

# **LOW SPEED MODELING AND SIMULATION OF GAVIA AUV**

January 2012

**Bjarni Helgason**

Master of Science in Mechanical Engineering





# LOW SPEED MODELING AND SIMULATION OF GAVIA AUV

**Bjarni Helgason**

Master of Science

Mechanical Engineering

January 2012

School of Science and Engineering

Reykjavík University

**M.Sc. RESEARCH THESIS**







# **Low Speed Modeling and Simulation of Gavia AUV**

by

Bjarni Helgason

Research thesis submitted to the School of Science and Engineering  
at Reykjavík University in partial fulfillment of  
the requirements for the degree of  
**Master of Science in Mechanical Engineering**

January 2012

Research Thesis Committee:

Leifur Þór Leifson, Supervisor  
Assistant Professor, School of Science and Engineering

Indriði Sævar Ríkharðsson, Supervisor  
Assistant Professor, School of Science and Engineering

Copyright  
Bjarni Helgason  
January 2012

The undersigned hereby certify that they recommend to the School of Science and Engineering at Reykjavík University for acceptance this research thesis entitled **Low Speed Modeling and Simulation of Gavia AUV** submitted by **Bjarni Helgason** in partial fulfillment of the requirements for the degree of **Master of Science in Mechanical Engineering**.

---

Date

---

Leifur Þór Leifson, Supervisor  
Assistant Professor, School of Science and Engineering

---

Indriði Sævar Ríkharðsson, Supervisor  
Assistant Professor, School of Science and Engineering

The undersigned hereby grants permission to the Reykjavík University Library to reproduce single copies of this research thesis entitled **Low Speed Modeling and Simulation of Gavia AUV** and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the research thesis, and except as herein before provided, neither the research thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatsoever without the author's prior written permission.

---

Date

---

Bjarni Helgason  
Master of Science

# Low Speed Modeling and Simulation of Gavia AUV

Bjarni Helgason

January 2012

## Abstract

Today, many torpedo shaped autonomous vehicles (AUVs) are unable to perform low speed control. For that reason, underwater vehicle dynamics and control at low speed is an important research topic. In this work, the equations of motion of a torpedo shaped autonomous underwater vehicle at low speed in 4 degrees of freedom (DOF) are derived. The mathematical model forms the basis of simulator developed in Simulink.

The simulator is structured as four non inter-acting subsystems for surge, sway, heave and yaw consequently. The inputs of the simulator are the physical properties of the AUV, where hydrodynamics coefficients except the linear drag, are calculated, based on properties of the AUV. A script was made which allows the user to run multiple tests with different settings, as well as using the optimization and sensitivity analysis tools that Matlab has to offer.

The simulator was validated by experiments using a commercially available AUV, the Gavia AUV. A complete external thrust unit with control was developed for that process. The Python programming language, supported by Robot Operation System (ROS) was used to program the control. Numbers of experiments, in both open and closed loop control, were performed in Laugardalslaug Swimming Pool and a small test pool, located at the Tele-dyne Gavia facility.

Comparison was made between the results of these experiments and the simulator to adjust the simulator. The adjusted parameters in the simulator, which were the added mass, the linear drag, and the non-linear drag, were tuned by multiplicative blocks. The results of tuning these parameters manually were compared to the real data until the difference between simulated and real data had been minimized. After a little tuning the sway, heave and yaw models fitted well with the experimental data but the surge model did not fit as well and needs more analysis.

# **Hermun og líkanagerð á kafbátnum Gaviu á litlum siglingarhraða**

Bjarni Helgason

Janúar 2012

## **Útdráttur**

Sjálfráðir tundurskeytalaga kafbátar eru til en almennt er ekki hægt að stjórna þeim á litlum siglingarhraða. Þess vegna er hreyfiaflfræði slíkra kafbáta áhugavert rannsóknarefni. Hér verða hreyfijöfnur tundurskeytalagaðs sjálfráðs kafbáts, sem ferðast eftir fjórum frelsisgráðum á litlum siglingarhraða, leiddar út. Stærðfræði líkan er notað sem grunnur að hermi sem gerður er í Simulink.

Hermirinn er gerður úr fjórum óháðum hlutum, þar sem hver hluti líkir eftir hreyfingu einnar frelsisgráðu. Samhliða var skrifaður kóði, sem gerir kleift að gera margar prófanir í einu með mismunandi stillingum sem og notkun bestunar- og næmnistóls í matlab við vinnslu úr niður- stöðum hermisins.

Fjöldi tilrauna var gerður til að staðfesta niðurstöður hermisins. Hannaður var utanáliggjandi framdriftarbúnaður, sem er stýrt með hefðbundnum PID regli í þessum tilgangi. Stýringin var forrituð í Python og keyrð gegnum Robotic Operating System (ROS). Tilraunir voru gerðar í Laugardalslaug og lítilli laug staðsettri í Teledyne Gavia.

Bornar voru saman niðurstöður tilrauna við niðurstöður hermisins. Hermirinn var stilltur til að samræma niðurstöðurnar. Breyturnar sem stilltar voru í herminum voru viðbættur massi, línulegur núningskraftur og ólínulegur núningskraftur. Niðurstöður hermisins, sem fengnar voru með því að laga til stilltar breytur voru bornar saman við niðurstöður úr prófunum, þar til hegðun hermisins samræmdest niðurstöðum prófanna. Góðar niðurstöður fengust þegar hermirinn var stilltur fyrir hliðar færslu, köfun og snúning. Aftur á móti fékkst ekki eins góð niðurstaða fyrir áfram færslu og þarfnast hún frekari rannsókna.

# Acknowledgements

I would like to thank the people who made this thesis possible. First I would like to thank my supervisors at the School of Science and Engineering, Reykjavik University, Leifur Þór Leifsson and Indriði Sævar Ríkharðsson, for their support and assistance. I would also like to extend my gratitude to Teledyne Gavia and their staff, especially for their technical support in the use of the Gavia AUV, as well as allowing the use of their testing facility. I owe my gratitude to the University of Iceland, which made all the experiments possible, by lending their Gavia AUV during the thesis. I am in gratitude to Reykjavik University, which made the experiments possible with financial contribution for the experimental equipment. I extend my gratitude to the staff at Laugardalslaug Swimming Pool, who allowed the use of the indoor pool for the experiments. Finally, I would like to thank a few students at the Reykjavik University, Brynjólfur Guðmundsson, Eiríkur Jónsson, Elín Adda Steinarsdóttir and Guðmundur Viktorsson for their assistance in the AUV experiments. Without the help I have been given, my thesis would never have been completed.





# Contents

<b>List of Figures</b>	<b>x</b>
<b>List of Tables</b>	<b>xv</b>
<b>List of Abbreviations</b>	<b>xvii</b>
<b>List of Symbols</b>	<b>xix</b>
<b>List of Simulink Blocks</b>	<b>xxi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 The Gavia AUV . . . . .	2
1.2 Other low-speed AUV's . . . . .	4
1.3 Research Objectives . . . . .	6
1.4 Thesis Outline . . . . .	7
<b>2 Equations of Motion</b>	<b>9</b>
2.1 Statics and dynamics . . . . .	9
2.2 General Equations of Motion . . . . .	11
2.3 System of Equations for Low-Speed Motion . . . . .	15
2.3.1 Surge Dynamics . . . . .	18
2.3.2 Sway Dynamics . . . . .	19
2.3.3 Heave Dynamics . . . . .	20
2.3.4 Yaw Dynamics . . . . .	20
2.4 Summary of equations for 4 DOF . . . . .	23
<b>3 Simulator Design</b>	<b>25</b>
3.1 The design premises . . . . .	25
3.2 Overview of the Simulator . . . . .	26
3.3 Implementation . . . . .	32

3.4	Common elements . . . . .	34
3.4.1	Input elements . . . . .	34
3.4.2	PID controllers and thrusters . . . . .	34
3.4.3	Added Mass . . . . .	36
3.4.4	Drag . . . . .	36
3.4.5	Sensors . . . . .	37
3.5	Specific Implementation in Surge Model . . . . .	38
3.6	Specific Implementation in Sway and Heave Models . . . . .	39
3.7	Specific Implementation in Yaw Model . . . . .	40
3.8	Open Loop Version of the Simulator . . . . .	41
3.9	Running the Simulator . . . . .	41
3.10	Parametric and Optimization Features of the Simulator . . . . .	44
<b>4</b>	<b>Open and Closed Loop Experiments with the Gavia AUV</b>	<b>45</b>
4.1	Experimental Approach . . . . .	45
4.2	Experimental Setup . . . . .	47
4.2.1	Mechanical and Electrical Equipment . . . . .	47
4.2.2	Computer Software . . . . .	50
4.3	Experimental Testing . . . . .	53
4.3.1	Open Loop Tests . . . . .	53
4.3.2	Closed Loop Tests . . . . .	55
4.3.3	Example Results . . . . .	56
4.4	Discussion . . . . .	57
<b>5</b>	<b>Model Validation and Tuning</b>	<b>59</b>
5.1	Experimental Data Processing . . . . .	59
5.2	Surge Model Validation . . . . .	62
5.3	Sway Model Validation . . . . .	64
5.4	Heave Model Validation . . . . .	65
5.5	Yaw Model Validation . . . . .	66
5.6	Model Validation Plots . . . . .	68
<b>6</b>	<b>Design Studies</b>	<b>83</b>
6.1	Case Study 1 . . . . .	83
6.2	Case Study 2 . . . . .	89
6.3	Case Study 3 . . . . .	90
6.4	Case Study 4 . . . . .	91
<b>7</b>	<b>Concluding Remarks</b>	<b>93</b>

7.1	Summary . . . . .	93
7.2	Suggested Future Work . . . . .	95
<b>Appendices</b>		<b>98</b>
<b>A</b>	<b>Equations of motion</b>	<b>99</b>
A.1	Dynamics . . . . .	99
A.2	Equations of Motion . . . . .	100
A.2.1	Translational Motion . . . . .	100
A.2.2	Rotational Motion . . . . .	102
A.3	Governing Equations . . . . .	104
<b>B</b>	<b>Open Loop Version of the Simulator</b>	<b>107</b>
<b>C</b>	<b>Test Equipment Specification</b>	<b>113</b>
C.1	Seabotix Thruster Specification . . . . .	113
C.2	TSL Technology Tunnel Thruster Specification . . . . .	115
C.3	Arduino Mega Specification . . . . .	120
C.4	DCDC Mini-Box Regulator Specification . . . . .	128
C.5	Motor Controller Devantech MD22 Specification . . . . .	134
C.6	Doppler Velocity Log Specification (DVL) . . . . .	141



# List of Figures

1.1	Gavia AUV at Kleifarvatn, Iceland. ( <i>Courtesy of Teledyne Gavia, 2011</i> ) .	2
1.2	Gavia Base Vehicle Set-Up. ( <i>Courtesy of Teledyne Gavia, 2011</i> ) . . . . .	3
1.3	Gavia Payloads. ( <i>Courtesy of Teledyne Gavia, 2011</i> ) . . . . .	3
1.4	Photographs of the Kalamer-E and Kraken developed and made by University of Colorado and the Delphin2 AUV made by University of Southampton . . . . .	4
1.5	Different versions of thrusters for vertical and horizontal movement of AUV	5
1.6	Maco, low speed box configured AUV. ( <i>auvac.org, 2011</i> ) . . . . .	6
2.1	Coordinate System, both body-fixed and earth-fixed. . . . .	10
2.2	Rotation showing linear and angular velocities. Boat A rotates heading angle $\gamma$ about $Z_3$ , Boat B rotates pitch angle $\theta$ about $Y_2$ and boat C rotates roll angle $\phi$ about $X_1$ . . . . .	11
2.3	Force diagram of a torpedo shaped AUV moving forward . . . . .	12
2.4	The added mass effect due to inertia of surrounding fluid is shown by the red regions . . . . .	12
2.5	Remaining DOF for low speed demonstration . . . . .	15
2.6	Ellipsoid with semi-axis a, b and c . . . . .	17
2.7	Cylinder in cross-flow and forces location . . . . .	21
2.8	Drag coefficients of smooth bodies at low Mach numbers . . . . .	22
3.1	Graphical user interface of the simulator, closed loop . . . . .	27
3.2	Surge model, closed loop . . . . .	28
3.3	Sway model, closed loop . . . . .	29
3.4	Heave model, closed loop . . . . .	30
3.5	Yaw model, closed loop . . . . .	31
3.6	Basic block set-up used through all four decoupled degrees of freedom . .	32
3.7	Available thrusters response as function of input signal to motor controllers	33
3.8	Option of higher order input function . . . . .	35

3.9	Element 2: PID and thrust implementation . . . . .	35
3.10	Propeller direction implementation in element 2 . . . . .	36
3.11	Element 4: Quadratic drag implementation . . . . .	37
3.12	Element 5: Measuring meter implementation . . . . .	37
3.13	Surge thrust control implementation . . . . .	38
3.14	Implementation on coefficient $\alpha_0$ (Eq. 2.20) for surge direction . . . . .	38
3.15	Implementation on coefficient $\beta_0$ (Eq. 2.21) and added mass term for sway and heave directions . . . . .	39
3.16	Implementation on added inertia by the ellipse approach (cf. Section 2.3) for yaw rotation . . . . .	40
3.17	Implementation on added mass by the two dimensional approach (cf. Sec- tion 2.3) for yaw rotation . . . . .	41
3.18	Selection required going through running process of simulator in batch mode . . . . .	42
3.19	An example of running the batch mode script for one arbitrarily type of settings . . . . .	43
3.20	An example of minimizing area between two curves, one is from simula- tor and other based on real data . . . . .	44
4.1	Implementation of forces to validate each DOF of model . . . . .	46
4.2	A sketch of the brackets and thrusters mounted on to the hull of the Gavia AUV . . . . .	48
4.3	Thruster equipment . . . . .	49
4.4	Assembled experimental equipment . . . . .	49
4.5	RX Graph from ROS command line . . . . .	51
4.6	Graphical User Interface (GUI) . . . . .	52
4.7	Image from yaw testing . . . . .	54
4.8	Image from sway testing . . . . .	55
4.9	Experimental example results . . . . .	56
4.10	Layout of experimental equipment . . . . .	57
4.11	One suggested set-up of the experimental equipment if the experiments will be continued in the future . . . . .	58
4.12	Horizontal and vertical tunnels in the Delphin2 AUV <sup>3</sup> . . . . .	58
5.1	GPS tracking of open loop sway experiment . . . . .	60
5.2	Original Data from one of the open loop sway experiments . . . . .	60
5.3	Post-processed data from one of the open loop sway experiments . . . . .	61
5.4	Final Data . . . . .	61

5.5	Comparison on open loop surge model, with and without higher order term	63
5.6	Surge open loop response, ellipse method applied for added mass, the model is not equipped with higher order activity . . . . .	68
5.7	Surge open loop response, two dimensional method applied for added mass, the model is not equipped with higher order activity . . . . .	69
5.8	Surge open loop response, two dimensional method applied for added mass, the model is equipped with higher order transfer function . . . . .	70
5.9	Sway open loop response, the ellipse method applied. . . . .	71
5.10	Sway open loop response, the two dimensional method applied. . . . .	72
5.11	Sway closed loop response, the ellipse method applied. . . . .	73
5.12	Sway closed loop response, the two dimensional method applied. . . . .	74
5.13	Yaw open loop response, the ellipse method applied. . . . .	75
5.14	Yaw open loop response, the two dimensional method applied. . . . .	76
5.15	Yaw closed loop response, the ellipse method applied. . . . .	77
5.16	Yaw closed loop response, the two dimensional method applied. . . . .	78
5.17	Surge open loop velocity response, the ellipse method. . . . .	79
5.18	Surge open loop velocity response, the two dimensional method. . . . .	80
5.19	Surge open loop velocity response, the two dimensional method and the higher order term active. . . . .	81
6.1	Mapping around an interesting point . . . . .	84
6.2	Tunnel thruster comparison. Both thrusters were IntegratedThrusters from TSL Technology. The smaller one has diameter of 50mm while the larger one has diameter of 100mm . . . . .	86
6.3	Tunnel thruster location from center of gravity comparison. This is 100mm IntegratedThrusters from TSL Technology . . . . .	88
6.4	Comparison between the same system, with a different time working and publishing data from sensors . . . . .	89
6.5	The effect of different restoring forces acting on the boat while diving to a fixed reference value of 1m. Thrusters in the model are 100mm Integrated Thrusters from TSL Technology . . . . .	90
6.6	The effect of adding a higher order reference values to the system while travelling to a fixed length of 2 m in surge direction. The Thruster in the model is the original thruster pre-installed in Gavia . . . . .	91
A.1	Body-fixed reference frame and earth-fixed reference frame . . . . .	101
B.1	Surge open loop . . . . .	108
B.2	Sway open loop . . . . .	109

B.3	Heave open loop . . . . .	110
B.4	Yaw open loop . . . . .	111



# List of Tables

1.1	Gavia base vehicle specification . . . . .	3
2.1	Marine notation . . . . .	9
2.2	Vector Presentation . . . . .	10
2.3	Two dimensional added mass coefficient for slender bodies . . . . .	17
4.1	RPM values for surge . . . . .	54
4.2	Thrust values for sway and heave . . . . .	54
4.3	Thrust values for yaw . . . . .	54
4.4	PID and ref. for sway and heave . . . . .	55
4.5	PID and ref. for yaw . . . . .	55
5.1	Comparison performed for surge DOF . . . . .	62
5.2	Transfer function . . . . .	63
5.3	Results of adjusting the surge model to the experimental data . . . . .	64
5.4	Comparison performed for sway DOF . . . . .	64
5.5	Results of adjusting the sway model to the experimental data . . . . .	65
5.6	Tests performed for yaw DOF . . . . .	66
5.7	Results of adjusting the yaw model to the experimental data . . . . .	67
6.1	Values used for test Case 1 . . . . .	85
6.2	Values used for test Case 2 . . . . .	85
6.3	Meter Specifications for test Cases 1 and 2 . . . . .	85



# List of Abbreviations

AUV	Autonomous underwater vehicle
CG	Center of gravity
COTS	Commercial-off-the-shelf
DOF	Degrees of freedom
DVL	Doppler velocity log
GPS	Global positioning system
INS	Inertial navigation system
ROS	Robot operating system
ROV	Remotely operated vehicle
RPM	Revolutions per minute
UUV	Unmanned underwater vehicle
VRT	Vortex ring generator
















# List of Symbols

$m$	Mass	$[kg]$
$I$	Inertia	$[kg \cdot m^2]$
$X$	X-direction force	$[N]$
$Y$	Y-direction force	$[N]$
$Z$	Z-direction force	$[N]$
$K$	Moment about x-axis (roll)	$[Nm]$
$M$	Moment about y-axis (pitch)	$[Nm]$
$N$	Moment about z-axis (yaw)	$[Nm]$
$x$	X-direction Euler position	$[m]$
$y$	Y-direction Euler position	$[m]$
$z$	Z-direction Euler position	$[m]$
$\phi$	Euler angles about x-axis	$[rad]$
$\theta$	Euler angles about y-axis	$[rad]$
$\psi$	Euler angles about z-axis	$[rad]$
$u$	Linear velocity in x-direction	$[m/s]$
$v$	Linear velocity in y-direction	$[m/s]$
$w$	Linear velocity in z-direction	$[m/s]$
$p$	Angular velocity about x-axis	$[rad/s]$
$q$	Angular velocity about y-axis	$[rad/s]$
$r$	Angular velocity about z-axis	$[rad/s]$
$x_G$	X location of CG	$[m]$
$y_G$	Y location of CG	$[m]$
$z_G$	Z location of CG	$[m]$
$X_{\dot{u}}$	Added mass (surge)	$[kg]$
$Y_{\dot{v}}$	Added mass (sway)	$[kg]$
$Z_{\dot{w}}$	Added mass (heave)	$[kg]$
$N_{\dot{r}}$	Added mass (yaw)	$[kg]$
$X_u$	Linear Drag (surge)	$[kg/s]$
$Y_v$	Linear Drag (sway)	$[kg/s]$
$Z_w$	Linear Drag (heave)	$[kg/s]$
$N_r$	Linear Drag (yaw)	$[kg \cdot m^2/s]$
$X_{u u }$	Non-linear Drag (surge)	$[kg/m]$
$Y_{v v }$	Non-linear Drag (sway)	$[kg/m]$
$Z_{w w }$	Non-linear Drag (heave)	$[kg/m]$
$N_{r r }$	Non-linear Drag (yaw)	$[kg \cdot m^2]$

$\boldsymbol{\eta}_1$	Fixed position vector	$[m]$
$\boldsymbol{\eta}_2$	Fixed attitude vector	$[rad]$
$\boldsymbol{\tau}_1$	Force vector acting on the vehicle in the body fixed frame	$[N]$
$\boldsymbol{\tau}_2$	Moment vector acting on the vehicle in the body fixed frame	$[Nm]$
$\boldsymbol{v}_1$	Fixed linear velocity vector	$[m/s]$
$\boldsymbol{v}_2$	Fixed angular velocity vector	$[rad/s]$
$\tau_{Thrust}$	Thrust force	$[N]$
$\rho$	Density	$[kg/m^3]$
$\omega_n$	Natural frequency	$[rad/s]$
$\zeta$	Damping ratio	$[kg/m^3]$
$A$	Area	$[m^2]$
$D$	Diameter	$[m]$
$l$	Length	$[m]$
$L$	Length	$[m]$
$\alpha_0$	Constant for added mass	$[-]$
$\beta_0$	Constant for added mass	$[-]$
$C_D$	Drag coefficient	$[-]$
$e$	Eccentricity of ellipsoid	$[-]$

# List of Simulink Blocks

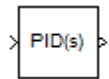
	Constant block, generates a real or complex constant value
	Gain block, it multiplies the input by a constant value (gain)
	Reciprocal block, it output the reciprocal of its input signal
	Sum block, it performs addition or subtraction on its inputs
	Production block, it outputs the multiplication of its input signals
	Power block, it outputs u in the the power of v
	Absolute block, it outputs the absolute value of the input
	Integrator block, it outputs the integral of its input at the current time step
	Relational operator block, it performs specified relational operation on its inputs
	Saturation block, it limits the range of signals
	Quantizer block, it holds the current value until next higher or lower discrete has been reached
	Transport delay block, it delays the input by given amount of time
	Zero-Order hold block, it samples the signal and holds that value for a predetermined sample period, then it re-sample and repeats



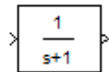




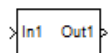
Switch block, it passes a signal through either input, based on the control signal



PID controller block, it simulates continuous- or discrete-time PID controllers



Transfer function, it models a linear system by the transfer function



Subsystem block, it represents a system within another system



Step function block, it generates a step function



Scope, displays signals generated during a simulation



Inport, creates input port for a subsystem or a external input



Outport, it creates an output port for a subsystem or an external output



Vector concatenate block, it concatenates the input signals of same data type to create contiguous output signal



# Chapter 1

## Introduction

Unmanned underwater vehicles (UUVs) are vehicles that are able to operate underwater without a human occupant. UUVs differ in sizes, they range from man portable lightweight vehicles to large and heavy vehicles. They fall into two categories, remotely operated underwater vehicles (ROVs) and autonomous underwater vehicles (AUVs). The ROV is connected to a command platform (usually ship or harbour) with a tether, which ensures energy supply and data communication between user and the ROV. An autonomous underwater vehicle (AUV) is a robot which travels underwater without requiring any input from an operator. Their tasks and missions are constantly evolving, becoming more complicated. Among users are; commercial-, military-, research- and hobby users. A typical commercial job for the AUV is, e.g., to construct detailed maps of the seafloor before building subsea infrastructure in the oil and gas industry. A typical military mission for an AUV is to map an area and determine if there are any mines, or to monitor a protected area for unidentified objects. Scientists use AUVs to study lakes, the ocean and the ocean floor.

The objective of this research is to investigate the low-speed characteristics of torpedo shaped AUV's and their control. We will use the commercially available Gavia AUV as a testbed. In this chapter, we will introduce the Gavia AUV and its key features and characteristics. Other low-speed AUV's will be introduced as well. The research objectives will be stated and the layout of the thesis will be given.



Figure 1.1: Gavia AUV at Kleifarvatn, Iceland. (*Courtesy of Teledyne Gavia, 2011*)

## 1.1 The Gavia AUV

Gavia is a small unmanned torpedo shaped modular AUV, designed by Teledyne Gavia<sup>1</sup>, shown in Fig. 1.1. Their contribution to the AUV market are three different versions of the boat, as well as any custom made versions based on customer demand. The standard versions are the Offshore-, Scientific- and Defence AUVs. The Offshore version is used for bathymetric surveys, pipeline inspections, environmental surveys, exploration and post hurricane inspection. The Scientific version is used for oceanography, limnology habitat assessment, hydrography, bathymetric surveys, archeology and wreck finding. And finally the Defence version is used for mine counter measure, anti submarine warfare training, environmental assessment, surveillance, search and recovery, security and researches. The minimum configuration of the Gavia platform is the Base Vehicle. That vehicle consist of nose, battery, control and propulsion modules. Other expansion payloads in the form of modules and sensors can be added to this base platform in order to provide additional capability. The Gavia Base Vehicle, shown in Fig. 1.2, is equipped with a compass, GPS, WLAN and Iridium connection as well as a ruggedized laptop PC for controlling the vehicle. The physical properties of the Base Vehicle are given in Table 1.1, but because of some customer needs in selection of payload, their boat can easily reach in excess of 3 meters.

Several payloads modules are available, shown in Fig. 1.3. Physical properties of the boat, e.g., length, weight and mass vary, with more payload installed. The configuration of Gavia is different for each customer.

<sup>1</sup> [www.gavia.is](http://www.gavia.is)

Table 1.1: Gavia base vehicle specification

Physical Property	Value
Mass	49 <i>kg</i>
Length	1.8 <i>m</i>
Diameter	0.2 <i>m</i>



Figure 1.2: Gavia Base Vehicle Set-Up. (Courtesy of Teledyne Gavia, 2011)

All of the Gavia modules are plug and play modules which can be configured in the field by inserting the required module for the task at hand. All modules are interconnected using the built-in Quick Lock system that snaps the modules together, both mechanically and electrically. All modules are splash proof, facilitating on-deck replacement but some sensors are mounted internally and are not field swappable.

The Gavia is capable of solving different types of tasks. Due to limited control during low speed, all tasks requires the boat to cruise at speed above 1.5 *m/s*. Consequently, Gavia is not capable of solving tasks that involves low speeds, such as surveillance during no motion and homing to a charging station.

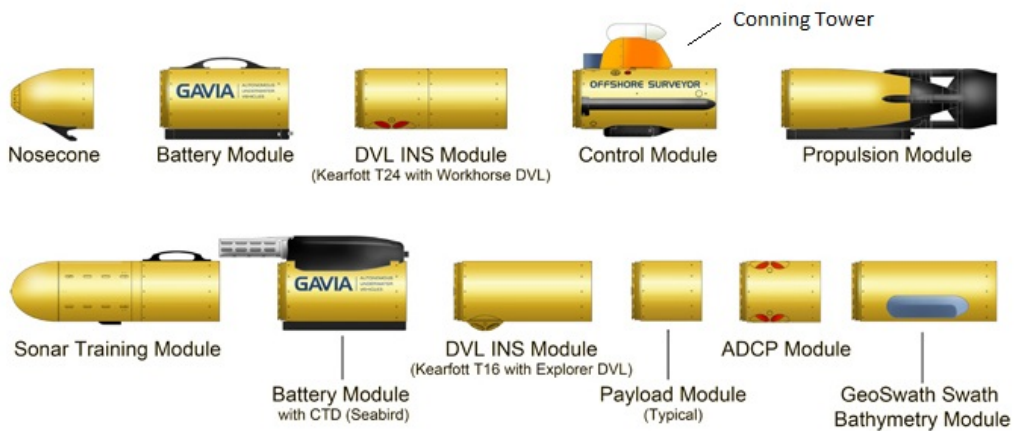


Figure 1.3: Gavia Payloads. (Courtesy of Teledyne Gavia, 2011)

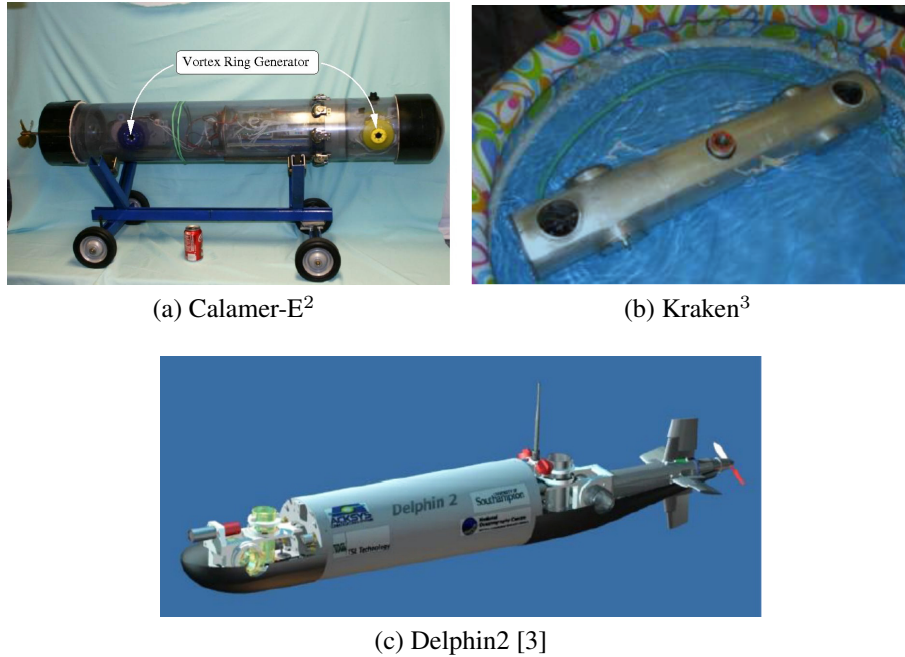


Figure 1.4: Photographs of the Calamer-E and Kraken developed and made by University of Colorado and the Delphin2 AUV made by University of Southampton

Among features that Gavia needs are:

- Maintain heading at low or zero speed
- Maintain depth at low speed
- Maintain location at zero speed

## 1.2 Other low-speed AUV's

The implementation of a low-speed control in a high-speed, long range underwater vehicles has been studied by few groups during recent years. All groups are university related, driven by professors or students doing their research. As an example of such projects are the Calamer-E<sup>2</sup>, Kraken<sup>2 3</sup> and Delphin2<sup>4</sup> projects [1, 2, 3].

CALAMER-E (The Cavity Actuated Low-speed Actively Manoeuvrable Aquatic Rover Experiment), shown in Fig. 1.4a, heritage dates back to 2002. It was developed by engineering students at the University of Colorado. In 2005, its main goal was to build a submarine with a novel jet actuators. The vehicle had no traditional steering but used

<sup>2</sup> <http://enstrophy.mae.ufl.edu/mohseni/Platforms1.html#Underwater1>, December 2011

<sup>3</sup> [http://aeroprojects.colorado.edu/07\\_08/kraken/SFR/SFR\\_KRAKEN\\_trunc.pdf](http://aeroprojects.colorado.edu/07_08/kraken/SFR/SFR_KRAKEN_trunc.pdf), December 2011

<sup>4</sup> <http://www.leo-steenenson.com/delphin2/intro.html>, December 2011

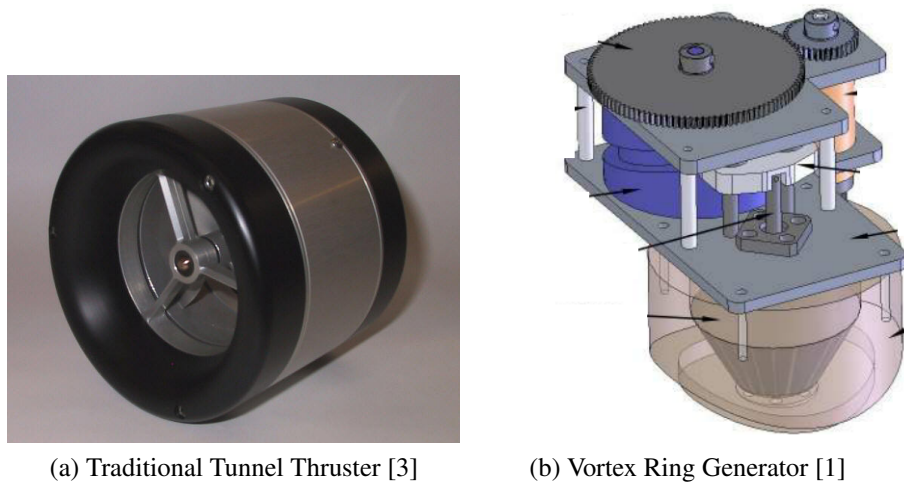


Figure 1.5: Different versions of thrusters for vertical and horizontal movement of AUV

thrusters for all movements instead. The propeller at the back had a standard motor driven thrust equipment while the thrust equipments at sides, top and bottom had synthetic jet actuators [1].

In 2008, a team of engineering students from the University of Colorado developed and made KRAKEN AUV (the kinematically Roving Autonomously Controlled Electro-Nautic), shown in Fig. 1.4b. The vehicle incorporates Vortex Ring Thrusters (VRTs) previously designed by Professor Mohseni and his students at the university, shown in Fig. 1.5b. The VRTs, which mimic the propulsive mechanism of squids, could propel the vehicle laterally and allow it to execute zero-radius yaw rotation while maintaining a low external profile with minimal drag [2]. Its design was modelled in Simulink.

The Delphin2 AUV was designed and built at the University of Southampton, UK, shown in Fig. 1.4c. The vehicle is over-actuated with two horizontal tunnel thrusters, a rear ducted propeller, and four independent control planes. The tunnel thrusters here are 70 mm TSL brushless 50W traditional electrical tunnel thrusters, shown in Fig. 1.5a [3].

The performance of tunnel thrusters in an AUV hull form over the full range of operational vehicle speeds and yaw have been investigated by isolating the thrusters from the hull [4]. These results showed a variation in thrust of around 15 % over the range of forward speeds tested. However no common modelling approach for the performance of a tunnel thruster is readily available. This is thought to be due to the complexity of the interactions involved and the uniqueness of each of the configurations.

Low-speed manoeuvrability and modelling of a box configured AUV has been studied in more detail. These are AUVs equipped with multiple thrust-fans creating vehicles that



Figure 1.6: Maco, low speed box configured AUV. (*auvac.org*, 2011)

manoeuvre accurately and make precise attitude adjustments. These vehicles are incapable of high speed travelling due to their high drag. Among those modelling projects are the decoupled modelling and control for the hybrid AUV, Maco [5]. Each degree of freedom was modelled separately in Simulink, where coefficients were found by experiments. The Maco AUV is shown in Fig. 1.6.

### 1.3 Research Objectives

As discussed, the manoeuvring of AUVs is an important research topic. In particular, there is a need to develop robust and accurate mechanisms for the control of torpedo shaped AUVs moving at low-speeds. The research objectives of this thesis are the following:

1. Derive the equations of motion for torpedo shaped AUVs moving at low-speed.
2. Implement the equations of motion in a computational framework, i.e., develop a simulator.
3. Develop open loop and closed loop control systems for the torpedo shaped AUVs at low-speed.
4. Validate the simulator and control systems through physical experiments.
5. Demonstrate how the developed methodology can be applied to realistic design scenarios.



## 1.4 Thesis Outline

The remainder of the thesis is organized as follows. Chapter 2 covers the AUV dynamics and kinematics of the AUV. In Chapter 3, the simulator design and implementation of the simulator are discussed. Chapter 4 is about experiments performed and equipment needed to validate the simulator. Chapter 5 describes different types of design criteria using the simulator and Chapter 6 covers the concluding remarks and suggests future work. Appendix A is a support to the dynamic, carried out in chapter 2. Appendix B shows the open loop control subsystems for each DOF, in the open loop control version of the simulator. Appendix C covers equipment specification of test equipment.



## Chapter 2

# Equations of Motion

In this chapter we will describe the general motion dynamics of underwater vehicles. Based on that, we will derive the equations of motion for a torpedo shaped AUV moving at low-speed in four degrees of freedom.

### 2.1 Statics and dynamics

It is common to divide the study of dynamics into two parts: kinematics and kinetics. Kinematics treats only geometrical aspects of motion or the physical properties of the vehicle, but kinetics is the analysis of the forces causing the motion usually called hydrodynamic forces. Since 6 independent coordinates are necessary to determine the position and orientation of a rigid body, a general torpedo shaped AUV has 6 degrees of freedom (DOF). Table 2.1 lists the important abbreviations used through this chapter.

Table 2.1: Marine notation

DOF	Description	Positions and Euler angles	Forces and Moments	Linear and Angular Velocity
Surge	Motions in the x-direction	$x$	$X$	$u$
Sway	Motions in the y-direction	$y$	$Y$	$v$
Heave	Motions in the z-direction	$z$	$Z$	$w$
Roll	Rotations about the x-axis	$\phi$	$K$	$p$
Pitch	Rotations about the y-axis	$\theta$	$M$	$q$
Yaw	Rotations about the z-axis	$\psi$	$N$	$r$

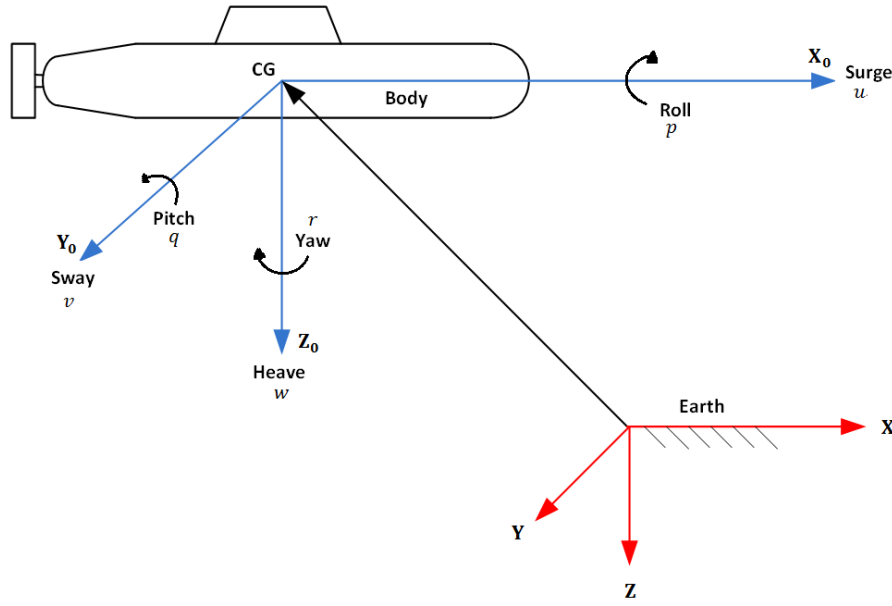


Figure 2.1: Coordinate System, both body-fixed and earth-fixed.

It is convenient to define two coordinate frames when analysing the motion of marine vehicles is 6 degrees of freedom. The moving coordinate frame  $X_0Y_0Z_0$  is conveniently fixed to the vehicle and is called the body-fixed reference frame and the fixed coordinate frame  $XYZ$  is the earth coordinate system [6], see Fig. 2.1. The fixed coordinate system is earth coordinate system while the moving one follows vehicle heading and location each time. The  $X$  axis is in line with length of boat and center of gravity where positive movement means forward and vice versa. All movement in line with  $X$  axis is surge movement while rotation about the axis is roll rotation. Axis  $Y$  and  $Z$  then follow the  $X$  axis as in regular coordinate system. Movement in line with  $Y$  axis is sway while rotation about same axis is pitch rotation. Movement in line with  $Z$  axis is heave, while rotating about same axis is yaw rotate. Figure 2.1 describes the general motion of a marine vehicle moving in 6 DOF using notation from Table 2.2 [7].

Here  $\eta$  denotes the earth fixed position and attitude vector,  $v$  denotes the body fixed linear and angular velocity vector and  $\tau$  is used to describe the forces and moments acting on

Table 2.2: Vector Presentation

Location	Velocity and Ang. Velocity	Forces and Moments
$\eta = [\eta_1^T, \eta_2^T]$	$v = [v_1^T, v_2^T]$	$\tau = [\tau_1^T, \tau_2^T]$
$\eta_1 = [x, y, z]^T$	$v_1 = [u, v, w]^T$	$\tau_1 = [X, Y, Z]^T$
$\eta_2 = [\phi, \theta, \psi]^T$	$v_2 = [p, q, r]^T$	$\tau_2 = [K, M, N]^T$

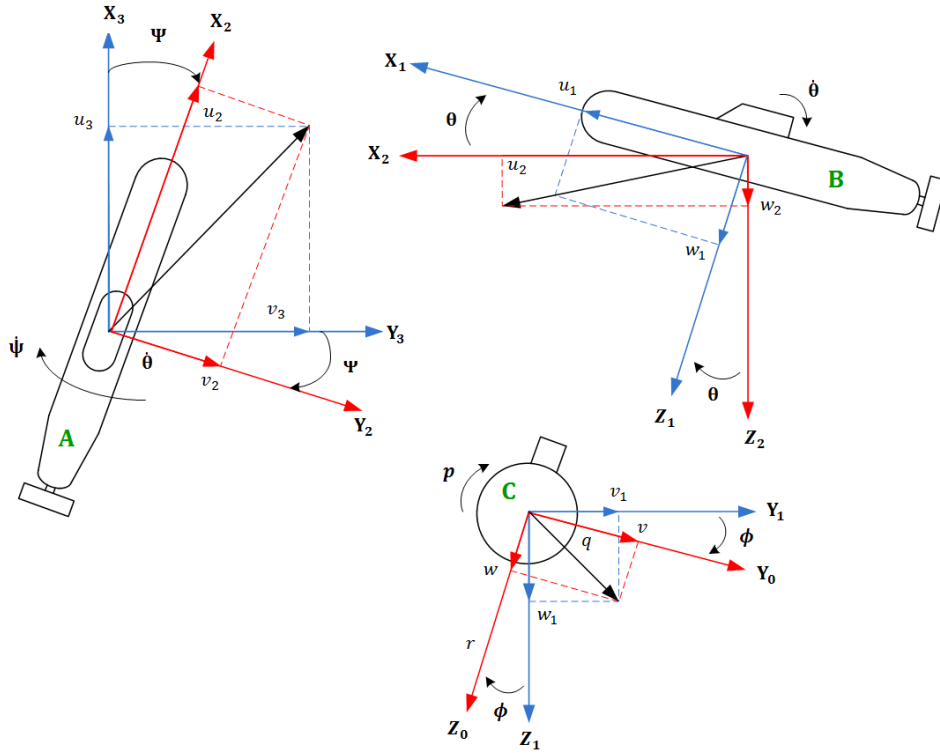


Figure 2.2: Rotation showing linear and angular velocities. Boat A rotates heading angle  $\gamma$  about  $Z_3$ , Boat B rotates pitch angle  $\theta$  about  $Y_2$  and boat C rotates roll angle  $\phi$  about  $X_1$ .

the vehicle in the body-fixed frame [7]. For further understanding on notation and coordinates used through this thesis, Fig. 2.2 describes coordinate system of an underwater vehicle.

## 2.2 General Equations of Motion

Motion of rigid body can be represented with Newton's equations of motion. For a rigid body of any type moving in 6 degrees of freedom, the equation of motion are,

$$\begin{aligned}
 m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] &= \sum X_{ext} \\
 m[\dot{v} - wp + ur - y_G(r^2 + p^2) + z_G(qr - \dot{p}) + x_G(qp + \dot{r})] &= \sum Y_{ext} \\
 m[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q}) + y_G(rq + \dot{p})] &= \sum Z_{ext} \\
 I_x \dot{p} + (I_z - I_y)qr + m[y_g(\dot{w} - uq + vp) - z_G(\dot{v} - wp + ur)] &= \sum K_{ext} \\
 I_y \dot{q} + (I_x - I_z)rp + m[z_g(\dot{u} - vr + wq) - x_G(\dot{w} - uq + vp)] &= \sum M_{ext} \\
 I_z \dot{r} + (I_y - I_x)pq + m[x_g(\dot{v} - wp + ur) - y_G(\dot{u} - vr + wq)] &= \sum N_{ext},
 \end{aligned} \tag{2.1}$$

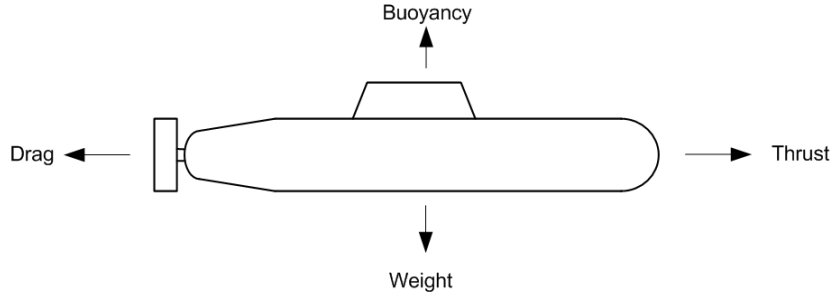


Figure 2.3: Force diagram of a torpedo shaped AUV moving forward

where  $m$  is the rigid body mass,  $I$  is the rigid body inertia about specific axis,  $x_G$ ,  $y_G$  and  $z_G$  are each axis distance from rigid body center of gravity. Other symbols are shown in Table 2.1. A detailed derivation of this equation is given in Appendix A. This equation describes coupled motion of underwater vehicle moving in 6 degrees of freedom. The equation is general and applies to all underwater vehicles.

Hydrodynamic forces and moments are imposed on an AUV by water flowing against and around it. Among the forces and moments are positive frontal pressure against the structure, drag effect along the sides, and negative pressure in the downstream side. Figure 2.3 shows force diagram of an AUV moving forward. The figure shows how the weight and buoyancy forces affects the vertical movement of the boat, while drag and propeller forces affects the horizontal movement. Hydrodynamic forces and moments are; restoring forces due to weight and buoyancy, added mass due to inertia of surrounding fluid, shown in Fig. 2.4 , radiation induced potential damping due to energy carried away by generated surface waves, environmental forces due to ocean currents, waves, wind and finally the propulsion forces due to the propulsion equipment.

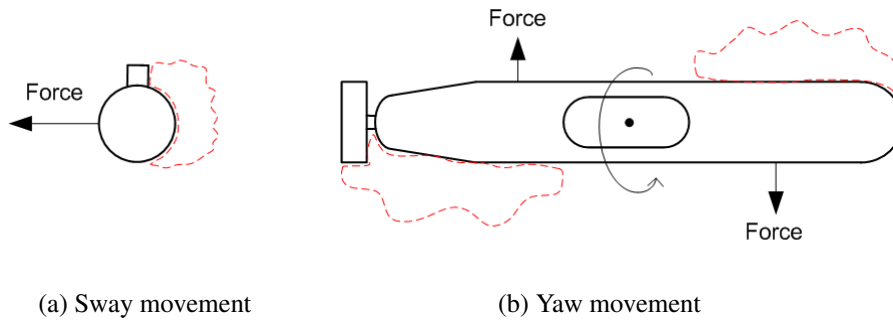


Figure 2.4: The added mass effect due to inertia of surrounding fluid is shown by the red regions

By examining the vector form of Eq. 2.1,

$$M_{RB}\dot{v} + C_{RB}(v)v = \tau_{RB}, \quad (2.2)$$

the total inertia matrix and Coriolis terms are,

$$M \triangleq M_{RB} + M_A, \quad (2.3)$$

$$C(v) \triangleq C_{RB}(v) + C_A(v), \quad (2.4)$$

where  $M_{RB}$  denotes the rigid body mass matrix and  $M_A$  denotes the inertia matrix due to added mass, defined for 6 DOF as [6],

$$M_A \triangleq - \begin{bmatrix} X_{\dot{u}} & X_{\dot{v}} & X_{\dot{w}} & X_{\dot{p}} & X_{\dot{q}} & X_{\dot{r}} \\ Y_{\dot{u}} & Y_{\dot{v}} & Y_{\dot{w}} & Y_{\dot{p}} & Y_{\dot{q}} & Y_{\dot{r}} \\ Z_{\dot{u}} & Z_{\dot{v}} & Z_{\dot{w}} & Z_{\dot{p}} & Z_{\dot{q}} & Z_{\dot{r}} \\ K_{\dot{u}} & K_{\dot{v}} & K_{\dot{w}} & K_{\dot{p}} & K_{\dot{q}} & K_{\dot{r}} \\ M_{\dot{u}} & M_{\dot{v}} & M_{\dot{w}} & M_{\dot{p}} & M_{\dot{q}} & M_{\dot{r}} \\ N_{\dot{u}} & N_{\dot{v}} & N_{\dot{w}} & N_{\dot{p}} & N_{\dot{q}} & N_{\dot{r}} \end{bmatrix}. \quad (2.5)$$

$C_A(v)$  is the hydrodynamic Coriolis and centripetal matrix due to added mass, defined as [6],

$$C_A(v) \triangleq \begin{bmatrix} 0 & 0 & 0 & 0 & -a_3 & a_2 \\ 0 & 0 & 0 & a_3 & 0 & -a_1 \\ 0 & 0 & 0 & -a_2 & a_1 & 0 \\ 0 & -a_3 & a_2 & 0 & -a_6 & a_5 \\ a_3 & 0 & -a_1 & a_6 & 0 & -a_4 \\ -a_2 & a_1 & 0 & -a_5 & a_4 & 0 \end{bmatrix}, \quad (2.6)$$

where

$$\begin{aligned} a_1 &= X_{\dot{u}}u + X_{\dot{v}}v + X_{\dot{w}}w + X_{\dot{p}}p + X_{\dot{q}}q + X_{\dot{r}}r \\ a_2 &= X_{\dot{v}}u + Y_{\dot{v}}v + Y_{\dot{w}}w + Y_{\dot{p}}p + Y_{\dot{q}}q + Y_{\dot{r}}r \\ a_3 &= X_{\dot{w}}u + Y_{\dot{w}}v + Z_{\dot{w}}w + Z_{\dot{p}}p + Z_{\dot{q}}q + Z_{\dot{r}}r \\ a_4 &= X_{\dot{p}}u + Y_{\dot{p}}v + Z_{\dot{p}}w + K_{\dot{p}}p + K_{\dot{q}}q + K_{\dot{r}}r \\ a_5 &= X_{\dot{q}}u + Y_{\dot{q}}v + Z_{\dot{q}}w + K_{\dot{q}}p + M_{\dot{q}}q + M_{\dot{r}}r \\ a_6 &= X_{\dot{r}}u + Y_{\dot{r}}v + Z_{\dot{r}}w + K_{\dot{r}}p + M_{\dot{r}}q + N_{\dot{r}}r \end{aligned} \quad (2.7)$$

$C_{RB}(v)$  is the hydrodynamic Coriolis and centripetal matrix caused by rigid body, defined as [6],

$$C_{RB} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}, \quad (2.8)$$

where,

$$C_{11} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad (2.9)$$

$$C_{12} = \begin{bmatrix} -m(y_G q + z_G r) & -m(x_G q - w) & -m(x_G r + v) \\ -m(y_G p + w) & m(x_G r + x_G p) & -m(y_G r - u) \\ -m(z_G p - v) & -m(z_G q + u) & m(x_G p + y_G q) \end{bmatrix}, \quad (2.10)$$

$$C_{21} = \begin{bmatrix} -m(y_G q + z_G r) & m(y_G p + w) & m(z_G p + w) \\ m(x_G q - w) & -m(z_G r + x_G p) & m(z_G q + u) \\ m(x_G r + v) & m(y_G r - u) & -m(x_G p + y_G q) \end{bmatrix}, \quad (2.11)$$

$$C_{22} = \begin{bmatrix} 0 & I_{yz}q - I_{xx}p + I_z r & I_{yz}r + I_{xy}p - I_y q \\ I_{yz}q + I_{xx}p - I_x r & 0 & -I_{xx}r - I_{xy}q + I_x p \\ -I_{yz}r - I_{xy}p + I_y q & I_{xz}r + I_{xy}q - I_x p & 0 \end{bmatrix}. \quad (2.12)$$

From Eq. 2.2, the hydrodynamic forces and moments of a rigid body can be expressed as,

$$\tau_E + \tau + \tau_H = \tau_{RB}, \quad (2.13)$$

where  $\tau_E$  are forces caused by the environment,  $\tau$  are the forces caused by propulsion equipment and  $\tau_H$  is the sum of forces caused by added mass, potential damping, restoring force, drag force, waves and vortex damping.

$$\tau_H = -M_A \dot{v} - C_A(v)v - D(v)v - g(\eta), \quad (2.14)$$

Here, the  $-M_A \dot{v} - C_A(v)v$  terms refers to the added mass,  $-g(\eta)$  terms refers to the buoyancy force and  $-D(v)v$  refers to the hydrodynamic damping, represented as,

$$D(v) \triangleq D_p(v) + D_s(v) + D_W(v) + D_M(v), \quad (2.15)$$

where  $D_P(v)$  represents the potential damping,  $D_s(v)$  represents the drag due to form and friction,  $D_W(v)$  represents the wave drift and  $D_M(v)$  represents the vortex shedding.



## 2.3 System of Equations for Low-Speed Motion

All the theoretical principles and equations that have been introduced here apply for all underwater vehicles, independent of their shape and size. However, most AUV related work at low speed can be performed using only four degrees of freedom. Those degrees are; surge, sway, heave and yaw. In this case, the governing equations of motion can be reduced to the following equations

$$\begin{aligned}
 m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] &= \sum X_{ext} \\
 m[\dot{v} - wp + ur - y_G(r^2 + p^2) + z_G(qr - \dot{p}) + x_G(qp + \dot{r})] &= \sum Y_{ext} \\
 m[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q}) + y_G(rq + \dot{p})] &= \sum Z_{ext} \\
 I_z \dot{r} + (I_y - I_x)pq + m[x_g(\dot{v} - wp + ur) - y_g(\dot{u} - vr + wq)] &= \sum N_{ext}
 \end{aligned} \quad (2.16)$$

When Gavia cruises at speed above 1.5 m/s it is capable of controlling several DOFs. The surge motion is the only controllable uncoupled DOF while others are coupled. All motion of the Gavia AUV requires the surge motion, e.g., changes in depth request requires the boat to surge forward and pitch until new depth is obtained and changes in heading request requires the boat to surge forward and yaw until new heading is obtained. During low speed (below 1.5 m/s) the steering forces become too small to control the boat causing the surge motion to be the only controllable DOF. Decoupled control design for traditional torpedo shaped AUVs equipped with one propeller, suggest that the 6 DOF linear equations of motion can be divided into three non-interacting subsystems for speed control (surge), steering (surge, sway and yaw) and diving (surge, heave and pitch) for a regular torpedo shaped AUV [6]. For a decoupled system the off-diagonal is small causing that the Coriolis and centripetal terms, Eqs. 2.6 and 2.8, will be neglected. Equation

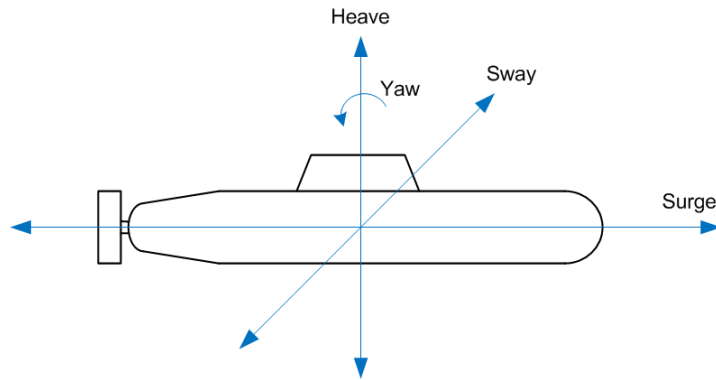


Figure 2.5: Remaining DOF for low speed demonstration

2.14 therefore reduces to,

$$\tau_H = -M_A \dot{v} - D(v)v - g(\eta). \quad (2.17)$$

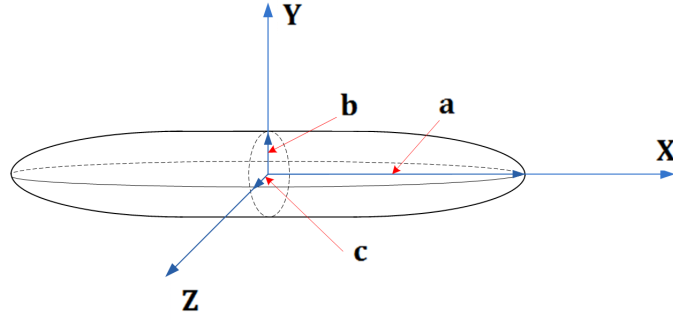
Based on the fact that Gavia is a regular torpedo shaped AUV, aimed to be controlled with four decoupled degrees of freedom, we will divide Eq. 2.16 to four non-interacting subsystems. Figure 2.5 shows the four remaining DOFs. By decoupling the control problem, individual controller design is greatly simplified, but it has been proven that decoupled control system architecture is effective in controlling at low speed and that a complex model is not necessary [8]. The surge, sway, heave and yaw degrees of freedom will be implemented here further as decoupled control system. By doing each equation non-interacting from each other all velocity and acceleration parts that belongs to other equation will be eliminated. That means that the left hand side of Eq. 2.16 simplifies. Modifications are required to the AUV hull to control the four DOFs, further discussed in Chapter 4. By assuming that the Gavia AUV is only the hull and neglecting the conning tower, the propeller, the side sonar and other items, it has three planes of symmetry. Having three planes of symmetry means that the contribution from the off diagonal elements in Eq. 2.5 can be neglected. Equation 2.5 therefore reduces to,

$$M_A \triangleq \begin{bmatrix} -X_{\dot{u}} & 0 & 0 & 0 & 0 & 0 \\ 0 & -Y_{\dot{v}} & 0 & 0 & 0 & 0 \\ 0 & 0 & -Z_{\dot{w}} & 0 & 0 & 0 \\ 0 & 0 & 0 & -K_{\dot{p}} & 0 & 0 \\ 0 & 0 & 0 & 0 & -M_{\dot{q}} & 0 \\ 0 & 0 & 0 & 0 & 0 & -N_{\dot{r}} \end{bmatrix}, \quad (2.18)$$

where the notation  $X_{\dot{u}}$  means the partial derivative of  $X$  with respect to  $\dot{u}$ , i.e.,  $\frac{\partial X}{\partial \dot{u}}$ .

Many methods have been developed for calculating the added mass. Here, we will apply two methods, the former one is based on an ellipse eccentricity which assumes that the hull has shape of an ellipse. The latter one assumes the boat has a slender body and uses the strip theory to find the added mass. The assumption that the boat is either formed as an ellipse or a slender body pretty accurate and should give a rough estimate of the added mass. However, an even better way to find the added mass, is performing physical experiments in a tow-tank and/or by using computational fluid dynamics (CFD) modeling.

The ellipse method is based on hull shape, shown in Fig. 2.6. It shows an semi-axis submerged ellipsoid with the origin at the centre. The added mass derivatives can be

Figure 2.6: Ellipsoid with semi-axis  $a$ ,  $b$  and  $c$ 

calculated using eccentricity  $e$  as well as constants  $\alpha_0$  and  $\beta_0$  represented in next three equations [6],

$$e = 1 - \frac{b^2}{a^2}, \quad (2.19)$$

$$\alpha_0 = \frac{2(1 - e^2)}{e^3} \frac{1}{2} \ln \frac{(1 + e)}{(1 - e)} - e, \quad (2.20)$$

$$\beta_0 = \frac{1}{e^2} - \frac{(1 - e^2)}{2e^3} \ln \frac{(1 + e)}{(1 - e)}, \quad (2.21)$$

where  $a$  and  $b$  are given in Fig. 2.6.

The strip theory for finding the added mass term is based on two dimensional added mass coefficient. The theory will be referred as the two dimensional method during this work. We will consider the AUV as a slender body with a characteristic length in one directions that is considerably longer than the length in other two dimensional. Submerged part of the vehicle is divided to number of strips where two dimensional coefficient can be found for each strip. To obtain the total effect or the 3D added mass coefficient, the effects of all individual strips are integrated along the length of AUV where  $A_{ij}^{2D}$  indicates the two dimensional added mass usually given in table for simple shape parts, shown in Table 2.3.

Table 2.3: Two dimensional added mass coefficient for slender bodies

$A_{11}^{(2D)} = 0.1m$ $A_{22}^{(2D)} = \pi \rho r^2$ $A_{33}^{(2D)} = \pi \rho r^2$	
--	--

The damping forces are mainly drag forces based on the velocity of the vehicle. The two main drag components for a low speed underwater vehicle are linear skin friction and quadratic (non-linear) skin friction. The linear damping  $D(v)_L$  refers to skin friction due to laminar boundary layers while the quadratic form  $D(v)_Q$  contains skin friction due to turbulent boundary layers [9]. At low velocities, the linear forces and moments dominate the non-linear damping and vice versa. The linear drag then has to decay with velocity to reflect that, at higher speeds, the non-linear drag is dominated by higher order effects [10].

For an underwater vehicle with non-coupled control, the total damping term  $D(v)$  from Eq. 2.15, can be represented as [6],

$$D(v) = (-diag[X_u, Y_v, Z_w, K_p, M_q, N_r] - diag[X_{u|u|}, Y_{v|v|}, Z_{w|w|}, K_{p|p|}, M_{q|q|}, N_{r|r|}]) , \quad (2.22)$$

or simplified as,

$$D(v) = D(v)_L + D(v)_Q, \quad (2.23)$$

where  $X_{u|u|}$  means the partial derivative of  $X$  with respect to  $u|u|$ . Besides the added mass, the drag coefficients can also be estimated using the strip theory. Derivatives using this approach usually are inaccurate and often unsatisfactory [11]. The quadratic term will be estimated theoretically in the next section. The linear term will be found later by examine and compare simulated data to experimental data, in Chapter 5.

Further simplifications will be performed in the next sections, where each DOF will be discussed separately and constants from Eqs 2.19, 2.20 and 2.21 will be used as well as the strip theory to estimate the added mass derivative.

### 2.3.1 Surge Dynamics

Surge dynamics are described by,

$$m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] = \sum X_{ext}. \quad (2.24)$$

In the surge direction, the equation simplifiers to,

$$m\dot{u} = \sum X_{ext}. \quad (2.25)$$

External forces or hydrodynamic forces in the surge direction are forces due to, added mass, thrust and drag,

$$m\dot{u} = -M_A\dot{u} - D(u)u + T_{thrust}, \quad (2.26)$$

where  $T_{thrust}$  stands for thrust forces and  $D(v)$  is represented by Eq. 2.22. The added mass derivative for surge, based on the ellipse method is [6],

$$X_{\dot{u}} = -\frac{\alpha_0}{(2 - \alpha_0)}m. \quad (2.27)$$

Added mass based on the two dimensional approach is [6],

$$X_{\dot{u}} = -\int_{-l/2}^{l/2} A_{11}^{(2D)}(y, z)dx \simeq -0.1m, \quad (2.28)$$

where  $A_{11}^{(2D)}$  is given in Table 2.3. The quadratic damping term is represented by [9, 11, 12]

$$D(v)_Q = \frac{1}{2}\rho A_{surge} C_{Dsurge} v|v|, \quad (2.29)$$

where  $\rho$  is the density,  $A_{surge}$  is the surface area for surge,  $C_{Dsurge}$  is the drag coefficient for surge and  $v$  is the velocity in surge direction. Hauksson [13] determined the surge drag coefficient for Gavia by using computational fluid dynamics (CFD), yielding  $C_{Dsurge} = 0.37$ .

### 2.3.2 Sway Dynamics

Sway dynamics are described by (after simplifications),

$$m\dot{v} = -M_A\dot{v} - D(v)v + T_{thrust}. \quad (2.30)$$

The added mass derivative for sway based on the ellipse method is,

$$Y_{\dot{v}} = -\frac{\beta_0}{(2 - \beta_0)}m. \quad (2.31)$$

While added mass based on the two dimensional approach is,

$$Y_{\dot{v}} = -\int_{-l/2}^{l/2} A_{22}^{(2D)}(y, z)dx \simeq -\pi\rho r^2 l, \quad (2.32)$$

where  $A_{22}^{(2D)}$  is given in Table 2.3. The quadratic damping term is represented same way as for the surge direction, Eq. 2.29. CFD analysis has not yet been performed to other than surge direction so no drag coefficient is available from CFD for the sway DOF. It can be taken as 1.2 for an infinite long circular cylinder [14].

### 2.3.3 Heave Dynamics

The shape of the boat for heave and sway direction are similar except for the conning tower, see Figs. 1.3 and 2.5. However, in the heave direction we have the gravitational and the buoyancy forces (Gavia is positively buoyant). Heave dynamics are described by (after simplification),

$$m\dot{w} = -M_A\dot{w} - D(w)w - (f_g + f_b) + T_{thrust}. \quad (2.33)$$

The added mass derivative term  $Z_{\dot{w}}$  and the quadratic damping term  $Z_{w|w|}$  for heave, can be assumed the same as for sway because of various symmetry in Gavia. Same value will also be used for the drag coefficient.

### 2.3.4 Yaw Dynamics

Yaw dynamics are described by (after simplification),

$$I_z\dot{r} = \sum N_{ext}. \quad (2.34)$$

External moments or hydrodynamic moments in yaw rotation are moment due to added mass and thrusters,

$$I_z\dot{r} + D(r)r = M_{thrust}. \quad (2.35)$$

The added mass derivative for yaw, based on the ellipse method is [6],

$$N_{\dot{r}} = -\frac{1}{5} \frac{(b^2 - a^2)^2(\alpha_0^2 - \beta_0^2)}{2(b^2 - a^2) + (b^2 + a^2)(\beta_0 - \alpha_0)} m. \quad (2.36)$$

It requires use of both coefficients from Eqs. 2.20 and 2.21. Added mass, based on the two dimensional approach is [6],

$$\begin{aligned}
 N_{\dot{r}} &= - \left( \int_{-B/2}^{B/2} y^2 A_{11}^{(2D)}(x, z) dy + \int_{-l/2}^{l/2} x^2 A_{22}^{(2D)}(y, z) dx \right) \\
 &\simeq - \frac{1}{12} (0.1mr^2 + \rho\pi r^2 L^3) \\
 &\simeq - \frac{1}{12} \rho\pi r^2 L^3
 \end{aligned} \tag{2.37}$$

where  $A_{11}^{(2D)}$  and  $A_{22}^{(2D)}$  are given in Table 2.3. The term  $0.1mr^2$  is neglected in the last step because in case of small torpedo shaped AUVs the term  $0.1mr^2$  is much smaller than  $\rho\pi r^2 L^3$  ( $0.1m \ll \rho\pi L^3$ ). For a real submarine this term is huge and can not be neglected.

The quadratic damping term is represented in the same way as for the surge direction, Eq. 2.29. Assume that each half is a cylinder in cross-flow and the force is acting in the middle, Fig. 2.7. The drag coefficient is then determined from graph in Fig. 2.8 [15]. The graph, which describes  $C_D$  based on Reynolds number and the  $L/D$  ratio, yielded  $C_{Dyaw} = 0.55$ . The  $L/D$  ratio of Gavia is from 9 to 13, based on different set-ups, but in the graph the ratio used was 5.

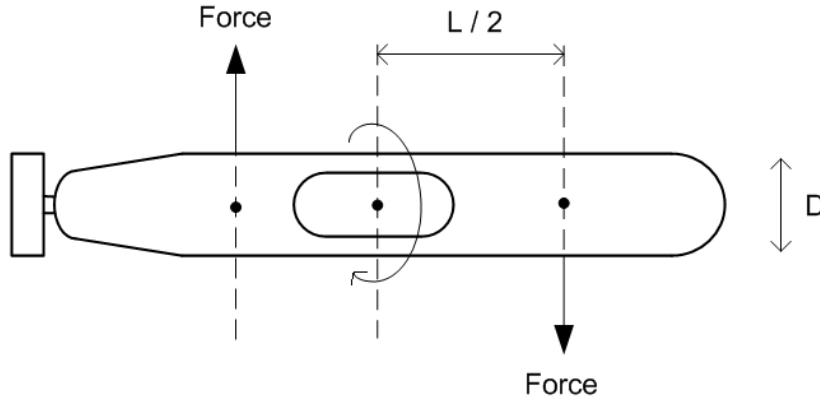


Figure 2.7: Cylinder in cross-flow and forces location

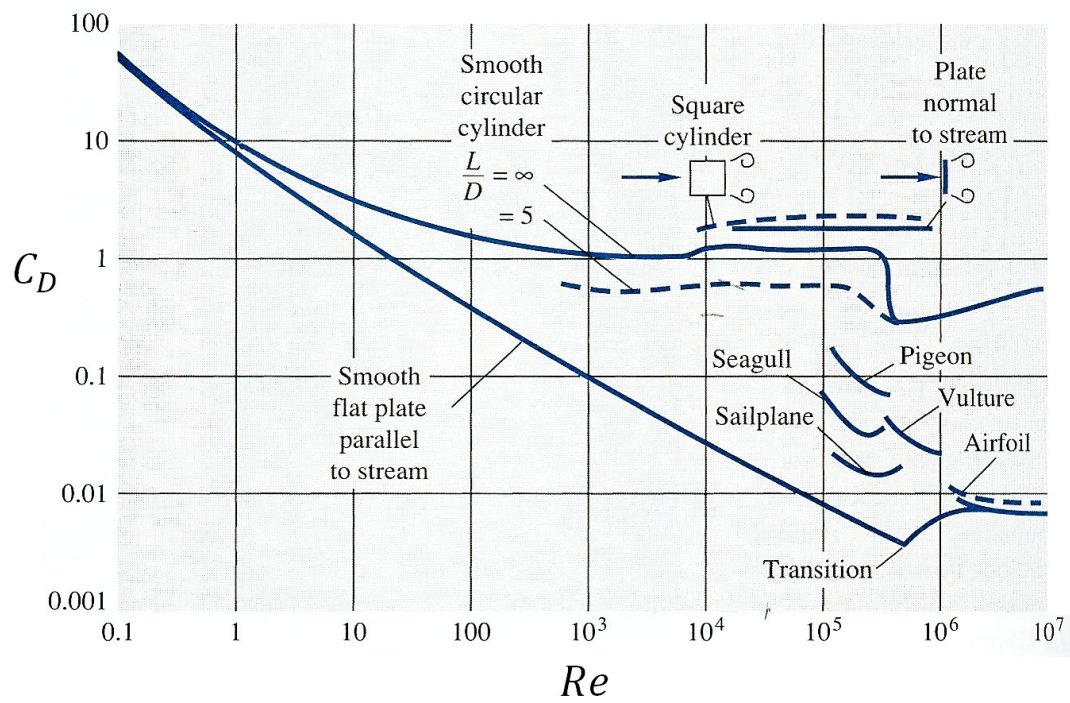


Figure 2.8: Drag coefficients of smooth bodies at low Mach numbers



## 2.4 Summary of equations for 4 DOF

### Surge DOF

$$\begin{aligned} m\dot{u} &= -M_A\dot{u} - D(u)u + T_{thrust}, \\ X_{\dot{u}(Ellipse)} &= -\frac{\alpha_0}{(2 - \alpha_0)}m, \\ X_{\dot{u}(2D)} &= -0.1m, \\ D(u)_Qu &= \frac{1}{2}\rho A_{surge}C_{Dsurge}u|u|. \end{aligned}$$

### Sway DOF

$$\begin{aligned} m\dot{v} &= -M_A\dot{v} - D(v)v + T_{thrust}, \\ Y_{\dot{v}(Ellipse)} &= -\frac{\beta_0}{(2 - \beta_0)}m, \\ Y_{\dot{v}(2D)} &= -\pi\rho r^2l, \\ D(v)_Qv &= \frac{1}{2}\rho A_{sway}C_{Dsway}v|v|. \end{aligned}$$

### Heave DOF

$$\begin{aligned} m\dot{w} &= -M_A\dot{w} - D(w)w - (f_g + f_b) + T_{thrust}, \\ Z_{\dot{w}(Ellipse)} &= -\frac{\beta_0}{(2 - \beta_0)}m, \\ Z_{\dot{w}(2D)} &= \pi\rho r^2l, \\ D(w)_Qw &= \frac{1}{2}\rho A_{heave}C_{Dheave}w|w|. \end{aligned}$$

### Yaw DOF

$$\begin{aligned} I_z\dot{r} + D(r)r &= M_{thrust}, \\ N_{\dot{r}(Ellipse)} &= -\frac{1}{5}\frac{(b^2 - a^2)^2(\alpha_0^2 - \beta_0^2)}{2(b^2 - a^2) + (b^2 + a^2)(\beta_0 - \alpha_0)}m, \\ N_{\dot{r}(2D)} &= \frac{1}{12}\rho\pi r^2l^3, \\ D(r)_Qr &= \frac{1}{2}\rho A_{yaw}C_{Dyaw}r|r|. \end{aligned}$$



## Chapter 3

# Simulator Design

In this chapter we will describe the design of a simulator for torpedo shaped AUVs. A simulator is a program that simulates real system behaviour. The simulator is a tool the designer can use to perform a variety of experiments in a virtual environment. This saves both time and money during the production.

### 3.1 The design premises

AUV simulators are usually built in such a manner that depends on variety of experiments. The data from those experiments is then analysed and used to correct and tune the model to a correct output response. Also the simulators are often decoupled in many separated file systems, which can be confusing to handle. Another common fact about simulators is that they are often only usable through their graphical user interface (GUI), which also can be time consuming for complex systems, as well as it is easy to make an error while typing in several parameters. An important part of making a simulator is validating and testing as much as possible for different criteria.

Various software can be used to implement the simulator, e.g., Simulink<sup>1</sup> by Mathworks [16]. Simulink is an environment for multi domain simulation and Model-Based Design for dynamic and embedded systems. Simulink provides an interactive graphical environment and a customizable set of block libraries to design, simulate, implement, and test a variety of time-varying systems, including communications, controls, signal processing, video processing and image processing [16].

The following guidelines were used in the construction of the simulator

<sup>1</sup> [www.mathworks.com](http://www.mathworks.com)

- The simulator should be simple and user friendly
- The simulator should be reliable and validated as much as possible
- The simulator should take in design premises from a torpedo shaped AUV
- The simulator should only have a few or none unexplainable constants that only fits special set-up of the AUV
- The simulator should be runnable through the GUI (Graphical User Interface)
- An option to run the simulator in batch mode where all design premises can be easily changed without entering each block in current diagram should be available
- An option should be to save all output in the simulator to a chosen folder
- An option should be to export all parameter in the simulator to a given file or folder
- An option should be to import settings from different boat
- An option should be to run the simulator in loops for, e.g., 1000 different design values
- An option should be to plot each output of the simulator.
- An option should be to set trajectories as input to each DOF.
- Use Matlab Simulink to do the simulator, mainly due to its flexibility and reliability.

## 3.2 Overview of the Simulator

Taking into account the design premises from section 3.1, the simulator was built in a plain block diagram environment where each differential equation for each DOF was implemented. The simulator structure is based on four non inter-acting subsystem, each corresponding to a separate DOF. We then combine those four models to one complete and user friendly model. As an alternative the user can select the DOF he's interested in by simply triggering its subsystem button as well as triggering two, three or four buttons if he's interested in more systems at a time. During each run all output exports automatically to the workspace where it can easily be processed further. The main GUI is shown in Fig. 3.1, and the four non inter-acting subsystems for, surge, sway, heave and yaw are consequently shown in Figs. 3.2, 3.3, 3.4 and 3.5. We will discuss each element of the simulator in the following sections.

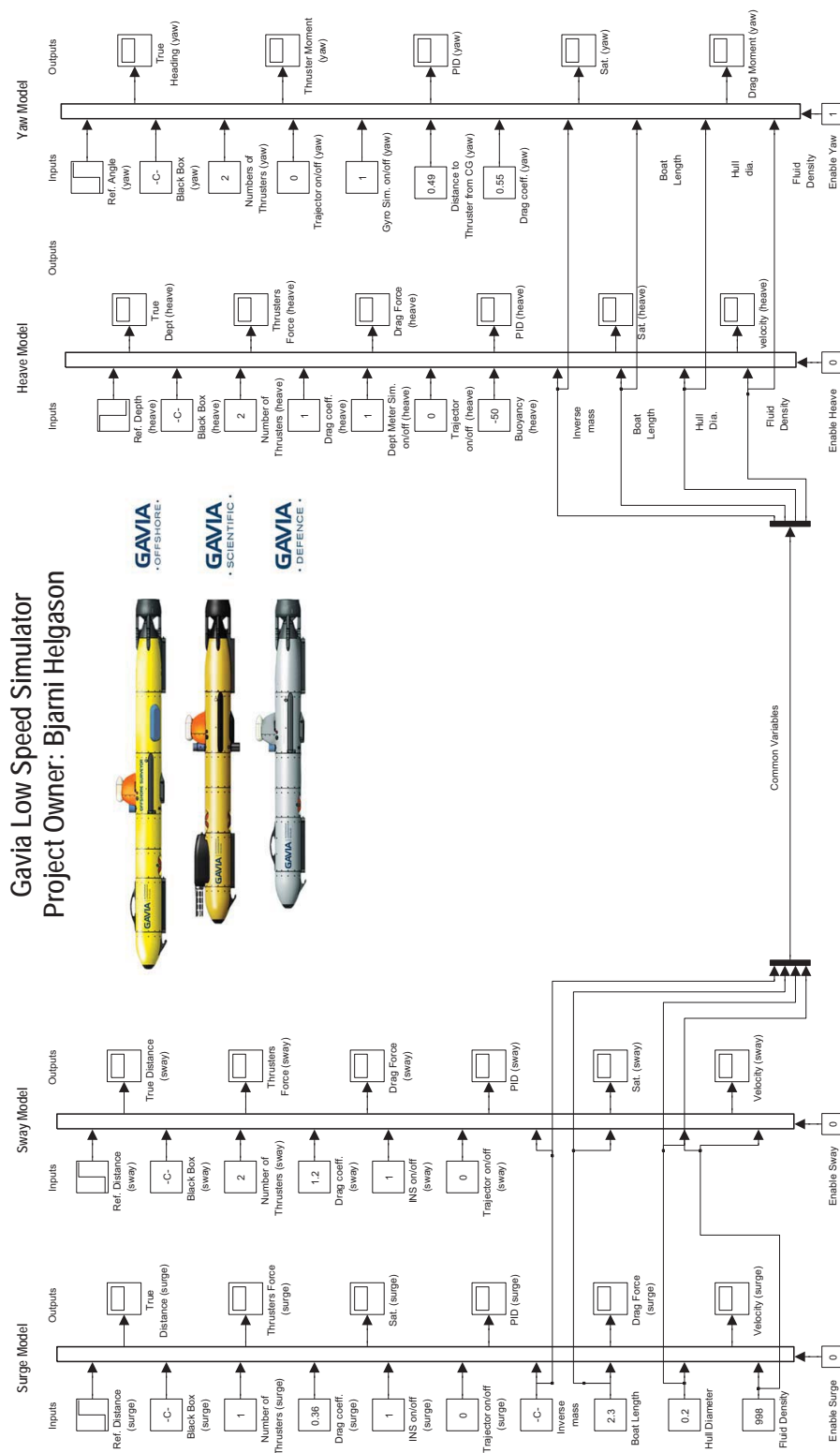


Figure 3.1: Graphical user interface of the simulator, closed loop

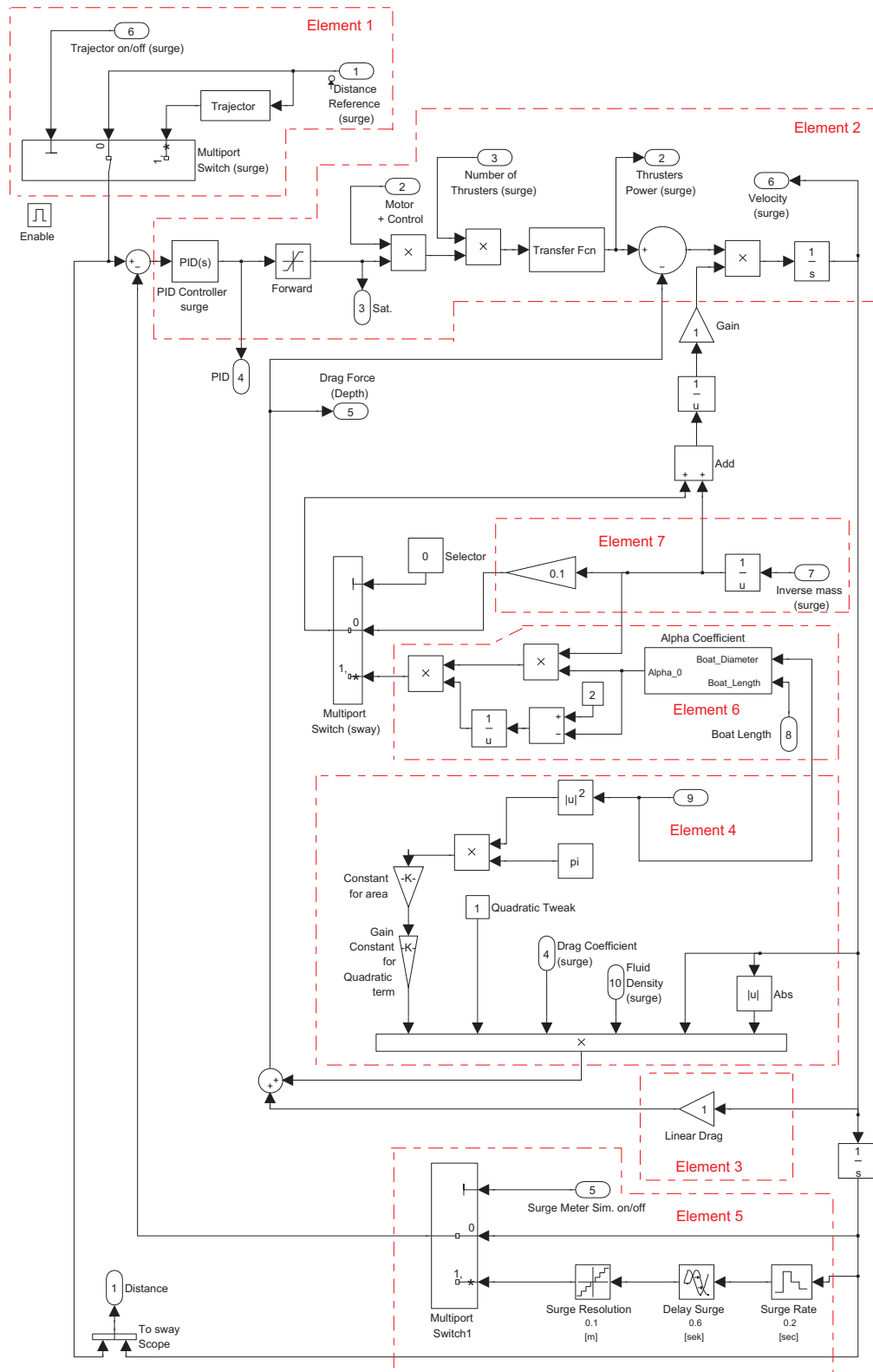


Figure 3.2: Surge model, closed loop

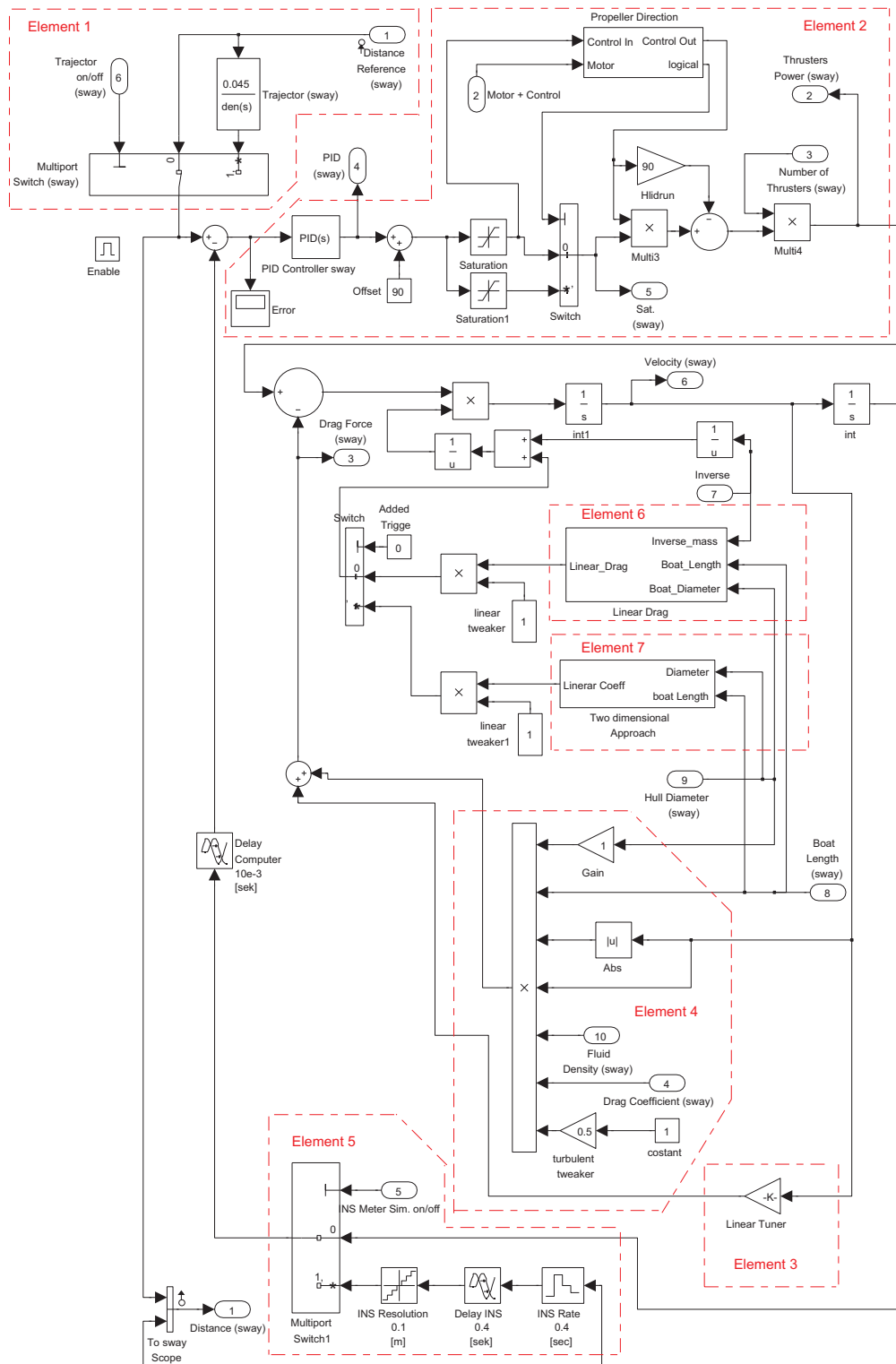


Figure 3.3: Sway model, closed loop

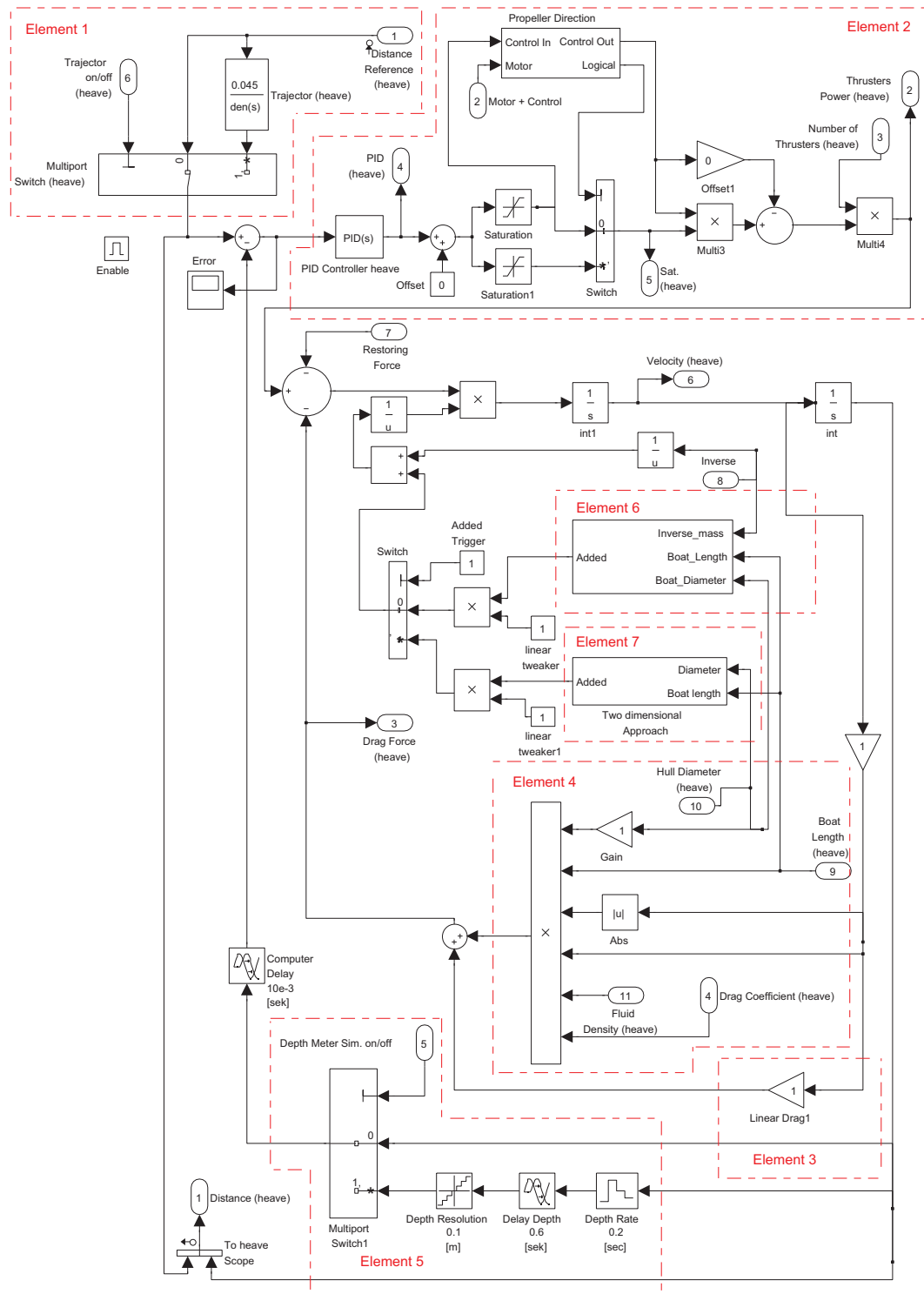


Figure 3.4: Heave model, closed loop



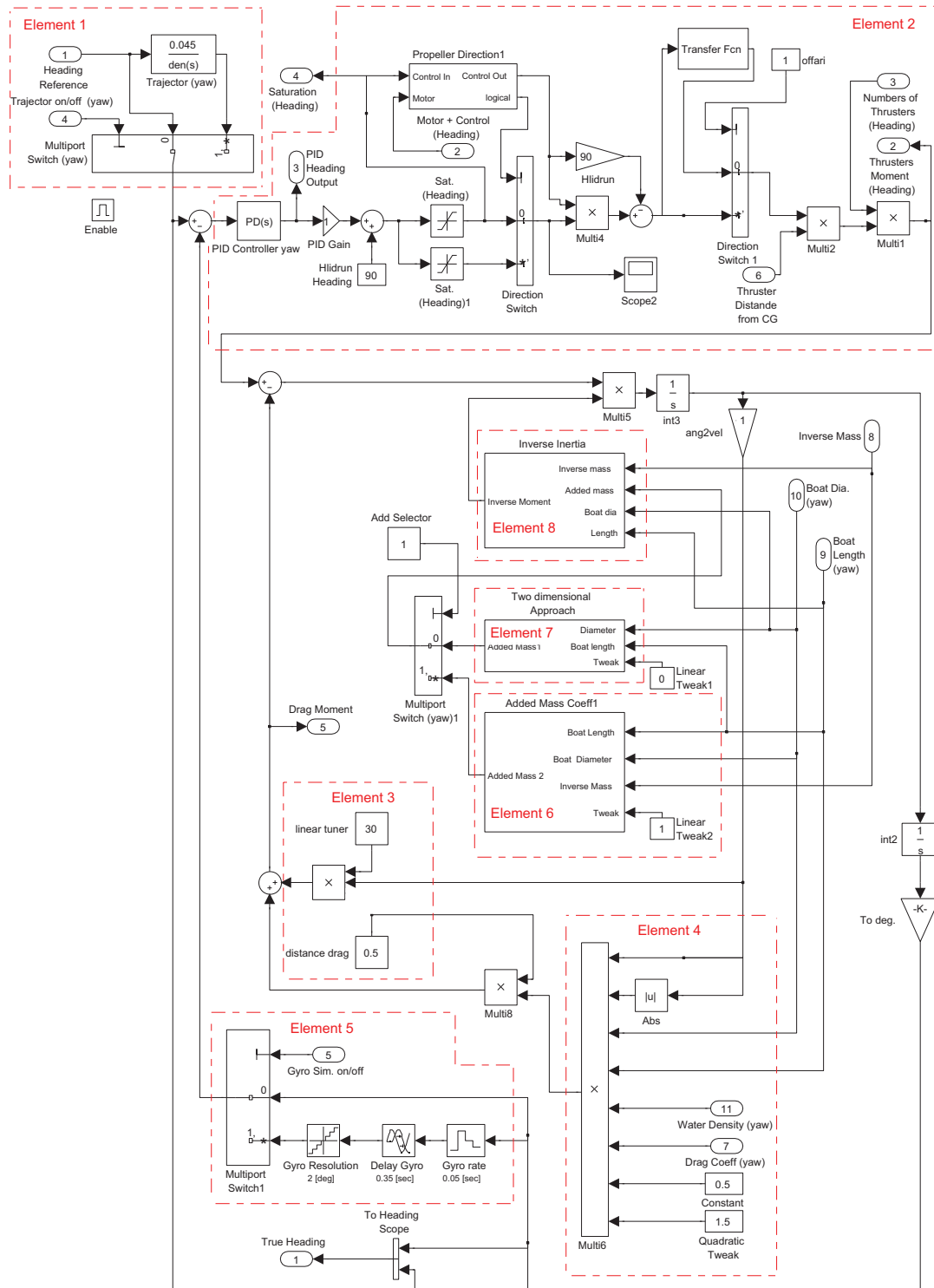


Figure 3.5: Yaw model, closed loop

### 3.3 Implementation

The equations of motion for the four DOFs, derived in Chapter 2, need to be treated and implemented in a certain way in the simulator. Block implementations for each DOF are similar, but differ slightly. During the next subsections we will carefully explain implementation of each model. For the introduction for the main concept we will start, by looking at general simple block diagram, Fig. 3.6, and then go through each implementation.

Figure 3.6 has several coloured blocks. The green block represents the input function, various input signal can be selected. The black block, represents the PID regulator. A normal PID regulator is not necessarily the ideal regulator for an AUV, but for simplification it will be used here. It is easy to change and see how other types of regulators might work for current set-up after the simulator has been validated. Speed control system, with inner loop PI control system, and outer-loop velocity control system, is an example of other type of regulator.

The red block is the motor and controller block. It can be any type of motor and controller specified by the user. One way is to find the propeller force as function of given input signal by doing a linear estimate of its relationship at a given design velocity. Another way is to find transfer function for each part of the thrust equipment, e.g., regulators, control equipment, motor, gear and propeller. Those transfer functions can then be combined to find the exact transfer function for this block. The former option is used in this work. On board Gavia there is thrust equipment for the surge (forward / backward) motion. The thruster is powerful, intended for cruising speeds and not for low velocity manoeuvring. Teledyne Gavia already had specification of their own thrust equipment. Unfortunately the working range we required for low velocity is on the non-linear part of that curve. The thrust given by the propeller as a function of the propeller shaft rotation is shown in Fig. 3.7.

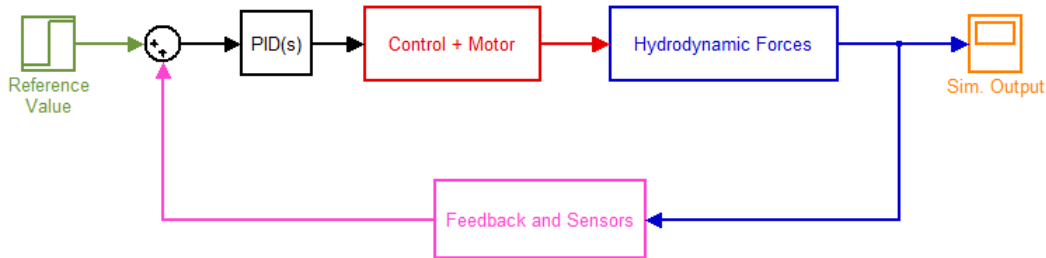


Figure 3.6: Basic block set-up used through all four decoupled degrees of freedom

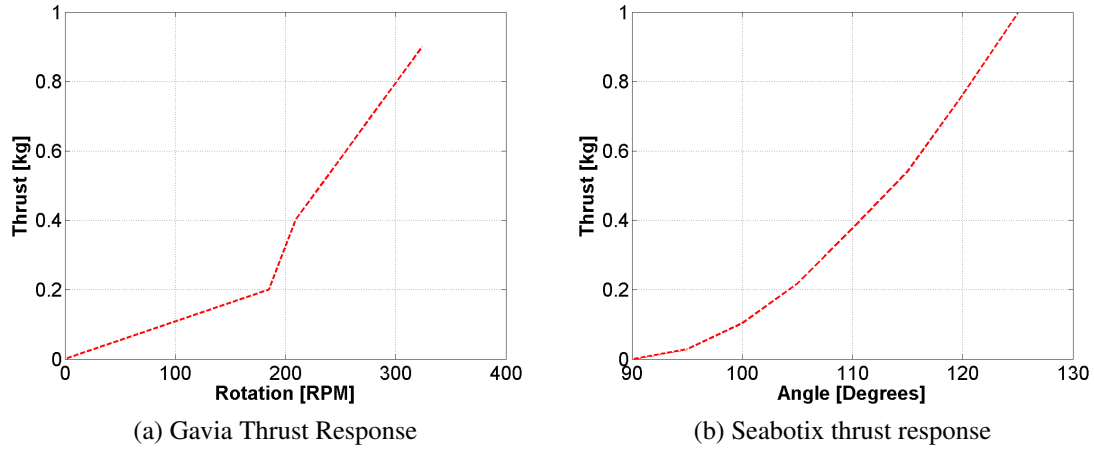


Figure 3.7: Available thrusters response as function of input signal to motor controllers

Thrusters from Seabotix<sup>2</sup> were used for other DOFs, its specification is shown in Appendix C. Data similar to the Gavia thrust data, thrust force as function of test equipment inputs (degrees) was mapped and plotted. This was performed by rigging up the test equipment and measuring the thrust while the input signal was fixed. The thrust was measured with an electrical force meter. The angles inputs to the test equipment were on the interval  $0 - 180^\circ$  where  $0^\circ$  represented full backward thrust,  $90^\circ$  represented stop position and  $180^\circ$  represented full forward thrust. The Equipment that was used is further discussed in the experiment set-up, Chapter 4. Results are shown in Fig. 3.7. Both slopes from Figs 3.7a and 3.7b are then the input to the thrust block of current DOF being analysed.

The blue block in Fig. 3.6 represents the hydrodynamic forces. These forces are difficult to simulate due to their non-linearity. We will implement the three of them in this work; drag due to laminar flow, later called linear drag, drag due to turbulent flow, later called the quadratic drag and the restoring force caused by buoyancy. Other hydrodynamic forces are due to; surface waves, environmental forces, oceans currents and wind, each depending on location and the environment. These forces are easy to add but should only be applied if designing AUV to, e.g., location with known oceans currents or other known environmental causes. In the previous chapter two methods of finding the added mass are explained. Both methods are implemented for each DOF in the simulator. The selected method is set by a trigger. Quadratic drag is also derived in the previous chapter and is implemented for each DOF in the simulator.

<sup>2</sup> [www.seabotix.com](http://www.seabotix.com)

The orange block in Fig. 3.6 represents the output of the model, in our case depth, degrees or distance. The magenta block represents the feedback loop. The feedback loop has an option to simulate a current sensor, as well as any delay in the feedback loop. Together these blocks represent an implementation of each dynamic equation of motion.

Various subsystems were used to combine those four models in one system. The main GUI (Graphical User Interface) is a set of four triggered subsystems which individually or together can be selected to run at the same time.

## 3.4 Common elements

Now we will look at the common elements of each DOF models in Figs. 3.2, 3.3, 3.4 and 3.5.

### 3.4.1 Input elements

Element 1 on all DOFs models in the simulator, shown in Fig. 3.8a, represents a step and trajectory input to the system. Those two input blocks are linked with a trigger. Generally a step function is not used. Usually an appropriate trajectory or a step response of a higher order is set as reference value because it overshoots less and is easier to regulate by the PID. By selecting a 2nd-order reference model

$$\psi_d(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \psi_r(s), \quad (3.1)$$

one can select a damping ratio and an angular frequency to attain a better response. Example responses are shown in Fig. 3.8b.

### 3.4.2 PID controllers and thrusters

Part of the AUV dynamics is carried by Element 2 and is shown in Fig. 3.9. The PID controller is first to the left. The sway, heave and yaw model are controlled with Arduino Mega, using inputs, as mentioned previously, from  $0 - 180^\circ$ . This causes a need of the adder that adds  $90^\circ$  to signal from PID controller for a correct zero. The difference between the real value and the expected value is the input to the PID controller, therefore if the difference is zero, which means that expected value is reached, then the thrust out should be zero. By adding those  $90^\circ$  to zero we assert that the motor controllers keep zero

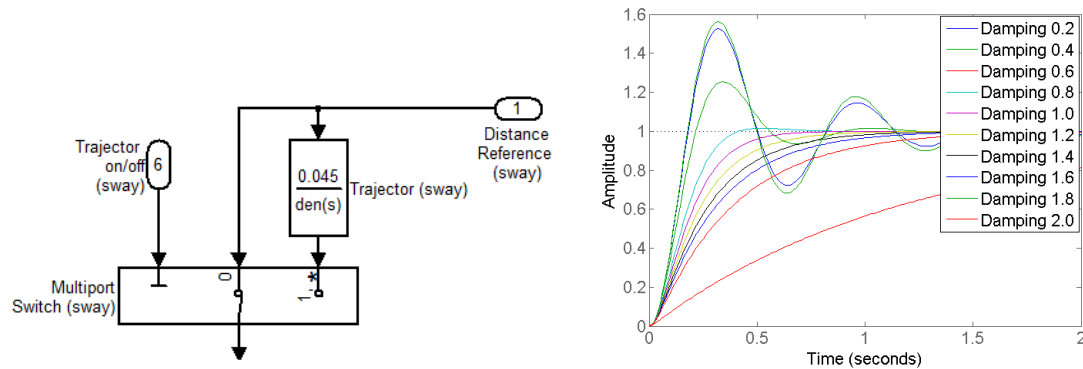


Figure 3.8: Option of higher order input function

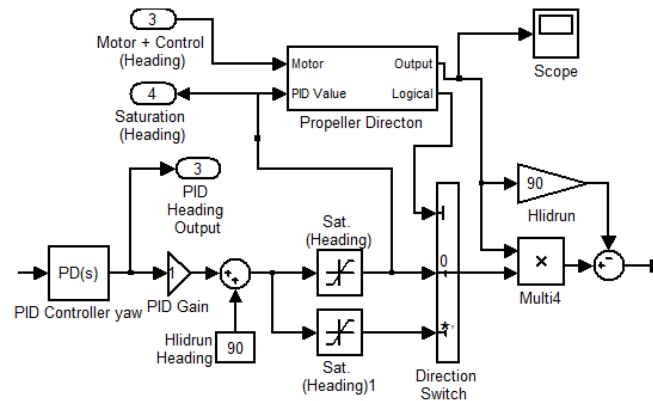


Figure 3.9: Element 2: PID and thrust implementation

thrust from the thruster, because  $90^\circ$  means stop. The saturation block is next to the adder and assures the control signal does not go above or below a limited value. By making the thrust curve linear, as in Fig. 3.7, a limit has to be set. A selector that chooses the correct slope to the motor and control block comes next. The selection depends on the propeller direction. Generally many thrusters don't have exactly the same thrust force curve in both directions, which means that the slope on the given design interval, as in Fig. 3.7, is not the same. We use this feature to limit thrust for backward motion of thruster due to the set up of the validation equipment (Chapter 5). The subsystem that controls the logical value to the switch is named Propeller Direction. Its implementation is shown in Fig. 3.10, where input 1 is the control signal to motor controllers, an if-elseif-else block then sees if the signal indicates forward or backward motion. The choice of the current subsystem depends on the direction which controls the output. Input to all subsystems here is the thrust slope, each subsystem is then capable of modifying the thrust slope, depending on

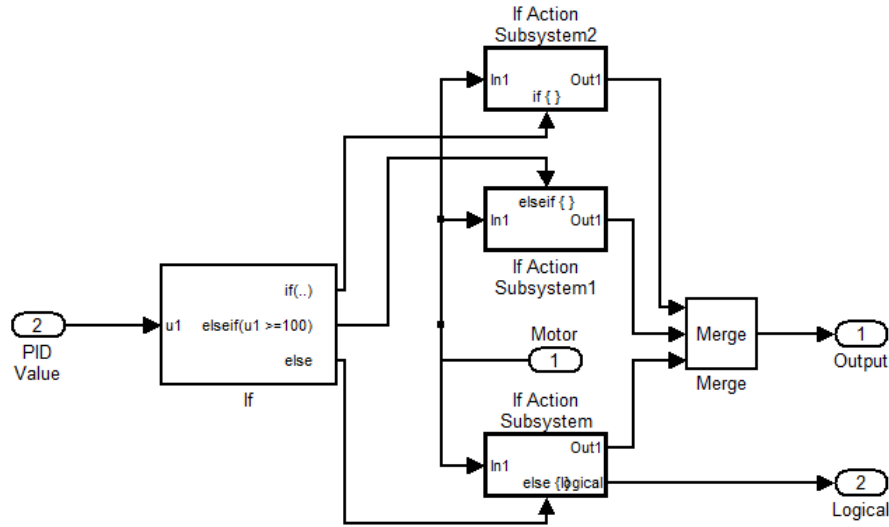


Figure 3.10: Propeller direction implementation in element 2

the user configuration. Another function the subsystem can also do is exporting logical value 0 or 1 to control the switch that controls which saturation block to use. This also means that the thrust can be limited there. Another implementation due to the Arduino Mega angles is next and to right of the switch. If, e.g., it is  $120^\circ$  forward signal then we can not multiply that number by thrust slope because value for the slope is only  $30^\circ$ . The subtract block then corrects the thrust to given Arduino Mega input.

### 3.4.3 Added Mass

Elements 6 and 7 estimates the added mass. In previous chapter two different methods are described to estimate that term. Methods are implemented differently between each DOF and will be discussed separately under each DOF in next sections.

### 3.4.4 Drag

Both linear and quadratic damping is included in all models. Element 3 carries the linear part and element 4 carries the quadratic drag. Both are implemented similarly in each DOF. However the area calculation and drag coefficient for the quadratic drag differ in each DOF. Figure 3.11 shows implementation on the quadratic drag part. Due to the non linearity in the general drag formula, where the velocity is in the power of two, the velocity is multiplied by the absolute value of itself, shown in the lower right hand corner [9, 11, 17]. The density of water the sub is travelling in comes next. After that comes the drag coefficient, quadratic tweak constant and finally the area. During high velocity

the quadratic term is dominating and vice versa. The quadratic tweaker is considered an adjustable parameter if one wants to reduce or neglect that term during low speed modelling.

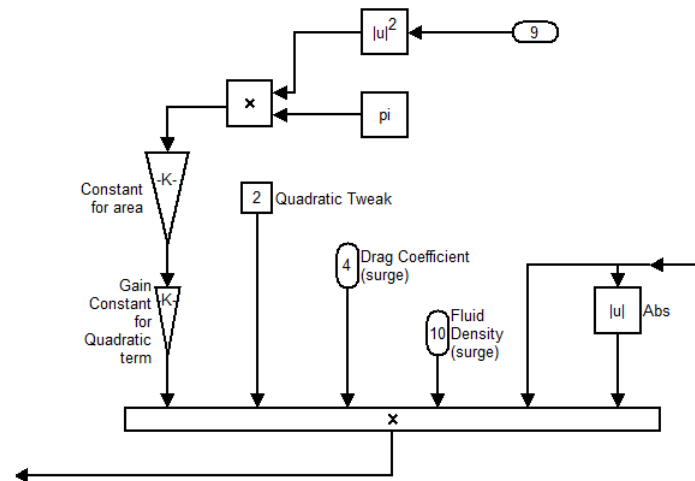


Figure 3.11: Element 4: Quadratic drag implementation

### 3.4.5 Sensors

Element 5 carries the feedback loop that also carries an implementation on measuring equipment available, see Fig. 3.12. The Zero-order hold block is the first block to the right, which holds its input for the sample period the user specifies. This simulates the rate of the measurement equipment. The transport delay is the next block to left, which simulates delay time required by sensor to handle data. The quantizer is next. It passes its input signal through a stair-step function which simulates the resolution of current sensor. Those three blocks combined implement a simple sensor.

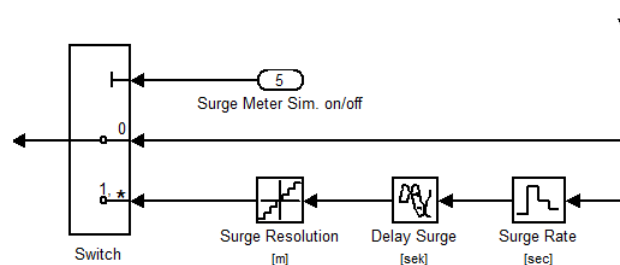


Figure 3.12: Element 5: Measuring meter implementation

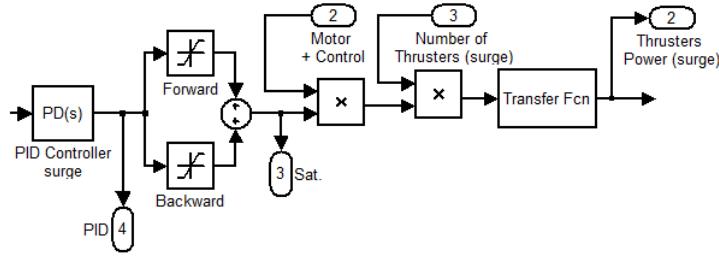
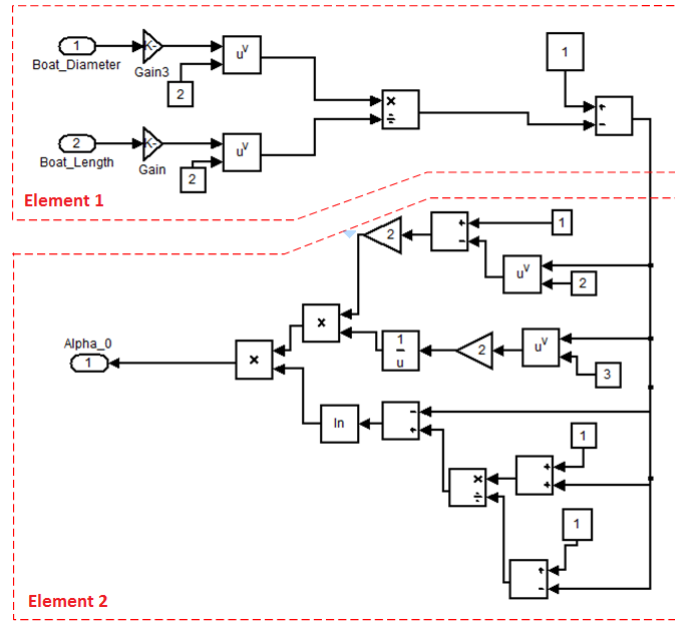


Figure 3.13: Surge thrust control implementation

Figure 3.14: Implementation on coefficient  $\alpha_0$  (Eq. 2.20) for surge direction

### 3.5 Specific Implementation in Surge Model

The surge model is shown in Fig. 3.2. Here the input signal is revolutions per minutes on the propeller shaft, which means there is no offset like for the Arduino Mega set-up, discussed earlier. This means that plus control signal indicates forward motion while minus control indicates backward motion. The saturation block has a limit for plus and minus control signals. The implementation is shown in Fig. 3.13. The other blocks in this element are similar to same elements of the other three DOFs model, excluding the transfer function block on the right hand side. It is a function that was added during the validation process and will be discussed further in Chapter 5.

Two methods are available to estimate the added mass and a trigger that chooses between them. Element 7 on Fig. 3.2 represents the added mass based on the two dimensional method (cf. Section 2.3). Element 6 on same figure carries the ellipse method (cf. Section 2.3) to the added mass. This approach is more complicated than the two dimensional one.



Figure 3.14 shows a subsystem that finds the necessary constant  $\alpha_0$  (Eq. 2.20) to calculate the added mass. This subsystem uses the boat diameter and the boat length to estimate its coefficient. Element 1 in that figure is implementation on eccentricity of the ellipse but element 2 uses the eccentricity to calculate the coefficient  $\alpha_0$ .

### 3.6 Specific Implementation in Sway and Heave Models

Here we discuss the sway and heave models (Figs. 3.3 and 3.4). The form of a torpedo shaped AUV is nearly symmetric for sway and heave movement. The main difference is the conning tower which generates drag moving in sway direction. The conning tower on Gavia AUV (see Figs 1.1-1.3) is small and we will use same set up during modelling for sway and heave movements. The only difference is that we will use buoyancy force during heave movement that does not affect sway movement. Element 7 represents the added mass estimated by two dimensional approach and element 6 represents the added mass based on the ellipse approach. This element, similar to surge, has a subsystem to find a coefficient, as well as complete, implementation of the added mass. Its implementation is shown in Fig. 3.15.

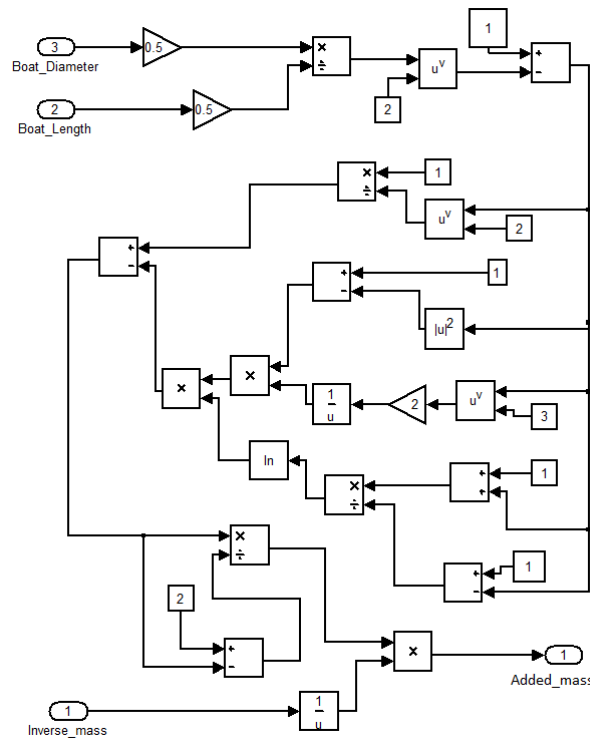


Figure 3.15: Implementation on coefficient  $\beta_0$  (Eq. 2.21) and added mass term for sway and heave directions

### 3.7 Specific Implementation in Yaw Model

Here we discuss the yaw model (Fig. 3.5). Element 6 represents the two dimensional approach (cf. Section 2.3) to the added inertia, it is a subsystem shown in Fig. 3.17. Element 7 represents the ellipse approach (cf. Section 2.3) to the added inertia. As seen it also carries both subsystem, used previous to calculate coefficients  $\alpha_0$  and  $\beta_0$ , shown in Fig. 3.16. Element 8 is a subsystem used to estimate the mass moments of inertia about the rotating axis. It is assumed that the AUV is a circular cylindrical shell without a conning tower. The element also adds the added inertia given from elements 6 or 7.

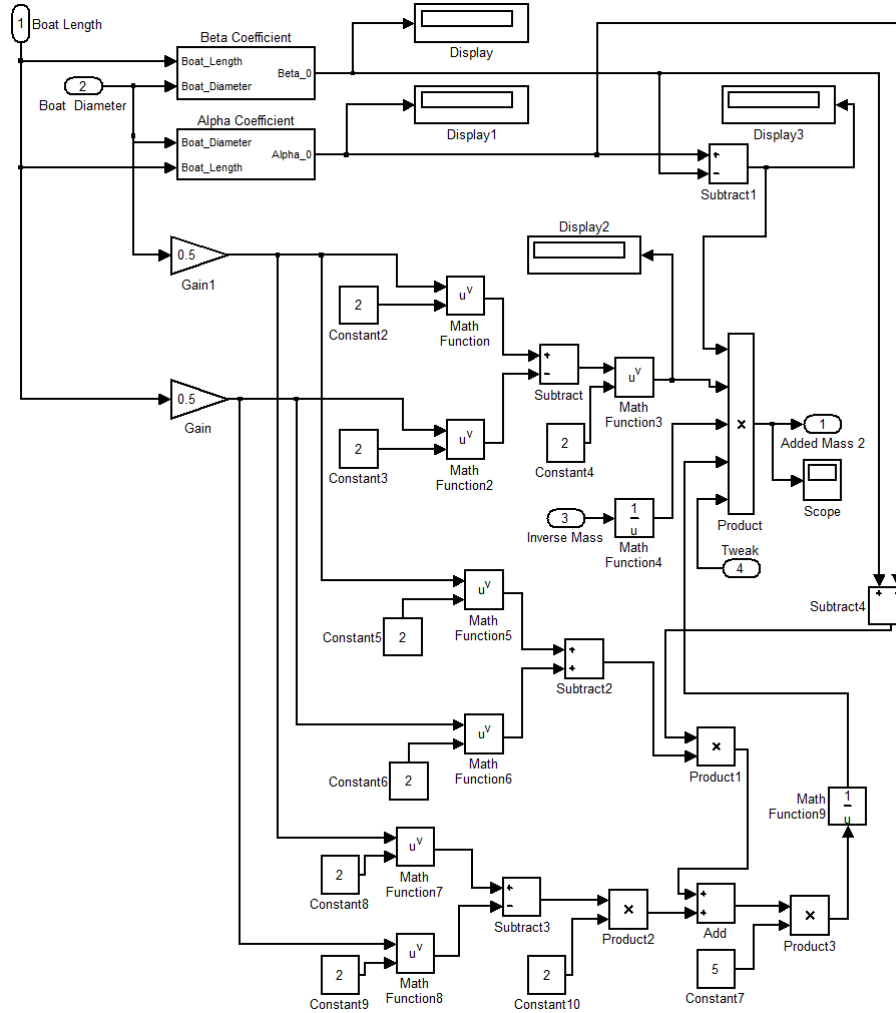


Figure 3.16: Implementation on added inertia by the ellipse approach (cf. Section 2.3) for yaw rotation

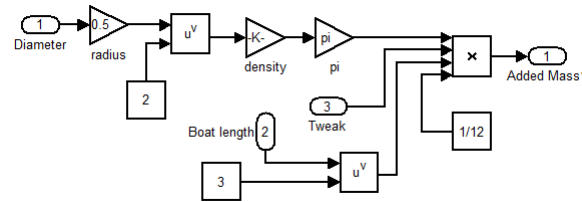


Figure 3.17: Implementation on added mass by the two dimensional approach (cf. Section 2.3) for yaw rotation

### 3.8 Open Loop Version of the Simulator

An open loop control system is a control system without any feedback from sensors. Open looped systems are the simplest available form of control and do not contain any limiters. By inserting an input signal for a given time to a AUV open loop control, the dynamics of the vehicle tells the system where it is at all times. Also an Open loop simulator with all of the closed loop parameters was designed. All earlier documentation about the closed loop simulator further explain the functionality of the open loop simulator. All block diagrams of the open loop simulator, as well as its main GUI, can be seen in Appendix B.

### 3.9 Running the Simulator

Simulink<sup>3</sup> has a graphical user interface, so every parameter can be modified by double clicking its block on the display. This process can be time consuming as well as complicated for multiple simulation. It can also be difficult to focus where each value is located, if the system is complicated.

A script Matlab code was written parallel to the design of this simulator. The purpose of the code is to make the simulator more user friendly and more robust. The code basically consists of a case loop, with option to run the simulator by selecting a DOF, read all values from the simulator and write them to file as well as load each AUV configure from file to run the simulator with new settings. For each mode mentioned above, the user has multiple choices.

In the running mode the user can select a number of triggers and settings. Most triggers are only logical triggers with value 0 and 1. This process can best be described by Fig. 3.18. First a dimension is selected with a trigger, next the reference value is set and finally the simulation time is selected. Next the user is presented with an option to set a trajectory

<sup>3</sup> [www.mathworks.com](http://www.mathworks.com)

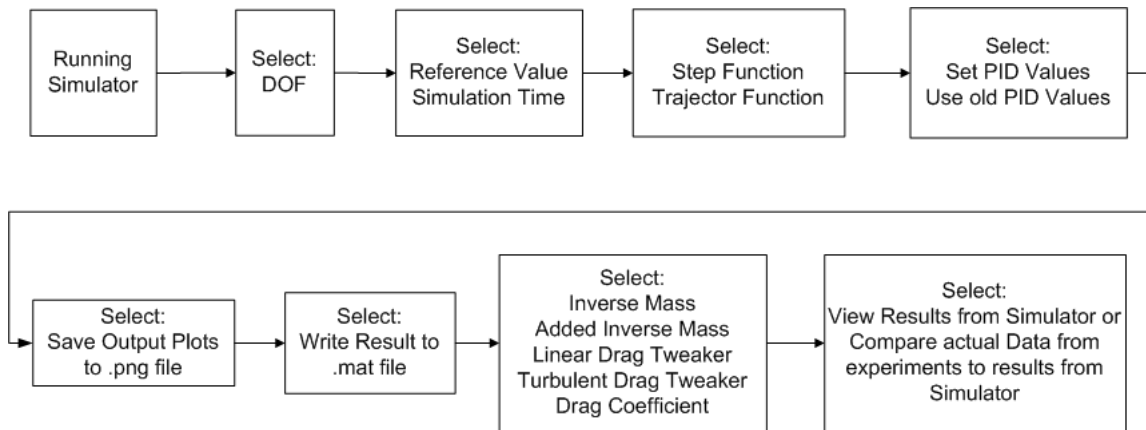


Figure 3.18: Selection required going through running process of simulator in batch mode

trigger, if not set, the model uses a normal step function. An option to set new PID values for the simulation run is set by a trigger. Then the user can select whether to save the results to a plot or data file. Finally there is a trigger that chooses between a plain output or a comparison to data. For comparisons the user must have experimental data available for input to the simulation.

This script has multiply options all though some of them are not fully implemented. A very useful option allows the user to run 1000 of different values overnight using a for loop. All data is then saved and can be processed further in the following days.

Another important feature of the script is that it returns a multi-plot, the user does not have to go to every scope in the simulator and does not have to scale or adjust the scope every time. The script output is always as expected, with labelled axis, title, legend and so on. Figure 3.19 shows an example of running one iteration through the batch mode. Users can adjust the sub-plots to fit his use of the simulator. This is just an example set-up and is the default output. Further simulator results are shown in Chapter 5.

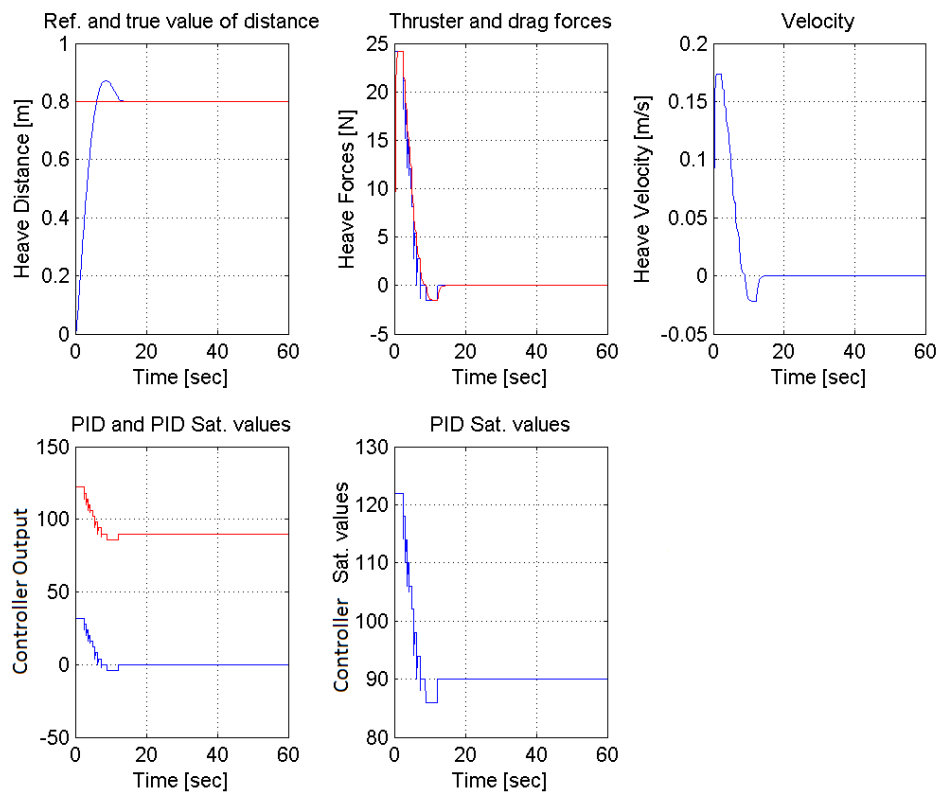


Figure 3.19: An example of running the batch mode script for one arbitrarily type of settings

### 3.10 Parametric and Optimization Features of the Simulator

An important option of this simulator is the access to all data and results via batch mode. It allows the user to implement optimization runs of various parameters. The validation process is an example of such use.

The optimization is not a part of this work but due to its importance it will be discussed here briefly. Matlab offers a variety of algorithms for standard and large-scale optimizations. These algorithms solve constrained, unconstrained, continuous and discrete problems. The optimization can be used for portfolio, device selection and data fitting. Variables during optimization can be physical parameters from the AUV such as length, mass and thruster sizes. Constrains maybe cost, manufacturing limitations, timing requirements, drag, speed and propeller.

Figure 3.20 shows an example of use during the validation process of the simulator. The main objective is to find the values of parameters in the model that minimize the area between the two curves, i.e., the experimental data and the simulator output. Because that simulator responses are often of lower order than the real systems, a transfer function is added to their model for higher order response. Parameters to optimize are; angular velocity, damping factor and inertia. Such an optimization is difficult to do manually because it requires that user modifies more than one variable at a time during each iteration. Therefore, optimization algorithms should be used.

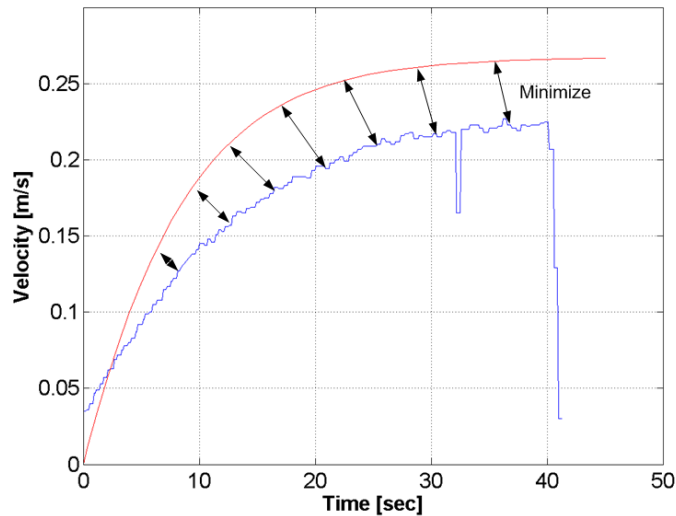


Figure 3.20: An example of minimizing area between two curves, one is from simulator and other based on real data

## Chapter 4

# Open and Closed Loop Experiments with the Gavia AUV

In this chapter, we will describe experiments performed to validate the simulator. We will also describe the experimental equipment, as well as the set-up for each DOF.

### 4.1 Experimental Approach

Validating an AUV simulator, that simulates certain movements, that the testbed AUV is not capable of doing such can be difficult. The set-up of Gavia AUV is like most other types of torpedo shaped AUVs, it has a propeller at the back controlling the surge direction, together with control planes that control other DOFs. The simulator assumes that the AUV is equipped with a couple of control thrusters controlling each DOF instead of the traditional control planes. Therefore, using Gavia AUV as a testbed requires that additional thrusters are added to control each DOF. The minimum requirement for surge is one thruster, and the minimum requirement for sway, heave and yaw are two thrusters.

There are two ways to validate the simulator, one is to design, build and install those thrusters as payload in the boat, make all control on board and do testing. Due to the time frame of this work it is not a feasible option. The other option is to implement those forces directly from the hull. That method was used here. The Gavia hull has a great design for adding all kinds of test equipment. Each module has 12 M6 holes around its circumference, which makes it easy to install test equipment or sensors on the hull. The only premises are that this equipment should be neutral in water and it should make minimum drag so it will not disturb measurements during experiments. Figure 4.1 shows

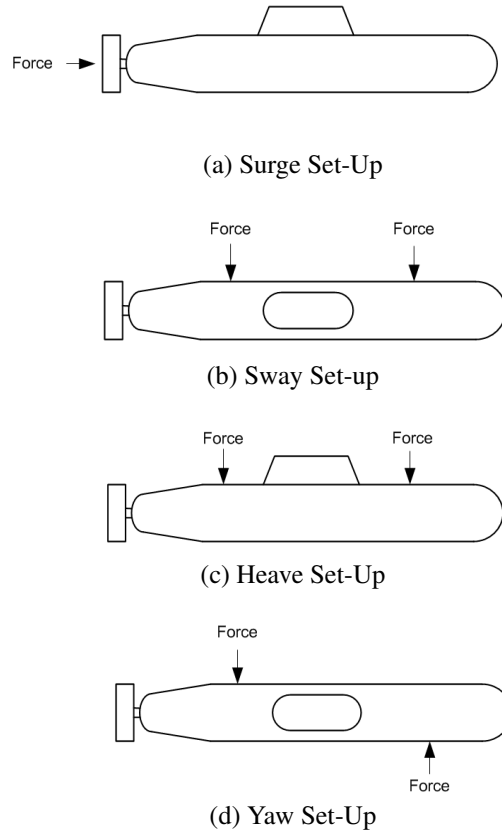


Figure 4.1: Implementation of forces to validate each DOF of model

the set-up used and the idea about adding forces directly to the hull to approximate the forces from onboard thrust equipment. With this method no modifications are required to the Gavia AUV. By locating the thrust equipment at the back of the hull in every run minimizes drag and should give a good response estimate. But this set-up requires one test at a time with only minimum equipment installed. Using this set-up also requires the AUV to be perfectly balanced to minimize any pitch, roll or rotation motion. The boat is designed to be neutral in water, but due to the test equipment it had to be balanced again during each DOF test.

For the experiments, we borrowed a Gavia AUV from the University of Iceland. That vehicle is a base vehicle with one extra payload. The payload is the doppler velocity log (DVL INS T24) module which is a multi-function commercial-off-the-shelf (COTS) acoustic sensor that provides highly accurate velocity information. The DVL specification is shown in Appendix C. In addition to providing speed over ground and speed through water, the instrument uses other sensors to provide position updates such as sensors giving depth, heading and acceleration. This version of DVL is not equipped with an inertial navigation system (INS). Due to the experimental set-up and the high component number (AUV, external thrusters, controllers, regulator, two laptops, tools, power cables and ether-



net cables) the experiments were complex and time consuming. A Number of experiments were made during this work where data was collected and worked on later. In Iceland there is no special facility to test AUVs indoor. For experiments two places were used both located in Reykjavik, Iceland. The first one is an indoor swimming pool. The length is 50 meters, the width is 25 meters and the depth is 2 meters. Tests were performed during night due to public opening all day. The other place was a small dolphin pool located at the Teledyne Gavia facility, it is a circle shaped pool with diameter approximately 4 meters and depth of 1 meter.

Unfortunately, the boat was leaced to Australia with a short notice. Not all necessary measurements could therefore be finished. But most of the scheduled measurements were made and gave good results. Due to technical problem regarding reading a value from the depth meter, neither open loop tests nor closed loop tests were performed for heave DOF. But because of symmetric and other properties regarding the sway and heave dynamics, all the data from the sway experiments will be used for the heave DOF. This will be discussed further during Chapter 5. One other measurement was also skipped, that was the closed loop test on the surge direction.

During all testing, open and closed loop, the test equipment needed to be relocated and adjusted for each DOF. It was important to place the thrusters cg precisely relative to the AUV cg. This displacement was later on also set in the simulator for comparison. The displacement of the thrusters also needed to be the same. Otherwise the boat lost heading for sway testing and pitch for heave testing.

## **4.2 Experimental Setup**

To perform the experiments, the experimental equipment had to be designed and fabricated. In this section, we will describe the experimental equipment, as well as the software developed to control the equipment.

### **4.2.1 Mechanical and Electrical Equipment**

The thrusters selected for the experiments were from Seabotix, type BTD150, shown in Fig. 4.3d. The BTD150 specifications are located in Appendix C. Those thrusters were mounted on the hull of Gavia with a custom designed bracket. The thrust equipment (thruster and bracket) must be easily installed and removed. And each bracket should serve testing for the sway, heave and yaw directions. Each thruster should also be ad-

justable on its bracket, e.g., it should be designed so it could easily be moved couple of centimetres to left or right. It should also be capable of pushing the boat below the middle of the hull, or near its centre of gravity to forestall any movement other than being tested. Figure 4.2, shows the bracket designed on a boat. The bracket is fastened with two bolts and can be fastened anywhere on each module perimeter. It also has two sets of mounting holes to select whether thrust is required to the middle of a hull or a little lower. And finally, it has a number of options to slide the thruster along the boat to adjust its distance from centre for all degrees of freedom. This bracket was made of aluminium and weighed 427 grams. Each motor weighs 705 grams, which means that total weight of each unit without an electrical cord is 1132 grams. To make this equipment neutral in water, 50 mm PVC pipes were added as shown in Fig. 4.2.

The Electrical equipment used to control and run the thrusters were

- Motor controller Devantech MD22 (Fig. 4.3a)
- Arduino Mega to control the motor controller (Fig. 4.3b)
- DC DC Mini-box regulator to supply controllers required voltage (Fig. 4.3c)
- Laptop with an autopilot to control the equipment mentioned above
- AC DC Transformer to supply power to the regulator and other smaller equipments.

The electrical equipment was installed and connected into an electrical control box. Figure 4.4 shows the equipments assembled and ready for use. Figure 4.4a shows where thrusters, PVCs pipes and thrust brackets have been assembled, Fig. 4.4b shows one thrust unit equipped with a thruster and a PVC pipe, Fig. 4.4c shows where all electrical circuits boards have been installed to the control box.

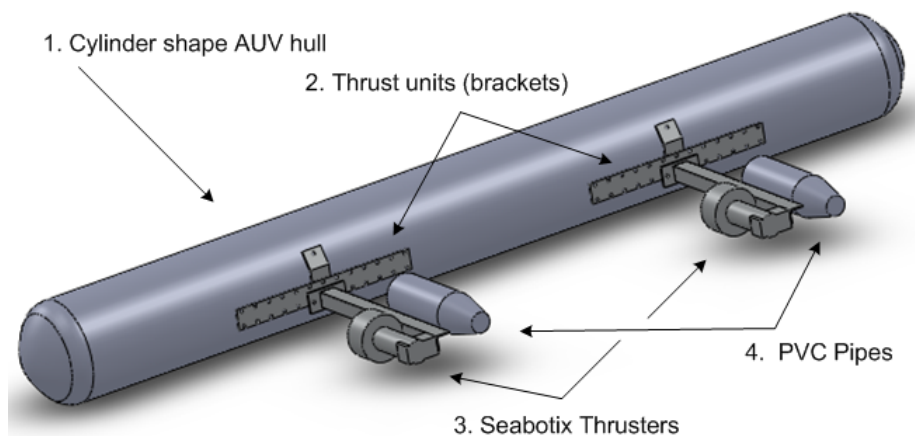


Figure 4.2: A sketch of the brackets and thrusters mounted on to the hull of the Gavia AUV

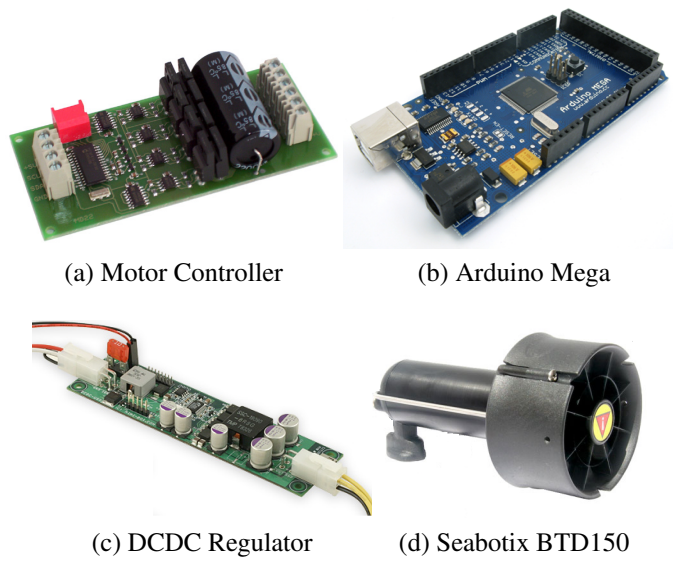
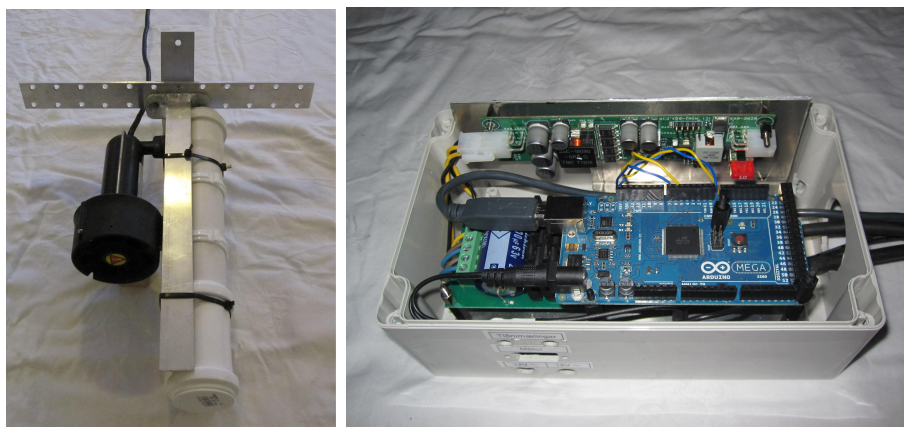


Figure 4.3: Thruster equipment



(a) Thrust units and control box



(b) Thrust Unit

(c) Inside the control box

Figure 4.4: Assembled experimental equipment

### 4.2.2 Computer Software

Robot Operating System<sup>1</sup> (ROS) was used as the software framework for the experiments. ROS is a software framework for robot software development developed by Stanford Artificial Intelligence Laboratory<sup>2</sup>. ROS is based on graph architecture where processing takes place in nodes that may receive, post and multiplex sensor, control, state, planning, actuator and other messages. Python was the main programming language and the ROS system was running on the Ubuntu Linux operating system during the experiments.

The experimental software stack was built up of two modules, which were the ROS modules and the Arduino module. ROS modules are codes in the ROS environment written in Python programming language, while Arduino module are code written in the C++ programming language and imported to the Arduino board. Arduino is an open-source single-board micro-controller, descendant of the open source Wiring platform, designed to make the process of using electronics in multidisciplinary projects more accessible. The Arduino is connected with usb and controlled with serial communication. The role of the Arduino was to translate the signals from the autopilot and forward them to the controllers controlling the thrusters.

The main ROS modules made were; Autopilot, Arduino communicator, Gavia link and Graphical User Interface (GUI). The Autopilot is a module that takes in requests about what state the Gavia should be in, e.g., reference value in each of the four DOF. The Autopilot also reads real values from the Gavia link module, which is a communication link between the Gavia and the Autopilot. Based on a real value from the Gavia link and a reference value chosen by the user, the Autopilot reacts according to the output of the PID controller. The PID controller is a part of the Autopilot. The Autopilot also has an option to log all data, both data coming from the Gavia link as well as data accessible through the ROS topics. The data is logged to a text file. The GUI module allows easier control of the boat during experiments. Through the GUI, the user can log values to a file, change feedback control constants, set reference values, read sensors values as well as turning the Autopilot on and off. The Autopilot publishes thruster data to a topic which then the Arduino communicator subscribes (listens) to. All thrust values during ROS modulus are numbers on the interval  $-1$  to  $1$ . The interval  $-1$  to  $1$  is a normalized interval where, e.g.,  $0.2$  means  $20\%$  of full power. The module Arduino communicator translates these values to a correct input for motor controllers controlling the thrusters.

<sup>1</sup> [www.ros.org](http://www.ros.org)

<sup>2</sup> <http://ai.stanford.edu/>

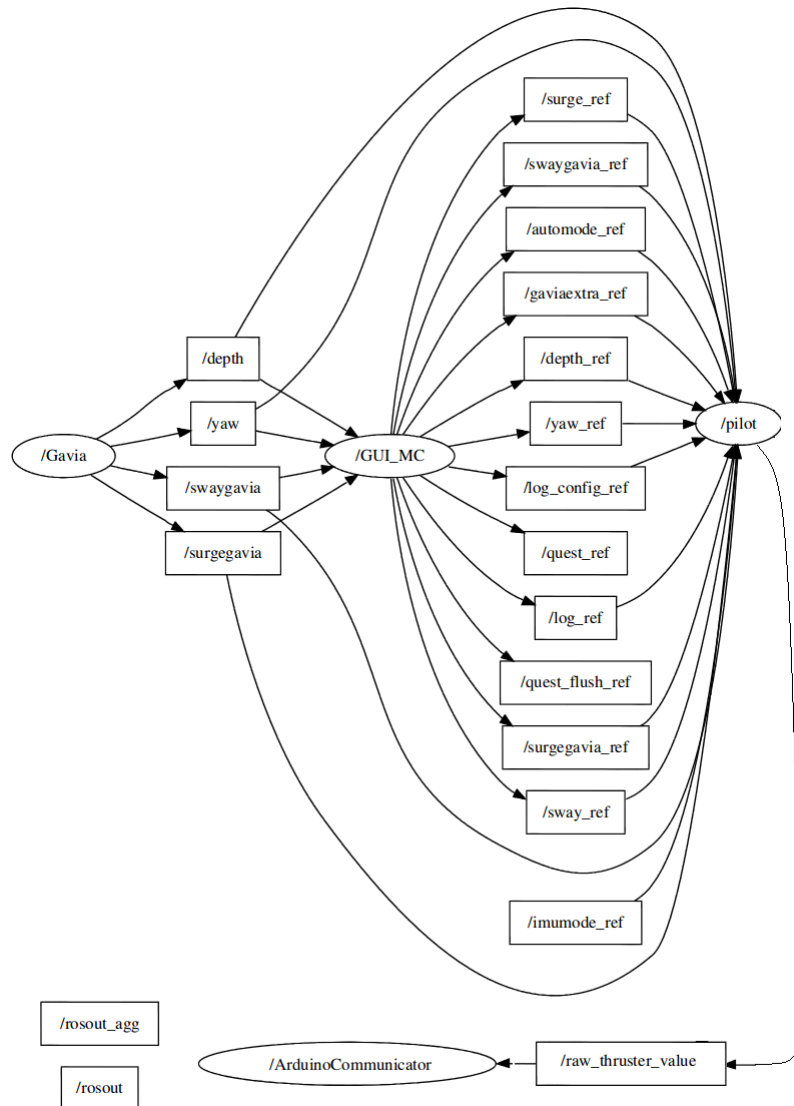


Figure 4.5: RX Graph from ROS command line

Rxgraph is a command-line tool for visualizing a ROS computation graph. Figure 4.5 shows the ROS computational graph of the designed software. Figure 4.5 shows boxes and ellipses. The ellipses represent modules in the script while the boxes represent the topics that the modules are publishing or subscribing to. The figure shows the Gavia link module labelled as `/Gavia`, the GUI module labelled as `/Gui MC`, the Autopilot module labelled `/pilot` and the Arduino communicator module. The Gavia link module publishes depth in meters to `/depth` topic, heading in degrees to `/yaw` topic, latitude in degrees to `/swaygavia` topic and longitude in degrees to `/surgegavia` topic. The modules subscribing those topics are `/GUI MC` and `/pilot`. The GUI uses those values to show the user current location, depth and heading, while the Autopilot uses those values to handle requests by the user. The GUI publishes surge reference value in meters to

/surge ref topic, sway reference value in meters to /sway ref topic, depth reference value in meters to /depth ref topic, heading reference value in degrees to /yaw ref topic, Autopilot requests to automode ref topic, log handling to log config ref and log ref topics, and a current location in degrees when a request to a reference value is set to /gaviaextra ref topic. The Autopilot subscribes all topics published by Gavia link and GUI and then publishes its final value to /raw thruster value topic.

Figure 4.6 shows the GUI. The upper part contains the log settings, where the user selects what to log and a log name. Below the log is the PID settings for each DOF. The bottom of the figure shows in the first column the four reference values set by the user, depth in meter, yaw in degrees, surge distance in meters and sway distance in meters. Below the reference values are option for a direct thrust settings to thrusters. In the right column is

Testbench for low speed modeling of Gavia AUV  
Bjarni Helgason

Enter the file name for logging  
test\_sway\_1.txt

☒ Log Thrusters

☒ Log Depth ☒ Log Surge

☒ Log Heading ☒ Log Sway

Start Logging Stop Logging

Logging status: none

SurgeDist PID	SwayDist PID	Heave PID	Yaw PID
P: <input type="text"/>	P: 15	P: <input type="text"/>	P: <input type="text"/>
I: <input type="text"/>	I: 0	I: <input type="text"/>	I: <input type="text"/>
D: <input type="text"/>	D: 0	D: <input type="text"/>	D: <input type="text"/>
<input type="button" value="commit"/>	<input type="button" value="commit"/>	<input type="button" value="commit"/>	<input type="button" value="commit"/>

Desiwhite values	Sensor values
Depth: <input type="text"/>	Depth: 0
Yaw: <input type="text"/>	Yaw: 0
SurgeDist: <input type="text"/>	Longitude : 0
SwayDist: 0.8	Latitude : 0
Surge: <input type="text"/>	
Sway: <input type="text"/>	
<input type="button" value="commit"/>	

Pilot Switch

Figure 4.6: Graphical User Interface (GUI)

information about location, depth and heading. In the lower right corner there are buttons to activate and deactivate the pilot.

During all the experiments, the system values were logged through ROS for processing. The first line of each log contained PID values, then the actual log started. If all information were selected for logging, then line 2 contained all thrusters values and time, line 3 contained real depth values, reference depth and time, line 4 contained real longitude and time, line 5 contained real latitude and time and then the same routine continues until the log was turned off.

## **4.3 Experimental Testing**

This section describes the experimental testing. Test cases are summarized in tables with the test values used for each run. All data logged during the experiments will be used to validate the simulator later on. A few examples of results are given here, but all results are shown in the next chapter where it is compared with the simulator results

### **4.3.1 Open Loop Tests**

Open loop testing means that the thrusters are driven without feedback at a constant value. All data from the measuring equipment were logged during this procedure. Open loop tests for surge were performed differently than for the other DOFs. To gain control of the Gavia AUV propeller one has to go through the Gavia Software located in the Gavia Laptop Computer. The Gavia software also has different ways of logging, all data are logged to XML files in a completely different way than it was set up for other DOFs. Desired values were the RPM at the thruster. Table 4.1 lists test settings used during surge experiments. Experiments for other DOF were performed with Robot Operating System (ROS), explained later in this chapter. Tables 4.2 and 4.3 lists test settings used during those tests. Figure 4.7 shows the Gavia being tested at the Teledyne Gavia facility.



Table 4.1: RPM values for surge

Experiment	RPM
1	187
2	197
3	203
4	218
5	242

Table 4.2: Thrust values for sway and heave

Experiment	Thrust Value
1	0.25
2	0.3
3	0.4
4	0.5
5	0.7
6	0.8

Table 4.3: Thrust values for yaw

Experiment	Thrust Value
1	0.2
2	0.4
3	0.5
4	0.6
5	0.9

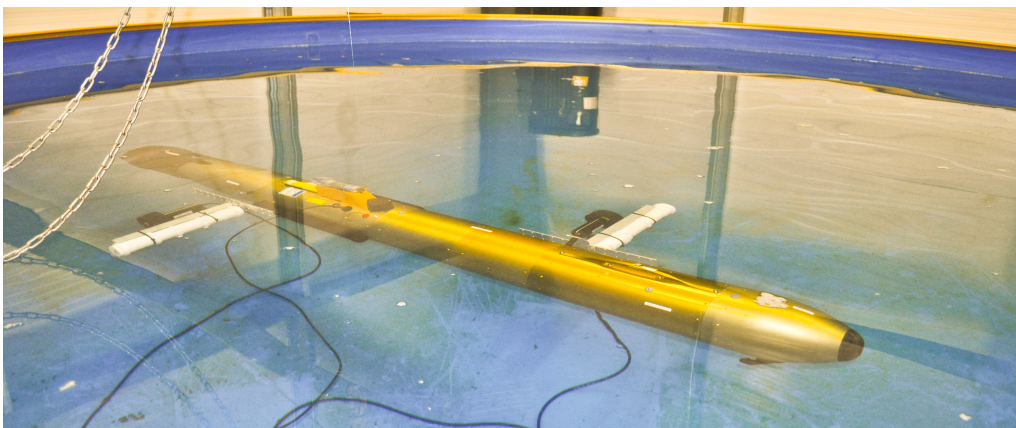


Figure 4.7: Image from yaw testing



### 4.3.2 Closed Loop Tests

Closed loop tests are tests performed with the feedback loop active. The PID regulator gets the error between reference value and true value. Based on that error it selects an appropriate value to control the thrusters. During those tests all data were logged for further analysis as for the open loop test. Tables below lists settings used for closed loop testing. Figure 4.8 shows the Gavia AUV being tested in the Reykjavik swimming pool. The PID values selected for the experiments were not considered the best values to control the boat, they were rather chosen as appropriated values to compare the behaviour while the boat was overshooting and oscillating about its reference value. For better performance, e.g., less overshoot and shorter settling time which is the time it takes for the system to converge to its steady state the Ziegler-Nichols [18] tuning rule can be applied.

Table 4.4: PID and ref. for sway and heave

Experiment	P-Value	I-Value	D-Value	Ref. [m]
1	30.0	0.0	0.0	0.8
2	30.0	0.0	0.4	0.8
3	40.0	0.0	0.4	0.8
4	40.0	0.0	20.0	0.8
5	50.0	0.0	0.0	0.8

Table 4.5: PID and ref. for yaw

Experiment	P-Value	I-Value	D-Value	Ref. [deg]
1	1.0	0.0	0.0	40
2	2.0	0.0	0.0	40
3	2.0	0.0	0.4	40
4	3.0	0.0	0.0	40
5	3.0	0.0	0.4	40

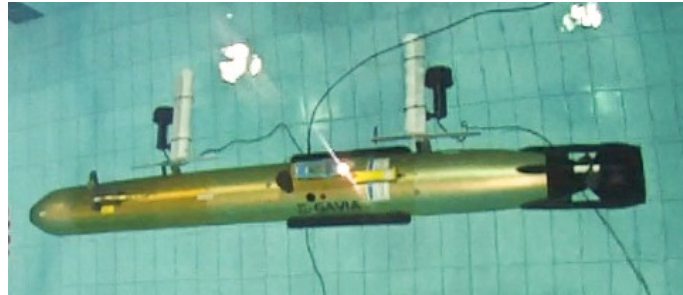
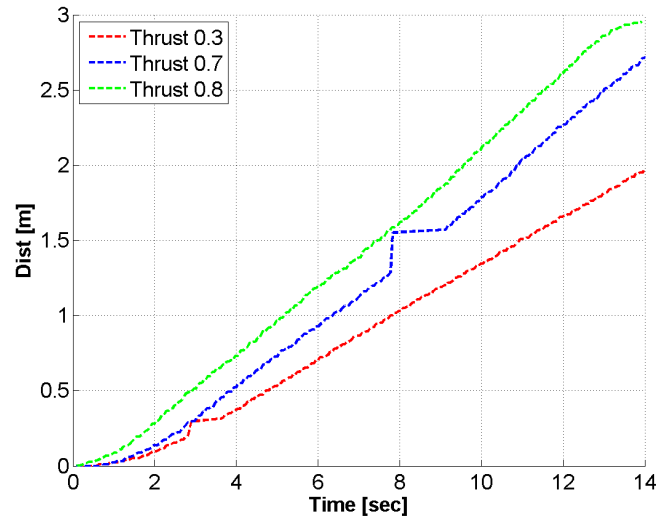


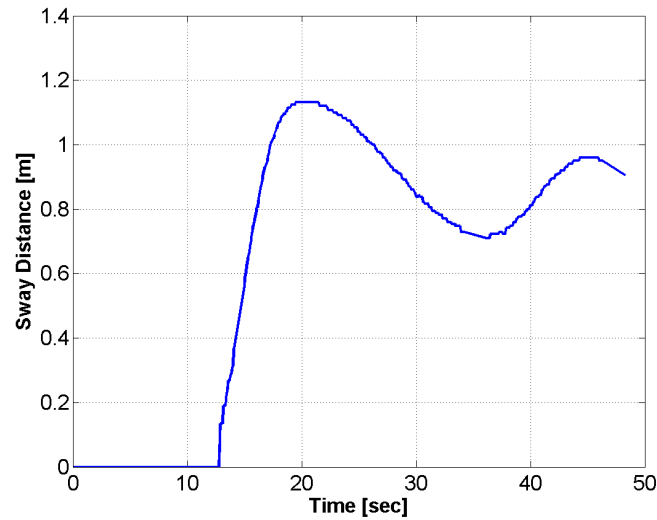
Figure 4.8: Image from sway testing

### 4.3.3 Example Results

Figure 4.9 shows an example of results based on both the open loop experiment and the closed loop experiment in sway DOF. Figure 4.9a shows results from three of the open loop sway experiments. Figure 4.9b shows a result from one of the closed loop sway experiments.



(a) Open loop sway experiment



(b) Closed loop sway experiment

Figure 4.9: Experimental example results

## 4.4 Discussion

The chosen method to generate forces required to move the boat was by adding an additional propulsion thrusters as mention earlier. This method has certain restraints due to the additional thrust units that are not included in the equations of motion already derived in Chapter 2. The additional units are though designed and located to affect the drag as little as possible. The units are located at the back of the hull during forward motion. The direction on forward and backward motion is shown in Fig. 4.10. In the open loop experiments, forward motion was applied but both forward and backward motion in the closed loop experiments. During backward motion the distance  $L$  in Fig. 4.10 causes part of the exhaust to hit the AUV hull which reduces the thrust. This function was implemented in the simulator by decreasing the slope of the propulsion equipment during backward motion.

The drag effect of the experimental units in the yaw DOF is more than for other DOFs. It is because larger part of the propeller units are located in the outer flow during rotation. As an alternative method for the experiments we suggest two different setups of the additional thrust units if further experiments will be performed.

One method is to use 4 units instead of 2 and locate them as shown in Fig. 4.11. This method will increase the stability of the AUV during movement. This method also requires that the drag generated by these 4 units should be should be accounted for in the analysis.

Another method is to make a complete thrust module that can be connected using the built-in Gavia Quick Lock system. This is a more complex method that requires no additional units outside the hull. The drag caused by the additional modules will now be in the from of a longer boat.

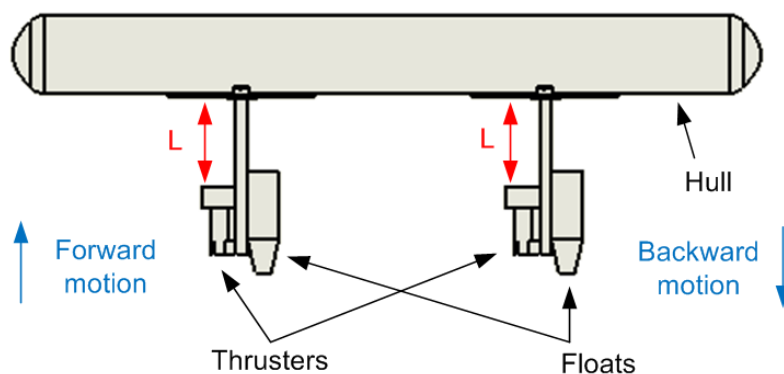


Figure 4.10: Layout of experimental equipment

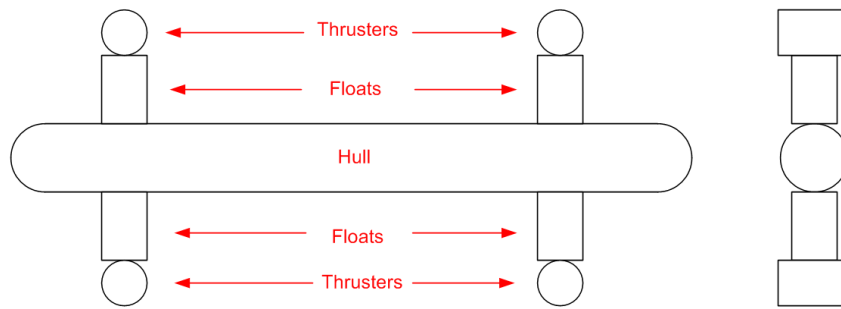


Figure 4.11: One suggested set-up of the experimental equipment if the experiments will be continued in the future

Figure 4.12 shows an example from the Delphin2<sup>3</sup> AUV thrust equipment mentioned in Chapter 1. The figure shows both vertical and horizontal tunnels where tunnel thrusters are installed.

The boat center of gravity is located under the center of buoyant. Roll movement during sway testing were minimized manually by adjusting the height of the point where the sway thrust forces were affecting the hull. The correct point was estimated little above the center of gravity due to the conning tower, which generates moment about the center of gravity during sway movement.

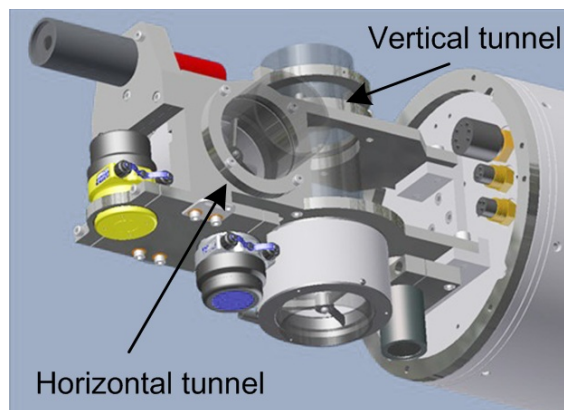


Figure 4.12: Horizontal and vertical tunnels in the Delphin2 AUV<sup>3</sup>

<sup>3</sup> <http://www.leo-steenson.com/delphin2/tslthrusters.html>, January 2012

## Chapter 5

# Model Validation and Tuning

Several experiments were performed to obtain data to validate the simulator. The experiments are described in Chapter 4. In this chapter, we are going to discuss the results and compare the data to the simulator data. Due to the amount of results, the plots discussed here are located at the end of this chapter, both open and closed loop comparisons.

### 5.1 Experimental Data Processing

Even though all the data were logged during the experiments, a lot of work and effort was put in the post-processing. All data from text files (log files) were imported and handled in Matlab. Values were read and sorted to a database, either for surge, sway, heave or yaw. The surge and sway data required more work than the other data due to coordinate work. To be able to request the boat to go 2 meters ahead, one has to calculate every position from the position when reference value was set. It means steady calculation of distance to the reference value. All data processing started by tracking the boat position in longitude and latitude coordinates. Figure 5.1 shows maps of the experiments performed during the open loop sway experiments. All the experiments started at the exact same location indicating a small drift in the measurement equipment. The distance between the first and the last experiment was a couple of meters and the time between them was approximately 30 minutes. The drift will be neglected due to the short time each experiment takes. Figure 5.2 shows a sorted data from one of the open loop experiment in Fig. 5.1. The distance between every set of latitude and longitude points were calculated and compared to log time to estimate the exact start and stop during the experiment. Figure 5.3 shows an example of such a work were a complete experiment is shown and unnecessary data from Fig. 5.2 has been deleted. The distance as a function of time is shown in Fig. 5.4. The

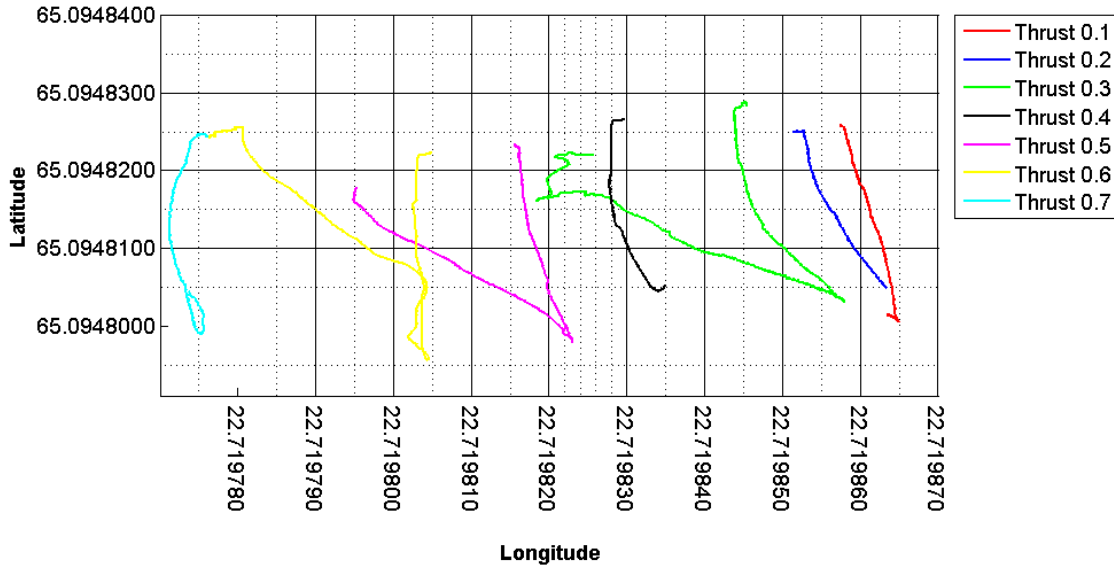


Figure 5.1: GPS tracking of open loop sway experiment

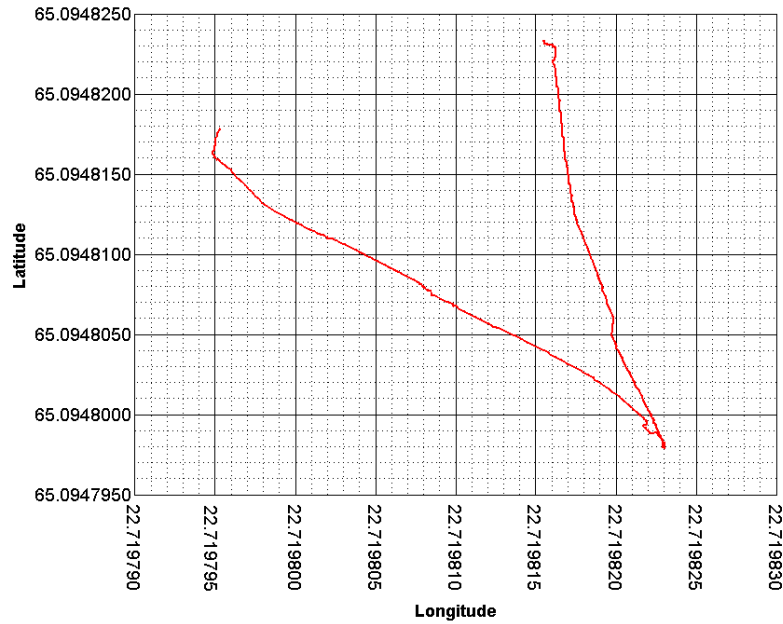


Figure 5.2: Original Data from one of the open loop sway experiments

same process was performed for the closed loop experiments were the distance between every latitude and longitude was calculated. When setting the reference value, a new zero of distance was made and all distance calculation calculated from that zero.

The remainder of this chapter is split up to four sections, one for each DOF. All the models are first compared to the results from the open loop experiments. When that comparison

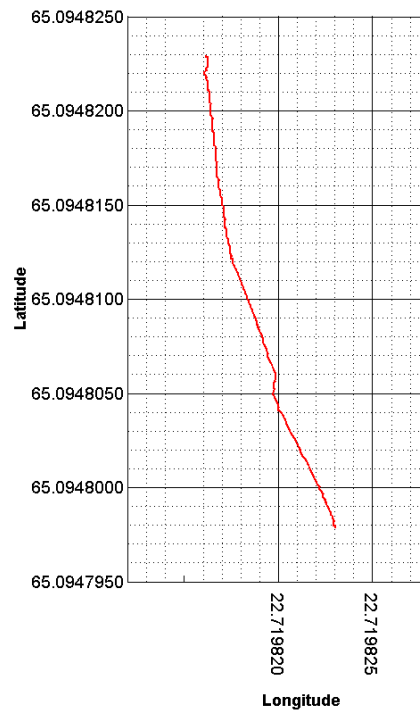


Figure 5.3: Post-processed data from one of the open loop sway experiments

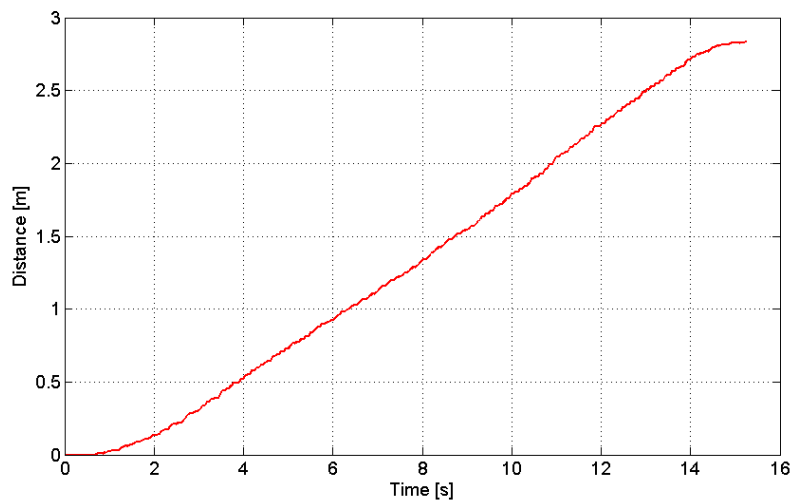


Figure 5.4: Final Data

is good, it will be compared to the result of the actual closed loop experiments. We may have to repeat this a few times to get good results. It soon became obvious that more delay was on the Gavia link during all the coordinate calculation than expected. The Gavia link, explained in Chapter 4, transferred all the data from the Gavia AUV to the computer onshore. The data communication was through a UDP link. The data was sent from the Gavia to the computer. Handling this data caused a delay varying from 0.2 and up to 0.4 seconds depending on what was being tested. This delay will be implemented

for each model with a transfer delay block due to the reason it strongly affects the output response in a closed loop comparison.

## 5.2 Surge Model Validation

The results logged during the open loop control of surge DOF were analysed. The added mass coefficient was found both by using two dimensional approach and the ellipse approach. A list of the surge results are given in Table 5.1.

The resolution in latitude and longitude was low in the surge experiments. Different experimental equipment was used during surge than for other DOFs. Data was logged through the Gavia Software interface to a XML data structure. We found out that each latitude and longitude sets were logged with one fewer digits than other logging during the work. The response made from the experimental data looked like a response from a sensor with low resolution which caused the response graph to have a ladder type of curve. By doing a curve fit to the data the response looked more like expected.

Figures with results, given by comparing real data to the simulator with a fixed propeller revolutions, from 187 RPM to 243 RPM are given in Section 5.6. Figures 5.6 and 5.7 shows how both linear and quadratic terms are underestimated, using the ellipse and two dimensional approaches without adjustments. It is easily seen that the data on the lowest RPM setting does not fit at all. The thrust data for the Gavia AUV is not available for low speed causing uncertainty about the sloped used. The lowest value possible during the experiment was 187 RPM.

Several adjustments were performed to correct the model. The results of adjusting the hydrodynamic parts; added mass, linear drag and non-linear drag, in a way of increasing the hydrodynamic forces, yielded a fit in the distance plots but not in the velocity plots.

Table 5.1: Comparison performed for surge DOF

Case	Results from:	Transfer Function	Comparison	Figure
1	Ellipse method	No	Location	Figure 5.6
2	Two dim. method	No	Location	Figure 5.7
3	Two dim. method	Yes	Location	Figure 5.8
4	Ellipse method	No	Velocity	Figure 5.17
5	Two dim. method	No	Velocity	Figure 5.18
6	Two dim. method	Yes	Velocity	Figure 5.19



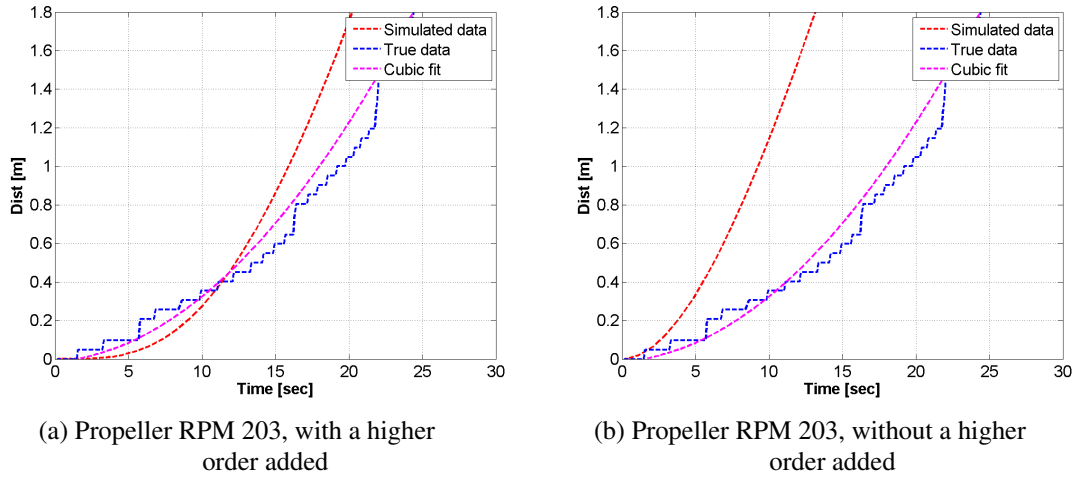


Figure 5.5: Comparison on open loop surge model, with and without higher order term

An attempt to attain a higher order response was performed by adding a second order transfer function, shown in Table 5.2, into the simulator. The transfer function was tuned manually and we got better results, shown in Figs. 5.8 and 5.19.

For comparison, Fig. 5.5a shows results given from the surge model equipped with higher order transfer function and Fig. 5.5b shows results given from the surge model without higher order transfer function. Based on a limited amount of experimental data in surge DOF, it is difficult to predict what exact transfer function suits the AUV best. Based on the available data, it is likely that the model is of lower order, and a transfer function of higher order will be needed to correct the system. Suggestive method to find appropriate transfer function is to use optimization as described in the simulator chapter, chapter 3.

Comparison of the velocity in Figs. 5.17 and 5.18 to 5.19, shows how the higher order term affects the velocity. Fig. 5.19c shows a velocity comparison where the area between the two curves has been minimized during manual tuning of parameters in the transfer function.

The hydrodynamic variables obtained by having the transfer function active are given in Table 5.3.

Table 5.2: Transfer function

Method	Transfer function
Two dim.	$\frac{(1/5)^2}{s+2*1*(1/5)+(1/5)^2}$

Table 5.3: Results of adjusting the surge model to the experimental data

Parameter	Symbol	Ellipse approach	Two dim approach	Units
Mass	$m$	63.0	63.0	$kg$
Added mass	$X_{\dot{u}}$	0.05	6.3	$kg$
Linear drag	$X_u$	9	9	$kg/s$
Quadratic drag	$X_{u u }$	0	0	$kg/m$
Quadratic tuner	-	0	0	-

### 5.3 Sway Model Validation

Results logged during sway analysis, both in an open and closed loop mode were analysed. The added mass coefficient was found by using both two dimensional approach and the ellipse approach. Results are in section 5.6 and are listed in Table 5.4.

During tuning, the linear drag term had the starting value of known AUV vehicles. The model was then adjusted to the experimental data by affecting the linear and turbulent drag. Table 5.5 shows the best results during manual optimization on the model. Untweaked open loop experiments, for both ellipse and two dimensional methods for the added mass, shown in Figs. 5.9 and 5.10 are almost identical. Similar comparison between two dimensional approach and the ellipse approach in a closed loop system is shown in Figs. 5.11 and 5.12, they also look identical. The reason is that the difference between the added mass based on those two methods is small. Values used during this validation are listed in Table 5.5. The table also shows the estimated quadratic drag is reduced to less than half of its estimated size due to low speed.

Table 5.4: Comparison performed for sway DOF

No.	Results from:	Added mass	Figure
1	Open loop	Ellipse	Figure 5.9
2	Open loop	Two dim.	Figure 5.10
4	Closed loop	Ellipse	Figure 5.11
5	Closed loop	Two dim.	Figure 5.12

Table 5.5: Results of adjusting the sway model to the experimental data

Parameter	Symbol	Ellipse approac	Two dim approach	Units
Mass	$m$	63	63	$kg$
Added mass	$Y_{\dot{v}}$	59.6	72	$kg$
Linear drag	$Y_v$	12	12	$kg/s$
Quadratic drag	$Y_{v v }$	123.9	123.9	$kg/m$
Quadratic tuner	-	0.45	0.45	—
Delay	-	400	400	$\mu s$

The delay from reading the sub and calculating the distance between every latitude and longitude points caused a minor uncertainty. Approximated delay during the experiment was  $300\mu s$  to  $400\mu s$  due to a steady coordination calculation as a limited data communication. The delay was imported to the delay block during all validation. When the boat was heading to its reference value, the simulated output response was good but the time period of the sinusoidal response did not always fit perfectly. It was easy to fit all the data by varying the time delay but all the figures in Figs. 5.11 and 5.12 have the same delay.

## 5.4 Heave Model Validation

No experimental data is available for heave DOF. But because of hull symmetry, the model for heave and sway are almost the same. The conning tower affects, in sway, is neglected causing its dynamics to act like heave. The only difference is the restoring force, or the buoyancy force, affecting the sub while diving. By having the boat neutral in water, the restoring force is zero and all validation already performed for the sway model, also works for the heave model. By adding some values to the restoring term in the model, it simply means that the boat goes up, even without any thrusters or steering at all. It also changes the output response so that the sub always goes faster up than it goes down, specially if the same PID values are used during up and down movement. An example of this is in the next chapter.

## 5.5 Yaw Model Validation

Results logged during the yaw analysis, both in an open loop and closed loop mode were analysed. The added mass coefficient was found by using both two dimensional approach and the ellipse approach. Results are in section 5.6 and are listed in Table 5.6. Before figures will be compared and analysed, following assumptions will be made.

- The center of rotation was fixed to the the center of gravity. For simplification the system is thought of as a point mass at certain distance from the cg, where it is rotated about its axis.
- The total linear and quadratic forces are acting in a fixed distance, located at  $1/4$  of boat length from the cg.

Figures 5.13 and 5.14 shows results from an open loop experiment performed with ellipse and two dimensional methods for added inertia applied consequently. The figures shows that both methods gives promising results but Fig. 5.13 where the ellipse method is applied is better.

By examining Figs. 5.15 and 5.16 for the closed loop, quite a difference is noticed. Table 5.6 shows values used during those experiments. As seen, there is a big difference between the added mass term depending on what method is used. High added inertia, resulted in longer time period of the oscillatory response. It was longer than the experimental data period. A solution to use high theoretical estimated added inertia is to adjust the signal processing delay or downgrade the added inertia with an adjuster for better results when comparing the experimental data to the simulator. Figures 5.15 and 5.16 uses the original added inertia based on theoretical results. One main parameter was adjusted to tune the model, it was the linear drag term. Due to drag behaviour, the quadratic drag was also increased a bit. The added inertia based on ellipse approach obviously shows better results here, even though the added inertia based on two dimensional can be downgraded and used as well. Delay estimated here was smaller than for sway due to no coordination

Table 5.6: Tests performed for yaw DOF

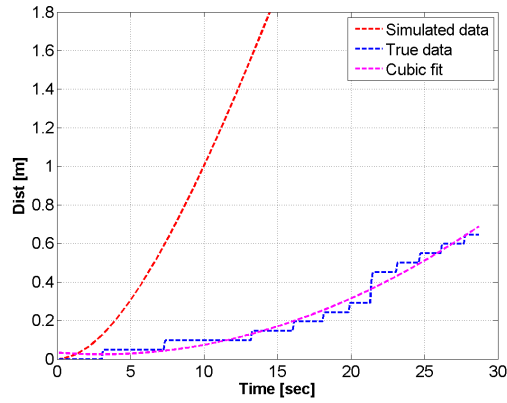
No.	Results	Method	Figure
1	Open loop	Ellipse	Figure 5.13
2	Open loop	Two dim.	Figure 5.14
3	Closed loop	Ellipse	Figure 5.15
4	Closed loop	Two dim.	Figure 5.16

Table 5.7: Results of adjusting the yaw model to the experimental data

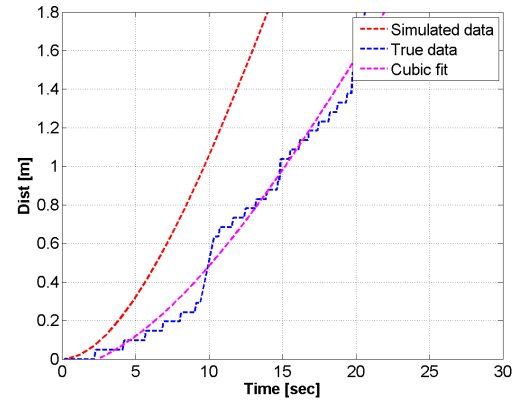
Parameter	Symbol	Ellipse approach	Two dim approach	Units
Inertia	$I_{zz}$	28.5	28.5	$kg \cdot m^2$
Added inertia	$N_{\dot{r}}$	13.3	31.8	$kg \cdot m^2$
Linear drag	$N_r$	12	12	$kg \cdot m^2/s$
Quadratic drag	$N_{r r }$	15	15	$kg \cdot m^2$
Quadratic tuner	-	1.5	1.5	-
Delay	-	350	350	$\mu s$

calculation, it was approximated on a interval  $250\mu s$  to  $350\mu s$ . Manual method was used to adjust the model in this section.

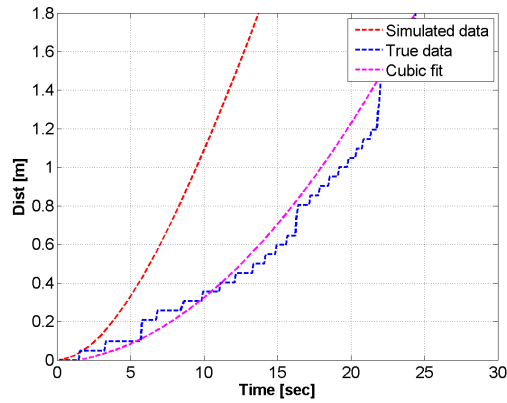
## 5.6 Model Validation Plots



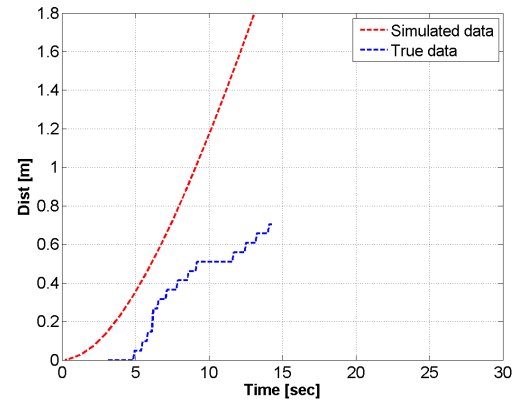
(a) Propeller RPM 187



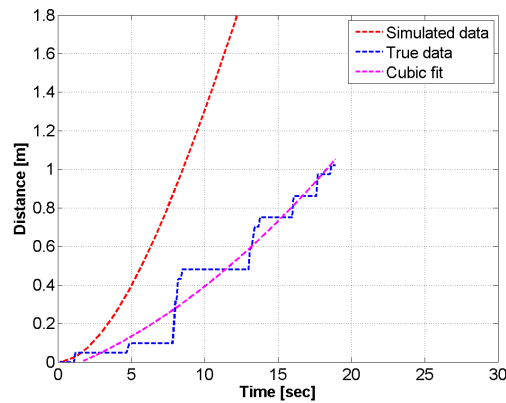
(b) Propeller RPM 197



(c) Propeller RPM 203

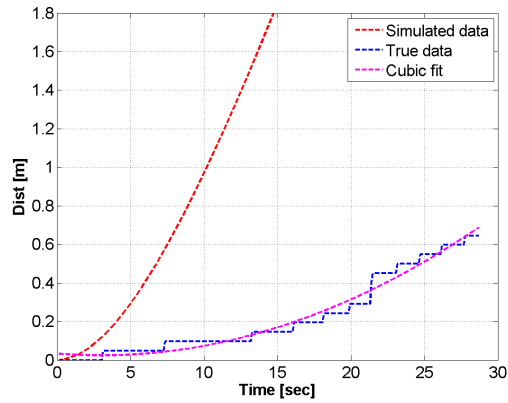


(d) Propeller RPM 218

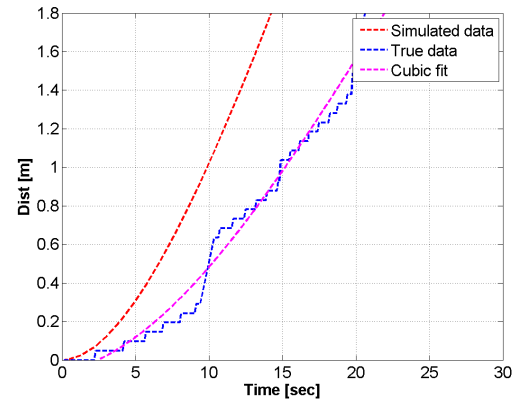


(e) Propeller RPM 242

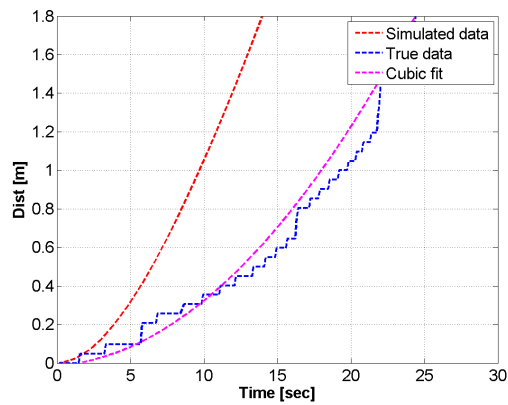
Figure 5.6: Surge open loop response, ellipse method applied for added mass, the model is not equipped with higher order activity



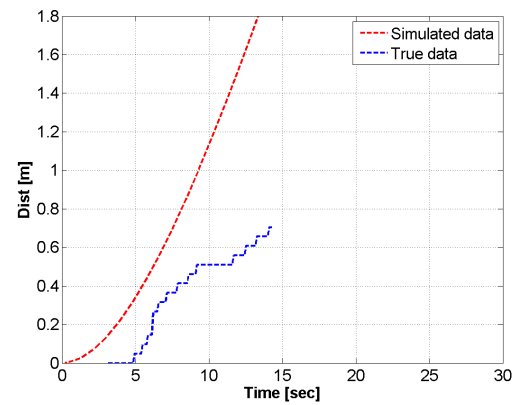
(a) Propeller RPM 187



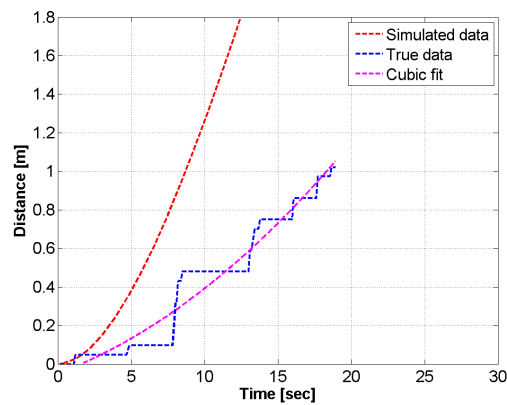
(b) Propeller RPM 197



(c) Propeller RPM 203

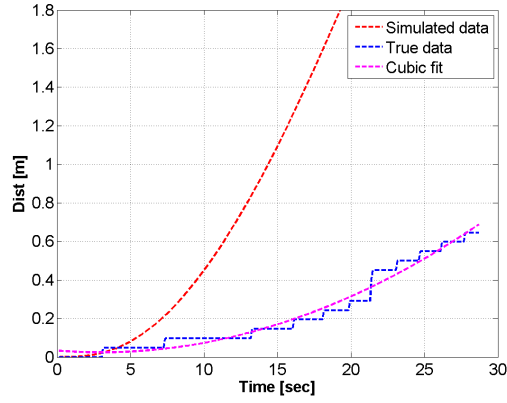


(d) Propeller RPM 218

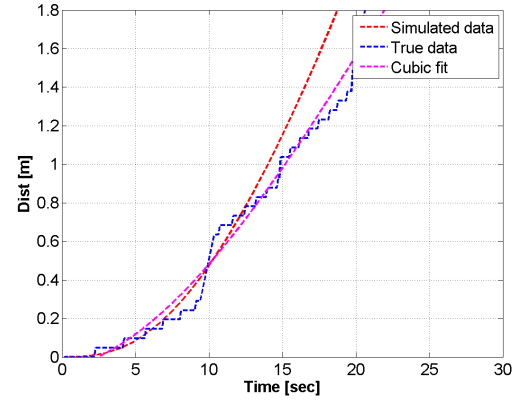


(e) Propeller RPM 242

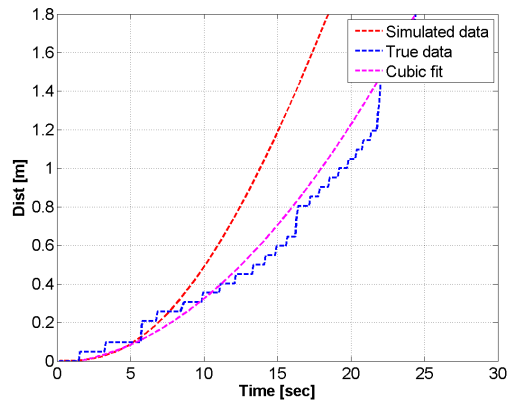
Figure 5.7: Surge open loop response, two dimensional method applied for added mass, the model is not equipped with higher order activity



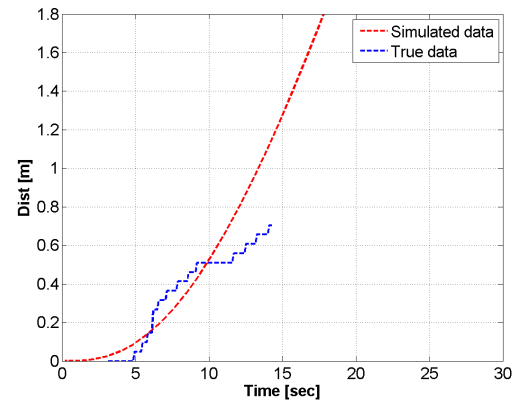
(a) Propeller RPM 187



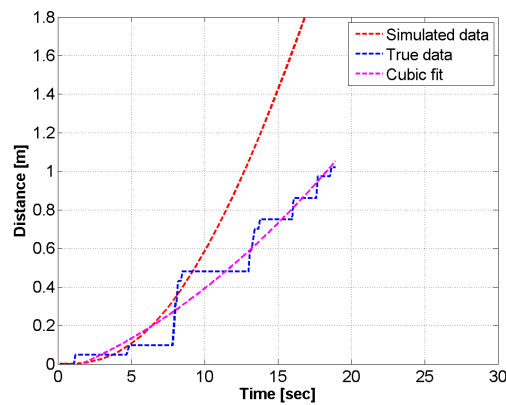
(b) Propeller RPM 197



(c) Propeller RPM 203



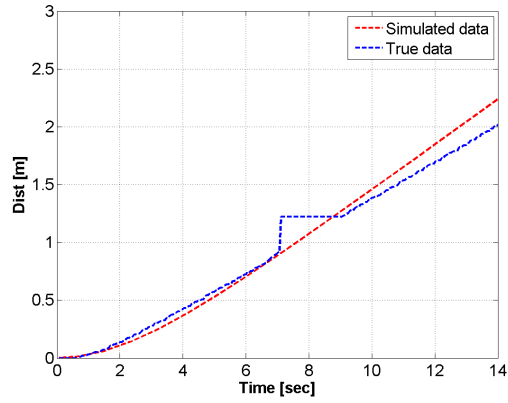
(d) Propeller RPM 218



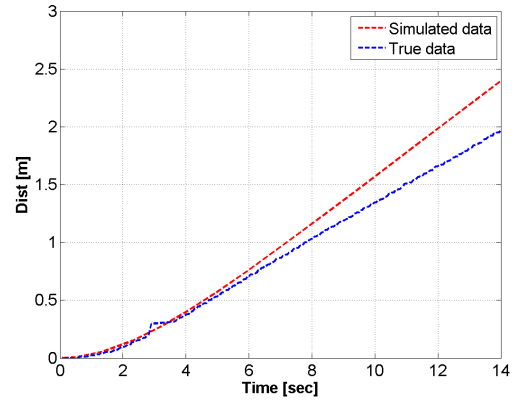
(e) Propeller RPM 242

Figure 5.8: Surge open loop response, two dimensional method applied for added mass, the model is equipped with higher order transfer function

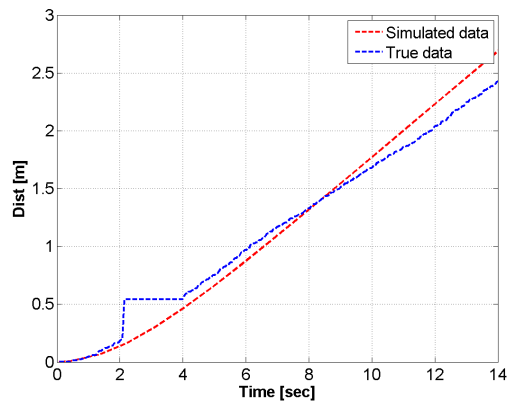




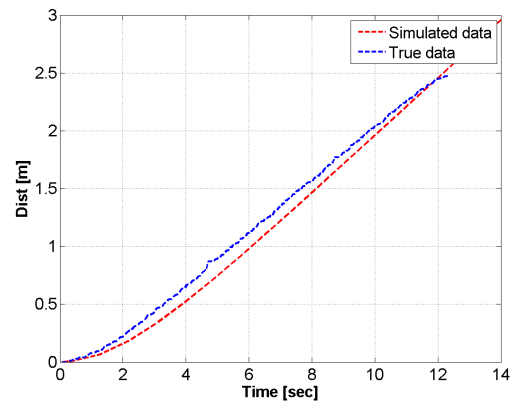
(a) Constant Thrust 0.25



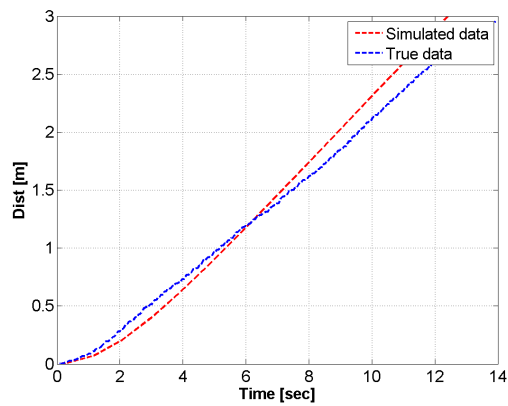
(b) Constant Thrust 0.3



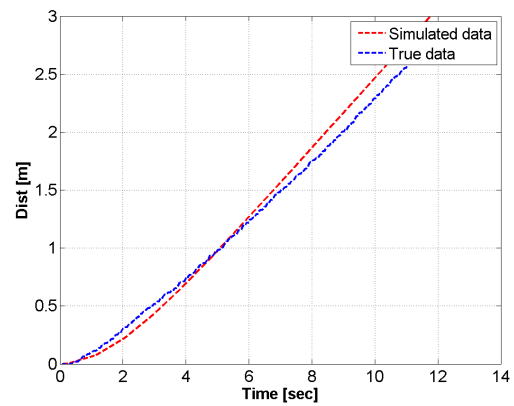
(c) Constant Thrust 0.4



(d) Constant Thrust 0.5

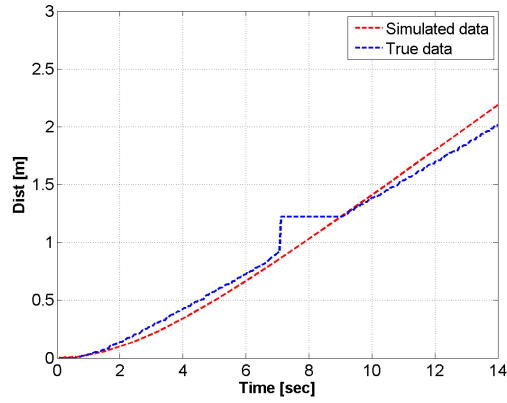


(e) Constant Thrust 0.7

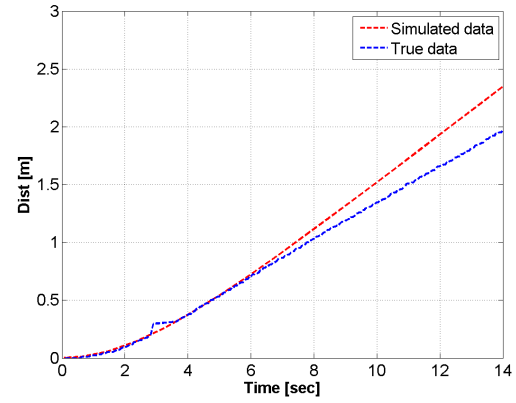


(f) Constant Thrust 0.8

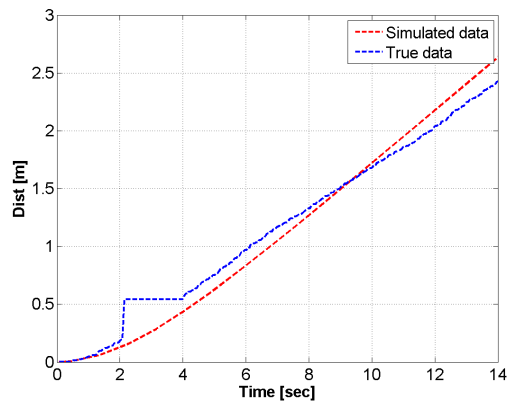
Figure 5.9: Sway open loop response, the ellipse method applied.



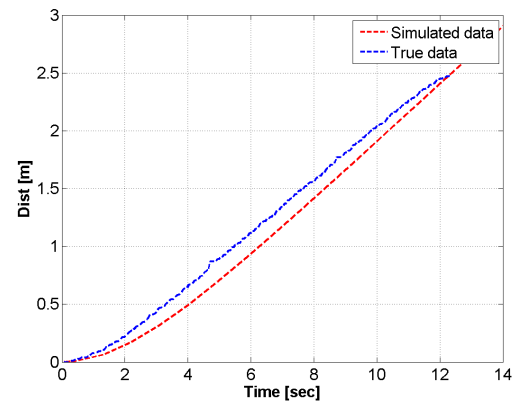
(a) Constant Thrust 0.25



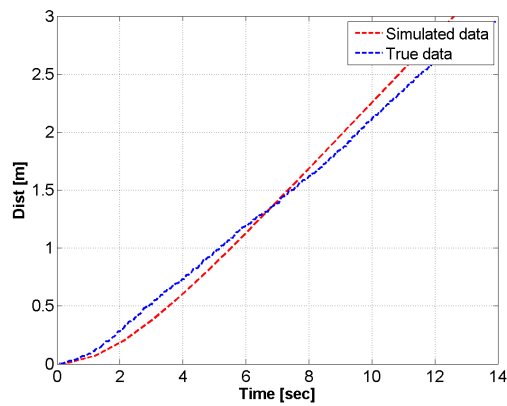
(b) Constant Thrust 0.3



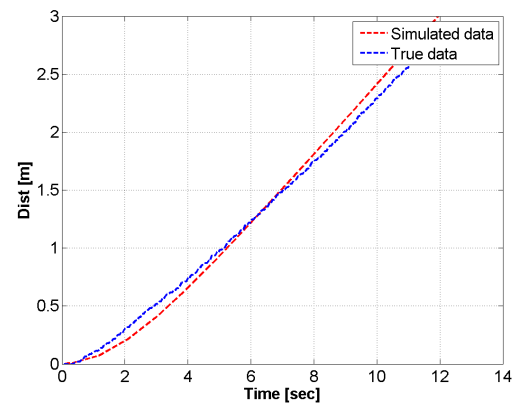
(c) Constant Thrust 0.4



(d) Constant Thrust 0.5



(e) Constant Thrust 0.7



(f) Constant Thrust 0.8

Figure 5.10: Sway open loop response, the two dimensional method applied.

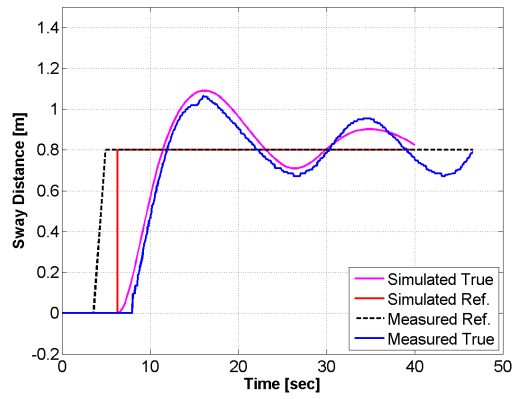
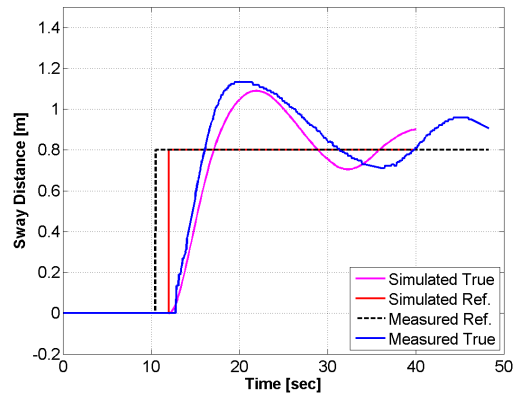
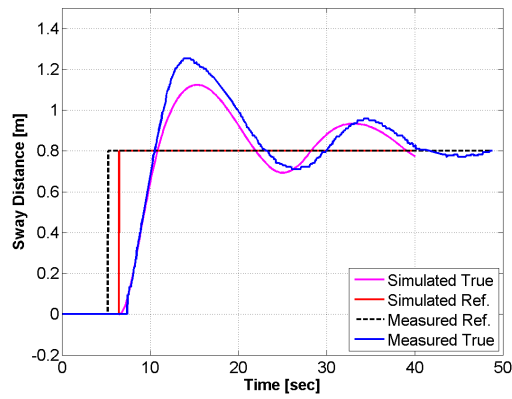
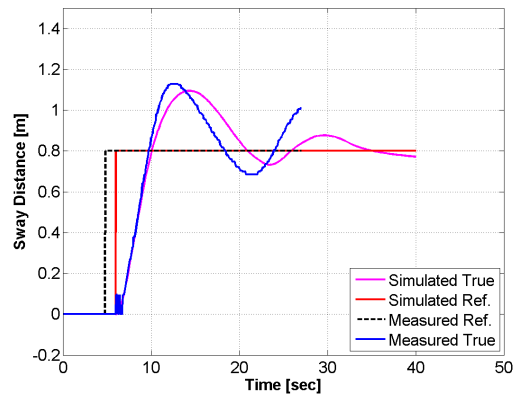
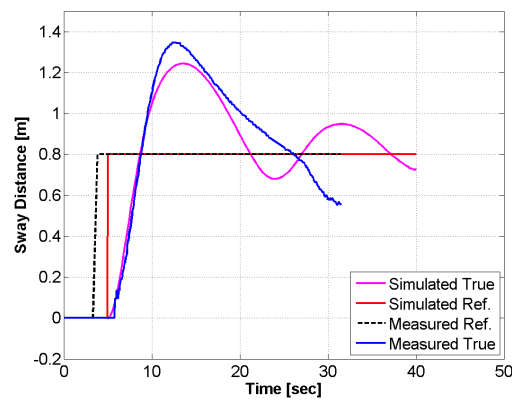
(a) PID Values:  $P = 30$ ,  $I = 0$  and  $D = 0$ (b) PID Values:  $P = 30$ ,  $I = 0$  and  $D = 0.4$ (c) PID Values:  $P = 40$ ,  $I = 0$  and  $D = 0$ (d) PID Values:  $P = 40$ ,  $I = 0$  and  $D = 20$ (e) PID Values:  $P = 50$ ,  $I = 0$  and  $D = 0$ 

Figure 5.11: Sway closed loop response, the ellipse method applied.

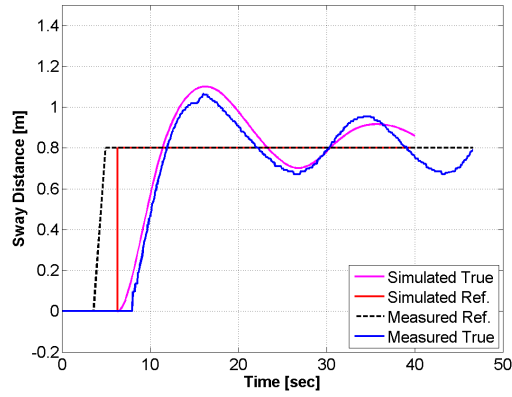
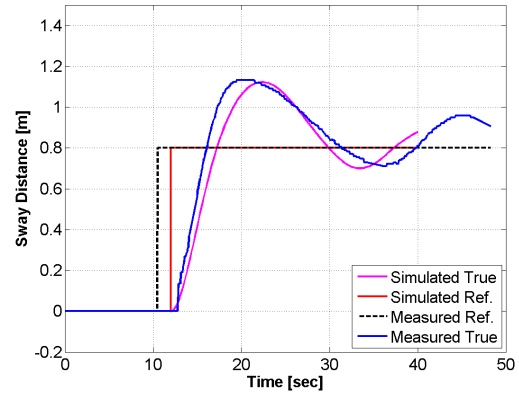
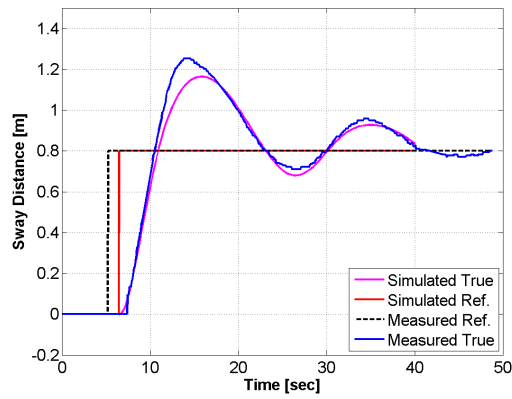
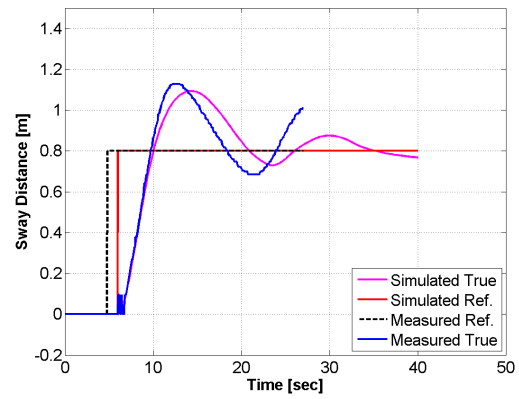
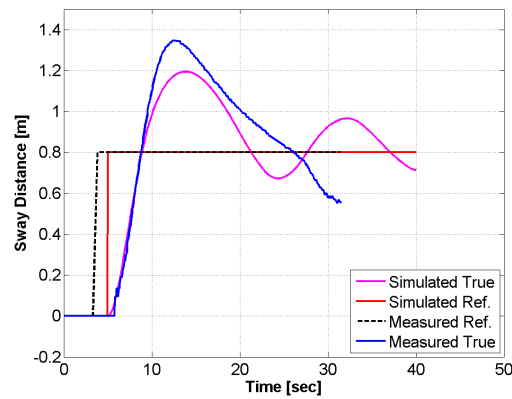
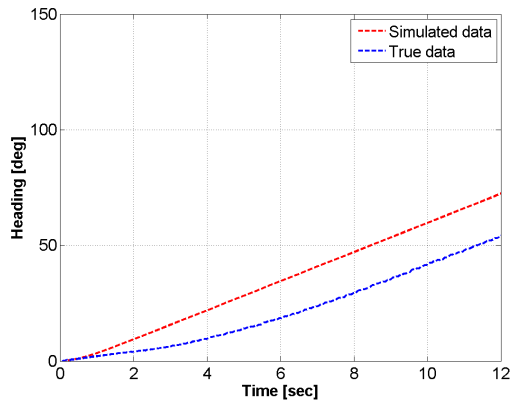
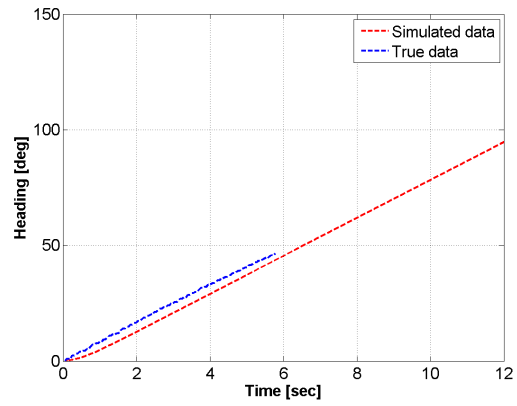
(a) PID Values:  $P = 30$ ,  $I = 0$  and  $D = 0$ (b) PID Values:  $P = 30$ ,  $I = 0$  and  $D = 0.4$ (c) PID Values:  $P = 40$ ,  $I = 0$  and  $D = 0.4$ (d) PID Values:  $P = 40$ ,  $I = 0$  and  $D = 20$ (e) PID Values:  $P = 50$ ,  $I = 0$  and  $D = 0$ 

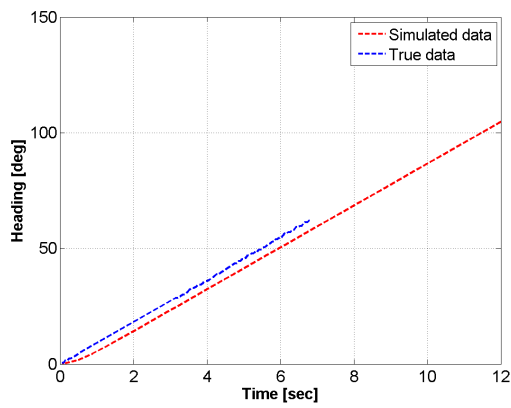
Figure 5.12: Sway closed loop response, the two dimensional method applied.



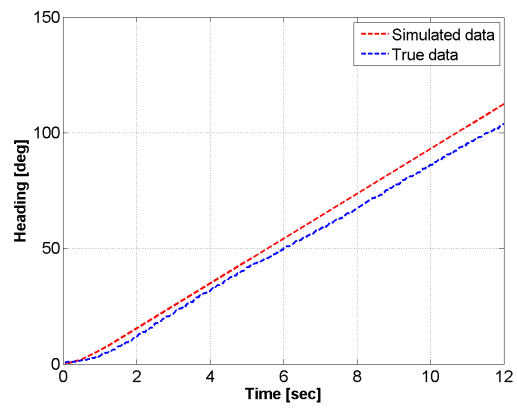
(a) Constant Thrust 0.2



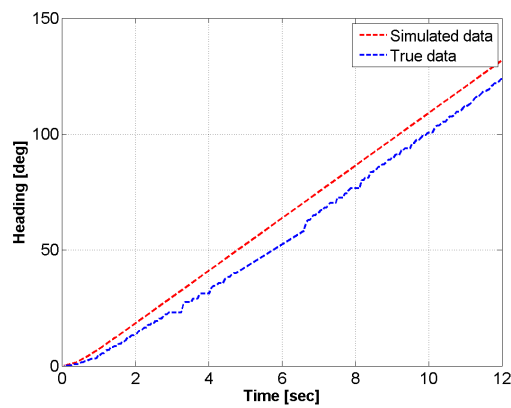
(b) Constant Thrust 0.4



(c) Constant Thrust 0.5

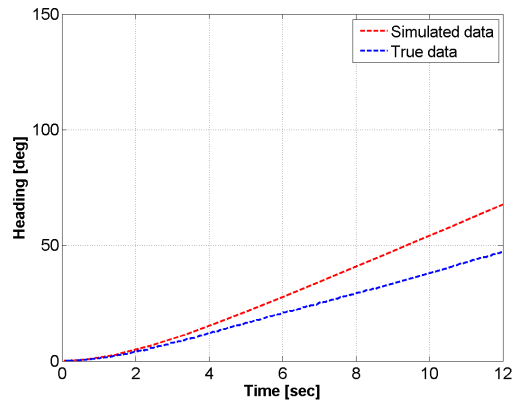


(d) Constant Thrust 0.6

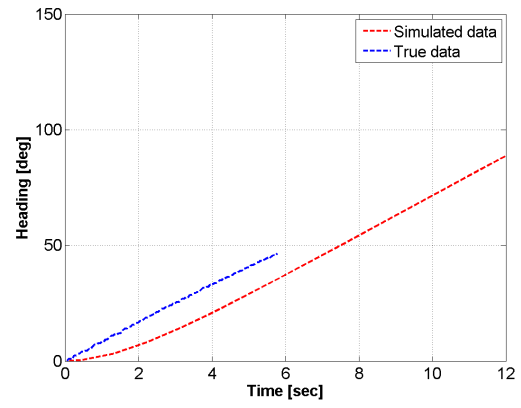


(e) Constant Thrust 0.9

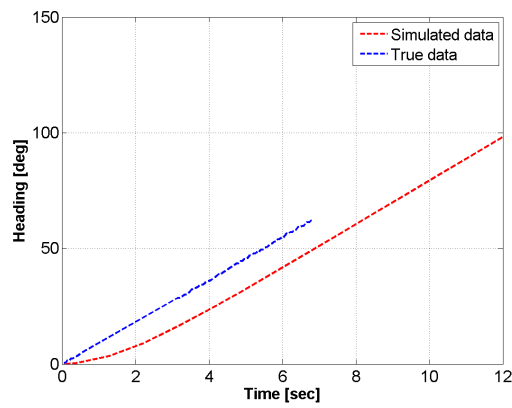
Figure 5.13: Yaw open loop response, the ellipse method applied.



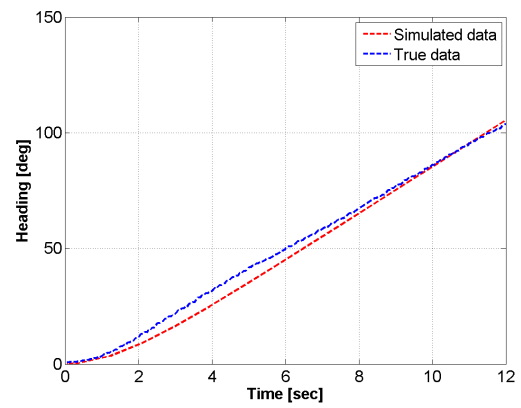
(a) Constant Thrust 0.2



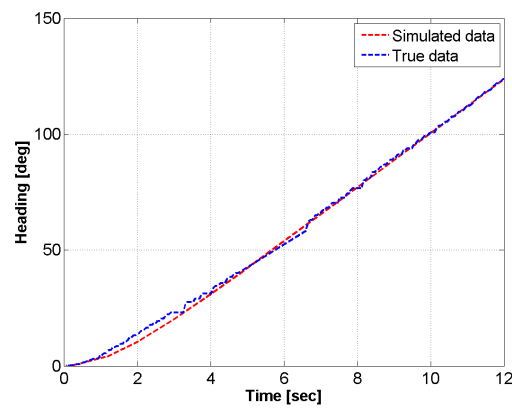
(b) Constant Thrust 0.4



(c) Constant Thrust 0.5



(d) Constant Thrust 0.6



(e) Constant Thrust 0.9

Figure 5.14: Yaw open loop response, the two dimensional method applied.

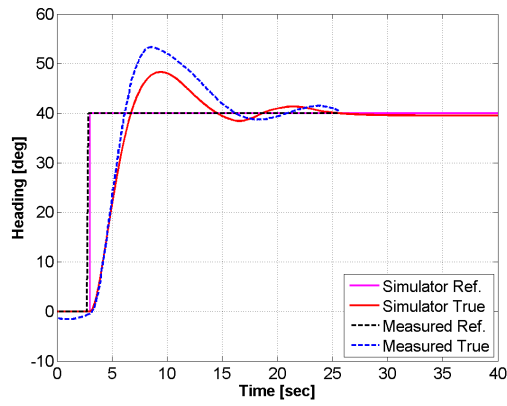
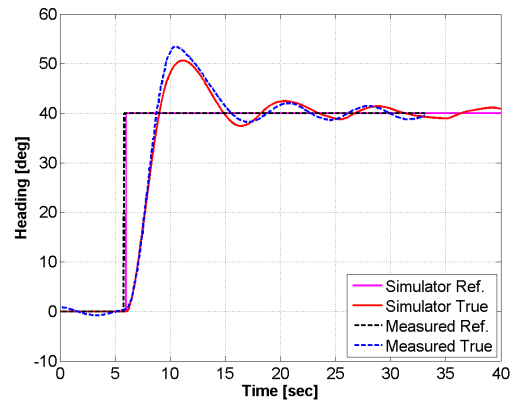
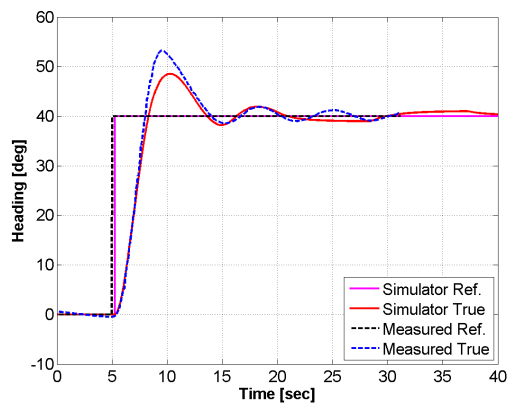
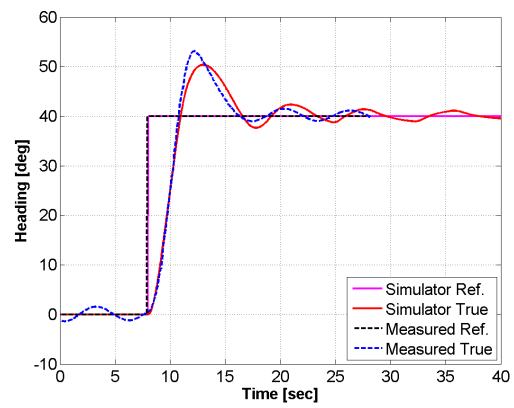
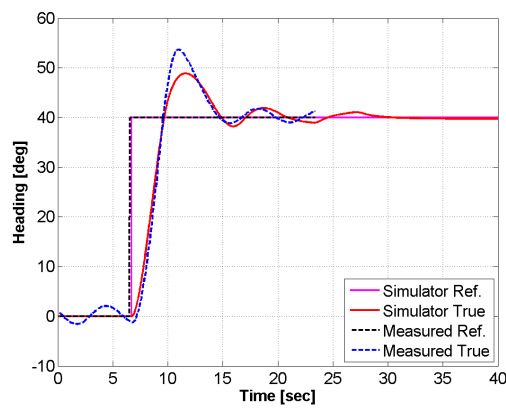
(a) PID Values:  $P = 1$ ,  $I = 0$  and  $D = 0$ (b) PID Values:  $P = 2$ ,  $I = 0$  and  $D = 0$ (c) PID Values:  $P = 2$ ,  $I = 0$  and  $D = 0.4$ (d) PID Values:  $P = 3$ ,  $I = 0$  and  $D = 0$ (e) PID Values:  $P = 3$ ,  $I = 0$  and  $D = 0.4$ 

Figure 5.15: Yaw closed loop response, the ellipse method applied.

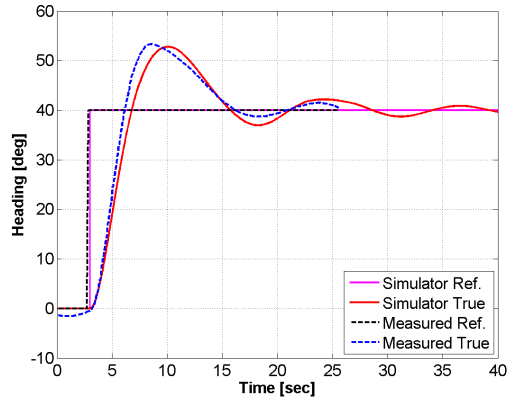
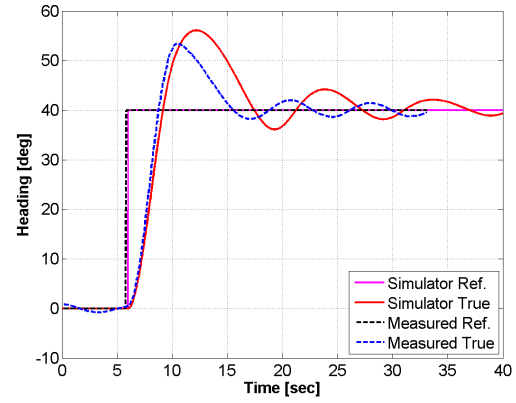
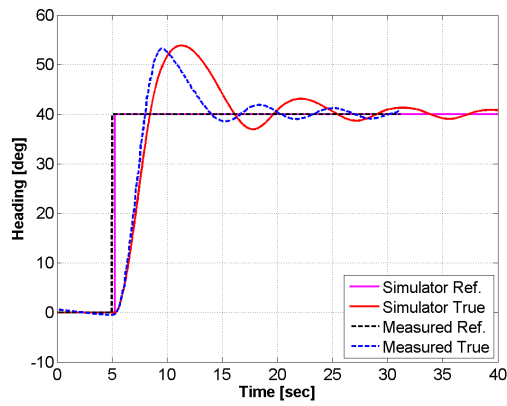
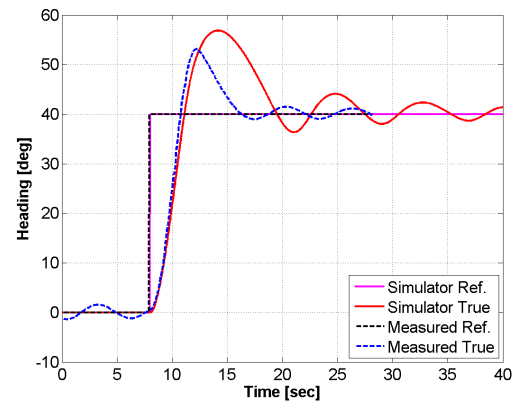
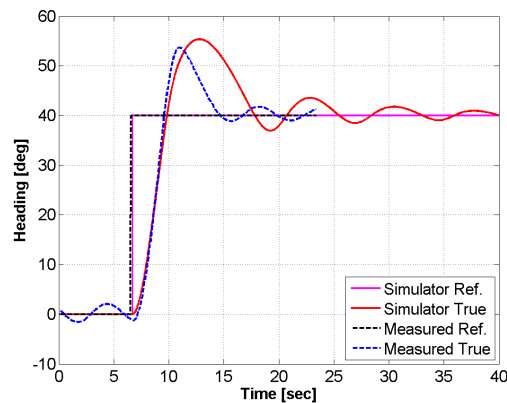
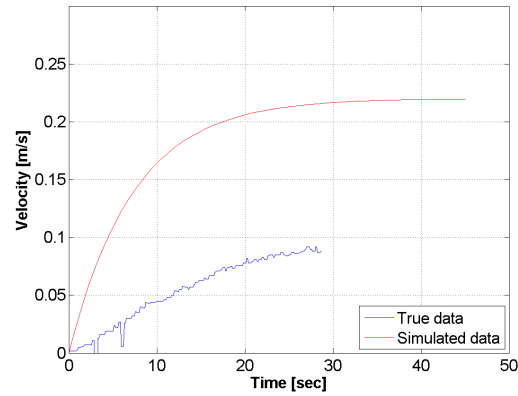
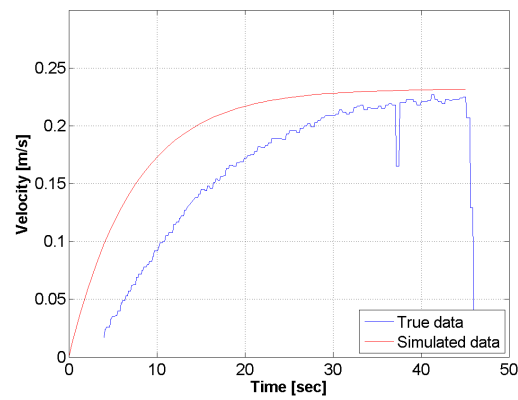
(a) PID Values:  $P = 1$ ,  $I = 0$  and  $D = 0$ (b) PID Values:  $P = 2$ ,  $I = 0$  and  $D = 0$ (c) PID Values:  $P = 2$ ,  $I = 0$  and  $D = 0.4$ (d) PID Values:  $P = 3$ ,  $I = 0$  and  $D = 0$ (e) PID Values:  $P = 3$ ,  $I = 0$  and  $D = 0.4$ 

Figure 5.16: Yaw closed loop response, the two dimensional method applied.

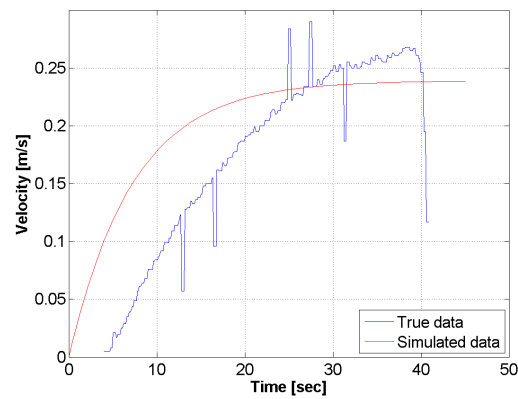




(a) Surge velocity during open loop test. Reference value 187 RPM, ellipse method applied

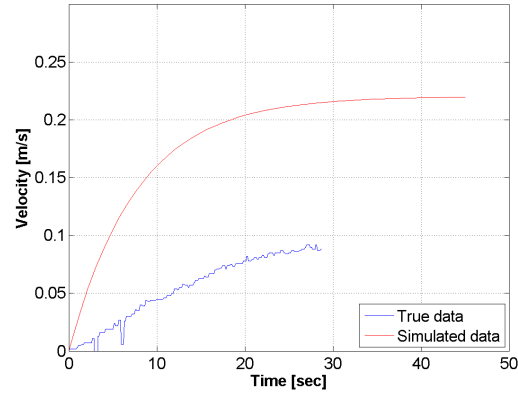


(b) Surge velocity during open loop test. Reference value 197 RPM, ellipse method applied

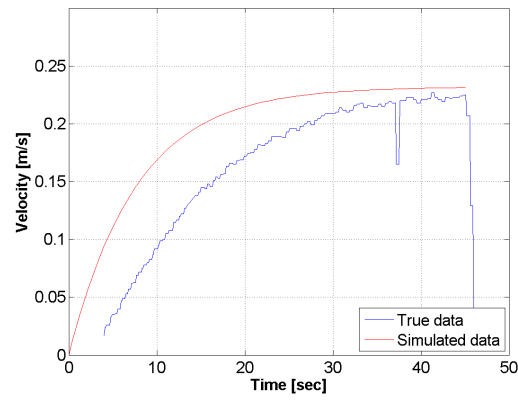


(c) Surge velocity during open loop test. Reference value 203 RPM, ellipse method applied

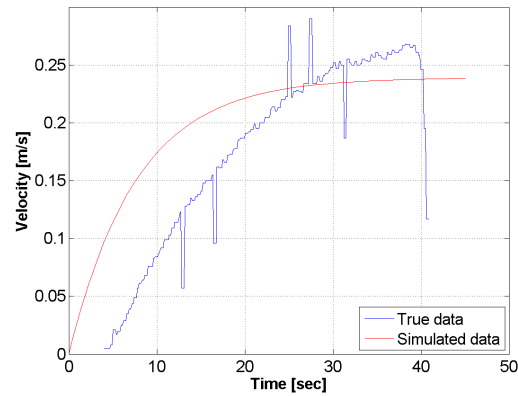
Figure 5.17: Surge open loop velocity response, the ellipse method.



(a) Surge velocity during open loop test. Reference value 187 RPM, two dimensional method applied

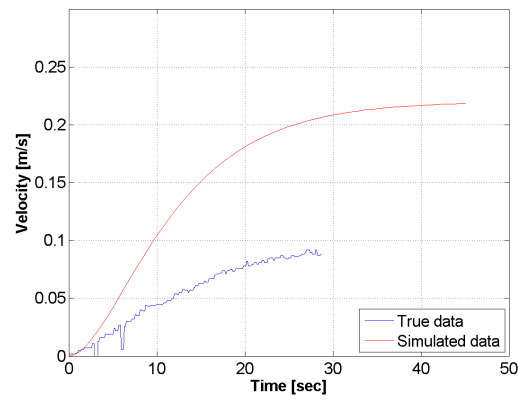


(b) Surge velocity during open loop test. Reference value 197 RPM, two dimensional method applied

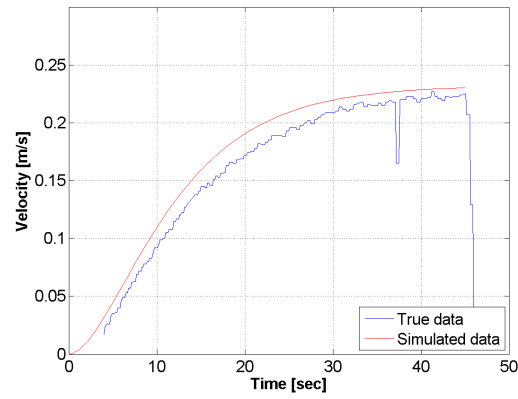


(c) Surge velocity during open loop test. Reference value 203 RPM, two dimensional method applied

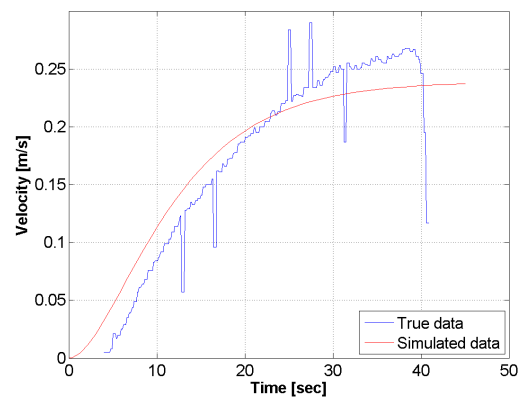
Figure 5.18: Surge open loop velocity response, the two dimensional method.



(a) Surge velocity during open loop test. Reference value 187 RPM, two dimensional method applied



(b) Surge velocity during open loop test. Reference value 197 RPM, two dimensional method applied



(c) Surge velocity during open loop test. Reference value 203 RPM, two dimensional method applied

Figure 5.19: Surge open loop velocity response, the two dimensional method and the higher order term active.



# Chapter 6

## Design Studies

The simulator's capabilities are demonstrated here through several design cases. Though only one or two DOF are examined during each case, the simulator is capable of doing the same tasks for other DOFs. The design cases consider vehicles longer than the vehicle used in the experiment. Ideally, experiments with longer vehicles would yield a better and more accurate model. However, we still apply the model to these longer vehicles for demonstration purposes. Further experiments will be conducted in the future.

### 6.1 Case Study 1

Let's say that an important client wants a custom made boat. The boat should be a torpedo shaped AUV with both high speed manoeuvring and low speed manoeuvring as an option. When the boat finds interesting areas during surveying it should log its coordinate and then complete its surveying mission. Instead of sending a group of divers to investigate the interesting areas already found, the boat should be able to do a second mission to map the place around already found coordinate with its inbuilt camera. The mapped place should be rectangle of a given size with its interesting spots coordinates in its middle. To be able to do this mapping the boat obviously has to be able to do sway and yaw without moving in surge direction. Therefore the boat has to be implemented with thruster equipment capable of doing such movement. Figure 6.1 shows how the boat should map the ocean floor after the finding of interesting spots. By first approach based on the client request about the selected payload and equipment, the custom made boat will have a diameter of 20 cm, length of 3.5 meters and weight of 95 kg.

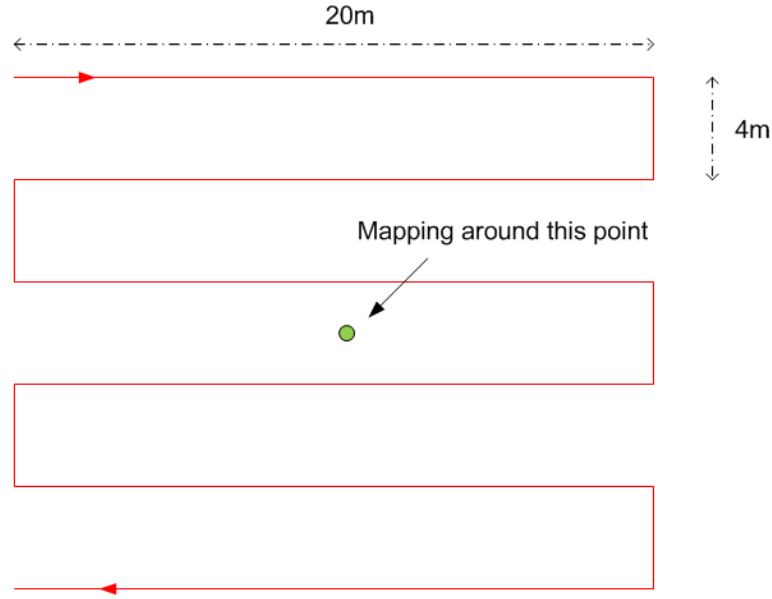


Figure 6.1: Mapping around an interesting point

To respond to the request from the client, we will start by examining the custom made boat behaviour fully in the simulator. We will simulate two test cases during the analyses. Due to the reason that all Gavia AUVs are equipped with the same propeller equipment at the back, we will use it for surge movement. But for other DOFs, sway, heave and yaw, we will consider tunnel thrusters. Two tunnel thrusters are needed for each DOF. The tunnel thrusters selected for the modelling are the same version as Leo Steensson uses for the experiment of the Delphin2 AUV [19]. The tunnel thrusters are made by TSL Technology and are available in different sizes. Figure 1.5a shows such a thruster. Since TSL Technology offers couple of sizes of those thrusters, we will model two sizes, labelled test Case 1 and test Case 2 hereafter. Test Case 1 uses the smallest one or a 50 mm version while test Case 2 uses 100 mm version.

The thrusters specifications are listed in Appendix C. Tables 6.1 and 6.2 lists physical properties of each case. These tables show that all the settings of the boat in the simulator are the same except it has a different tunnel thruster for each case. Specifications about the measuring equipment used in the model are listed in table 6.3. The simulator gives a lot of interesting plots to export and examine. To focus on what it is capable of, we will export some figures for sway and yaw, that are important for such a design and analyses. Due to our interest in modelling the boats behaviour and selection of appropriate propeller equipment we will focus on that and leave all other design processes out.

We started by analysing the sway DOF. Figure 6.2 lists four figures selected to describe the boat behaviour with two different types of tunnel thrusters, both available from TSL

Table 6.1: Values used for test Case 1

Physical Properties	Value
Mass [kg]	95
Diameter [m]	0.2
Length [m]	3.5
Tunnel thruster size [mm]	50

Table 6.2: Values used for test Case 2

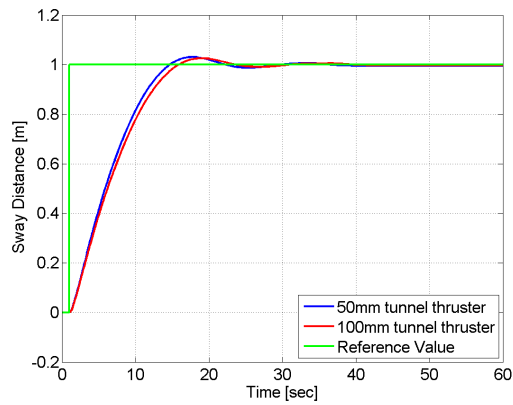
Physical Properties	Value
Mass [kg]	95
Diameter [m]	0.2
Length [m]	3.5
Tunnel thruster size [mm]	100

Table 6.3: Meter Specifications for test Cases 1 and 2

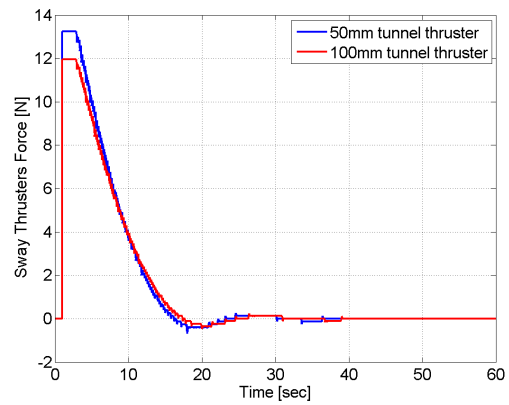
DOF	Resolution	Delay [ms]	Computational Delay [ms]	Rate[s]
Surge	0.1 m	250	1000	0.01
Sway	0.1 m	250	1000	0.01
Heave	0.1 m	250	500	0.01
Yaw	0.6 deg.	250	500	0.01

Technology. One is  $100W/12N$  and  $50mm$  in diameter while the other is  $300W/56N$  and  $100mm$  in diameter. Even though the larger one is obviously more powerful, the comparison is set-up to compare them while they are delivering same amount of work to the system. That results in similar output response as listed in Figure 6.2a. Here the boat mission is to move 1 meter in sway direction. The PID regulator was set to certain values, only P part of the PID was used. For the smaller propeller test, the P value was set to 70 while it was set to 30 for the bigger one. Figure 6.2b shows the thrust delivered by those thrusters, those curves are similar even though the smaller thrusters are of slightly more peak value. Figure 6.2c shows the location error as function of time, both curves are similar. Figure 6.2d is probably the most important of those four. The figure shows the power per thruster in  $W$  as a function of time. The main result based on that image is that the  $100mm$  thruster uses lot less power than the  $50mm$  one. Average power based on this certain test for smaller thruster was  $10W$  while it was only  $5W$  for the bigger one. An important thing in the design process is the actual lifetime of the batteries.

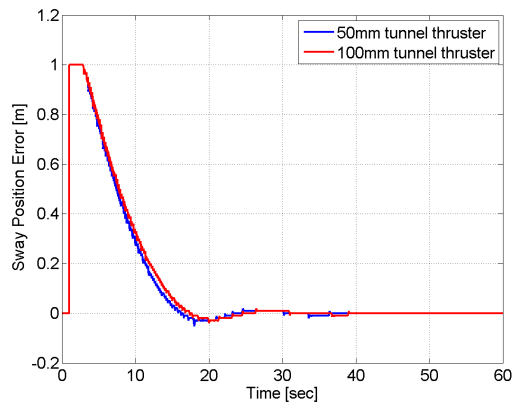
The AUV should be able to do mapping of the sea floor, as listed in the criteria above. Here, based on the power consumption one can estimate, e.g. for mapping, how much power the AUV needs to complete one mapping. An example of mapping can be surge  $20m$ , sway  $4m$  and rotate  $180^\circ$ . Then the user can estimate the power required for one such movement and count the movements required to complete the mission.



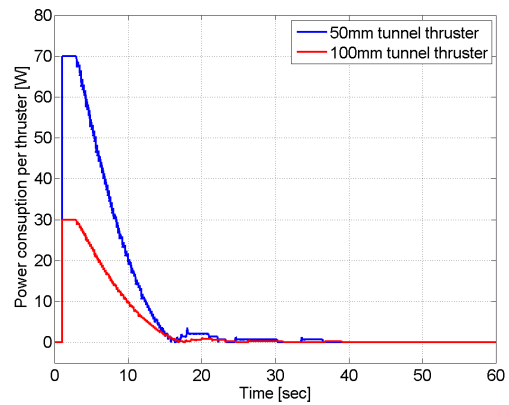
(a) Sway distance response comparison



(b) Sway thrust response comparison



(c) Sway error position comparison



(d) Sway power consumption per thruster comparison

Figure 6.2: Tunnel thruster comparison. Both thrusters were IntegratedThrusters from TSL Technology. The smaller one has diameter of 50mm while the larger one has diameter of 100mm

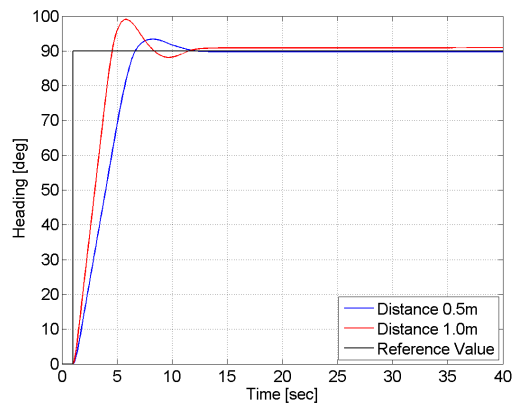


By looking at the yaw movement, the location of the thruster from the centre of gravity (cg) is important. Another kind of comparison is done here, now with same P value in all cases, but both I and D values are set to zero. Two places were selected for the 100mm tunnel thruster as a fixed location. The 3.5m long boat is now turned 90° on the spot. Results are shown in Figure 6.3a. One location was 0.5m from the cg while the other was 1m. This location of the thruster is most important for yaw DOF, but less important for other DOFs. As seen in Figure 6.3a the distance of 1m is resulting in a quicker response, but it has slightly more overshoot, which is mainly caused by the same PID setting between different scenario.

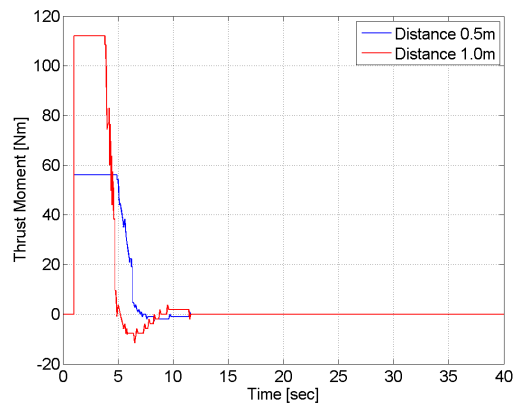
Teledyne Gavia is already manufacturing versions of the Gavia that are in excess of 3 meters, and are equipped with only one thruster at the back. This version has a huge turning radius and limited manoeuvring abilities due to its set-up.

Figure 6.3b shows the moment generated by the two tunnel thrusters. The peak moment caused by the thruster located at 0.5m from cg is 56Nm which is the rated thrust given by one thruster. Because of the distance from cg and the relationship between torque, distance and force, this torque is obviously correct. The same can be stated about the twice times more torque for twice times distance from cg. During the developing process PID values are also of special interest, Fig. 6.3c shows PID output during this experiment set-up. Comparison of the PID values from those two simulations shows that the PID regulator for the longer distance location to cg reaches the final value quicker. The overall time to the final value is nearly the same. This is mainly caused by the set-up of this experiment. If the PID would have been tuned specially to the longer distance case it would also act differently there.

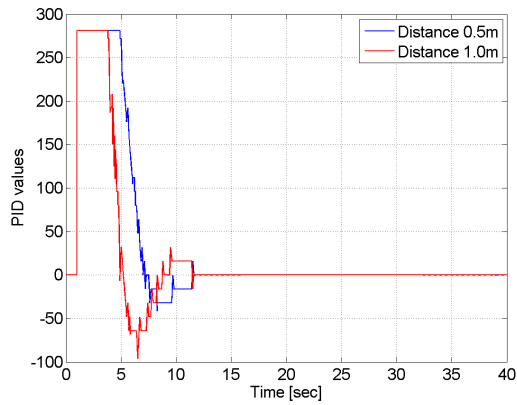
Figure 6.3d shows the power as a function of time for one 90° turn. As seen in the figure, the power used for longer distance is less than the power used for shorter distance. As before one can count number of turns, if that is part of the mission and estimate the power required for such a mission. The ideal set-up of such a case is to have the two thrusters located at same distance from cg.



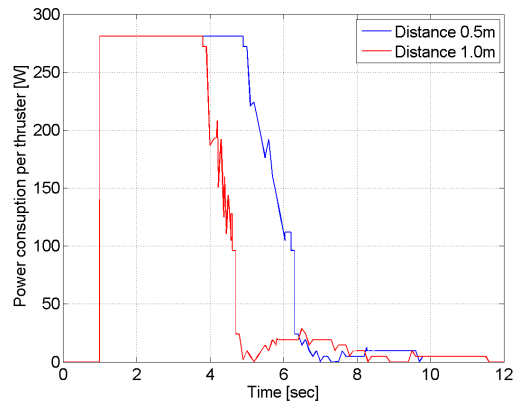
(a) Yaw heading response comparison



(b) Yaw thrust moment response comparison



(c) Yaw PID output comparison



(d) Yaw power consumption per thruster comparison

Figure 6.3: Tunnel thruster location from center of gravity comparison. This is 100mm IntegratedThrusters from TSL Technology

## 6.2 Case Study 2

A designer has been developing an autopilot system for a new generation of a torpedo shaped AUV. Due to a special need, by an important client, it has to be able to calculate position from a certain point, based on latitude and longitude coordinate many times a second, scale and interpolate many signals also many times per second. The AUV is still in the design process and has not been built yet. The designers code runs smoothly during desk tests but can not be tested since the boat is not available. Delay in automatic system usually causes trouble, mainly because the regulator or autopilot is constantly reading old values. Simple things like steering a vehicle go straight is a problem if the delay is high. The designer has estimated its autopilot time, its time do work data, read from sensor, data work in autopilot and finally the publishing process to a topic. He wants to see how that time affects the next generation of AUV he's working on.

The simulator has a computer and sensor delays, so a couple of delays can be pre-installed to a matrix in the simulators script and results of each of them can be examined. Based on those results the designer can estimate his delay affect on the boat regulation and see if its autopilot needs to be improved or not. Figure 6.4 shows an example of such a run from the simulator. It uses the same test case scenario used in Case Study 1, and the 100mm tunnel propeller. PID and reference values are the same for all runs, the only value changed was the time delay in the feedback loop. That delay both indicates the delay in the sensor itself as well as all delay handling and working the data in autopilot. As seen in the Fig. 6.4

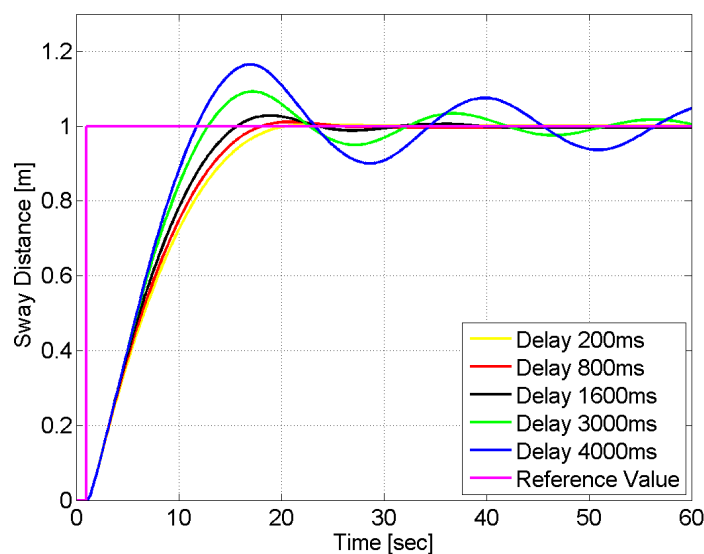


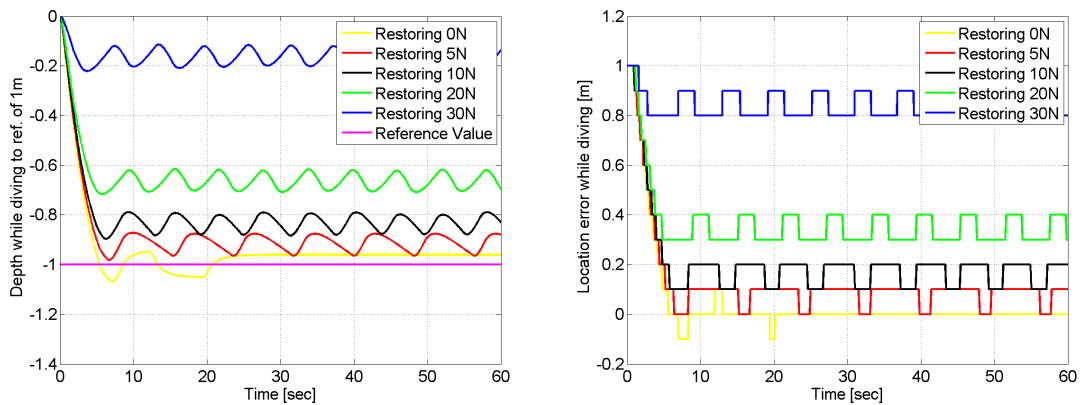
Figure 6.4: Comparison between the same system, with a different time working and publishing data from sensors

it is easier to tune a system with low time delay than others. One way to handle such a delay is to add a higher order reference value to the system instead of the step function. That is implemented in design criteria 4 and examined there further.

### 6.3 Case Study 3

A designer wants to select a tunnel thruster to a torpedo shaped AUV for heave movement. The boat is designed to be neutral in fresh water but due to different types of tasks solving each time, it sometimes comes slightly buoyant. The designer prefers a small tunnel thruster, because it is easier to design and install small thruster. He's also concerned it might be too small to dive if the boat will become buoyant. A solution to this is to run this test case scenario with the simulator set at a different restoring force (buoyant forces) each run.

Figure 6.5 shows an example of such a run based on all the settings given in design criteria 1. Two 100mm tunnel thrusters were used and a P value was set to 150. Figure 6.5a shows how the two 100mm tunnel thrusters handle the diving to 1 meters well with the boat neutral. But as soon as the restoring force gets bigger the AUV withdraws more and more from the reference value. When the restoring force is about 30N the thrusters are only able to dive down to 20cm at this PID value. This means that if the buoyancy is a variable then the PID values also need to be a variable. But as soon as the buoyancy gets the same value as rated thrust power, the thrusters are not capable of pushing the boat down.



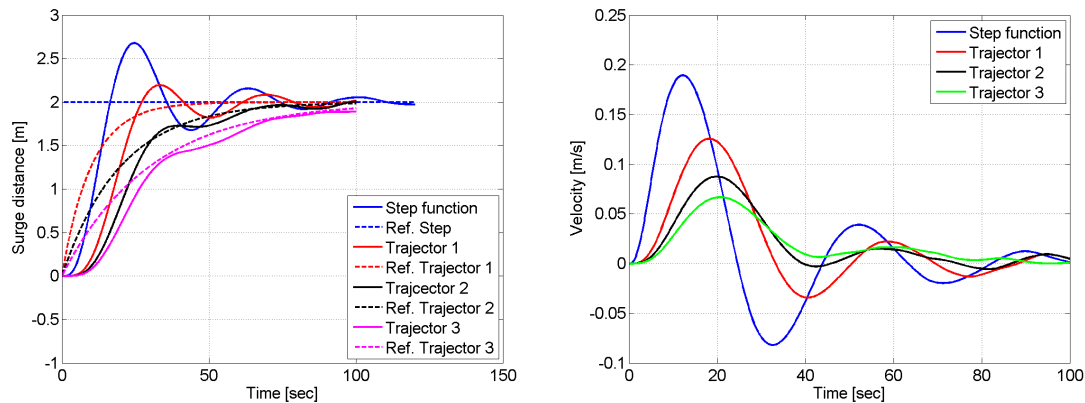
(a) Depth during diving to fixed reference value of 1m with different reference values (b) Depth error while diving to fixed reference value of 1m with different reference values

Figure 6.5: The effect of different restoring forces acting on the boat while diving to a fixed reference value of 1m. Thrusters in the model are 100mm Integrated Thrusters from TSL Technology

## 6.4 Case Study 4

A designer has completed his design and has also made a prototype torpedo shaped AUV. During testing it became obvious that the sub had problems reaching set reference values without overshooting a bit. This happened during surge direction. The designer wants a better regulation and has spent a lot of time and effort in experiments with different PID settings each time. Those experiment show no improvement regarding huge overshoot when final value is reach.

It can be desirable to use higher order reference model to generate smooth trajectory for the reference value to follow. The real reference value follows a certain trajectory to its requested reference value set by the user or autopilot. Such reference trajectory can be of many type, which leads to many time consuming experiments to find the one that actually suits each AUV. The simulator has a trigger that allows the user to use a reference trajectory instead of a traditional step function as a reference value. Then the user can select one or more trajectories and plots and save the results. An example of such a test, where three different trajectories were tested as input trajectories is given in Figure 6.6. Here the final reference value is 2 and the user can see how different trajectories affects the performance in surge direction. The three test trajectories are given in Eqs. 6.1, 6.2 and 6.3 on next page.



(a) Surge distance comparison by adding higher order input functions (b) Surge velocity comparison by adding higher order input functions

Figure 6.6: The effect of adding a higher order reference values to the system while travelling to a fixed length of 2 m in surge direction. The Thruster in the model is the original thruster pre-installed in Gavia

$$\frac{1/10}{s + (1/10)}, \quad (6.1)$$

$$\frac{1/20}{s + (1/20)}, \quad (6.2)$$

$$\frac{1/30}{s + (1/30)}. \quad (6.3)$$

# Chapter 7

## Concluding Remarks

This chapter concludes the work presented in this thesis. It first summarizes the topics and main conclusions from each chapter, and then suggests future work.

### 7.1 Summary

The equations of motion of a torpedo shaped autonomous underwater vehicle travelling in 4 DOF's are derived. Hydrodynamic forces are estimated theoretically. Two different methods were used to estimate the added mass. The quadratic drag was calculated theoretically and the linear drag was estimated during the simulator validation.

Environmental and ocean forces were ignored. The equations describing the boat's behavior for each DOF were implemented in a simulator. Here, linear control technique is used with PID controller that regulates and stabilizes the system.

Parallel to the block diagram development, a script was made to run the simulator in batch mode. The script is capable of running several simulations with different settings each time. It could, e.g., easily run 1000 different values of drag coefficient during one run. An open loop simulator was also made with similar batch mode script as for the closed loop simulator.

To validate the simulator, numbers of experiments were made. A commercially available AUV, the Gavia AUV, was used as a testbed during all the experiments. To perform the experiments a complete external thrust unit with control capabilities was programmed in Python.

The simulator was compared to the experimental data. By making non-linear control system linear, the linear control system can be successfully controlled but with a reduced functional behavior. Generally, to fit simulated data to experimental data, minor simulator tuning is needed, where adjusted parameters are, e.g., hydrodynamic forces, input trajectories and higher order transfer functions

The simulated and experimental comparison for surge DOF indicated that the model was of lower order than the real system, so several adjustments were made. Adjusting the hydrodynamic forces yielded good results for the distance travelled, while it yielded bad results for the AUV velocity. Adding a higher order transfer function into the system resulted in a slightly better fit for both the distance and the velocity comparisons. None of the adjustment mention above resulted in an acceptable fit. The transfer functions parameters were adjusted manually.

The simulated and experimental comparison for sway and heave DOFs indicated that hydrodynamic forces could be used to adjust the systems. A good fit between the simulator and the experimental data was reach by adjusting both the linear and non-linear drag adjusters. The two methods used for the estimation of the added mass, yielded similar results.

The simulated and experimental comparison for yaw DOF indicated that the ellipse method for the added inertia gave better results than the two dimensional method. A good result was given by adjusting both the linear and non-linear drag terms. Two important assumptions were made during the yaw validating process. One was to fix the center of rotation at the center of gravity and the other was to fix the distance were the drag force was acting during the AUV rotation. The points where the drag forces acted on were assumed mid way between cg and either end of the boat.

The simulator capabilities are demonstrated through several design cases. The design cases consider vehicles longer than the simulator is validated for. The simulator is only validated for the Gavia AUV set-up but is used here for demonstration process.



## 7.2 Suggested Future Work

During the development of this simulator, new problems and topics of interest for future research have arisen. Suggested future work is listed below:

- Do a closed loop test of surge movement. It also requires precise examining of thrust equipments input and output during low speed.
- Do same experiments already done, for longer and shorter AUV's.
- Develop a method to optimize the model coefficients.
- Use simulator to do a sensitivity analysis on certain parameters regarding the dynamics.
- Analyse use of low speed tunnel thrusters or vortex ring generators to also assist traditional control equipment during cruise at higher velocity.
- Implement vortex ring generator to thrust model and compare them to similar electrical tunnel thrusters.
- Repeat tests with a perfectly shape ellipse to confirm the ellipse method theory for the added mass.



# Bibliography

- [1] T. Curtis, S. Delanda, J. Denard, D. Koblick, M. Krieg, B. McArdle, S. Reed, T. Schultz, C. Turansky, D. Akos, and S. Biringen. Calamer-E group fall final report. Technical report, University of Colorado, 2006.
- [2] T. Clark, P. Klein, G. Lake, S. Lawrence-Simon, J. Moore, B. Rhea-Carver, M. Sotola, S. Wilson, C. Wolfskill, and A. Wu. Kraken: Kinematically roving autonomously controlled electro-nautic. *47th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Orlando, Florida, January 2009.
- [3] A.B. Phillips, L. Steenson, J. Liu, C.A. Harris, S.M. Sharkh, E. Rogers, S.R. Turnock, and M. Furlong. Delphin2: An over actuated autonomolus underwater vehicle for manoeuvring research. *International Journal of Maritime Engineering*, (In Press), Jan-Mar 2011.
- [4] A. Palmer, G.E. Hearn, and P. Stevenson. Experimental testing of an autonomous underwater vehicle with tunnel thrusters. *First International Symposium on Marine Propulsion*, Throndheim, Norway, September 2009.
- [5] J. Kennedy. *Decoupled modelling and controller design for the hybrid autonomous underwater vehicle: MACO*. PhD thesis, University of Victoria, 2002.
- [6] T.I. Fossen. *Guidance and Control of Ocean Vehicles*. John Wiley & Sons, 1994.
- [7] T.I. Fossen and O.E. Fjellstad. Nonlinear modelling of marine vehicles in 6 degrees of freedom. *Journal of Mathematical Modelling of Systems*, 1, 1995.
- [8] S.B. Williams, P. Newman, G. Dissanayake, J. Rosenblatt, and H. Durrant-Whyte. A decoupled, distributed auv control architecture. *International Symposium on Robotics*, pages 246–251, 2000.

- [9] El-Fakdi Sencianes. *Gradient-based reinforcement learning techniques for underwater robotics behavior learning*. PhD thesis, University of Girona, Girona, Spain, 2011.
- [10] J.E.G. Refsnes. *Nonlinear Model-Based Control of Slender Body AUVs*. PhD thesis, Norwegian University of Science and Technology, Department of Marine Technology, Trondheim, Norway, 2007.
- [11] W. Wang and C.M. Clark. Modeling and simulation of the Videoray Pro III underwater vehicle. *OCEANS 2006-Asia Pacific*, pages 1–7, May 2007.
- [12] P. Ridley, J. Fontan, and P. Corke. Submarine dynamic modeling. *2003 Australian Conference on Robotics and Automation*, Brisbane, Australia, December 2003.
- [13] Baldur Hauksson. Straumfræðigreining á neðansjávar djúpfarinu gavia. Technical report, Reykjavik University, Iceland, September 2009.
- [14] E.A. de Barros, J.L.D. Dantas, A.M. Pascoal, and E. de Sá. Investigation of normal force and moment coefficients for an auv at nonlinear angle of attack and sideslip range. *IEEE Journal of Oceanic Engineering*, 33(4):538–549, 2008.
- [15] Frank M. White. *Fluid Mechanics*. McGraw-Hill, 6 edition, 2008.
- [16] *Simulation and Model-Based Design*, Online, Accessed 05-December-2011. Available: <http://www.mathworks.se/products/simulink/index.html>.
- [17] Jeff Kennedy. Decoupled modelling and controller design for the hybrid autonomous underwater vehicles: Maco. Master’s thesis, Univeristy of Victoria, 2002.
- [18] Les Fenical. *Control Systems Technology*. Thomson Delmar Learning, 1 edition, 2007.
- [19] L. Steenson, A. Phillips, E. Rogers, M. Furlong, and SR Turnock. The performance of vertical tunnel thrusters on an autonomous underwater vehicle operating near the free surface in waves. *Second International Symposium on Marine Propulsors*, June 2011.

# Appendix A

## Equations of motion

### A.1 Dynamics

A more detail derivation of the equations of motion (cf. Section 2.2) used in this master thesis is given by Fossen and others [6, 7, 12]. In the following sections, it will be shown that the 6 degrees of freedom non-linear equation of motion can be expresses as,

$$M\dot{v} + C(v)v + D(v) + g(\eta) = \tau. \quad (\text{A.1})$$

For robot manipulators it is common to transform the desired state trajectory to joint space coordinates by applying the inverse kinematics. Since  $\int_0^t v d\tau$  has no physical interpretation the kinematic equations of motion must be included in the control design if position and attitude control are of interest. For this purpose it is common to apply Euler angle representation (*xyz*-convention)[7],

$$\dot{\eta} = J(\eta)v, \quad (\text{A.2})$$

where  $M$  is the inertia matrix, that also includes the added mass.  $C(v)$  is the matrix of Coriolis and centripetal terms, also including the added mass.  $D(v)$  is the damping matrix.  $g(\eta)$  is the vector of gravitational forces and moments,  $\tau$  is the vector of control input and  $J(\eta)$  is the Euler transformation matrix which projects the vehicles local linear velocity and angle rate vector to the global position and attitude coordinate system [6].

Following sections represents how Eq. A.1 is obtained.

## A.2 Equations of Motion

### A.2.1 Translational Motion

Before deriving the rigid body equation of motion lets look at Fig. A.1. The figure shows a body fixed coordinate system  $X_0Y_0Z_0$  rotating with an angular velocity  $\omega$  about an fixed coordinate system of earth  $XYZ$ . This notation will also be used further through the thesis. To derive the motion we need a relationship between the time derivatives of an arbitrary vector  $c$  in  $XYZ$  and same vector in  $X_0Y_0Z_0$ . It can be done as follows,

$$\dot{c} = \check{c} + \omega \times c, \quad (\text{A.3})$$

where the difference between  $\dot{c}$  and  $\check{c}$  is that  $\dot{c}$  is the time derivative in the earth-fixed reference frame  $XYZ$  and  $\check{c}$  is the time derivative in the moving reference frame  $X_0Y_0Z_0$  [6].

By inserting angular velocity in Eqs. A.3 and A.4 shows that the angular acceleration is equal in the body fixed and earth fixed frames

$$\dot{\omega} = \check{\omega} + \omega \times \omega = \check{\omega}. \quad (\text{A.4})$$

The moment of inertia or the measure of an objects resistance to changes to its rotation has the notation  $I$ . The body inertia tensor ( $I_0$ ) of a body-fixed coordinate system with origin  $O$  in the body fixed frame is

$$I_0 \triangleq \begin{bmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{yx} & I_y & -I_{yz} \\ -I_{zx} & -I_{zy} & I_z \end{bmatrix}. \quad (\text{A.5})$$

The same equation can also be implemented in vector form as

$$I_0\omega = \int_V r \times (\omega \times r) \rho_A dV. \quad (\text{A.6})$$

The mass of the body is defined as

$$m = \int_v \rho_A dV. \quad (\text{A.7})$$

And by assuming that the mass is constant in time ( $\dot{m} = 0$ ), we can also define the center of gravity  $r_G$  as

$$r_G = \frac{1}{m} \int_V r \rho_A dV. \quad (\text{A.8})$$

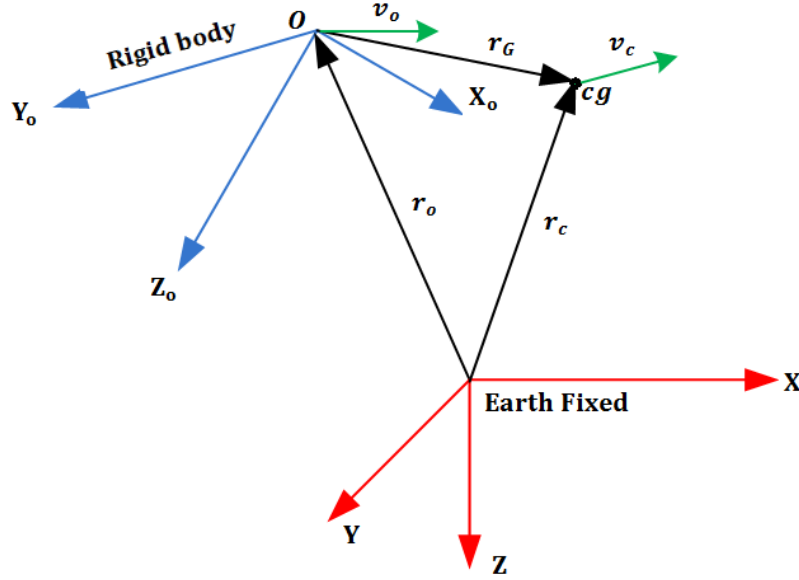


Figure A.1: Body-fixed reference frame and earth-fixed reference frame

From Fig. A.1

$$r_c = r_o + r_G. \quad (\text{A.9})$$

The velocity of CG (center of gravity) is

$$v_c = \dot{r}_c = \dot{r}_o + \dot{r}_G. \quad (\text{A.10})$$

Using the relationship between coordinate systems shown in Eq. A.3

$$\dot{r}_G = \check{r}_G + \omega \times r_G. \quad (\text{A.11})$$

Also using the fact that  $v_o = \dot{r}_o$  and  $\check{r}_G = 0$  for a rigid body we can rewrite velocity Eq. A.10

$$v_c = v_o + \omega \times r_G. \quad (\text{A.12})$$

Then we have the acceleration

$$\dot{v}_C = \dot{v}_0 + \omega \times r_G + \omega \times \dot{r}_G, \quad (\text{A.13})$$

which again can be stated as

$$\dot{v}_C = \ddot{v}_0 + \omega \times v_0 + \dot{\omega} \times r_G + \omega \times (\omega \times r_G). \quad (\text{A.14})$$

### A.2.2 Rotational Motion

To obtain the rotational equations of motion, let's return to the origin  $O$  in Fig. A.1. Then the angular momentum about  $O$  is

$$h_0 \triangleq \int_V r \times v \rho_A dV. \quad (\text{A.15})$$

By differentiating this expression with respect to time, we have

$$\dot{h}_0 = \int_V r \times \dot{v} \rho_A dV + \int_V \dot{r} \times v \rho_A dV, \quad (\text{A.16})$$

where the first term on right hand side is the moment vector which we are interested in,

$$m_0 \triangleq \int_V r \times \dot{v} \rho_A dV. \quad (\text{A.17})$$

From Fig A.1 we see that

$$v = \dot{r}_0 + \dot{r} \Rightarrow \dot{r} = v - v_0. \quad (\text{A.18})$$

Substituting Eqs. A.18 and A.17 into Eq. A.15 and using the fact that  $v \times v = 0$  gives

$$\dot{h}_0 = m_0 - v_0 \times \int_V \dot{r} \rho_A dV, \quad (\text{A.19})$$

which can be rewritten by differentiating Eq. A.8 with respect to time,

$$m \dot{r}_G = \int_V \dot{r} \rho_A dV. \quad (\text{A.20})$$



Using Eq. A.11 and the fact that  $\ddot{r}_G = 0$ , we have  $\dot{r}_G = \omega \times r_G$ . By substituting this to Eq. A.20 gives

$$\int_V \dot{\rho}_A dV = m(\omega \times r_G). \quad (\text{A.21})$$

Substituting this result into Eq. A.19 we have

$$\dot{h}_0 = m_0 - mv_0 \times (\omega \times r_G). \quad (\text{A.22})$$

Finding the absolute angular momentum from Eq. A.15, we have

$$h_0 = \int_V r \times v \rho_A dV = \int_V r \times v_0 \rho_A dV + \int_V r \times (\omega \times r) \rho_A dV. \quad (\text{A.23})$$

From Eq. A.8 the right hand side of Eq. A.23 can be rewritten

$$\int_V r \times v_0 \rho_A dV = \left( \int_V r \rho_A dV \right) \times v_0 = mr_G \times v_0. \quad (\text{A.24})$$

The right hand term or the second term in Eq. A.23 is recognized as the definition of Eq. A.6, Eq. A.23 then reduces to

$$h_0 = I_0 \omega + mr_G \times v_0. \quad (\text{A.25})$$

Assuming that  $I_0$  is constant with respect to time we can differentiate Eq. A.25 according to Eq. A.3

$$\dot{h}_0 = I_0 \ddot{\omega} + \omega \times (I_0 \omega) + m(\omega \times r_G) \times v_0 + mr_G \times (\dot{v}_0 + \omega \times v_0). \quad (\text{A.26})$$

Eliminating  $\dot{h}_0$  from Eqs. A.22 and A.26 by using the relation  $(\omega \times r_G) \times v_0 = -v_0 \times (\omega \times r_G)$ , we have

$$I_0 \ddot{\omega} + \omega \times (I_0 \omega) + mr_G \times (\dot{v}_0 + \omega \times v_0) = m_0. \quad (\text{A.27})$$

Equation A.27 can be simplified by chosen the origin O of the body-fixed coordinate system  $X_0 Y_0 Z_0$  to coincide with the vehicle's center of gravity. The simplified version

is

$$I_0 \ddot{\omega} + \omega \times (I_0 \omega) = m_0. \quad (\text{A.28})$$

We will use the simplified version through this thesis. The notation  $m_0$  is the moment of external forces about O and the notation  $m_C$  is the moment of external forces about CG [6].

### A.3 Governing Equations

Newton's equation of motion, for a rigid body with six degrees of freedom, relative to coordinates attached to the body at  $c_b$ , are  $\sum F = ma_G$  where  $m$  equals the mass of the submarine,  $a_G$  equals the acceleration of the centre of mass and the  $\sum F$  refers to hydrodynamic forces and moments, discussed further in next section of this chapter. Substituting for  $a_G$  from Eq. A.14 gives the following force equations in the  $X_{sub}$ ,  $Y_{sub}$  and  $Z_{sub}$  directions [12]

$$\begin{aligned} m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] &= \sum X_{ext} \\ m[\dot{v} - wp + ur - y_G(r^2 + p^2) + z_G(qr - \dot{p}) + x_G(qp + \dot{r})] &= \sum Y_{ext} \\ m[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q}) + y_G(rq + \dot{p})] &= \sum Z_{ext} \end{aligned} \quad (\text{A.29})$$

To find the moment equation in the  $X_{sub}$ ,  $Y_{sub}$  and  $Z_{sub}$  directions, Euler equation of motion are required. For a rigid body with six degrees of freedom, relative to coordinates attached to the body at CG, the sum of moment is

$$\sum M_B = \dot{H}_G + r_G \times ma_G. \quad (\text{A.30})$$

Equation A.28 gives the rate of change, of angular momentum about the centre of gravity. By substituting that equation to Eq. A.30 but neglecting small terms (e.g.  $x_G^2$ ) we have the moment equation in the  $X_{sub}$ ,  $Y_{sub}$  and  $Z_{sub}$  directions [12]

$$\begin{aligned} I_x \dot{p} + (I_z - I_y)qr + m[y_g(\dot{w} - uq + vp) - z_G(\dot{v} - wp + ur)] &= \sum K_{ext} \\ I_y \dot{q} + (I_x - I_z)rp + m[z_g(\dot{u} - vr + wq) - x_G(\dot{w} - uq + vp)] &= \sum M_{ext} \\ I_z \dot{r} + (I_y - I_x)pq + m[x_g(\dot{v} - wp + ur) - y_g(\dot{u} - vr + wq)] &= \sum N_{ext} \end{aligned} \quad (\text{A.31})$$

Now we have derived both the force and moment equation for a rigid body with 6 degrees of freedom. The governing equation for a system is the equation which determines the system's motion. Then the governing equation of motion for 6 DOF Rigid body like AUV

are shown in Eq. A.32. The three first equation represents the translational motion while the three last represents the rotational motion,

$$\begin{aligned}
 m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] &= \sum X_{ext} \\
 m[\dot{v} - wp + ur - y_G(r^2 + p^2) + z_G(qr - \dot{p}) + x_G(qp + \dot{r})] &= \sum Y_{ext} \\
 m[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q}) + y_G(rq + \dot{p})] &= \sum Z_{ext} \\
 I_x \dot{p} + (I_z - I_y)qr + m[y_g(\dot{w} - uq + vp) - z_G(\dot{v} - wp + ur)] &= \sum K_{ext} \\
 I_y \dot{q} + (I_x - I_z)rp + m[z_g(\dot{u} - vr + wq) - x_G(\dot{w} - uq + vp)] &= \sum M_{ext} \\
 I_z \dot{r} + (I_y - I_x)pq + m[x_g(\dot{v} - wp + ur) - y_G(\dot{u} - vr + wq)] &= \sum N_{ext}
 \end{aligned} \tag{A.32}$$

Equation A.32 can be simplified to a compact vector form

$$M_{RB}\dot{v} + C_{RB}(v)v = \tau_{RB}, \tag{A.33}$$

where  $M_{RB}$  represents the rigid body inertia matrix,  $v = [u, v, w, p, q, r]^T$  represents the body-fixed linear and angular velocity vector and  $\tau_{RB} = [X, Y, Z, K, M, N]^T$  represents the vector of hydrodynamic Forces and Moments discussed next [6].



## **Appendix B**

### **Open Loop Version of the Simulator**



Figure B.1: Surge open loop

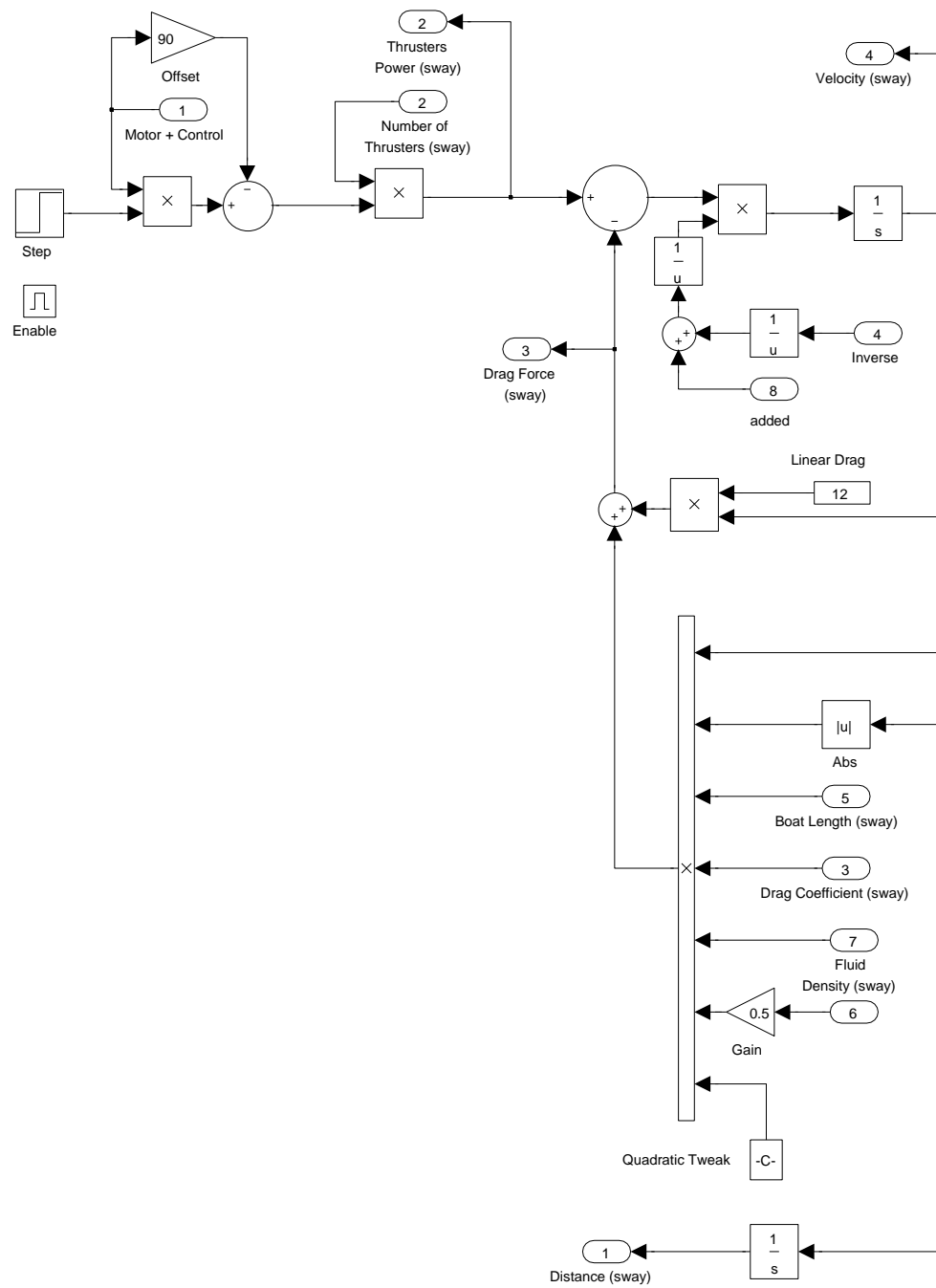


Figure B.2: Sway open loop

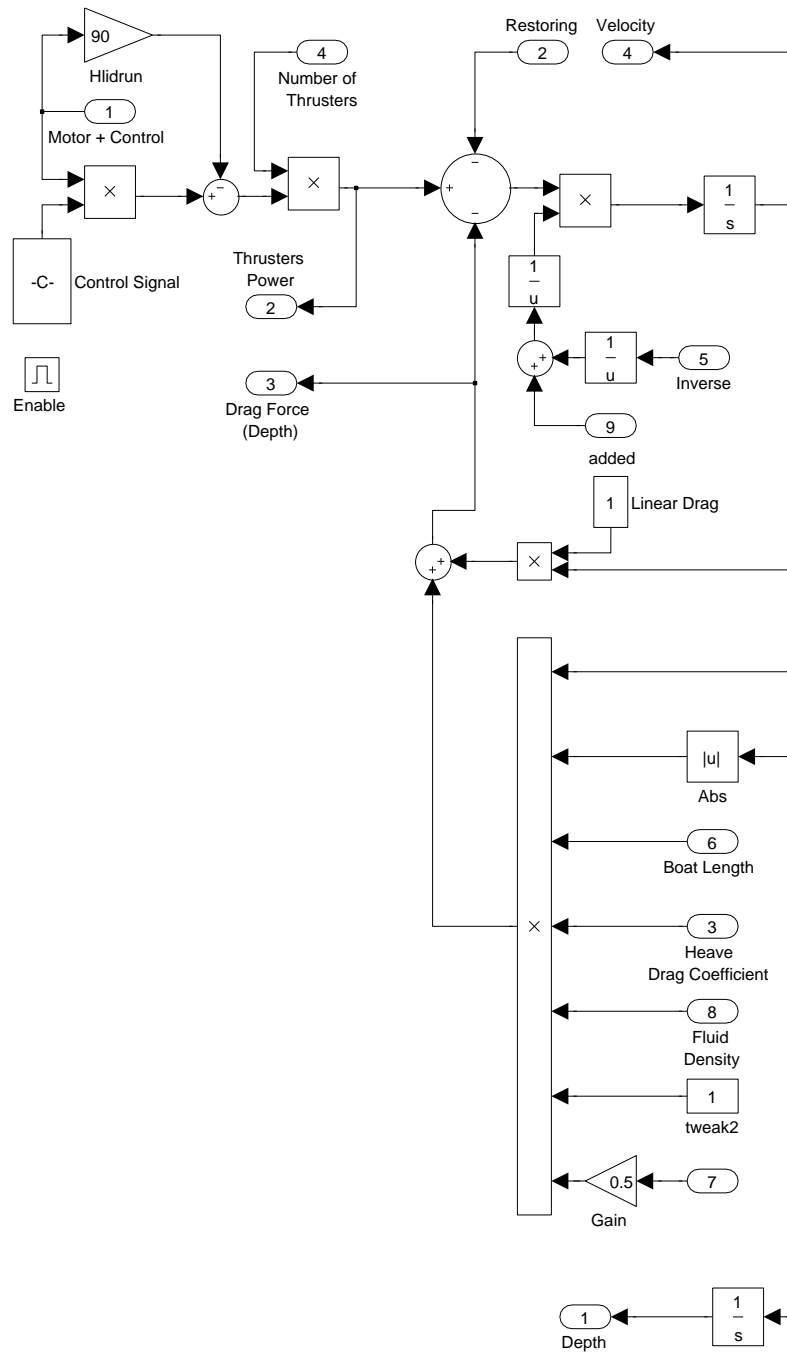


Figure B.3: Heave open loop





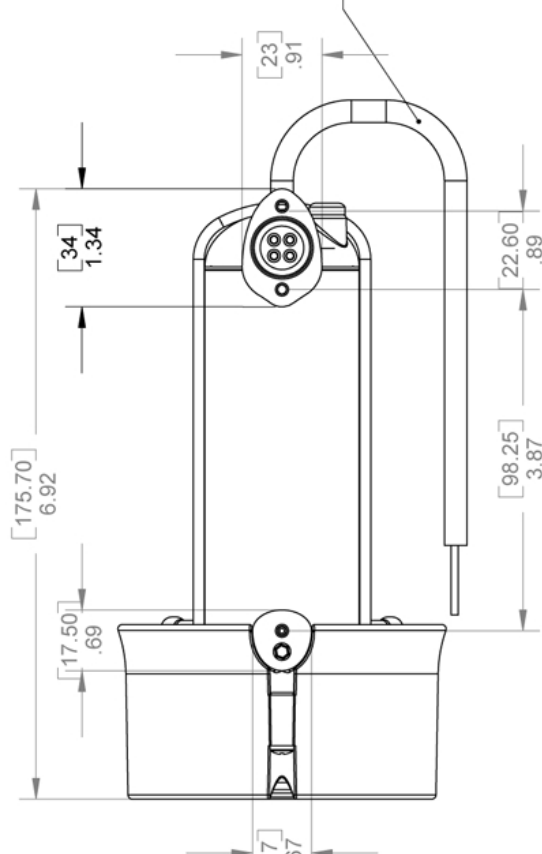
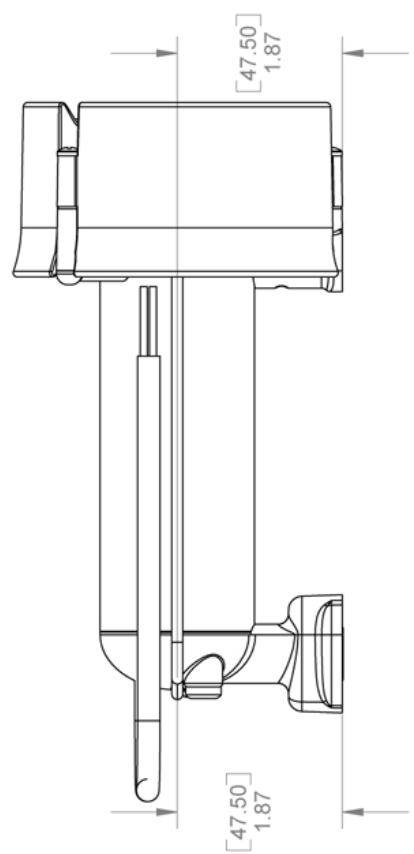
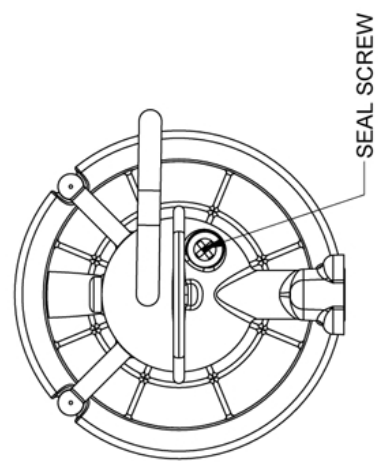
Figure B.4: Yaw open loop



## **Appendix C**

### **Test Equipment Specification**

#### **C.1 Seabotix Thruster Specification**



# BTD150 SPECIFICATIONS

## DEPTH RATING

FRESH WATER:  
150 METERS - 500 FEET

## MOUNTING HARDWARE

QTY: 3 (INCLUDED)  
TYPE: SHEET METAL SCREW, PAN HEAD PHILLIPS, 316 SS  
SIZE: #8 X 3/4"  
MCMASTER P/N: 90184A152 (OR EQUIVALENT)  
SEABOTIX P/N: FN308

## ELECTRICAL INTERFACE

VOLTAGE: +19.1V DC  $\pm 10\%$   
POWER: 110W MAXIMUM (DEPENDING UPON RPM OR DRAG)  
MAX AMPERAGE: 5.8 AMPS (30 SECOND DURATION)  
MAX CONTINUOUS AMPERAGE: 4.25 AMPS  
CABLE: 39IN (1 M)  
2 WIRE COLOR CODED

## WEIGHT

IN AIR: 705 GRAMS  
IN FRESH WATER: 350 GRAMS

## PERFORMANCE

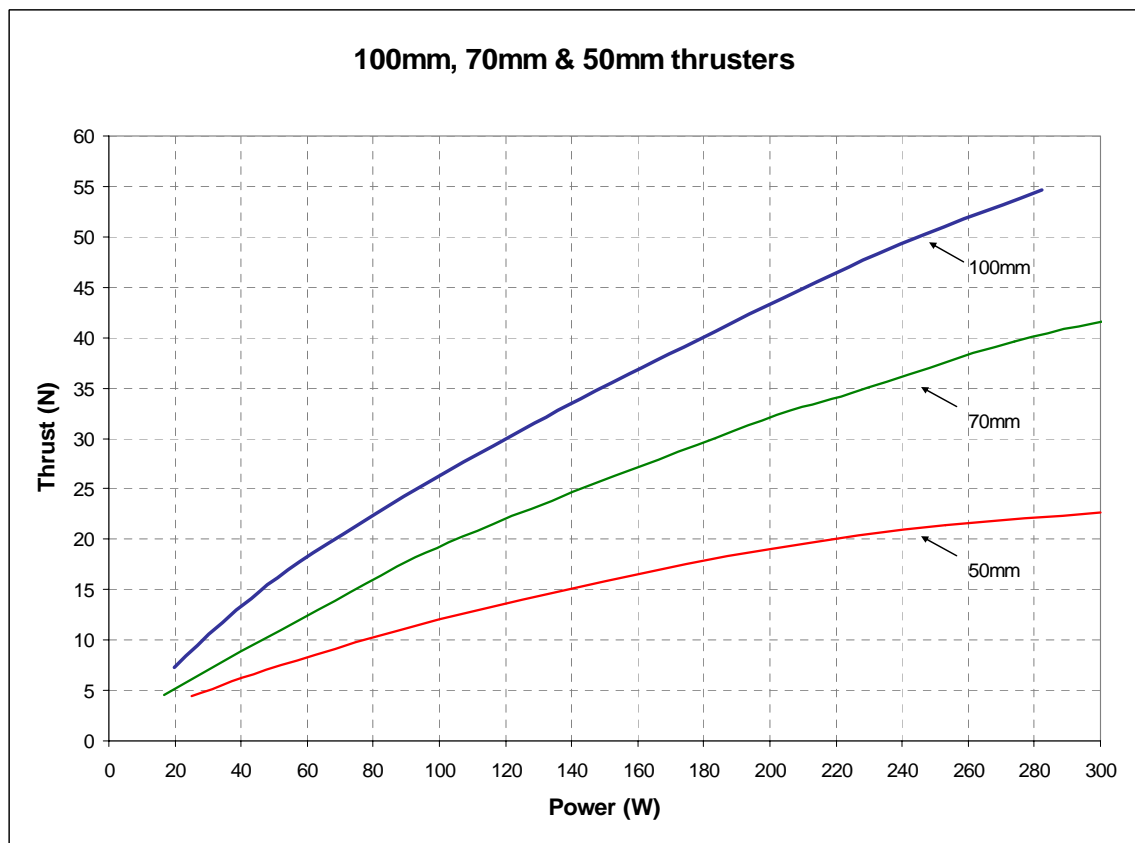
PEAK BOLLARD THRUST: 2.9 KGf (6.4 FT/LBS)  
CONTINUAL BOLLARD THRUST: 2.2 KGf (4.85 FT/LBS)

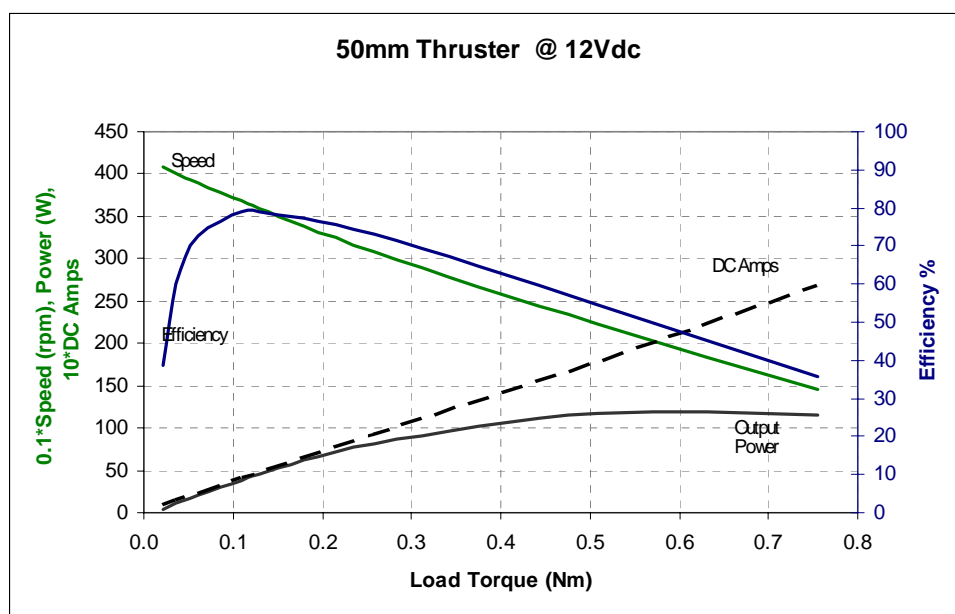
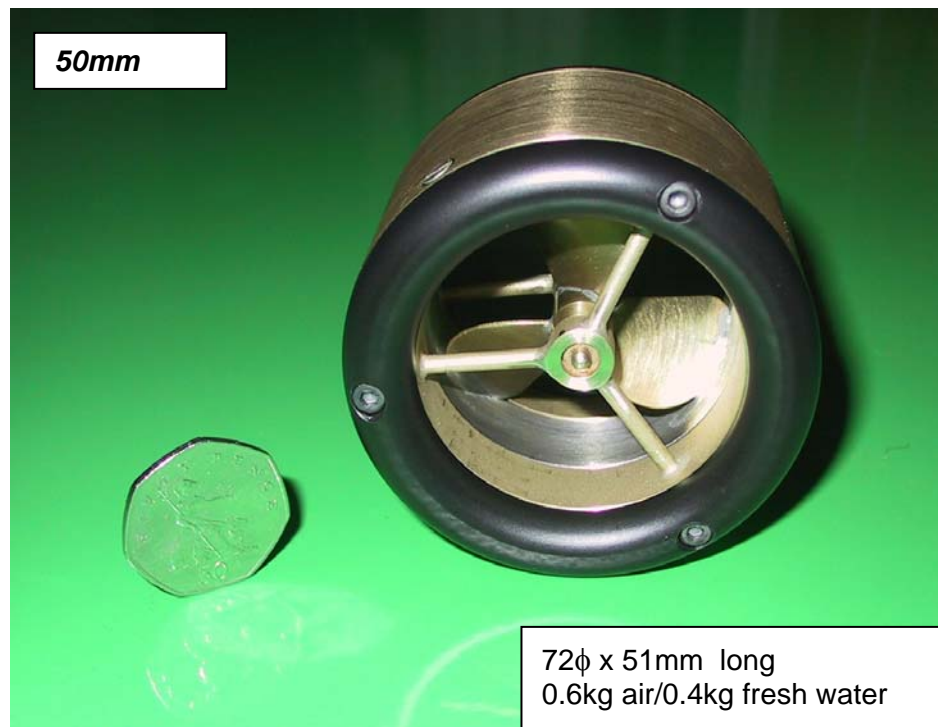
1 M (39") 2 WIRE CABLE

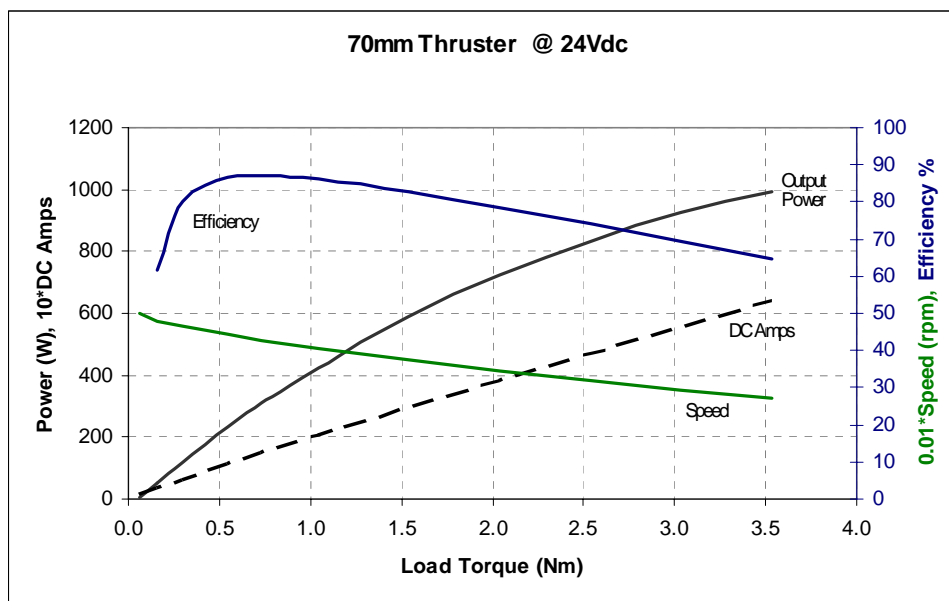
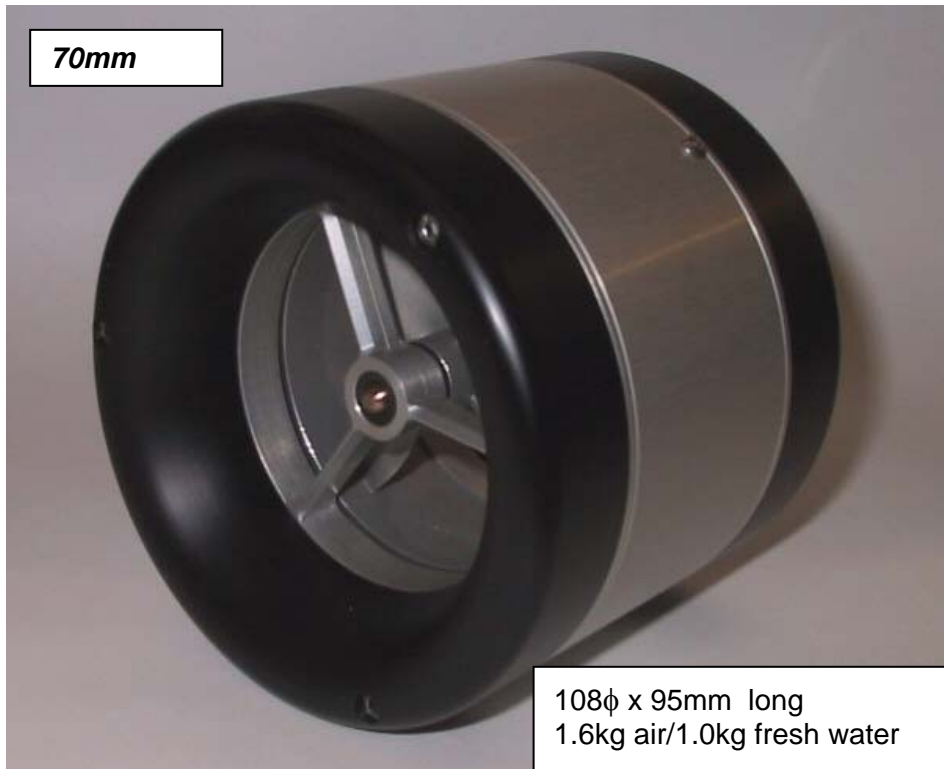
MATERIAL	UNLESS OTHERWISE SPECIFIED:			NAME	DATE
	DIMENSIONS ARE IN INCHES	DRAWN	J Rodocker	1 JAN07	SeaBotix Inc. 1425 Russ Blvd Suite T112D San Diego, CA 92101
	TOLERANCES:	CHECKED			
FINISH	ANGULAR: MACH 1" BEND $\pm 1^\circ$ ONE PLACE DECIMAL $\pm .100$ TWO PLACE DECIMAL $\pm .010$ THREE PLACE DECIMAL $\pm .005$	ENG. APPR.			
PROPRIETARY AND CONFIDENTIAL THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF SEABOTIX INC. ANY REPRODUCTION IN WHOLE OR IN PART WITHOUT THE WRITTEN PERMISSION OF SEABOTIX INC. IS PROHIBITED. © 2005 SEABOTIX INC.					
COMMENTS: UNLESS OTHERWISE SPECIFIED BREAK ALL EDGES R/C 0.01 DO NOT SCALE DRAWING. ALL DIMENSIONS APPLY AFTER FINISH.					
TITLE: Standard Thruster & 2 wire whip			SIZE	PART NO.	REV
			B	BTD150	A
			SCALE: 1:5	WEIGHT:	SHEET 1 OF 1

## **C.2 TSL Technology Tunnel Thruster Specification**

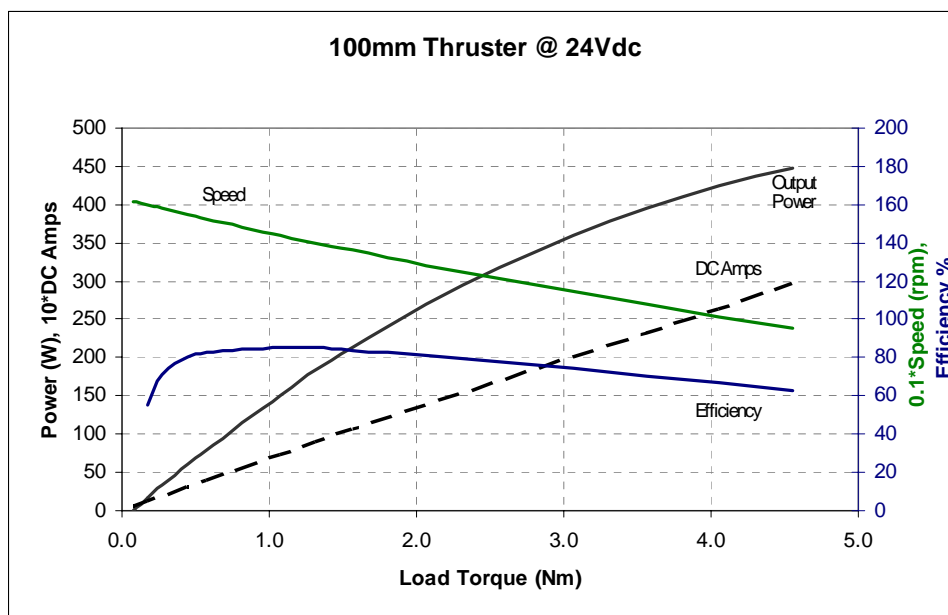
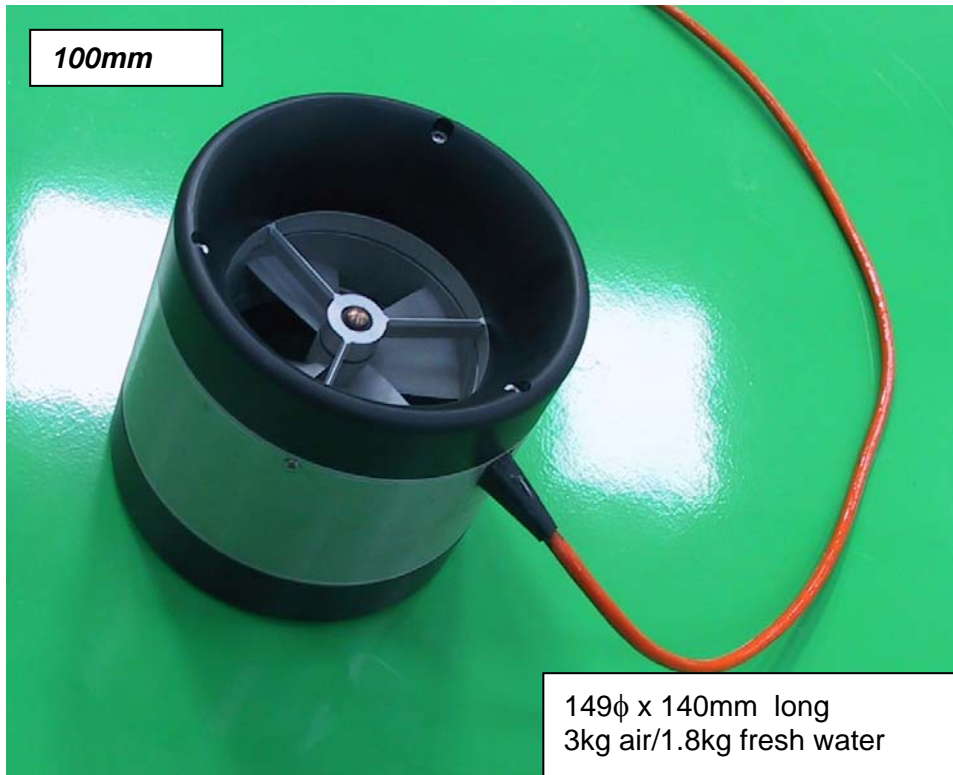
## IntegratedThruster™ Thrust Curves











### **C.3 Arduino Mega Specification**

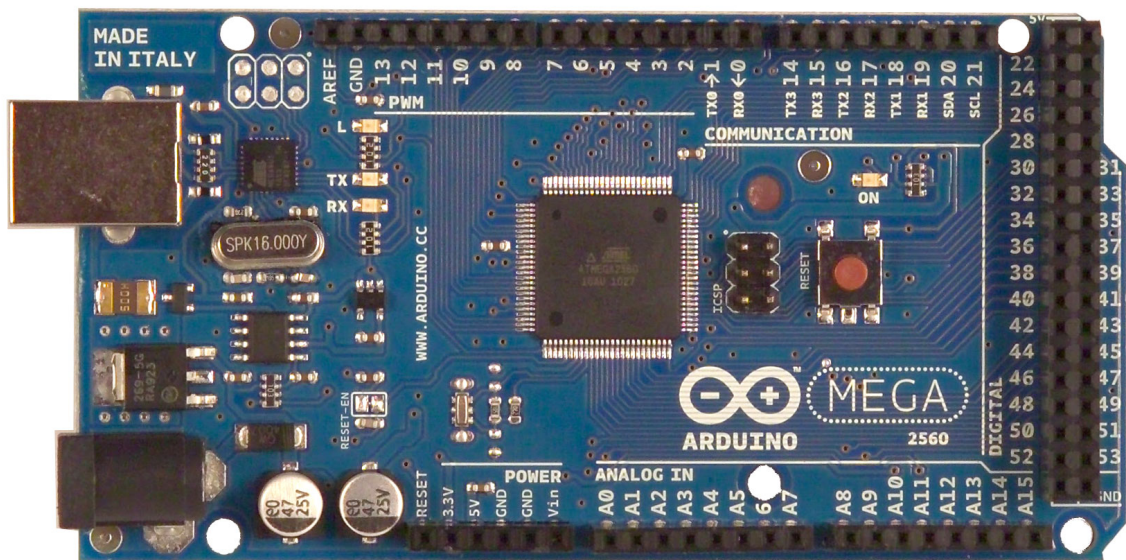


[www.robotshop.com](http://www.robotshop.com)

La robotique à votre service! - Robotics at your service!



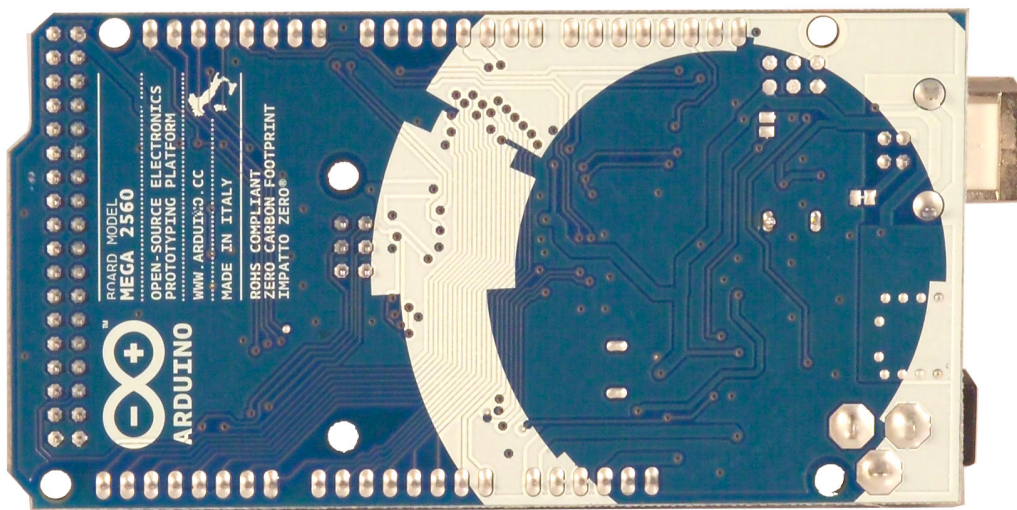
## Arduino Mega 2560 Datasheet





www.robotshop.com

La robotique à votre service! - Robotics at your service!



## Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 ([datasheet](#)). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

## Schematic & Reference Design

EAGLE files: [arduino-mega2560-reference-design.zip](#)



Schematic: [arduino-mega2560-schematic.pdf](#)

## Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

## Power

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.





The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

## Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the [EEPROM library](#)).

## Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using [pinMode\(\)](#), [digitalWrite\(\)](#), and [digitalRead\(\)](#) functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the [attachInterrupt\(\)](#) function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the [analogWrite\(\)](#) function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication using the [SPI library](#). The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH



value, the LED is on, when the pin is LOW, it's off.

- **I<sup>2</sup>C: 20 (SDA) and 21 (SCL).** Support I<sup>2</sup>C (TWI) communication using the [Wire library](#) (documentation on the Wiring website). Note that these pins are not in the same location as the I<sup>2</sup>C pins on the Duemilanove or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and `analogReference()` function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with [analogReference\(\)](#).
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

## Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically). The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A [SoftwareSerial library](#) allows for serial communication on any of the Mega2560's digital pins.

The ATmega2560 also supports I<sup>2</sup>C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I<sup>2</sup>C bus; see the [documentation on the Wiring website](#) for details. For SPI communication, use the [SPI library](#).

## Programming

The Arduino Mega can be programmed with the Arduino software ([download](#)). For details, see the [reference](#) and [tutorials](#).

The ATmega2560 on the Arduino Mega comes preburned with a [bootloader](#) that allows you to upload new code to it without the use of an external hardware programmer. It



communicates using the original STK500 protocol ([reference](#), [C header files](#)). You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see [these instructions](#) for details.

## Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega2560 contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see [this forum thread](#) for details.

## USB Overcurrent Protection

The Arduino Mega2560 has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

## Physical Characteristics and Shield Compatibility





The maximum length and width of the Mega2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega2560 and Duemilanove / Diecimila. *Please note that I<sup>2</sup>C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).*

## **C.4 DCDC Mini-Box Regulator Specification**

# **DCDC-USB**

**6-34V 10A, Intelligent DC-DC converter with USB interface**

## **Quick Installation Guide**

Version 1.0c  
P/N DCDC-USB

### **Introduction**

The DCDC-USB is a small yet powerful DC-DC power supply designed to power a wide variety of devices. This DC-DC has a range of intelligent functions not found in any tradition USB converters. Features include USB interface, programmable output voltage and scripting as well as automotive modes.

The DCDC-USB device is able to send ON/OFF 'pulse signals' to motherboards based on voltage levels or Ignition sensing, making it an ideal device for automotive or battery powered installations.

This unit has a wide input range (6-34V) and it can provide a tightly regulated output ranging from 6 to 24V (default set to 12V).

## Quick installation Instructions

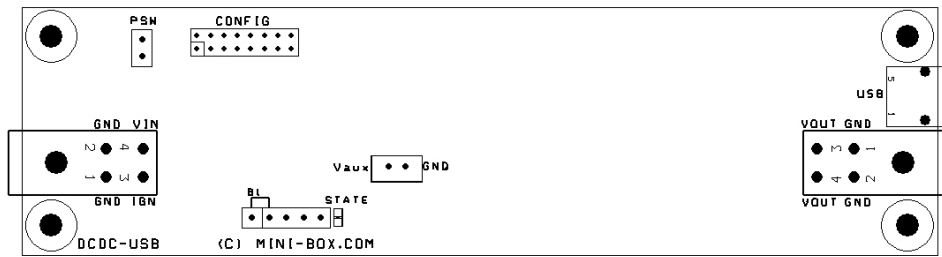


Fig 1.0, DCDC-USB layout

**CONFIG:** Configuration jumpers for Voltage, Mode and Timings.

**Left mini-FIT connector:** Power input, V(in), GND, Ignition.

**Right mini-FIT connector:** Power output V(out), GND.

**PSW:** Soft ON/OFF control for motherboard. Connect this to motherboard ON/OFF pins if you want the motherboard to be controlled by the unit.

**USB:** mini-USB type B jack. Connect this to a PC to access advanced settings.

**STATE:** State LED

**Vaux:** Provides unregulated switched input, to be used in automotive modes to power various peripherals.

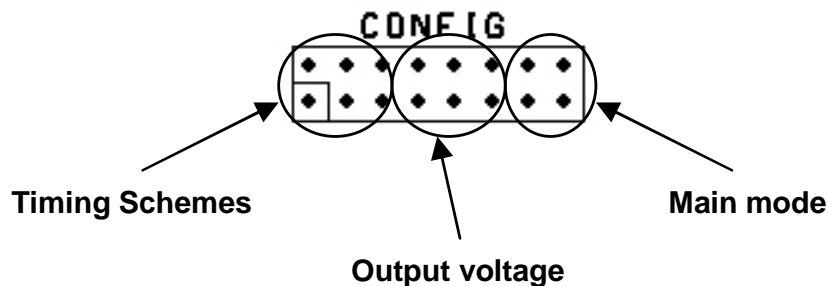
## Basic Operation

For basic operation, you would need to connect a power source to V(IN). V(IN) is on the left side of the board, near the 10A input fuse. Polarity is marked on the PCB (GND, VIN and Ignition). On the cable harness, GND is black, V(in) is red and Ignition is white. NOTE: Ignition is not needed for basic operation.

Without any further settings, V(out) will generate 12V regulated. V(out) is located on the right side of the PCB, near the USB connector. On the cable harness, Yellow is positive and GND is negative.

## Configuration jumpers

The configuration header (maked CONFIG) is the most important header in this board. It is divided in 3 sections:



**Main mode.** This header section sets one of the 3 main mode of operation: DUMB mode, Automotive mode and the SCRIPT mode.

**Output voltage.** This header section sets the output voltage of the unit.

**Timing Schemes.** This header section sets the OFFDELAY and HARDOFF timers, only available in the Automotive mode.

### Configuration, voltage settings

By default, the DCDC-USB module provides regulated 12V output. Should you need other voltage levels, you can change output voltages by setting jumpers 2, 3 and 4, see table 1.

After making a jumper change, the DCDC-USB unit needs to be power cycled in order for the new setting to take effect. **NOTE:** Finer voltage adjustments are available via USB settings, see Advanced USB Configuration manual.

Output voltage							
	0	1	2	3	4	5	6 7
12V							
5V							
6V							
9V							
13.5V							
16V							
19V							
24V							

### Configuration, Mode of Operation

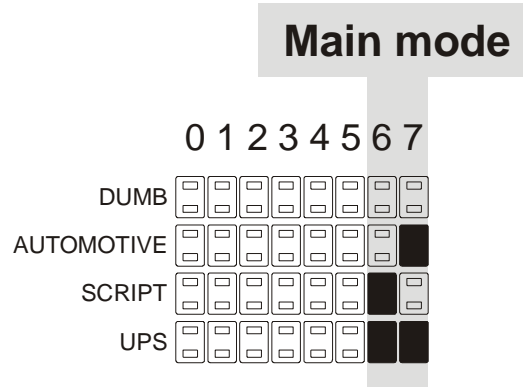
DCDC-USB has 4 modes of operation. This modes can be changed via jumpers.

**1) DUMB mode.** The units acts as a regular DC-DC converter. Only V(In) and V(out) and GND terminal are required. Unit will convert any input from 6-34V to a fixed voltage. Default voltage is set to 12V.

**2) AUTOMOTIVE MODE.** In this mode, the unit acts as as Intelligent DC-DC converter that is aware of Ignition state. In this mode the unit reads the Ignition terminal and based on Ignition statuts the unit sends ON/OFF pulses to a motherboard in order to start or stop. In this mode, two variables can be set: OFFDELAY and HARDOFF. See Default Timing Schemes for more information.

**3) SCRIPT MODE.** This is an advanced mode where unit can be scripted to perform various tasks based on user scripts. Please refer to Advanced USB programming manual for more details.

**4) UPS MODE.** Module is a DC UPS (Uninterruptible Power Source) module.



### Default Timing Schemes

When unit is set to operate in the Automotive Mode, 8 timing settings are available. These are combinations of OFFDELAY and HARDOFF timers. NOTE: These settings work only when the Ignition wire is used.

**OFFDEALAY** is the amount of time the unit waits until it sends an ON/OFF pulse the motherboard's ON/OFF pins after Ignition has been turned off.

**HARDOFF** is the amount of time the unit will still provide power after the ON/OFF pulse has been sent to the motherboard. If you have a battery sensitive application, set the HARDOFF to 1 minute to avoid battery drain. While in HARDOFF, the unit carefully monitors the battery and if battery voltage goes under 11.2V, power will be cut off in order to prevent battery drainage.

		<b>Default timing schemes</b>							
OffDelay	HardOFF	0	1	2	3	4	5	6	7
0	0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5s	60s	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5s	Never	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1min	1min	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
15 min	1min	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
15 min	Never	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
30 min	1 min	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2 hour	Never	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

### Electrical and Environmental Specifications

Minimum Input Operating voltage	6V
Maximum input Operating voltage	Electronic shut down at 34V (clamping will occur 34-36V)
Deep-Discharge shutdown threshold	11.2V
Input current limit (fuse protected)	10A (10A mini-blade fuse)
Max Output Power	100watts (limited by 10A input fuse) *
Regulation accuracy	2.5%
Operating temperature	-40 to +85 degrees Celsius
Storage temperature	-55 to +85 degrees Celsius
MTBF	>100,000 hrs @ 65C body temp.
Efficiency (Input 9-16V)	>95% (output = 12V 2A)
PCB size	135mx37mm
Input, output connectors	Right angle, Mini-FIT JR 4 pin

\*NOTE: At output power greater than 40watts or if unit temperature exceeds 65C, forced air ventilation will be required in order to prevent unit from excessive thermal stress for long period of times.

### 12V output max current charts

Input (V)	12V rail current	Input (V)	12V rail current
6V	4A	11V	8A
7V	5A	12V	8A
8V	6A	14V	8A (10A peak)
9V	7A	14-18V	8A (10A peak)
10V	8A	20-26V	8A

**Operating environment:** Temperature: -20 to 65 degree centigrade.  
Relative Humidity: 10 to 90 percent, non-condensing.

**Efficiency, MTBF:** MTBF >100K hours at PSU(temp) < 65 Celsius.  
NOTE: All solid polymer capacitor design, rated >50K hours at 85C or 500K hours at 65C.

**Shipping and storage:** Temperature -40 to +85 degree centigrade. Relative humidity 5 to 95 percent, non-condensing.

### Warranty

1 Year Limited Warranty statement. Warranty is void if maintenance or calibration is performed by end-user or by use in conjunction with power modules not provided by mini-box.com.

### Support

Email: [support@mini-box.com](mailto:support@mini-box.com)

Web Site: <http://www.mini-box.com>

## **C.5 Motor Controller Devantech MD22 Specification**



# MD22 - Dual 24Volt 5Amp H Bridge Motor Drive

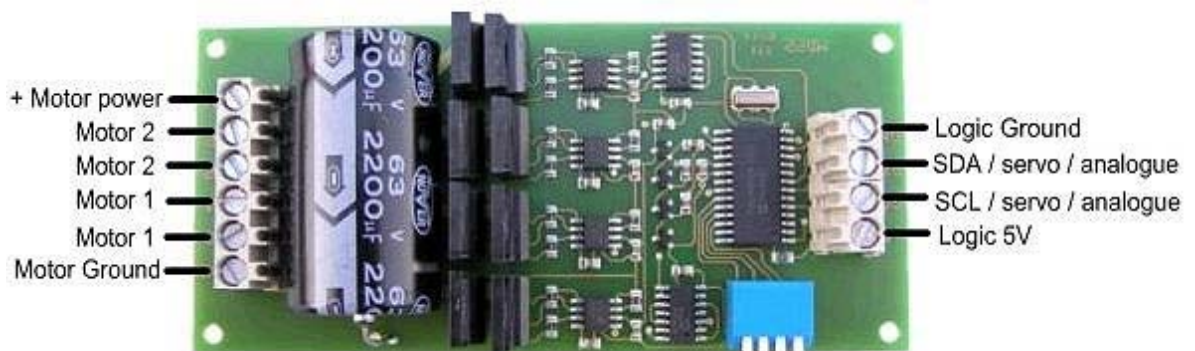
## Overview

The MD22 is a robust low/medium power motor driver, designed to supply power for two motors. Main features are:

1. Drives two motors with independent control.
2. Ease of use and flexibility.
3. The 15v MOSFET drive voltage is generated onboard with a charge pump, so the module requires only two supply voltages;
  - a) A standard 5V supply for the control logic, only 50mA maximum is required.
  - b) The H-Bridge has a rating of 60v allowing Motor voltages up to 24vdc.
4. Steering feature, motors can be commanded to turn by I2C register or input (Analogue + Servo).
5. Control of the module can be any of;
  - a) I2C bus, up to 8 MD22 modules, switch selectable addresses and 4 modes of operation including steering..
  - b) 2 independent 0v-2.5v-5v analog inputs. 0v full reverse, 2.5v center stop, 5v full forward.
  - c) 0v-2.5v-5v analog input for speed ,with the other channel for steering.
  - d) independent channel RC mode. Motors are individually Controlled directly from the RC receiver output.
  - e) RC mode with steering, allows speed control with one stick of radio control, and steering with the other.
6. Uses high current MOSFETs, making a very robust module.

## MD22 Connections

**Note -**  
**There is**  
**no fuse on**  
**the PCB.**  
**You should**  
**provide**  
**a 10A fuse**  
**in line with**  
**the +v**  
**battery**  
**terminal.**  
  
**Don't**  
**Ignore**  
**this, High**  
**currents**  
**can be**  
**dangerous!**



The Motor Ground and the Logic Ground are internally connected on the Module. Be sure to use cable rated for at least 10A for the Battery, Fuse and Motor leads.

## Motor Noise Suppression

Please note that using motors with the MD22 as with any other electronic device requires suppression of noise. This is easily achieved by the addition of a 10n snubbing capacitor across the motors. The capacitor should also be capable of handling a voltage of twice the drive voltage to the motor.

## Mode Switches

The 4 mode switches set the operating mode of the MD22. They are read once only when the module is powered up. You cannot switch modes while the unit is on.

Mode	Switch 1	Switch 2	Switch 3	Switch 4
I2C Bus - address 0xB0	On	On	On	On
I2C Bus - address 0xB2	Off	On	On	On
I2C Bus - address 0xB4	On	Off	On	On
I2C Bus - address 0xB6	Off	Off	On	On
I2C Bus - address 0xB8	On	On	Off	On
I2C Bus - address 0xBA	Off	On	Off	On
I2C Bus - address 0xBC	On	Off	Off	On
I2C Bus - address 0xBE	Off	Off	Off	On
0v - 2.5v - 5v Analog	On	On	On	Off
0v - 2.5v - 5v Analog + Turn	Off	On	On	Off
RC Servo	On	Off	On	Off
RC Servo + Turn	Off	Off	On	Off

## New modes from version 3(Dec 2004)

RC Servo, timeout on	On	On	Off	Off
RC Servo + turn, timeout on	Off	On	Off	Off

## New modes from version 9(Mar 2006)

Analogue turn mode 2	On	Off	Off	Off
RC Servo, turn mode 2, timeout on	Off	Off	Off	Off

Note that I2C addresses are the upper 7 bits. Bit 0 the the read/write bit, so addresses 0xB0/0xB1 are write/read respectively to the same address.

This range of I2C addresses is the same as those used by the MD03.

## Analog Mode - 0v-2.5v-5v

In this mode the motors are controlled independently by two 0v to 5v analog signal on the SCL (Motor1) and SDA (Motor2) lines.

0v is maximum reverse power

2.5v is the center stop position

5v is full forward power

There is a small (2.7%) dead band around 2.5v to provide a stable off position.

## Analog Mode - 0v-2.5v-5v with Differential drive

Both Motors speed is now controlled by the analogue voltage level on the SCL line. The SDA line is now responsible for offsetting the two speeds and thus controlling the degree of turn.

The voltage levels are the same as above but turn degree is:

0v is hard turn left

2.5v is the straight position

5v is hard turn right

There is the same dead band (2.7%) on the speed and the turn. .

## RC Servo Mode

This mode allows direct connection to standard model radio control receivers. Most receivers work from a 4.8v-6v battery pack and can be powered by 5v supply that powers the MD22 logic. The control pulses (Yellow) from the receiver should be connected to the SCL (Motor1) and SDA (Motor2) terminals. Connect the receiver supply (Red) to +5v logic supply and the receiver 0v ground (Black) to the MD22 logic ground. The output from an RC receiver is a high pulse 1.5mS wide when the joystick is central. The MD22 provides full control in the range 1mS to 2mS with 1.5mS being the center off position. There is a 7uS dead zone centered on 1.5mS for the off position. The Radio Transmitter centering control should be adjusted so that the motor is off when the joystick is released.

## RC Servo Mode with Differential drive

Again uses a standard radio control receiver module output to determine speed with the addition of the extremely useful steering function. the receivers Forward and Reverse channel should be wired to the SCL connection. And the steering (turn) through the SDA channel. Again fine adjustment to the transmitters offset may possibly be needed.

## RC Modes with timeout feature (from version 3)

An extra couple of modes have been added and operate in much the same way as the normal servo control. The difference is the addition of a new timeout feature. If the RC pulse is not detected on a channel 1 (SCL) for a period in excess of 200ms, then both of the motors will be stop being driven until a valid RC signal is received on channel 1.

## I2C Mode

I2C mode allows the MD22 to be connected to popular controllers such as the PICAXE, OOPic and BS2p, and a wide range of micro-controllers like PIC's, AVR's, 8051's etc.

I2C communication protocol with the MD22 module is the same as popular eeprom's such as the 24C04. To read one or more of the MD22 registers, first send a start bit, the module address (0XB0 for example - see mode switches) with the read/write bit low, then the register number you wish to read. This is followed by a repeated start and the module address again with the read/write bit high (0XB1 in this example). You are now able to read one or more registers. The MD22 has 8 registers numbered 0 to 7 as follows;

Register Address	Name	Read/Write	Description
0	Mode	R/W	Mode of operation (see below)
1	Speed	R/W	Left motor speed (mode 0,1) or speed (mode 2,3)
2	Speed2/Turn	R/W	Right motor speed (mode 0,1) or turn (mode 2,3)
3	Acceleration	R/W	Acceleration for i2c (mode 0,1)
4	Unused	Read only	Read as zero
5	Unused	Read only	Read as zero
6	Unused	Read only	Read as zero
7	Software Revision	Read only	Software Revision Number

The mode register defaults to 0, as does the acceleration register (slowest acceleration). No motor will move until directly after speed or speed2/turn registers are changed.

## Mode Register

The mode register selects which mode of operation and I2C data input type the user requires. The options being:

**0**, (Default Setting) If a value of 0 is written to the mode register then the meaning of the speed registers is literal speeds in the range of:

0 (Full Reverse) 128 (Stop) 255 (Full Forward).

- 1, Mode 1 is similar to Mode 0, except that the speed registers are interpreted as signed values. The meaning of the speed registers is literal speeds in the range of:  
-128 (Full Reverse) 0 (Stop) 127 (Full Forward).
- 2, Writing a value of 2 to the mode register will make speed control both motors speed. Speed2 then becomes the turn value (type 1).  
Data is in the range of 0 (Full Reverse) 128 (Stop) 255 (Full Forward).  
note - version 8+ speed controls the total power, the turn (speed 2) value is now with reference to this.
- 3, Mode 3 is similar to Mode 2, except that the speed registers are interpreted as signed values.  
Data is in the range of -128 (Full Reverse) 0 (Stop) 127 (Full Forward)  
note - version 8+ speed controls the total power, the turn (speed 2) value is now with reference to this.
- 4, (New from version 9) Alternate method of turning (type 2), the turn value being able to introduce power to the system.  
Data is in the range of 0 (Full Reverse) 128 (Stop) 255 (Full Forward).
- 5, (New from version 9) Alternate method of turning (type 2), the turn value being able to introduce power to the system.  
Data is in the range of -128 (Full Reverse) 0 (Stop) 127 (Full Forward)

### Speed Register

Depending on what mode you are in, this register can affect the speed of one motor or both motors. If you are in mode 0 or 1 it will Set the speed of the motor 1. The larger the number written to this register, the more power is applied to the motor. If mode is set to a turn mode it controls the speed and direction of both motors (subject to effect of turn register).

### Speed2/Turn Register

Again when in mode 0 or 1 this register operates the same as speed but controls the operation of the motor 2. When a turn mode is selected Speed2 becomes a Turn register, and any value in speed 1 is combined with the contents of this register to steer the device.

### Turn mode (up to version 7)

In software versions up to 7, the turn modes look at the speed channel or register to decide if the direction is forward or reverse. They then apply a subtraction or addition of the turn value on either motor.

so if the direction is forward  
 $\text{motor speed1} = \text{speed} - \text{turn}$   
 $\text{motor speed2} = \text{speed} + \text{turn}$

else the direction is reverse so  
 $\text{motor speed1} = \text{speed} + \text{turn}$   
 $\text{motor speed2} = \text{speed} - \text{turn}$

If the either motor is not able to achieve the required speed for the turn (beyond the maximum output), then the other motor is automatically changed by the program to meet the required difference.

### Turn Mode ( Version 8+ )

In turn mode 1 the power supplied to the motors is always with reference to the speed. Turn values are only applied with respect to the speed. The turn factor is determined by the equation below, where speed\_max is a program constant equating to the maximum possible motor speed.

$\text{turn factor} = \text{turn} * (\text{speed}/\text{speed\_max})$

And now the power to the motors can be calculated, remembering that a turn in either direction in a forward direction is the inverse in the reverse direction so:

if we are moving forwards and require a turn then

$\text{motor speed1} = \text{speed} - \text{turn factor}$

$\text{motor speed2} = \text{speed} + \text{turn factor}$

else if we are moving in reverse and require a turn then

$\text{motor speed1} = \text{speed} + \text{turn factor}$

$\text{motor speed2} = \text{speed} - \text{turn factor}$

If the either motor is not able to achieve the required speed for the turn (beyond the maximum output), then the other motor is automatically changed by the program to meet the required difference.

### Turn Mode 2 ( Version 9+ )

In turn mode 2 there is no link between the turn factor and the speed, this means when speed is at zero you can still turn. With this method is the backwards turns are inverted (left is right). The turn factor is now just :

$\text{turn factor} = \text{turn}$

And the power to the motor is now :

$\text{motor speed1} = \text{speed} - \text{turn factor}$

$\text{motor speed2} = \text{speed} + \text{turn factor}$

If the either motor is not able to achieve the required speed for the turn (beyond the maximum output), then the other motor is automatically changed by the program to meet the required difference.

### Acceleration Register (in direct motor control)

If you require a controlled acceleration period for the attached motors to reach there ultimate speed, the MD22 has a register to provide this. It works by inputting a value into the acceleration register which acts as a delay in the power stepping. The amount of steps is the difference between the current speed of the motors and the new speed (from speed 1 and 2 registers). So if the motors were traveling at full speed in the forward direction (255) and were instructed to move at full speed in reverse (0), there would be 255 steps.

The acceleration register contains the rate at which the motor board moves through the steps. At 0 (default) the board changes the power (accelerates) at its fastest rate, each step taking 64us. When the acceleration register is loaded with the Slowest setting of 255, the board will change the power output every 16.4ms.

So to calculate the time (in seconds) for the acceleration to complete :

$\text{time} = \text{accel reg value} * 64\text{us} * \text{steps}$ .

For example :

Accel reg	Time/step	Current speed	New speed	Steps	Acceleration time
0	0	0	255	255	0
20	1.28ms	127	255	128	164ms
50	3.2ms	80	0	80	256ms
100	6.4ms	45	7	38	243ms
150	9.6ms	255	5	250	2.4s
200	12.8ms	127	0	127	1.63s

255	16.32ms	65	150	85	1.39s
-----	---------	----	-----	----	-------

### Software Revision number

This register contains the revision number of the software in the modules PIC16F873 controller - currently 7 at the time of writing.

### Using the MD22 with popular controllers

One the easiest ways of connecting the MD22 to a standard controller, such as the BS2 Stamp, is to use RC Servo mode. Select normal (independent) or differential mode on the switches before powering the module. Now you can use the PULSOUT command to simulate the servo pulse and control the motors. The pulse needs to vary between 1mS (full reverse) to 2mS (full forwards) with 1.5mS being the center off position. Unlike servo's, which require the pulse to be repeated every 20mS or so, the MD22 need only be sent a new pulse when you want to change speed. With no pulses being sent it simply continues at the current speed. The timing parameter will vary depending on the controller. Here are some popular examples - all tested by us.

Controller	Pulsout Resolution	Full reverse	Center off	Full Forwards	Command example for Stop
BS2	2uS	500	750	1000	pulsout mot1, 750
BS2e	2uS	500	750	1000	pulsout mot1, 750
BS2sx	0.8uS	1250	1875	2500	pulsout mot1, 1875
BS2p	0.8uS *	1250	1875	2500	pulsout mot1, 1875
Atom	1uS	1000	1500	2000	pulsout mot1, 1500
BX-24	1.085uS	922	1382	1843	call pulseout(mot1, 1382, 1)

\* BS2p resolution is 0.8uS - rather than 0.75uS or 1.18uS as specified in earlier BS2p documentation. Parallax have confirmed this to us.

### Dec 2004 (from software version 3)

Pulse time in RC mode is now verified to assure it is within specified time period of 800us to 2.2ms. Pulses outside of this timing will act to stop the motor. There is also the addition of an extra two RC servo modes which will stop the motors if a valid pulse is not received on channel 1 for a period of 200ms.

### Mar 2006 (from Version 9)

The MD22 now includes two ways to implement the turn, the first uses the forward and backwards channel to control the power with channel two offsetting the power levels to turn. The second method allows channel two to introduce turn without any forward or backwards movement.

The latest (Version 10) md22.hex for the PIC16F873A is [here](#)

MD22 schematics are [md22sch1](#) and [md22sch2](#)

## **C.6 Doppler Velocity Log Specification (DVL)**





# Workhorse Navigator

## DOPPLER VELOCITY LOG (DVL)

### Precision Navigation for the Marine Environment

The Workhorse Navigator is the industry's first choice for precision navigation applications. Teledyne RDI's highly acclaimed Doppler Velocity Log (DVL) provides precise velocity and altitude updates for a wide variety of underwater tasks.

The highly flexible design allows the unit to be used in a standalone configuration or integrated with other navigation systems.

The compact and powerful Workhorse Navigator provides:

- Patented BroadBand processing technology, providing users with both short and long-term high-precision velocity data
- Reliable and accurate high-rate navigation and positioning data
- Proven bottom detection algorithms, and single ping bottom location, for robust and reliable bottom tracking over indeterminate terrain
- Superior low-altitude bottom tracking capability
- Real-time current profiling data

#### Navigator Applications:

- Subsea vehicle and surface vessel navigation
- Hydrographic, geophysical, and oceanographic survey positioning data
- LBL and USBL position aiding
- Spool piece metrology
- Inertial navigation correction and integration
- Cable burial operations
- Deep water positioning
- Station keeping and autopilot control
- Pipeline touchdown monitoring
- Dredge spoils, plume, and sediment tracking

#### Navigator full suite of capabilities:

- Bottom track velocity
- Water track velocity
- Altitude: 4 individual measurements
- Error velocity (data quality indicator)
- Temperature
- Heading/Tilt
- Acoustic echo intensity
- Pressure and depth (optional)
- Current profiling (optional)



**TELEDYNE  
RD INSTRUMENTS**

A Teledyne Technologies Company



# Workhorse Navigator

## DOPPLER VELOCITY LOG (DVL)



### Technical Specifications

Model	WHN 300	WHN 600	WHN 1200
Bottom Velocity			
Single-ping precision			
Std dev at 1m/s <sup>1</sup>	±0.3cm/s	±0.3cm/s	±0.3cm/s
Std dev at 3m/s <sup>1</sup>	±0.6cm/s	±0.5cm/s	±0.4cm/s
Std dev at 5m/s <sup>1</sup>	±0.8cm/s	±0.6cm/s	±0.5cm/s
Long-term accuracy	±0.4%±0.2cm/s	±0.2%±0.1cm/s	±0.2%±0.1cm/s
Minimum altitude <sup>2</sup>	1.0m	0.7m	0.5m
Maximum altitude <sup>2</sup>	200m	90m	30m
Parameters			
Velocity range <sup>3</sup>	±10m/s	±10m/s	±10m/s
Velocity resolution	0.1cm/s	0.1cm/s	0.1cm/s
Ping rate	7Hz max	7Hz max	7Hz max
Water Reference Velocity			
Accuracy	±0.4% ±0.2cm/s	±0.3% ±0.2cm/s	±0.2% ±0.1cm/s
Layer size	selectable	selectable	selectable
Minimum range	1m	0.7m	0.25m
Maximum range	110m	50m	18m
Environmental			
Operating temperature	-5 to 45°C	-5 to 45°C	-5 to 45°C
Storage temperature	-30 to 75°C	-30 to 75°C	-30 to 75°C
Depth rating	3000m or 6000m		
Weight in air:	3000m	15.8kg	12.4kg
	6000m	20.1kg	18.0kg
Weight in water:	3000m	8.8kg	6.1kg
	6000m	13.6kg	12.1kg
Power			
DC input	20–50VDC, external supply (48VDC typical)		
Current	0.4A minimum power supply capability		
Transmit <sup>4</sup>			
Peak power @ 24VDC	66w	21w	8w
Average power (typical)	8w	3w	3w

<sup>1</sup>Standard deviation refers to single-ping horizontal velocity, specified at half the maximum altitude.

<sup>2</sup>@5°C and 35 ppt, 42VDC.

<sup>3</sup>Maximum bottom-tracking range may be reduced due to flow noise at high speed and/or cavitation.

<sup>4</sup>@ 15% duty cycle at peak power (standby 1mW).

### Standard Sensors

**Compass:** ±2° @ 60° dip, 0.5g  
**Tilt:** ±0.5° up to ±15°  
**Temperature:** -5° to 45°C

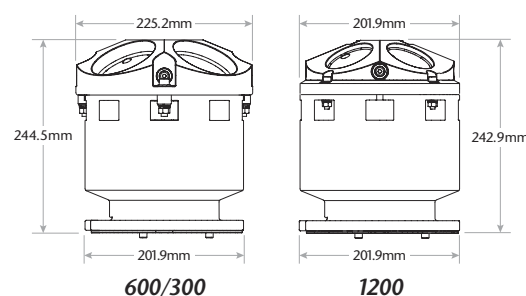
### Hardware

**Configuration:** 4-beam Janus array convex transducer, 30° beam angle  
**Communications:** NMEA0183, ASCII or binary outputs at 1200-115,200 baud user-selectable; serial port is switch-selectable for RS232 or RS422  
**Trigger inputs:** 1) ASCII; 2) RDS3; 3) low latency

### Options

- Current profiling firmware upgrade
- Integrated pressure sensor (±0.25% full scale)
- 25m serial/DC/computer cable
- 5m serial/DC/computer cable
- Internal memory cards (2GB max)
- Enhanced low altitude bottom tracking for model 1200

### Dimensions



**TELEDYNE**  
**RD INSTRUMENTS**

A Teledyne Technologies Company

[www.rdinstruments.com](http://www.rdinstruments.com)  
[www.dvlnav.com](http://www.dvlnav.com)



Free online product training



Free 24/7 emergency support

#### Teledyne RD Instruments

14020 Stowe Drive, Poway, CA 92064 USA

Tel. +1-858-842-2600 • Fax +1-858-842-2822 • E-mail: [rdsales@teledyne.com](mailto:rdsales@teledyne.com)

Les Nertieres 5 Avenue Hector Pintus 06610 La Gaude France

Tel. +33-49-211-0930 • Fax +33-49-211-0931 • E-mail: [rdie@teledyne.com](mailto:rdie@teledyne.com)



Specifications subject to change without notice.  
© 2006 Teledyne RD Instruments, Inc. All rights reserved. Nav-1004, Rev. 08/06







School of Science and Engineering  
Reykjavík University  
Menntavegur 1  
101 Reykjavík, Iceland  
Tel. +354 599 6200  
Fax +354 599 6201  
[www.reykjavikuniversity.is](http://www.reykjavikuniversity.is)  
ISSN 1670-8539