to get a realistic sensor noise and to make tuning of observers easier it is important to consider the measurement noise. The variance was obtained by logging 6796 measurements with a sampling frequency of $10Hz$ for each sensor. The uDrone was submerged with a weight to have realistic data measurements from the test basin. The variances are then calculated from the resulting time series by using MATLAB's "var" function. The results can be seen in Table 4.8. The time series and computations can be found in Appendix A.8.

Sensor	Measurement	Variance(σ^2)	Unit
Euler Angle	ϕ	1.0584e-06	$[rad^2]$
	θ	2.7771178e-07	$[\frac{m}{s}^2]$
Pressure Sensor	z	1.189622e-04	$[m^2]$
Qualisys	x	2.727425e-08	$[m^2]$
	y	2.584928e-08	$[m^2]$
	z	1.987803e-09	$[m^2]$
	ϕ	3.678414e-07	$[rad^2]$
	θ	8.842403e-08	$[rad^2]$
	ψ	1.641552e-06	$[rad^2]$

Table 4.8: Available sensors for uDrone in the test basin with there measurements. There corresponding measurement noise is also shown in the third column

4.5 Discussion

The estimation of the rigid body parameters was done as explained in Section 4.1 found by analyzing a CAD model. The estimates in Table 3.4 are not entirely accurate. For instance, the buoyancy from the estimation of the volume turned out to be several 100 g less positive than expected looking into the bias. The mass and buoyancy are also always changing depending on how much water the floating element absorbs an so forth.

Eidsvik's method gives a rough estimate of the added mass. This also happens since the ROV is not inside the ±10%. Therefore, more consideration should be taken.

The system identification of surge was done by using a thrust allocation that does not give an exact clean motion due to multiple problems as discussed in the Chapter 2. Further implementing PD controllers to provide more accurate movements is a possibility, but it still will not remove the errors in forces due to open loop thrust allocation. The result of the SI is however remarkably similar to Eidsvik's method. SI method is expected to be somewhat bigger than Eidsvik's method due to lack of tether modeling for both mass and added mass. These results could be worth exploring later in further work.

Finding damping parameters was more difficult than initially expected. It was assumed that no modifications were necessary on the rig. This turned out to be false, and adjustments were needed. When the testing started, there were problems with bending in the rig as mentioned in the thrust allocation chapter. This led to poor results for high forces, however due to only low-speed considerations, this never became a problem.

The test results for heave, surge and sway, shown in the Figures 4.5a, 4.6b and 4.6a are as expected when it comes to their size of the projected area as stated in Table 4.2. The heave forces are the largest, followed by sway and surge, respectively. It should be noted that the tether forces are not considered in the towing tests.

The yaw forces were not considered in towing due to the obtention of poor results. They were instead tested with constant moments by using the thrust allocation and observing the stationary ending velocities. They were then fitted as shown in Figure 4.5b. Test results in yaw are opposed to the results from the towing test, also considering the tether forces. However, since the forces will vary on cable length and buoyancy among other things it will only give a rough estimate compared with a towing test. The thrust allocation accuracy will also play a major role here. The roughness of the estimation can be seen in the third and fourth measuring points in Figure 4.5b. It should be noted that all coefficients are positive. This is exactly what one should be expecting in the "real world". Otherwise, the system would have negative damping which means that it is an unstable system.

Chapter 5

State Estimation and Control

The main reasons for which an ROV system like uDrone needs an observer (e.g. a state estimator) are listed in the following:

- Allows disturbance rejection for the measured states;
- Estimates states that are not measured;
- Gives state updates during sensor dropout.

This chapter will designed observer for uDrone, which is the discrete Kalman Filter. Simulation and experimental results will also be presented.

5.1 Observer Design

One linear model for use in estimating depth and heading. The sensors at hand here will be the pressure sensor and Qualisys.

5.1.1 Observer Model

The observer model developed is used in low velocities. This observer will be used together with autodepth and autoheading controls. The continuous model uses the linearized process model given in (3.1) and (3.3) which renders Equations (5.1), (5.2) , (5.3) and (5.4):

$$\dot{z} = w \tag{5.1}$$

$$\dot{\psi} = r \tag{5.2}$$

$$\dot{w} = -\frac{-Z_w}{M_w} w + \frac{1}{M_z} \mathscr{Z} \tag{5.3}$$

$$\dot{r} = -\frac{-N_r}{M_r} r + \frac{1}{M_r} \mathscr{N} \tag{5.4}$$

Since the model is not perfect, the process disturbances in the system need to be modeled. As already discussed, this is done by adding a first and second order Guass-Markov Process for slowly varying biases and noise in the velocity measurements. By defining the biases b_w and b_r and adding white noise to the kinetics Equations (5.3) and (5.4) we get the full observer model relating to the Gause-Markov Process from Equation (3.11). Equation (5.1), (5.1), (5.5), (5.6), (5.7) and (5.7) represent the full observer model.

$$\dot{w} = -\frac{-Z_w}{M_w} w + \frac{1}{M_z} \mathscr{Z} + \frac{1}{M_z} b_w + \frac{1}{M_z} w_w \tag{5.5}$$

$$\dot{r} = -\frac{-N_r}{M_r} r + \frac{1}{M_r} \mathscr{N} + \frac{1}{M_r} b_r + \frac{1}{M_r} w_r \tag{5.6}$$

$$\dot{b}_w = -\frac{1}{T_w} b_w + w_{b_w} \tag{5.7}$$

$$\dot{b}_r = -\frac{1}{T_r} b_r + w_{b_r} \tag{5.8}$$

Here, the stochastic processes have white noise $w \in \mathbb{R}^6$. The time constants are the last term in the square root as the β in Equation 3.11 with the Gaussian distribution $w \sim N(0, Q)$ which means $E[w] = 0$ and $Cov[w] = Q$. To find the covariance Van Loan Method, is used. This is further discussed in the design of the Kalman filter.

By defining the variables $\eta_{dh} = \begin{bmatrix} z & \psi \end{bmatrix}$, $\nu_{dh} = \begin{bmatrix} w & r \end{bmatrix}^T$ and $b_{dh} = \begin{bmatrix} b_z & b_\psi \end{bmatrix}^T$, the system can be written in matrix form as represented in Equation (5.9),(5.10) and(5.11).

$$\dot{\eta}_{dh} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \nu_{dh} \tag{5.9}$$

$$\dot{\nu}_{dh} = -\begin{bmatrix} \frac{-Z_w}{M_w} & 0 \\ 0 & \frac{-N_r}{M_r} \end{bmatrix} \nu_{dh} + \begin{bmatrix} \frac{1}{M_w} & 0 \\ 0 & \frac{1}{M_r} \end{bmatrix} [\tau + b_{dh} + w_m] \tag{5.10}$$

$$\dot{b}_{dh} = -\begin{bmatrix} \frac{1}{T_w} & 0 \\ 0 & \frac{1}{T_\psi} \end{bmatrix} b_{dh} + w_b \tag{5.11}$$

The "DH" is the abbreviation for Depth and Heading model. If the states are defined as x_{d_h}, u_{dh} and $w_d h$ as in Equation (5.12).

$$x_{dh} = \begin{bmatrix} \eta_{dh} \\ \nu_{dh} \\ b_{dh} \end{bmatrix}, \quad u_{dh} = \begin{bmatrix} 0 \\ \tau \\ 0 \end{bmatrix}, \quad w_{dh} = \begin{bmatrix} 0 \\ w_m \\ w_b \end{bmatrix} \tag{5.12}$$

Then the continuous state space model can be defined as in Equation (5.14).

$$\dot{x}_{dh} = A_{dh} x_{dh} + B_{dh} u_{dh} + E_{dh} w_{dh} \tag{5.13}$$

$$y_{dh} = H_{dh} x_{dh} + v_{dh} \tag{5.14}$$

where $y_{dh} \in \mathbb{R}^{3x1}$ is the measurement vector, $H \in \mathbb{R}^{mx6}$ depends on the measurement vector $y_{dh} \in \mathbb{R}^m$ and $v_{dh} \in \mathbb{R}^m$ is the measurment noise, where $v_{dh} \sim N(0, R)$. The matrices are shown in Equation (5.15).

$$A_{dh} = \begin{bmatrix} 0 & I & 0 \\ 0 & -M_{dh}^{-1} D_{L_{dh}} & M_{dh}^{-1} \\ 0 & 0 & -T_{dh}^{-1} \end{bmatrix} \in \mathbb{R}^{6x6}, \; B_{dh} = \begin{bmatrix} 0 \\ M_{dh}^{-1} \\ 0 \end{bmatrix} \in \mathbb{R}^{6x2}, \; E_{dh} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & M_{dh}^{-1} & 0 \\ 0 & 0 & I \end{bmatrix} \in \mathbb{R}^{6x6} \tag{5.15}$$

Observability

In order for the observer model in Equation (5.13) and (5.14) to be used by the Kalman Filter the systems observability needs to be checked. This means that for a measured signal y we can reconstruct all the states x in the model. In Chen (2013), the system is observable if the observability matrix have full rank. By calculation of Equation (5.16) it can be confirmed that the system is observable.

$$rank(\mathscr{O}) = rank(\begin{bmatrix} H \\ HA \\ \vdots \\ HA^6 \end{bmatrix}) = 6 \tag{5.16}$$

Discretization

The chosen observer design is a discrete Kalman Filter. A discrete state space model stated in Equation (5.17) and (5.18) is therefore needed.

$$x_{k+1} = \phi x_k + \Delta u_k + \Gamma w_k \tag{5.17}$$

$$y_k = H x_k + v_k \tag{5.18}$$

Where x_{k+1} and $x_k \in \mathbb{R}^6$ are the state vector with six states. $\phi \in \mathbb{R}^{6\times 6}$, $\Delta \in \mathbb{R}^{6\times 2}$ and $\Gamma \in \mathbb{R}^{6\times 6}$ is the state transition matrices relating a previous state, control input and white noise to the next state vector. The "exact discretization" can be used to find the discretization. It is given as in Chen (2013) in the following equation:

$$\phi = e^{A_{dh}T}, \quad \Delta = \int_0^T e^{A_{dh}\tau} d\tau B_d h, \quad \Gamma = \int_0^T e^{A_{dh}\tau} d\tau E_{dh}, \quad H = H_{dh} \tag{5.19}$$

The calculations have been done using MATLAB's "c2d" method which uses exact discretization. The calculations are shown in the MATLAB-script in Appendix A.3. The assumptions needed to have an correct exact discretization is constant control input over each time step. The time step used here are 0.1 seconds. This means that the thrust only changes once every

0.1 second. This is hard to do in reality, but it is still a good approximation in use with a controller. Calculations of the covariances in the process and sensor noise are also important. This is a hard task to do analytical, therefore a numerical procedure worked out in Loan (1978) and formulated in page 126 of Brown and Hwang (2012). The procedure will be restated here.

1. Find the 12×12 matrix F by Equation (5.20)
2. Use the MATLAB function "$expm(F)$" to get e^F, Equation 5.21 is obtained.
3. Obtian Q by multiplication of the upper right part of the matrix with the lower part as in Equation 5.22.

$$F = \begin{bmatrix} -A & EWE^T \\ 0 & A^T \end{bmatrix} \tag{5.20}$$

$$G = \begin{bmatrix} \dots & \phi^{-1}Q \\ 0 & \phi \end{bmatrix} \tag{5.21}$$

$$Q = \phi\phi^{-1}Q \tag{5.22}$$

The $W \in \mathbb{R}^{6\times 6}$ is a scaling factor for the power spectral density of the white noise. Q will be used in the discrete Kalman Filter and the design of process model. W are the tunable parameters in the stochastic processes along with the time constants T_{b_w} and T_{b_r}. They are tuned to fit the process in the experiment. R is given from the sensor variance measurement in Section 4.4. All the calculations are done in MATLAB and can be found in Appendix A.3.

5.1.2 Discrete Kalman Filter Design

The recursive discrete Kalman Filter can be developed on the optimization criteria of minimizing the mean-sqeare estimation error in the random process in Equation (5.17) and (5.18). The filter assumes that the covariances of the process noises are Gaussian distributed with $w \sim N(0, Q)$ and $v \sim N(0, R)$ and that they are uncorrelated with each other. The equations will

1. Initialization
$\bar{x}_0$ and $\bar{P}_0$
2. Compute Kalman Gain
$K_k = \bar{P}_k H^T (H\bar{P}_k H^T + R_k)^{-1}$
3. Correct former estimate
$\hat{x}_k \quad = \bar{x} + K[y_k - H\bar{x}]$
$\hat{P}_k \quad = [I - K_k H]\bar{P}_k[I - K_k H]^T + K_k R_k K_k^T$
4. Project ahead
$\bar{x}_k \quad = \phi\hat{x}_k + \Delta u_k$
$\bar{P}_k \quad = \phi\bar{P}_k\phi^T + \Gamma Q \Gamma^T$

Table 5.1: Corrector and Predictor of discrete Kalman Filter developed in Kalman (1960).

not be derivate here, but can be seen in for instance Brown and Hwang (2012). The Corrector and Predictor terms of the Kalman Filter from Kalman (1960) are given in Table 5.1.

In this notation, $\bar{x}_k$ and $\bar{P}_k$ are projected estimates for the previous step, while $\hat{x}_k$ and $\hat{P}_k$ is the corrected estimate for the states and covariances of step k, respectively. K_k is the Kalman gain computed by previous covariance estimates weighing the sensor updates against the noise in the process. As the equation shows, a small R compared with the covariances makes a large weight K_k thereby favoring the measurement over the model. The opposite happens, if R is large when compared with Q.

Sensor Fusion

There are two redundant sensor outputs when driving the uDrone. These are the Qualisys positioning system measuring the depth and heading with the pressure sensor and compass, respectively. Here, only the redundancy in heave is considered, due to the errorness measurements from the compass, probably because of the existence of many magnetic fields in the MC-lab. To merge two sensors it is important to consider the position of measurement, such that both sensors can relate to the same point. The desired measurement should be in VO, as defined in Chapter 3. Equation (5.23) defines the relationship between the measured position (p_m) and the VO (p_b).

$$p_b = p_m - R(\Theta)l \tag{5.23}$$

Where $l \in \mathbb{R}^3$ is the position of the pressure sensor or Qualisys relative to VO and $R(\Theta) \in \mathbb{R}^{3\times3}$.

The sensor fusion using the Kalman Filter is very simple, since the filter already defines the variances for all the sensors from their covariance, R. It is only necessary to alter the H, and the Kalman Gain weights them based on their variances. The chosen H and R for sensor fusion with the measurement $y \in \mathbb{R}^3$ in Equation (5.24), are defined as (5.25) and (5.26). The R matrix is obtained from the variance measurement given in Section 4.4.

$$y = \begin{bmatrix} z_{Qual} & z_{Pressure} & \psi_{Qual} \end{bmatrix} \tag{5.24}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.25}$$

$$R = \begin{bmatrix} 1.987803e-09 & 0 & 0 \\ 0 & 1.189622e-04 & 0 \\ 0 & 0 & 1.641552e-06 \end{bmatrix} \tag{5.26}$$

Dead Reckoning

When there is a partly or total sensor dropout it is necessary to change the R matrix also known as the sensor variance in the system. By setting a large variance on the failing sensor in comparison with redundant sensors and the process noise, the sensor is neglected in the algorithm. This means zero Kalman Gain, K_k, for the sensor's weight as discussed in Section 5.1. This method of dead reckoning is implemented here. As suggested in Sorensen (2013), checking signal variance can be a useful method. If the variance is higher or lower than a threshold, the signal is defined as erroneous and drop-out procedure is initiated. This is not implemented here, but is nice to mention as a possible procedure.

5.2 Control Design

To do some testing of the performance of the observer we need to implement some controllers. An autodepth and autoheading controller are implemented for the Kalman Filter, from which a LQG controller is obtained.

5.2.1 Control Model

To do this, a control model is needed and it is natural to use the same model as in the observer as in Equation (5.13) and (5.14). The difference from the observer model is the stochastic processes. This happens because the Kalman Filter should separate the model signal from external disturbances. However, the bias estimated in the Kalman filter is feed-forwarded from the observer directly into the LQR, since the bias is considered a constant error for the model relative to the reality. The control model renders the following equations.

$$A_{dh} = \begin{bmatrix} 0 & I \\ 0 & -M_{dh}^{-1} D_{dh_L} \end{bmatrix} \in \mathbb{R}^{4x4}, \quad B_{dh} = \begin{bmatrix} 0 \\ M_d h^{-1} \end{bmatrix} \in \mathbb{R}^{4x2}, \tag{5.27}$$

$$E_{dh} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4x4}, \quad H_{dh} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{4x6} \tag{5.28}$$

Discretization of the system is explained in Section 5.1.

5.2.2 Optimal Control

The same principle as in Sandøy (2015) is used, where the controller is defined by:

$$\tau = Pr - K_c \begin{bmatrix} \hat{\eta} \\ \hat{\nu} \end{bmatrix} + \hat{b} \quad \in \mathbb{R}^{2x1}. \tag{5.29}$$

Where $K_c \in \mathbb{R}^{2x4}$ renders a Hurwitz closed loop system and $P = [-H_c(A - BK_c)^{-1}B]^{-1}$. K_c is found by minimizing the cost function

$$J = \int_0^{\infty} (\begin{bmatrix} \hat{\eta} & \hat{\nu} \end{bmatrix} Q \begin{bmatrix} \hat{\eta} \\ \hat{\nu} \end{bmatrix} + \tau R \tau^T) dt \tag{5.30}$$

with the weighing matrices $Q \in \mathbb{R}^{4x4}$ and $R \in \mathbb{R}^{2x2}$. The cost function in Equation (5.30) is subjected to the control model. The bias $\hat{b}$ is feed forwarded from the Kalman filter since this is a constant bias also in the control model. This is a method to avoid constant biases instead of augmenting an integral state in the control model.

5.2.3 Tuning of LQR Controller

To tune the LQR controller the weighing matrices $Q \in \mathbb{R}^{4x4}$ and $R \in \mathbb{R}^{2x2}$ are changed. This is done with Byrson's Rule of Thumb which suggests that before the tuning the entries of both matrices R and Q should be composed by the inverse of the squared maximum of each state.

5.3 Simulation

To verify control design simulation studies are done in Simulink. The observer and process model developed in previous sections is used. The simulation studies demonstrate the designed observer's ability to filter measurement noise, estimating velocities and its fusion of the pressure and Qualisys measurement.

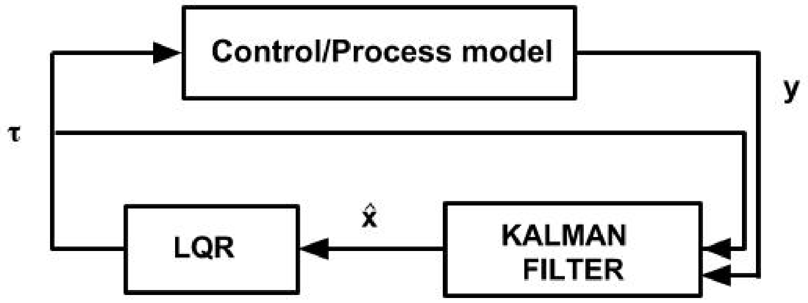


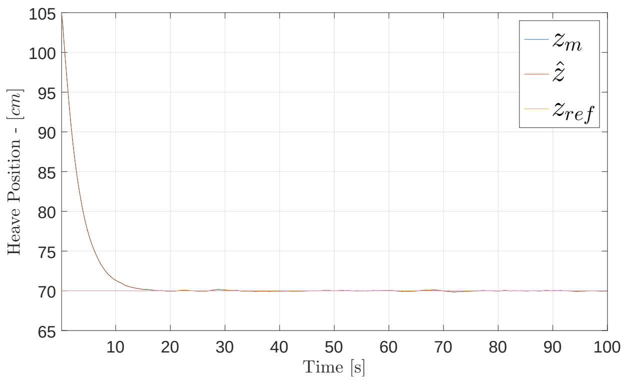
Figure 5.1: Block diagram describing signal flow. The measurement from the model are illustrated in Figure 3.1. The measurement then is sent into the Kalman filter, which also receives control input updates. The Kalman filter filters and fuses redundant measurements getting better prediction. In addition to estimates new states from the measurement. LQR uses the state estimated in the controller and decides an optimized input signal in relation with the model and its weighing matrices.

The discrete Kalman filter is implemented with an LQR for demonstration purposes are seen in Figure 5.1. Note that a Kalman filter in combination with an LQR controller is called LQG control. The same setup is used in Simulink for simulation of both process and control model. When testing the implementation in uDrone, the process model is removed and message blocks to interface with ROS are added. The message blocks are ready to use in the Robotics toolbox in Simulink. All implementation in both simulation and real system is given in Appendix A.7.

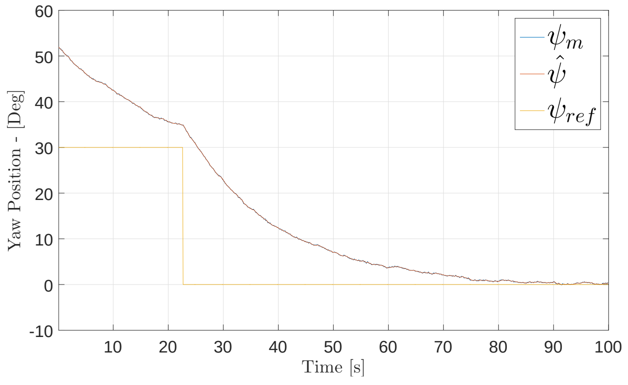
5.3.1 Simulation Case Study: Observer and Process Model

The simulation cases presented here include one simulation using the observer model and two simulations using the observer model. In Figure 5.2 a step response is tested with an initial condition of 105 cm in depht and 52 degrees in heading, as seen in Figures 5.2a and 5.2b, respectively. Reference r of the LQR controller, is set to 70 cm and 30 degrees the first 22 seconds for heave and heading. At 22 seconds the yaw reference is set to 0 degrees. The heave reference stays the same. The same simulation is ran for the process model in Figure 5.3. In Figure 5.4 the dropout functionality is tested along with the sensor fusion discussed in the previous sections. The drop out occurs at 20 seconds, before the measurement resumes at 40 seconds, where the heave uses the pressure sensor measurement in heave and goes into dead reckoning for yaw. Notation in the graph is lowercase m, ref and sim for measurement, reference in controller and value from simulation model, respectively. All simulations have the same parameters for both Kalman Filters Covariance matrices and the weighing matrices in the LQR. All simulations are shown in Appendix A. The time constant $T_{dh} \in \mathbb{R}^{2\times 2}$ in the bias model is set to be 100 times slower than the system mass matrix $M_{dh} \in \mathbb{R}^{2\times 2}$. The Covariance $R \in \mathbb{R}^{3\times 3}$ is the same as in Equation 5.26 and $Q \in \mathbb{R}^{6\times 6}$ is found from the Van Loan method and is given by:

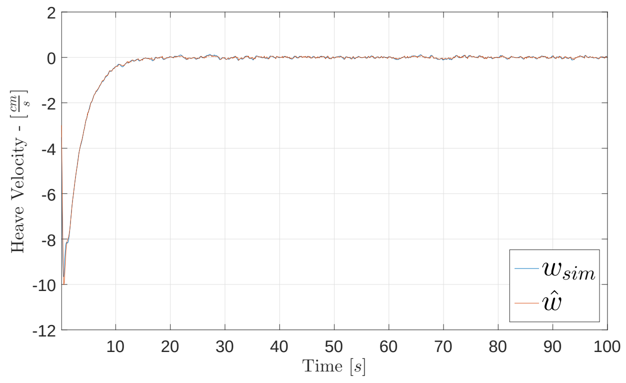
$$Q = \begin{bmatrix} 9.9e-09 & 0 & 1.5e-07 & 0 & 1.8e-06 & 0 \\ 0 & 7.3e-09 & 0 & 1.8e-07 & 0 & 6.3e-07 \\ 1.5e-07 & 0 & 3.1e-06 & 0 & 5.4e-05 & 0 \\ 0 & 1.8e-07 & 0 & 4.8e-06 & 0 & 1.9e-05 \\ 1.8e-06 & 0 & 5.4e-05 & 0 & 0.02 & 0 \\ 0 & 6.3e-07 & 0 & 1.9e-05 & 0 & 1.0e-04 \end{bmatrix} \tag{5.31}$$



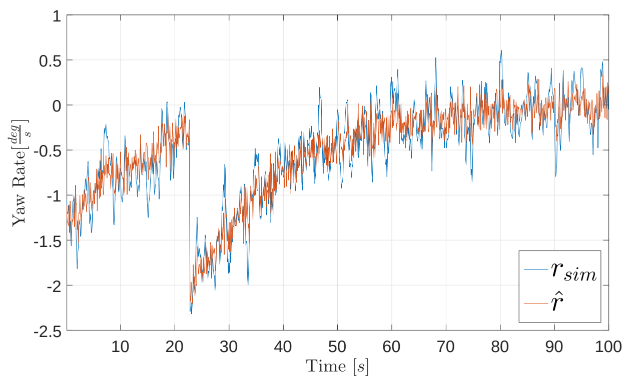
(a) Estimates for heave.



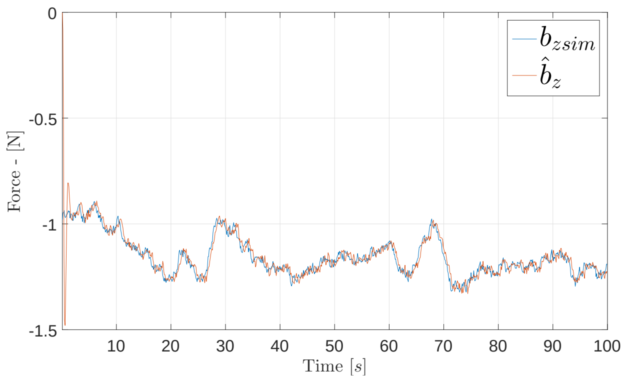
(b) Estimates for yaw.



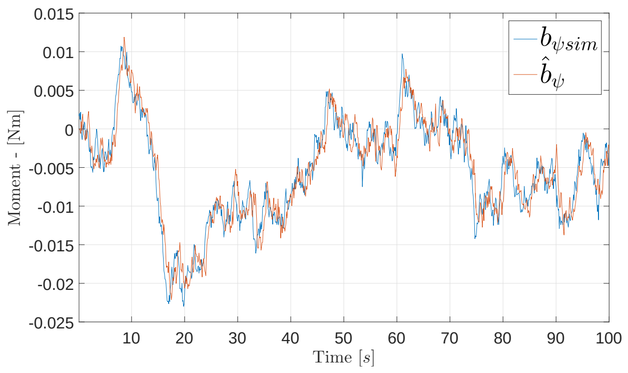
(c) Estimates for heave velocity.



(d) Estimates for yaw velocity.

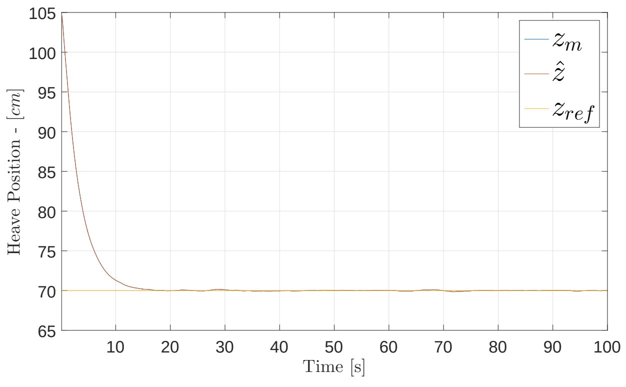


(e) Estimates for biases in heave.

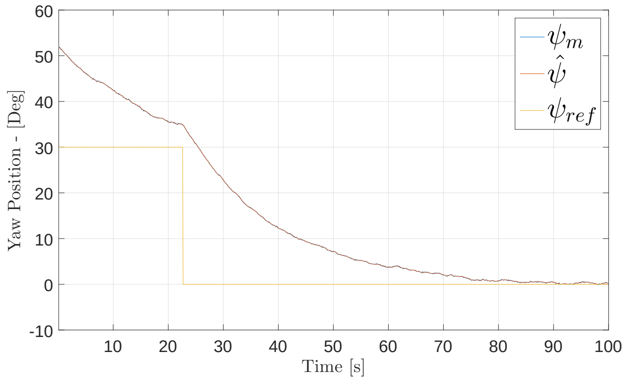


(f) Estimates for biases in yaw.

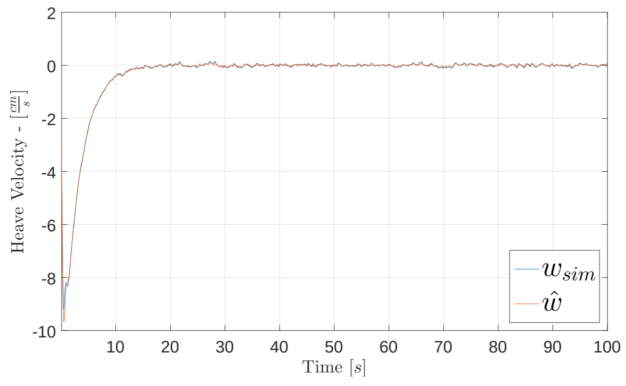
Figure 5.2: Step response for control model simulation. The kalman filter performance is very good in the perfect model. The biases measured are also clearly only due to the process noise.



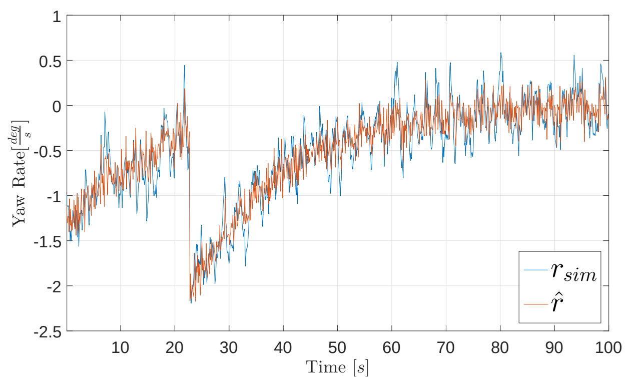
(a) Estimates for heave.



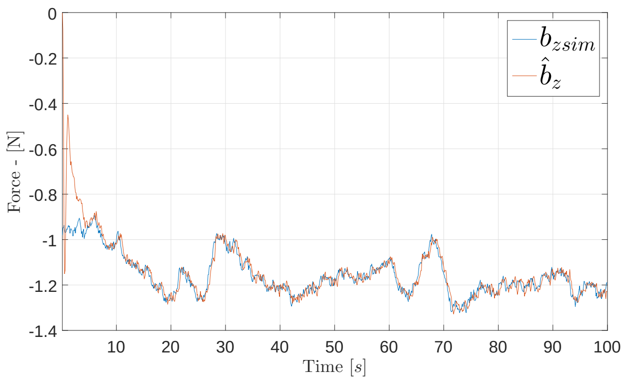
(b) Estimates for yaw.



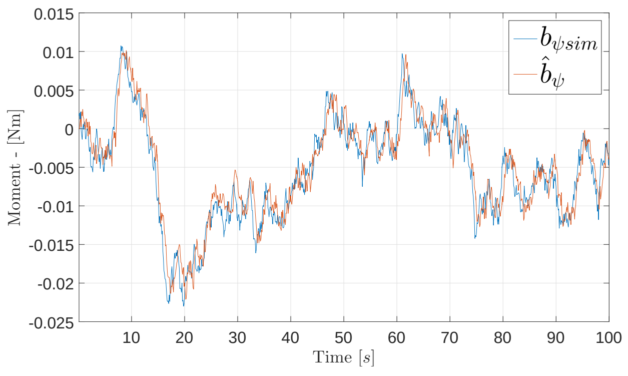
(c) Estimates for heave velocity.



(d) Estimates for yaw velocity.

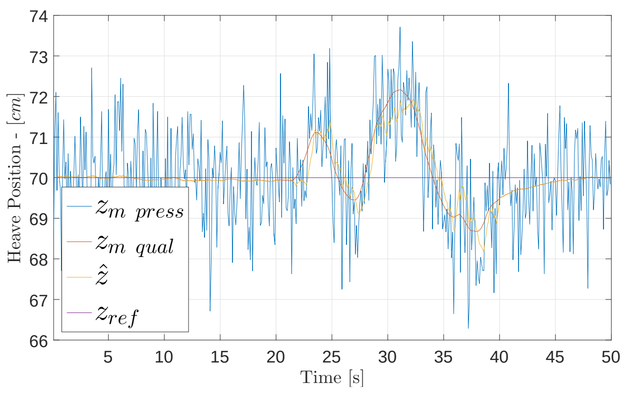


(e) Estimates for biases in heave.

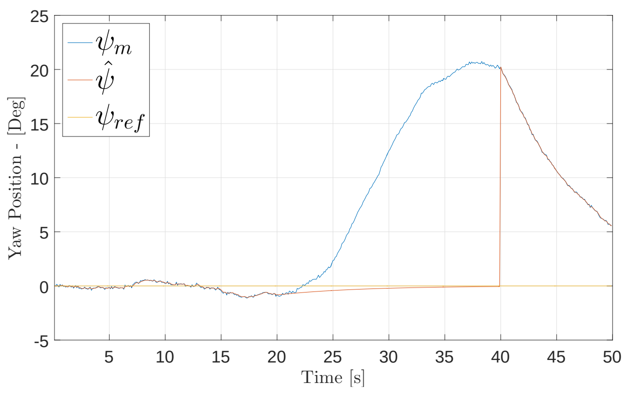


(f) Estimates for biases in yaw.

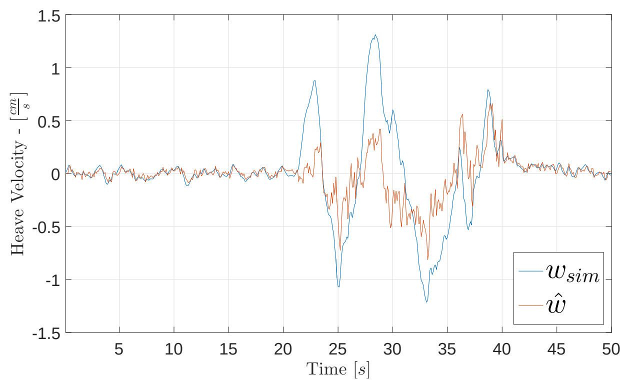
Figure 5.3: Step response in Process model simulation with Kalman Filter.



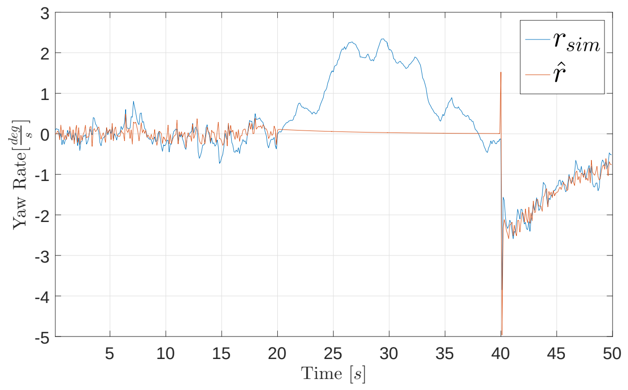
(a) Estimates for heave.



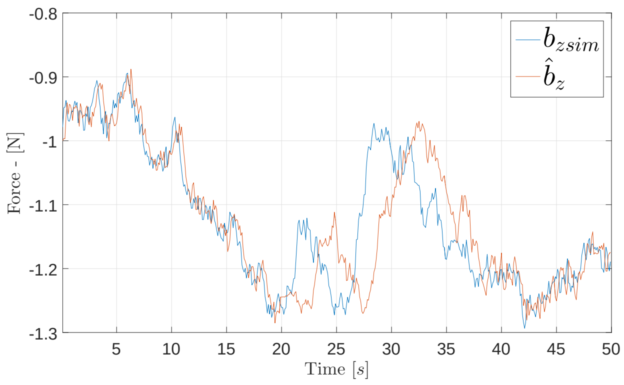
(b) Estimates for yaw.



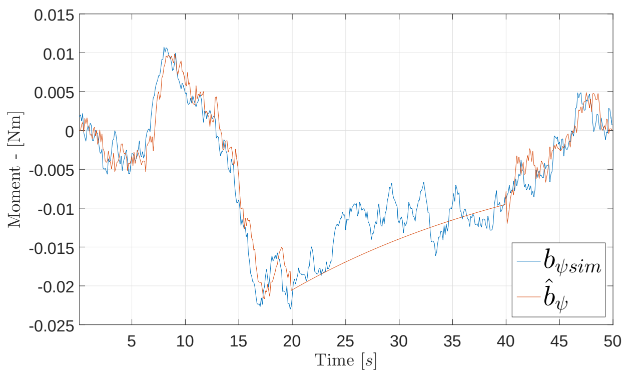
(c) Estimates for heave velocity.



(d) Estimates for yaw velocity.

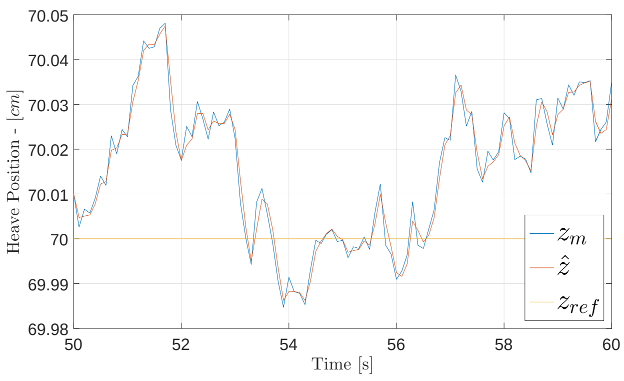


(e) Estimates for biases in heave.

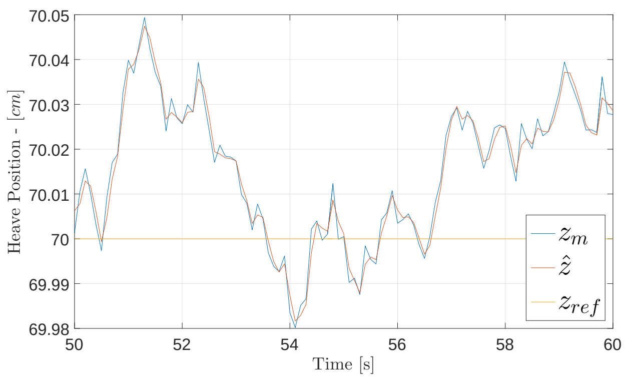


(f) Estimates for biases in yaw.

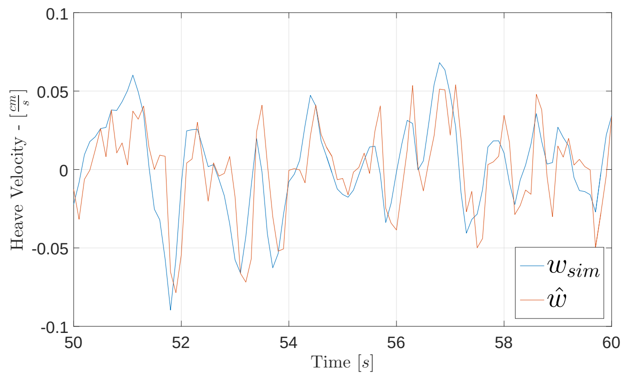
Figure 5.4: Drop out in Qualisys signal which means dead reckoning in yaw and switching to pressure sensor data only, for heave measurements. This is a simulation with process model, Kalman filter and the LQR. The drop-out starts at $t = 20$ sec and ends at $t = 40$ sec.



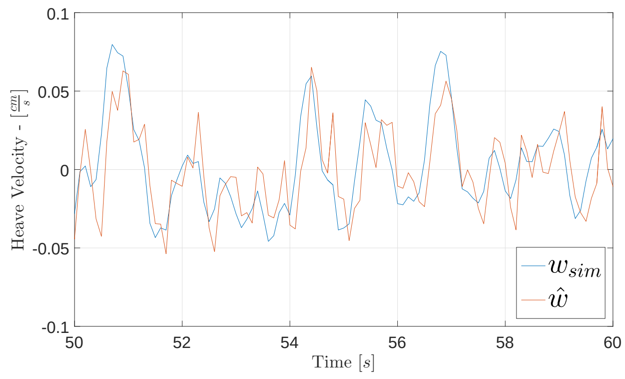
(a) Estimates for heave in observe model.



(b) Estimates for heave with process model.



(c) Estimates for heave velocity with observer model.



(d) Estimates for heave velocity with process model.

Figure 5.5: Close up filtering properties at 50 to 60 seconds in the process model step simulation. Sample time from the Figures 5.3a and 5.2a. The same is done for the velocity estimate in the Figures 5.3c and 5.2c.

5.3.2 Discussion of Simulation Results

In the Figures 5.2 and 5.3 the same step response experiment is executed with the observer and process model, respectively. The same process and sensor noise were applied using Monte Carlo simulation. Naturally, the difference is the nonlinear damping in the process model, which can be noticed in the bias estimation in the Figure 5.2e and 5.2f in comparison to 5.3e and 5.3f, respectively. This represents the deviations from the biases in the simulations. The biases could also be modeled with a higher time constant $T_{dh} \in \mathbb{R}^{2\times2}$, to have a less varying bias force. Modeling a constant is more the purpose of the bias rather than modeling the rapid changes.

The filtering properties of the control and process model are very similar, and are illustrated for the process model in Figure 5.5. Almost no lag can be seen in heave estimates in both process and observer control in the Figures 5.5b and 5.5a.

The velocities in heave for the observer and process model are shown in Figures 5.2c and 5.3c, respectively. They both have an overshoot in the velocity in the first seconds. The reason for this is the overshoot of the biases. They have an initial value set to zero and they therefore need the first seconds to properly converge. It should also be noted that the velocity of the observer model is bigger than in the process model due to less damping. After convergence of the velocity estimation, a close up is shown in Figure 5.5a and 5.5d from 50 to 60 seconds for the observer and process model, respectively. The filtering has the same behavior for the estimation in both models.

It should be mentioned that the time response from the controller is slow. This is done to assure that the Kalman filter dynamics are faster than the closed loop dynamics of the system and controller, which is of great importance to stability of the system.

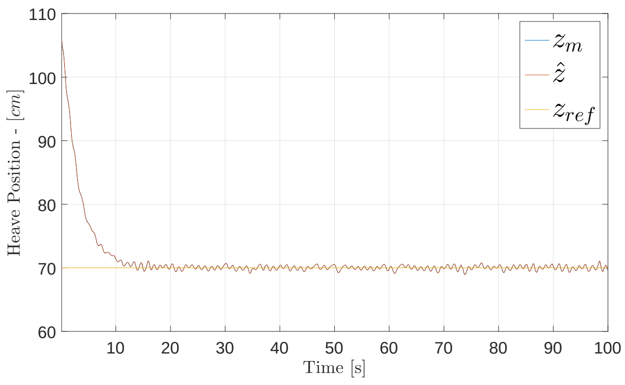
A simulation of a drop-out of the Qualisys signal with the process model is shown in Figure 5.4. The simulations have the same parameters as used in the step response simulation in Figure 5.3. Here also the sensor fusion algorithm is implemented which means that in Figure 5.4a the pressure measurement, denoted as $z_{m\ press}$, and Qualisys are merged into the estimate $\hat{z}$ and also used in the updates of velocity and bias of heave. However, this is not noticeable due to the big difference of variance between the pressure and Qualisys measurement. This simulates a usage of only $z_{m\ press}$ in heave, and dead reckoning in yaw. When going into the drop-out at 20 seconds, the heave position becomes more oscillatory and gets more error as can be seen in Figure 5.4a. This happens because the less accurate measurement lead to more lag in the bias estimates as seen in Figure 5.4e, which effects the performance of the controller directly since the bias is a feed-forward term. The estimate of velocity in Figure 5.4c becomes more filtered due to less "trust" in the measurement and lag in the bias. The dead reckoning in yaw is shown in Figures 5.4b, 5.4d and 5.4f. The bias estimate is smaller than in reality and the control input is therefore too large. This leads to positive deviations in both heading and angular velocities. An important curiosity is the transient that occurs when the Qualisys measurement returns at 40 seconds. It gives a major jump in the velocity, that is first positive and then negative. This can be unfortunate for a controller in real life, therefor a transition algorithm can here benefit the system. This can, for instance, be a transitional $R \in \mathbb{R}^{3\times3}$ instead of returning directly to the low variance of the Qualisys measurement.

5.4 Experiments

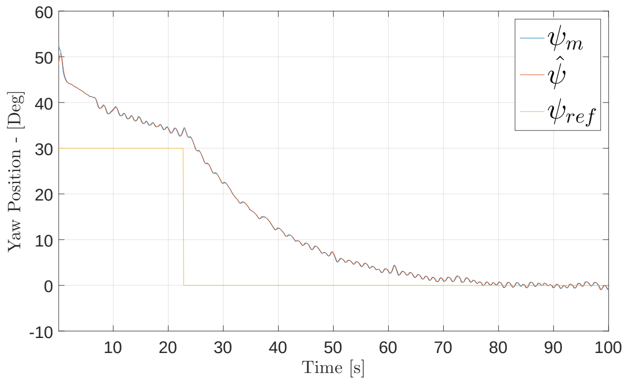
This section is intended to verify the simulations in the previous section. Experimental setups were ran in the MC-lab with the ROV uDrone. The implementation is done in Simulink and introduced in Appendix A.7. Tunable variables are here possible to change in the constant blocks during runtime. The initialization of the system parameters is carried out by the same scripts as used in the simulations, which are located in Appendix A.3 and A.4. The thrust allocation implemented used is presented in Appendix A.6. All parameters are possible to monitor live using the already implemented scope blocks in Simulink.

5.4.1 Experimental Results

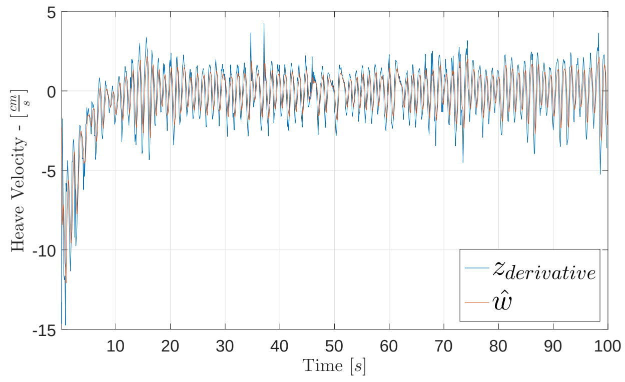
Results from the experiments are presented in Figures 5.8, 5.6 and 5.7. The two former figures, have the same setup as in the simulations in Figures 5.2 and 5.3. This way it is easier to compare the difference between simulation and experiment. The latter, is a drop-out test. There is no Qualisys at the first 45 seconds. This means that only the pressure sensor is used in the drop-out phase, then, after the 45 seconds the Qualisys sensor is resumed and considered more reliable. It should be noted that measurements of the Qualisys are not good in comparison to the pressure sensor. It could be removed by adjusting the global reference of Qualisys.
It should be noted that the notation used in the labels is the same as in the previous simulation section. However, here the simulated "perfect" velocity or angular velocity are not available, therefore the measured value and its derivative need instead to be the reference. Dirty derivatives can give quite noisy values, but due to the quality of the Qualisys measurement a good estimate is acquired. Naturally, the real bias not available in the experiment.



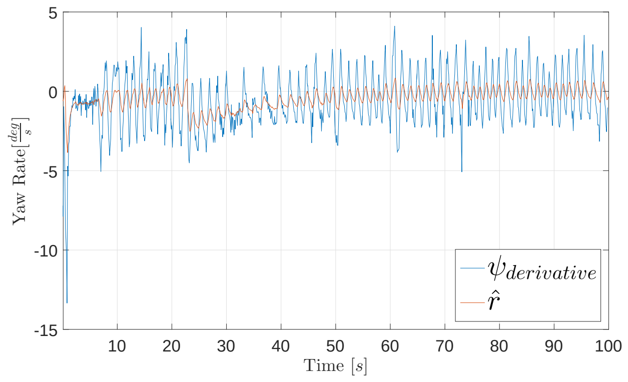
(a) Kalman filter estimates for heave with Qualisys measurements.



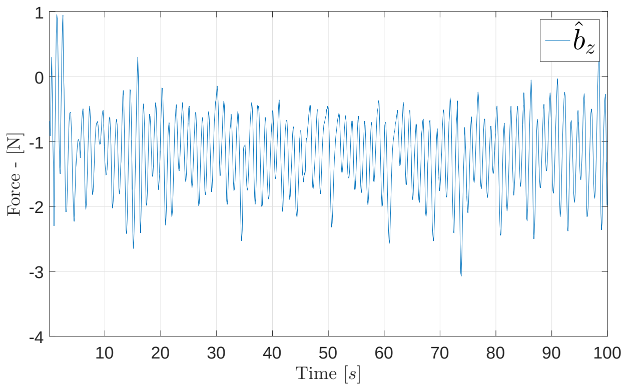
(b) Kalman filter estimates for yaw.



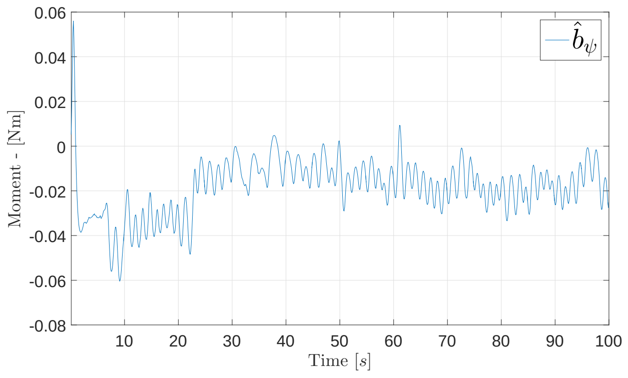
(c) Kalmanfilter estimates for heave velocity.



(d) Kalman filter estimates for yaw velocity.

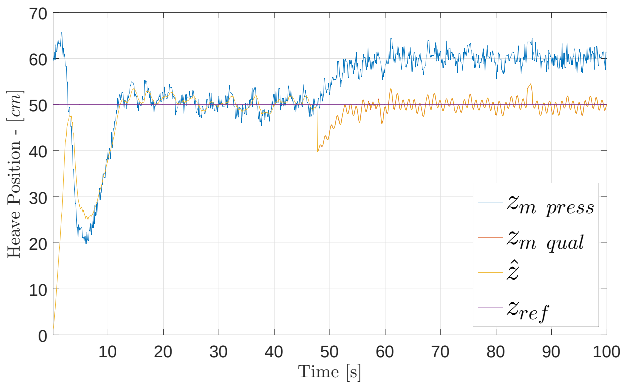


(e) Kalman filter estimates for heave velocity with Qualisys measurements.

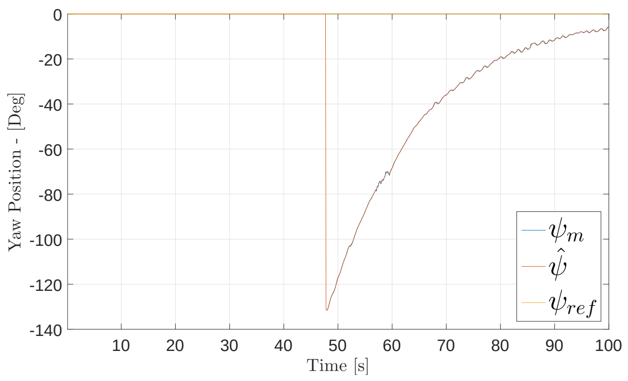


(f) Kalman filter estimates for yaw.

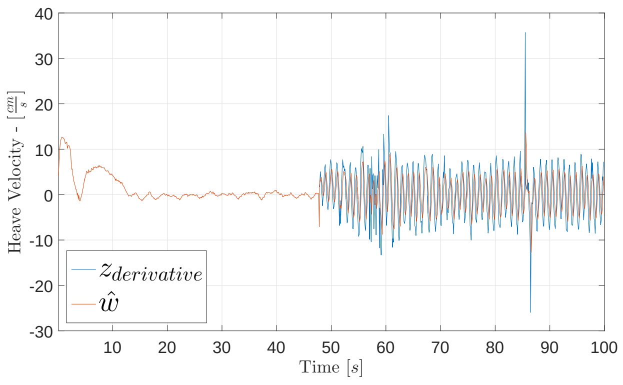
Figure 5.6: Experiment for step response.



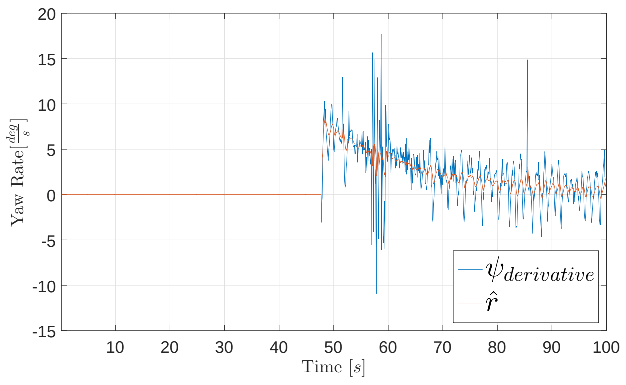
(a) Kalman filter estimates for heave with Qualisys measurements.



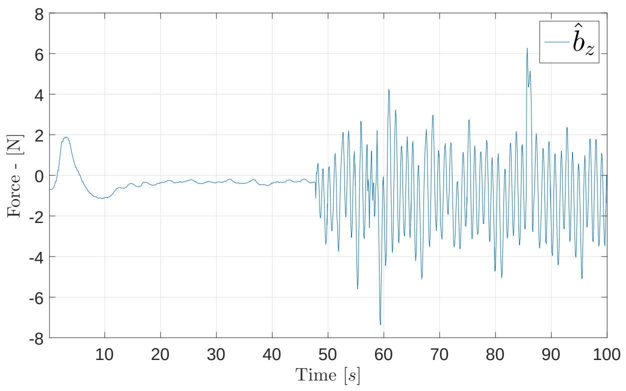
(b) Kalman filter estimates for yaw.



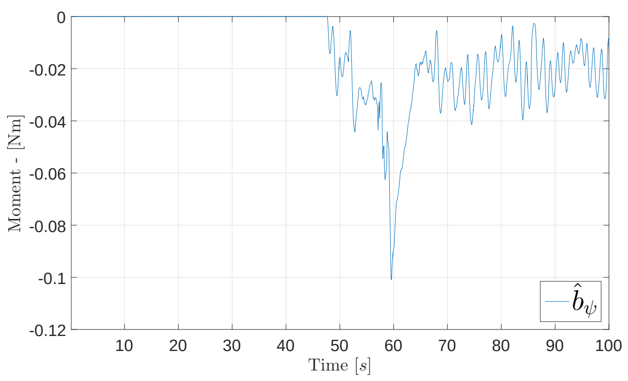
(c) Kalman filter estimates for heave velocity.



(d) Kalman filter estimates for yaw velocity.



(e) Kalman filter estimates for heave velocity with Qualisys measurements.



(f) Kalman filter estimates for yaw.

Figure 5.7: Experiment for step response.

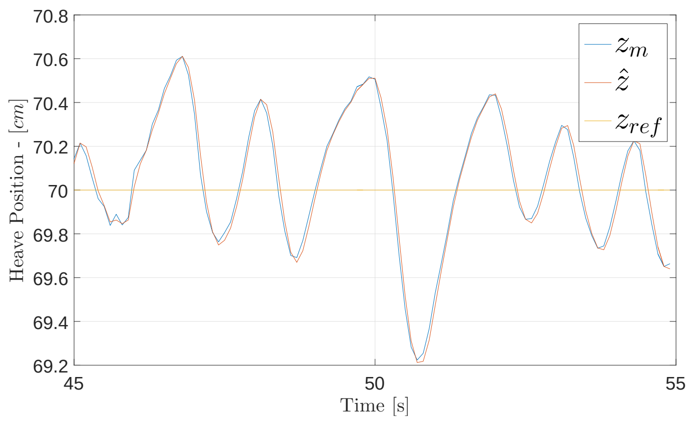


Figure 5.8: Filtering properties of heave position from Figure 5.6a.

5.4.2 Discussion of Experimental Results

As mentioned in Figure 5.6, the same step response setup is used as in the simulations and is compared with the response from the process model, in Figure 5.3. Figure 5.8 is a close up of Figure 5.6 to analyze the filtering properties. The noise is very small on this scale (the scale of the y-axis is large in relatively to the noise in Figure 5.5), but the lag is minor and there is some filtering early in the sample.

It is noticeable that the heave velocity is more varying in the experiment seen in Figure 5.6c than in the simulation in Figure 5.3c. This indicates that the noise power of vector $W \in \mathbb{R}^{6\times 6}$ should be higher for the heave velocity noise w_w. This may give a more filtered velocity like in the yaw velocity in Figure 5.6d. The angular speed in yaw has a more similar behavior with the simulation in Figure 5.3d, and the velocity noise varies around $\pm 2.5\frac{deg}{sec}$. It should be noticed that the time response of convergence in heave and yaw is strikingly similar with the simulation. This could be an indication on good approximations of the parameters found in Chapter 4. Also, the thrust allocation should give forces in the desired area. But there are also as discussed in the thrust allocation chapter some oscillation in pitch, which effects the thrust in pure heave motion considerably due to the tilting of the thrusters. There are also strongly varying biases in heave which can be due to the thrust allocation, or maybe the tether. The bias time constant could prohibit this rapid variation along with rise of the process noise. This could result in a less aggressive feedback to the controller and filter out more of the additional alternating velocity noise. However, for higher performance, a better thrust allocation could be an option.

The drop-out experiment is given in Figure 5.7. It is a demonstration of the dead reckoning algorithm in yaw and the pressure sensor usage in Qualisys drop-out together with the resuming signal. In Figure 5.7a, the first 10 seconds of convergence are due to initialization of the heave position at zero. Further, both the heave velocity and biases in 5.7c and 5.7e stabilize nicely. There are no estimates of a reference velocity in this period due to the lack of Qualisys signal. When the signal returns at roughly 47 seconds, the estimates trust more in the Qualisys measurement and the response is the same as discussed in the step response from Figure 5.6. In Figures 5.7b, 5.7d and 5.7f show dead reckoning mode in yaw. It stays at the initial state at zero heading, angular velocity and bias. This happens until the Qualisys signal is restored.

Chapter 6

Conclusion and Further Work

6.1 Conclusion

The main goal of this thesis was to implement a complete model-based state estimation algorithm for uDrone. This was accomplished by first developing a thrust allocation model and finding thruster characteristics by towing tests. A model of high fidelity called process model, was developed. The parameters of the model were obtained using various methods. The added mass terms were found by using a simple estimation method developed in Eidsvik (2015). The added mass in surge was tested by applying MATLAB's system identification toolbox. The damping of the model was found by doing towing tests. When the parameters were obtained, a simplified model was developed from the process model, also called observer model. The famous Kalman filter was developed from the observer model along with a sensor-fusion and dead reckoning algorithm. The estimator design was implemented and verified both in simulations and on the uDrone. The implementation was performed directly into Simulink, which was interfaced with ROS. The process was live tuned and monitored simply by using constant blocks for input and scopes as monitors.

The method for finding thrust and damping parameters turned out to have a rig that was not rigid enough to withstand high forces, however, this was acceptable for a low-velocity design. The developed theory for the Kalman filter gave great results in the simulations. The process noises and sensor noises were modeled perfectly to fit the Kalman filter. An LQR controller was also developed using a control model. Along with the Kalman filter, the control loop is known

as an LQG control which is optimal assuming Gaussian noise and a model that is accurate.
The experimental tests show that there are still some improvements required. Especially with the thrust allocation this also became clear during the system identification trials, where clean motions turned out to be a challenge. The pitch motion due to three heave thrusters was especially a problem. Feedback of the system is needed for better performance, and open loop allocation is not satisfying when it comes to accurate control at this level.

6.2 Further Work

In order to complete the work here developed, some future work can be made to more accurately investigate some of the questions that were mentioned along the text in all the chapters. Firstly, the thrust allocation algorithm can be optimized for better performance. This can be done by improving the already implemented system or finding a way to implement one with feedback. The latter should be preferable. Secondly, the implemented model could also be quite simply extended to a system for full dynamic positioning control since all parameters are already known. Some of this work was executed during this thesis, by the author, but due to lack of time it was not documented. The curious reader can look for the implementation in Appendix A.9. Finally, implementation of new state estimator and control algorithms development can enhance the uDrone in the future.

Bibliography

(2004). Beyond the kalman filter; particle filters for tracking applications.

Aras, M. S. M., Azis, F. A., Teck, L. W., Abdullah, S. S., and Rahman, A. F. N. A. System identification of a prototype small scale rov for depth control. In *Control Conference (ASCC), 2015 10th Asian*, pages 1–6.

BlueRobotics (2015a). Bluerov.

BlueRobotics (2015b). Thrust specifiaction for t200 blue robotics thrusters.

BlueRobotics (2016). T200.

Bosch (2014). Bno055 - intelligent 9-axis absolute orientation sensor - data sheet.

Brown, R. G. and Hwang, P. Y. C. (2012). *Introduction to Random Signals and Applied Kalman Filtering with Matlab Exercises, 4th Edition.* Introduction to random signals and applied Kalman filtering. John Wiley and Sons.

Chen, C.-T. (2013). *Linear system theory and design.* The Oxford series in electrical and computer engineering. Oxford University Press, New York, fourth edition. edition. Ch. 6.2 Controllability.

Christ, R. D. and Wernli, S. R. L. (2011). *The ROV Manual : A User Guide for Observation Class Remotely Operated Vehicles.* Elsevier Science, Burlington. Front cover; The ROV Manual: A User Guide for Observation-Class Remotely Operated Vehicles; Copyright page; Contents; Foreword; Preface; Acknowledgments; History and dedication; Introduction; Chapter 1 A bit of history; 1.1 Introduction; 1.2 What is an ROV?; 1.3 In the beginning; 1.4 Today's observation-class vehicles; Chapter 2 ROV design; 2.1 Underwater vehicles to ROVs; 2.2 Autonomy plus:

’why the tether?’; 2.3 The ROV; Chapter 3 ROV components; 3.1 Mechanical and electro/mechanical systems; 3.2 Primary subsystems; 3.3 Electrical considerations; 3.4 Control systems.

DNV-RP-H103 (2010). Modelling and analysis of marine operations.

Dukan, F., Ludvigsen, M., and Sorensen, A. J. (2011). Dynamic positioning system for a small size rov with experimental results. In *OCEANS, 2011 IEEE - Spain*, pages 1–10.

Eidsvik, O. (2015). *Identification of hydrodynamical parameters of ROV.* Thesis.

Fossen, T. I. (1987). *Nonlinear Modelling and Control of Underwater Vehicles.* Thesis.

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

Hegrenas, O., Berglund, E., and Hallingstad, O. Model-aided inertial navigation for underwater vehicles. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1069–1076.

Kalman, R. E. (1960). A new approch to linear filtering and prediction problems. *Journal of Basic Engineering JBE-83*, page 7.

Kjerstad, Øyvind Kåre ; Skjetne, R. (2016). Disturbance rejection by acceleration feedforward for marine surface vessels.

Ljung, L. (1999). *System identification : theory for the user.* Prentice Hall information and system sciences series. Prentice Hall PTR, Upper Saddle River, N.J, 2nd ed. edition.

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

Magill, D. (1965). Optimal adaptive estimation of sampled stochastic processes. *IEEE Transactions on Automatic Control*, 10(4):434–439.

Mahony, R., Hamel, T., and Pflimlin, J. M. (2008). Nonlinear complementary filters on the special orthogonal group. *IEEE Transactions on Automatic Control*, 53(5):1203–1218.

MS5837 (2015). Ms5837-30ba - ultra small gel filled pressure sensor.

Sandøy, S; Henriksen, A. (2015). *Hardware and Software Design of uDrone.* Thesis.

Sandøy, S. S. (2015). An implementation of lqr and luenberger observer for autodepth control.

SNAME (1950). *Code on maneuvering and special trials and tests 1950.* SNAME, New York.

SOLIDWOKRS (2016). Solid works.

Sorensen, A. J. (2013). *Propulsion and Motion Control of Ships and Ocean Structures.*

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

Appendix A

Matlab and Simulink Implementations

A.1 Implementaion of Control Model with LQG control

The implementation of the control model can be found in the folder:
"../Appendix/Appendix_A/1_ControlModel/kalman_filter"
in the attachments. The other folders in the same path are the simulation results obtained form the simulations.

A.2 Implementation Process Model with LQG Control

The implementation of the control model can be found in the folder:
"../Appendix/Appendix_A/2_ProcessModel/kalman_filter"
in the attachments. The other folders in the same path are the simulation results obtained form the simulations.

A.3 Init Kalman Filter

```
%% Time step
h = 0.1;

%% Finding M = M_A + M_RB
```

```
m     = 7.31;
I_zz  = 0.16;

Z_dw  = -11.13;
N_dr  = -0.1;

M_A    =  -diag([Z_dw N_dr]);
M_RB   =   diag([m I_zz]);
M = M_RB + M_A;
M_inv = inv(M);

%% Experimental drag coeff
Z_w = -5.18;
N_r = -0.06656;

D_L   = - diag([Z_w N_r]);

%%Identity matrix
Identity    = eye(2,2);

%% Bias term
T_b = M*100;
T_b_inv = inv(T_b);

%% A matrix
A = [zeros(2,2), Identity, zeros(2,2);
         zeros(2,2), -M_inv*D_L, M_inv;
         zeros(2,2), zeros(2,2), -T_b_inv];

C = eye(2,6);

%% B matrix
```

```
B = [zeros(2,2);
      M_inv;
      zeros(2,2)];

%% E matrix
E =[zeros(2,2), zeros(2,2), zeros(2,2);
    zeros(2,2),  M_inv,     zeros(2,2);
    zeros(2,2), zeros(2,2), Identity];

[Phi, Delta] = c2d(A,B,h); %exact discretization
[Phi, Gamma] = c2d(A,E,h); %exact discretization

%% R matrix Measurment Noise
R = diag([1.9878e-09 1.616e-6]);
R_monte_carlo = diag([1.9878e-09 1.189622e-04 1.616e-6]);

%% Init values
P_apriori_init= diag([0, 0, 0, 0, 0, 0]);
xhat_apriori_init = [1.05 52*pi/180 0 0 0 0]';

%% Calculations for monte carlo simulation
% W = amplitude of white noise
W  = diag([0 0 1e-2 1e-8 2e-1 1e-3]);

A2 = [-A*h, E*W*E'*h;
       zeros(6,6), A'*h];

B2 = expm(A2);

phi = B2(7:12,7:12)';
Q   = phi*B2(1:6,7:12);

```

```
CQ = chol(Q)';
CR = chol(R_monte_carlo)';
```

A.4 Init LQR

```
%% Time step
Ts = 0.1;

%% Finding M = M_A + M_RB
m     = 7.31;
I_zz  = 0.16;

Z_dw  = -11.13;
N_dr  = -0.1;

M_A    =  -diag([Z_dw N_dr]);
M_RB   =   diag([m I_zz]);
M = M_RB + M_A;
M_inv = inv(M);

%% Experimental drag coeff
Z_w = -5.18;
N_r = -0.06656;

D_L  = - diag([Z_w N_r]);

%%Identity matrix
Identity    = eye(2,2);

%% A matrix
A = [zeros(2,2), Identity;
```

```
      zeros(2,2), -M_inv*D_L];

%% B matrix
B = [zeros(2,2);
      M_inv];

C = eye(2,4);

Q_lqr = diag([1/1.5^2 1/(pi/6)^2 1/(0.5)^2 1/(pi/100)^2]);
R_lqr = diag([0.1*1/10^2 0.01*1/(pi/100)^2]);

sys_lqr = ss(A,B,C,0);

sys_d =c2d(sys_lqr,Ts);

[K_c,S,e] = dlqr(sys_d.a,sys_d.b,Q_lqr,R_lqr);

P = inv(-C*inv(A-B*K_c)*B);

init_kalman_filter
```

A.5 Added Mass Estimation

The computations of the added mass are found in the attachments on the path "../Appendix/Appendix_A/4_systemIdnetifikasjon/Added_mass_estimates_eidsviks_method". The script is taken from Eidsvik (2015).

A.6 Thrust Allocation

The simulink implementation of the thrust allocation is found in the attachments at the path "../Appendix/Appendix_A/5_thrust_allocation".

A.7 Experimental LQG Control Implementation

The implementation in Simulink and results can be viewed in the attachments in the folder "…/Appendix/Appendix_A/3_experimental_testing/Kalman_Filter".

A.8 Noise Measurements

The time series and calculations of variances during the noise sampling can be found in the attachments at "…/Appendix/Appendix_A/6_Noise_measurement".

A.9 Full Dynamic Positioning Simulations and Implementation

The implementation in Simulink and for use in ROS can be viewed in the attachments in the folders "…/Appendix/Appendix_A/3_experimental_testing/Extended_Kalman_Filter" and The implementation in Simulink and results can be viewed in the attachments in the folder "…/Appendix/Appendix_A/2_ProcessModel_testing/Extended_Kalman_Filter".

Appendix B

Hydrodynamical properties

B.1 Hydrodynamic Damping parameters

B.1.1 Thurst Characteristics Data

The experimental values and scripts for reading the results can be found in the folder: ".../Appendix/Appendix_B/thrust_characteristic_experiment_result".

B.1.2 Towing Test Data

The experimental values can be found in the folder: ".../Appendix/Appendix_B/towing_test_result".

B.2 Hydrodynamical Matrices

B.2.1 Inertia and Added Mass Forces

$$M = M_{RB} + M_A \tag{B.1}$$

$$M_{RB} = \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & m & 0 & mx_g \\ 0 & 0 & m & 0 \\ 0 & mx_g & 0 & I_{zz} \end{bmatrix} \tag{B.2}$$

$$M_A = -diag\left[X_{\dot{u}} \quad Y_{\dot{v}} \quad Z_{\dot{w}} \quad N_{\dot{r}}\right] \tag{B.3}$$

B.2.2 Damping Forces

$$D(\nu) = D_L + D_{NL}(\nu) \tag{B.4}$$

$$D_L = -diag\left[X_u \quad Y_v \quad Z_w \quad N_r\right] \tag{B.5}$$

$$D_{NL}(\nu) = -diag\begin{bmatrix} X_{|u|u}|u| + X_{uuu}uu & Y_{|v|v}|v| + Y_{vvv}vv \quad .... \\ Z_{|w|w}|w| + Z_{www}ww & N_{|r|r}|r| + N_{rrr}rr \end{bmatrix} \tag{B.6}$$

B.2.3 Corriolis Forces

$$C(\nu) = C_A + C_{RB} \tag{B.7}$$

$$C_A = \begin{bmatrix} 0 & 0 & 0 & Y_{\dot{v}}v \\ 0 & 0 & 0 & -X_{\dot{u}}u \\ -Y_{\dot{v}}v & X_{\dot{u}}u & 0 & 0 \end{bmatrix} \tag{B.8}$$

$$C_{RB} = \begin{bmatrix} 0 & 0 & 0 & -m(v + rx_g) \\ 0 & 0 & 0 & m(u - ry_g) \\ 0 & 0 & 0 & 0 \\ m(v + rx_g) & -m(u - ry_g) & 0 & 0 \end{bmatrix} \tag{B.9}$$

B.3 Thrust Mapping For uDrone

$$\tau = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & -1 & 0 & 0 & 0 \\ -l_{y_1} & -l_{y_2} & -l_{y_3} & 0 & 0 & -l_{z_6} \\ l_{x_1} & l_{x_2} & l_{x_3} & l_{z_4} & l_{z_5} & 0 \\ 0 & 0 & 0 & -l_{y_4} & -l_{y_5} & l_{x_6} \end{bmatrix} f_t \tag{B.10}$$

Note that the first columns have negative terms in compearison with Fossen (2011). This is due to the 90 degree rotation of the thrusters.

Appendix C

CAD and Drawings

All CAD models are found in the "Appendix_C/" folder. The parts found are the Thruster model, towing rig model with all configurations and a complete assembly of the uDrone. This is the model used to obtain the rigid body parameters! In Figure C.1 and C.2.

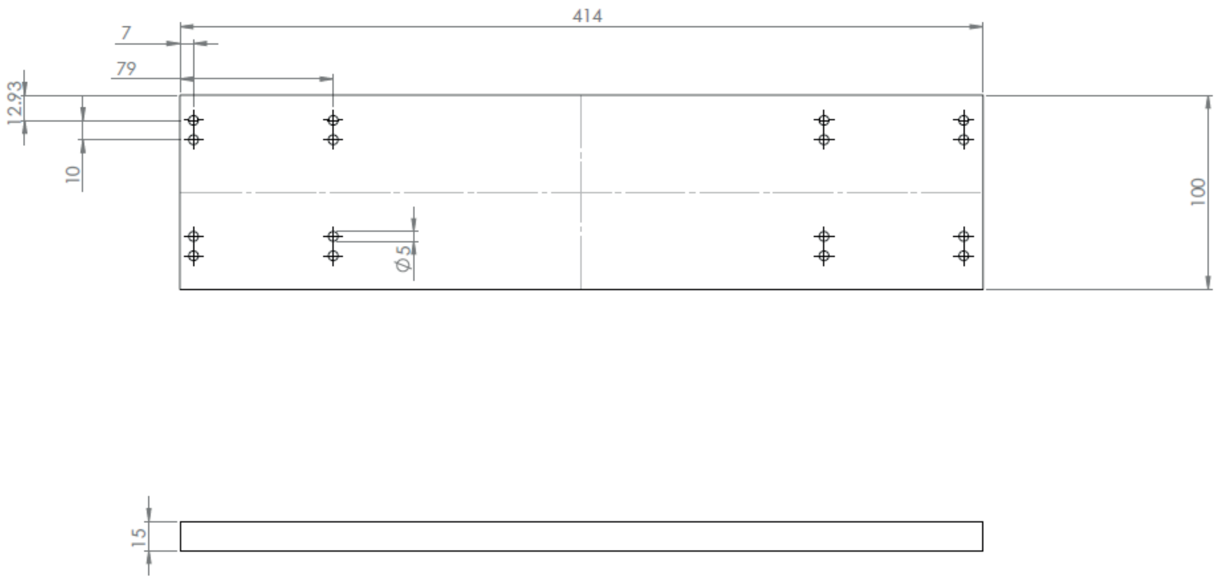


Figure C.1: Plate for more stiff rig setup. All holes where made with a tolerance of $+0.2mm$

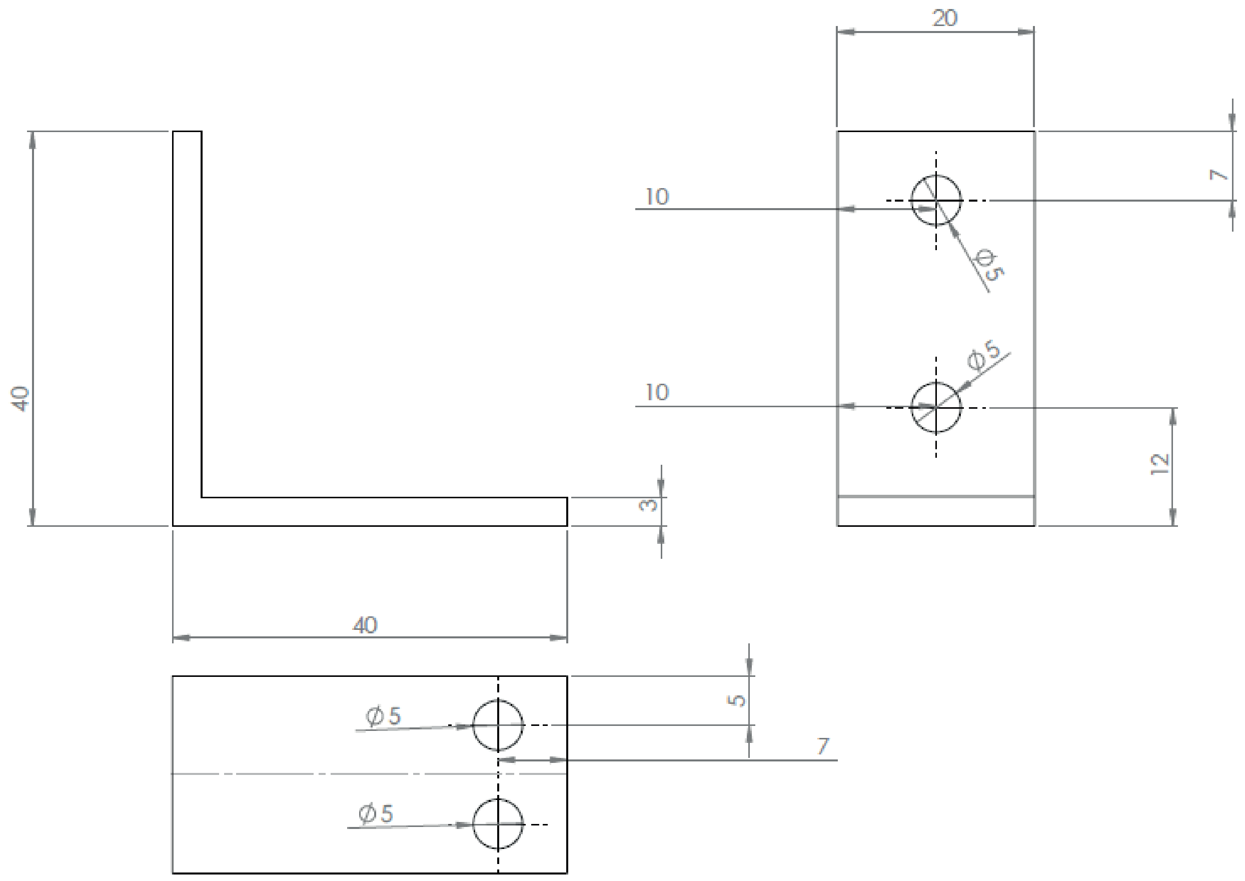


Figure C.2: ROV bracket for towing rig. All holes where made with a tolerance of $+0.2mm$.

Appendix D

Marin Cybernetics Lab

This Appendix contains information about MC-lab which was the laboratory used for all experiental setups. The information here is copied directly from : "http://www.ntnu.no/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.

D.1 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,

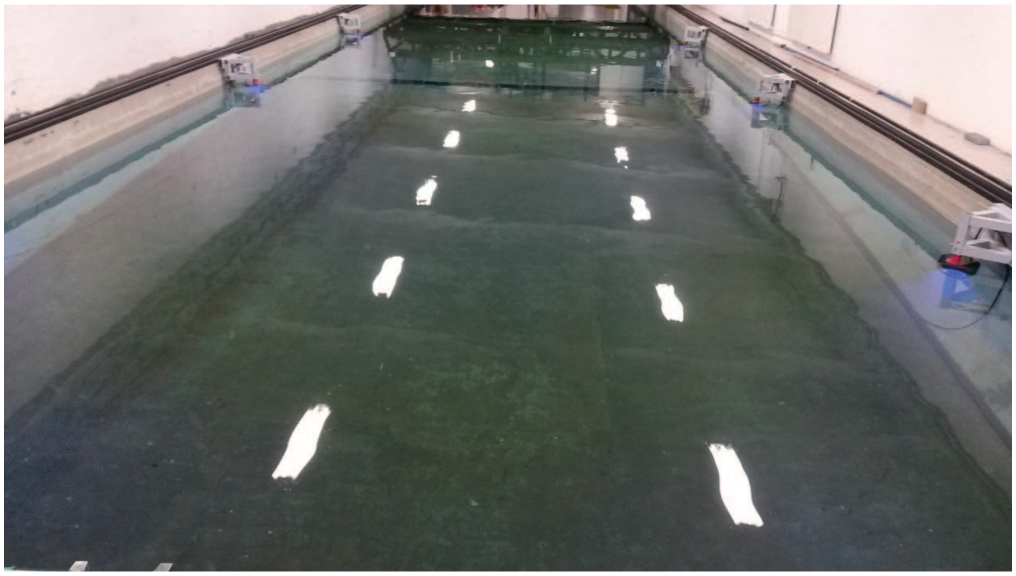


Figure D.1: MC lab's real time underwater motion system.

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. See Figure D.1

D.2 Towing Carriage

The carriage can be operated in manual or computer controlled mode. The manual operation is done from the consol on the carriage. There is made a special Labview and Opal application to setup regular or irregular movement of the different axes as seen in Figure D.2.

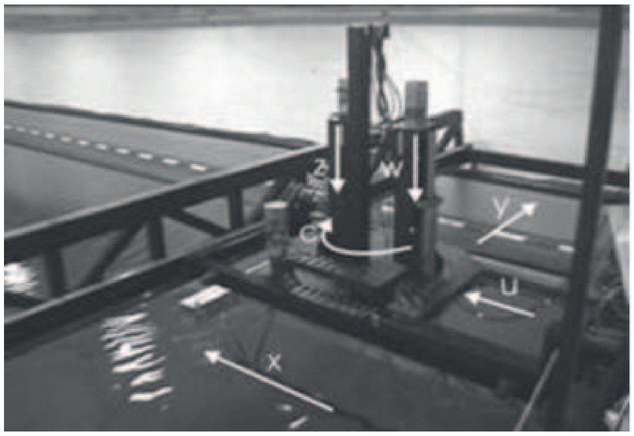


Figure D.2: Axes of movement of towing carriage, courtesy NTNU

Appendix E

External Libraries

E.1 ROS-Simulink interface

To interface Simulink with matlab is is the Robotics System Toolbox necessary. The setup and usage are documented at
http://se.mathworks.com/help/robotics/getting-started-with-robotics-system-toolbox.html
For running of real time simulation in ros is a real time pacer used. Its documation is found at
http://www.mathworks.com/matlabcentral/fileexchange/29107-real-time-pacer-for-simulink

E.2 Qualisys

To use the Qualisys along with ROS an already implemented library is used from KumarRobotics. The repository is found at https://github.com/KumarRobotics/qualisys.