NTNU
Det skapende universitet

Nonlinear Adaptive Motion Control and Model-Error Analysis for Ships

Simulations and MCLab experiments

Elias S. Bjørne

Master i kybernetikk og robotikk
Innlevert: juni 2016
Hovedveileder: Morten Breivik, ITK
Medveileder: Harald Martens, ITK
Mikkel Eske Nørgaard Sørensen, ITK

Norges teknisk-naturvitenskapelige universitet
Institutt for teknisk kybernetikk

I will dedicate this work to two heroes of mine, farfar and abuelo, for teaching me the joy of hard work and challenging problems

Problem Description

The candidate will consider the problem of designing, simulating, experimentally verifying and comparing nonlinear adaptive motion controllers for ships. In addition, techniques from the field of multivariate analysis shall be investigated, to find methods for analysing and predicting the model-error, which represents the dynamics and forces not included in the kinetic model of the ship. The following elements have to be considered:

1. The goal of the master thesis is to develop and experimentally verify a state-of-the-art adaptive control system capable of handling model uncertainties.
    - The model uncertainties to be investigated are the hydrodynamic coefficients especially the ones related to the rotation rate.
    - The considered surface vessel is the model-scale Cybership Enterprise 1.
2. Alternative adaptive control algorithms shall be implemented, numerically simulated and compared in Matlab/Simulink, based on a suitable numerical ship model.
    - In the comparative analysis, performance metrics and simulations will be used.
3. To compare the controllers in a real life scenario, model-scale experiments are to be conducted in the Marine Cybernetics Laboratory.
    - The experiment of the model-scaled ship will be performed for different trajectories in order to gather data for model-error analysis.
4. An analysis of the results will be done with system identification and multivariate analysis to find the structure of the model-error, which will be used to improve the simulation model and control of the ship.

Assignment given: January 11th 2016.
Supervisors: Morten Breivik, ITK, Mikkel Sørensen, ITK and Harald Martens, ITK

Summary

The ocean is an unreliable environment with nonlinearities and unpredictable disturbances, which is why having a controller that can handle the changing dynamics of the ship is essential to achieve precise and predictable tracking. Development of nonlinear adaptive controllers for a model-scaled ship is presented. This includes the design of controllers, simulations and experimental verification. Also the post-experimental analysis is presented, where conventional and new methods are tested for finding improvements for the simulator and the controllers. The experiments did not go as intended in this master thesis, and the main lesson from this report is how weakness in the basic design of the controllers has to be solved, before any fancy algorithm can improve the performance of the controller. It is also presented how analysing the model-error can be a tool for comparing and finding faults in adaptive controllers. And although the experiments did not go as intended, the developed concurrent learning (CL) controllers showed some promising results in the simulations.

In this report, several variants of the nonlinear adaptive controller CL, is developed for the model-scale ship Cybership Enterprise 1 (CSE1). Two storage algorithms are presented, namely the window (WIN) and singular value maximization (SVD). Also two error-signals for the adaptation, $\boldsymbol{z_2}$ and $\boldsymbol{\epsilon}$ are used. The difference in performance of the CL controllers is discussed, based on simulations of the controllers. It is also demonstrated how the controllers' abilities to estimate the model-error $\boldsymbol{\omega}$ affect their performance. The evaluations are done through a comparative analysis in which performance metrics are used. A comparative analysis is also done between CL controllers and the nonlinear controllers adaptive backstepping (ABS) and backstepping (BS).

A control system and a simulator for CSE1 has also been developed. As preparation for the real life basin experiments with CSE1 in the Marine Cybernetics Lab (MC-lab), a comparison of potential estimators are shown. A guidance system was designed, although an error in the late implementation of the system, made it produce a tracking signal $\boldsymbol{\eta_t}$ that was four times as fast as intended, which flawed the experiments. In total 16 tests are performed on CSE1, with 4 controllers, 2 trajectories and 2 sets of speeds. This is presented as a comparative analysis of the controllers, although difficulties to compare the controllers are seen, because of the flawed guidance system and insufficient adaptation possibilities in surge.

In parallel, the field of multivariate analysis (MVA) has been investigated, and MVA methods and techniques are presented, as well as multivariate statistics. The MVA is used in the post-experimental analysis, to clean the data, analyse the model-error, and predictor models of ω are proposed using partial least squares (PLS) regression. The models are compared with the experimental data, the model-errors are also implemented on simulator, and the results are presented.

In post experimental analysis, regular methods are used. An open loop simulation with experimental data is shown. This includes recouping of lost data and analysis of the adaptation problems the CL controllers had during the experiments. The CL controller with

error signal ϵ, an estimate of the deviation in the controller's model-error estimates, is proven capable for system identification (SysID), and a system identification scheme for identifying ω is shown. The effects of inefficient thrusters were added to the hydrodynamics in the uncertainty model proposed for SysID, and the models from SysID were also implemented in the simulator.

Results from this master thesis are:

- The CL controllers are proven to have the best performance against ABS and BS controllers in simulations, and this is supported by examining their superior adaptation ability of the model-error ω
- The CL controllers using the error signal ϵ have the best adaptation in simulations. Although, it should be noted that this is shown only during simulations, and the signal ϵ seems to be very difficult to estimate during a real life scenario.
- The speed of the CSE1 was too high during the experiments, which resulted in the dynamic model no longer being accurate. The result is a substantial model-error ω, especially in surge. This combined with the designed controllers inability to adapt in surge, resulted in bad tracking
- In the experimental results, CL controllers using z_2 showed the best performance, and CL controllers using the SVD storage algorithm were unstable.
- The CL controllers, as implemented in this project, are less robust than the BS and ABS controllers. Weaknesses of the CL controller is identified, such as its sensitiveness of spikes. These problems were addressed, and solutions provided.
- A weakness in the parametrization for all the controllers of the hydrodynamic error is identified. This made the controllers unable to adapt in the surge direction.
- Simulations were done of CSE1 following an elliptic trajectory, with the model-errors ωs implemented in the simulator. When the BS controller was simulated, its performance was equal to the one from the experiment. Problems with stability for the models from PLS regression were also found.
- A scenario is presented where SysID is able to identify thruster failure.
- A new parametrization of the model-error is proposed, and is implemented on the CL controller. It is shown, through simulations with ω implemented, that the modified CL controller is able to estimated and compensate for the disturbances.

Sammendrag

Havet er et upålitelig miljø fult av ulinearitеter og forstyrrelser. Dette øker viktigheten av å ha en regulator som kan håndtere den forandrende dynamikken til skipet for å oppnå førsteklasses styring. Et utviklingsscenario for en ulinær adaptiv regulator for et modellskip er presentert. Dette inkluderer design, simulering og eksperimentell verifisering av regulatoren. En analyse av eksperimentell data er også presentert, hvor både konvensjonelle og nye metoder for å finne forbedringer til simulatoren og regulatoren er brukt. I denne rapporten er flere varianter av den ulinære adaptive regulatoren, concurrent learning (CL) utviklet for modellskipet Cybership Enterprice 1 (CSE1). To lagrings algoritmer er foreslått, vindu (WIN) og singulær verdi maksimering (SVD), og to feilsignaler for adapsjonen, z_2 og ϵ. Forskjellen i CL regulatorenes ytelse er utforsket og evaluert gjennom simuleringer. Det er også demonstrert hvordan regulatorens evne til å tilpasse seg modellfeilen ω påvirket ytelsene deres. En komparativ analyse mellom et utvalg av CL regulatorene , og regulatorene adaptiv backstepping (ABS) og backstepping (BS) er også gjennomført.

Et regulatorsystem, og simulator for CSE1 ble utviklet. Som forberedelse for eksperimentene i bassenget med CSE1 i Marine Cybernetics Labben (MC-lab), ble et sett med observatorer evaluert opp mot hverandre. Et guidance system ble også utviklet, men en sen modifikasjon førte til at den gav ut følgesignaler som var fire ganger raskere en tiltenkt. Totalt ble 16 forsøk gjennomført, der 4 regulatorer ble sammenlignet, følgene etter 2 forskjellige baner med 2 set av hastigheter på hver. Dette ble presentert som en komparativ analyse mellom regulatorene , selv om resultatene gjorde det vanskelig å sammenligne regulatorene .

Imens har fagfeltet multivariat analyse (MVA) også blitt utforsket, og flere metoder og teknikker er presentert i rapporten, samt multivariat statistikk. Disse teknikkene er brukt under analysen av de eksperimentelle dataene, fra å sile data, til og både analysere og lage modeller av ω. Partiell minste kvadrats metode(PLS) regresjon er brukt for å lage modellene, og disse ble senere implementert i simulatoren for verifisering. Under analysen av de eksperimentelle dataene, ble mer konvensjonelle metoder også brukt, som system identifikasjon (SysID). Tapt data ble reprodusert i en åpen sløyfe simulator med de eksperimentelle dataene, samt at åpen sløyfe simuleringene av CL regulatorene gav bedre innsikt i hva som gikk galt under eksperimentene. CL regulatoren som brukte ϵ viste seg også å fungere til SysID, og dette ble også brukt for å lage en model av ω. I tillegg ble en modellering av pådrags feil også inkludert i SysID.

Resultater fra denne masteren er:

- CL regulatorer er vist å ha best ytelse sammenlignet med ABS og BS regulatorer under simulering, og dette er også bekreftet av deres overlegene evne til å tilpasse seg modellfeilen ω

- CL regulatorer som bruker ϵ feilsignalet viste seg å ha best adapsjon under simuleringer, men dette var bare tilfelle under simuleringer. Feilsignalet ϵ viste seg å være utfordrende å estimere under virkelige omstendigheter.
- Hastigheten til CSE1 var for høy under eksperimentene, noe som resulterte i at den dynamiske modellen til CSE1 ikke lenger var korrekt. Dette førte til en betydelig modellfeil ω, spesielt fremover. Dette kombinert med regulatorer som ikke kunne tilpasse seg krefter forfra, gav dårlig ytelse.
- Under experimentene var CL regulatorene som brukte z_2 som hadde best ytelse, og CL regulatorene som brukte SVD lagrings algoritme viste seg å være ustabil under virkelige omstendigheter.
- Under eksperimentene var CL regulatore, sånn som de var implementert i denne masteroppgaven, mindre robuste enn de øvrige BS og ABS regulatore. Svakheter til CL regulatorer ble identifisert, der sensitiviteten til signal pigger var et problem som ble adressert, og en løsning er foreslått.
- Simuleringer med de modellerte ωene ble gjennomført mens CSE1 fulgte en ellipse bane. Simuleringene av BS regulatorer på simulatorene med ω, ble sammenlignet med de eksperimentelle dataene, og likheten var påfallende. Det ble også adressert et stabilitests problem av modellene laget av PLS regresjonene under disse simuleringene.
- Et scenario der SysID ble brukt for å identifisere truster svikt er også presentert
- Etter analysen, ble parametriseringen av modellfeilen endret til å kunne tilpasse seg feil fremover, og denne ble brukt i CL regulatoren. Det ble under simuleringer vist at den modifiserte regulatoren klarte å tilpasse seg modell feilen ω.

Acknowledgements

"The first principle is that you must not fool yourself, and you are the easiest to fool"

— Richard Feynman

This work was performed at the Department of Engineering Cybernetics, Norwegian University of Science and Technology. First of all I want to thank Morten Breivik for excellent supervision and interesting discussions throughout this semester. I have highly benefited from his demands and structure, and critical reading of this report. I also want to thank my co-supervisor Mikkel Eske Nørgaard Sørensen for great support and stressful but fun times in the lab, and for a great eye for details. In addition, I would like to thank Harald Martens for great enthusiasm, exciting discussions and lots of optimism. I also want to thank Emilie Figenschou for giving me great feedback on my writing, and for being a good friend. A last thank you to my room mates Philip Røst Wegner, Jon Harald Bremdal and Markus Molander for providing a home of peace and prosperity.

"If I could explain it to the average person, I wouldn't have been worth the Nobel Prize"

— Richard Feynman

Table of Contents

List of Tables

List of Figures

Tables of Symbols

Table 1: Main states, disturbances and actuation for this master's thesis dynamical model, the "resolution" is used in the nominalization done in Section 3.2 for multivariate analysis, and the "Min" and "Max" was the min and max values of the states observed in the experiments.

No	Sym	Name	Type	Max	Resolution	Min	Comment	Unit
1	$\boldsymbol{\eta}$	Pose	states				mes	
2	x_n	North	state	6.5	10	3.5	mes	$[m]$
3	y_n	East	state	-2.5	10	2.5	mes	$[m]$
4	ψ	Yaw	state	π	10	-π	mes	$[rad]$
5	$\boldsymbol{\nu}$	Velocity	states				est	
6	u	Surge Speed	state	0.3	10	-0.02	est	$[m/s]$
7	v	Sway Speed	state	0.15	10	-0.15	est	$[m/s]$
8	r	Yaw rate	state	0.4	10	-0.4	est	$[rad/s]$
9	$\dot{u}$	Surge acceleration	state	0.1	5	-0.02	est	$[m/s]$
10	$\dot{v}$	Surge acceleration	state	0.02	5	-0.02	est	$[m/s^2]$
11	$\dot{r}$	Yaw acceleration	state	0.1	3	-0.1	est	$[rad/s^2]$
12	$\boldsymbol{\omega}^*$	Disturbance	function/ constant				est	
13	ω_n^*	Earth-fixed Disturbance	function/ constant				est	
14	ω_b^*	Body-fixed Disturbance	function/ constant				est	
15	$\boldsymbol{\tau}$	Thruster Force	actuation				req	
16	τ_u	Surge Force	actuation	1.2	5	-1.2	req	$[N]$
17	τ_v	Sway Force	actuation	3.8	5	-3.8	req	$[N]$
18	τ_r	Yaw Moment	actuation	2	5	-2	req	$[Nm]$

Table 2: Matrices and parameters used for the dynamical model in Chapter 2.

No	Symbol	Name	Type	Model	Precision
19	$\boldsymbol{M}^*$	Inertia Matrix	matrix	eq:(2.4)	
20	$\boldsymbol{M}_{RB}$	Rigid-Body Inertia Matrix	matrix	eq:(2.5)	
21	$\boldsymbol{M}_A$	Added Mass Matrix	matrix	eq:(2.5)	
22	$\boldsymbol{C}^*(\boldsymbol{\nu})$	Coriolis Matrix	matrix function	eq:(2.6)	
23	$\boldsymbol{C}_{RB}(\boldsymbol{\nu})$	Riged-body Coriolis Matrix	matrix function	eq:(2.7)	
24	$\boldsymbol{C}_A(\boldsymbol{\nu})$	Added Coriolis Matrix	matrix function	eq:(2.7)	
25	$\boldsymbol{D}^*(\boldsymbol{\nu})$	Damping Matrix	matrix function	eq:(2.8)	
26	$\boldsymbol{D}_L$	Linear Damping Matrix	matrix	eq:(2.9)	
27	$\boldsymbol{D}_{NL}(\boldsymbol{\nu})$	Nonlinear Damping Matrix	matrix function	eq:(2.9)	
28	m^*	Mass	parameter	14.79	certain
29	I_z^*	Moment of Inertia around z_b	parameter	1.76	uncertain
30	x_g^*	Distance from CG to O_b	parameter	0.375	certain

Table 3: Hydrodynamic coefficients used in in the dynamical model from Chapter 2. CS2:Values are similar to Cybership II's (Skjetne et al., 2004). LS: Valid for low speeds.

N0	Symbol	Name	Type	Model value	Precision	Comment
31	$X^*_{\dot{u}}$	Added mass surge	Hydro-dynamic Coefficient	-2.0	uncertain	CS2
32	$Y^*_{\dot{v}}$	Added mass sway	Hydro-dynamic Coefficient	-10.0	uncertain	CS2
33	$Y^*_{\dot{r}}$	Added mass yaw	Hydro-dynamic Coefficient	0	uncertain	CS2
34	$N^*_{\dot{v}}$	Added mass sway to yaw	Hydro-dynamic Coefficient	0	uncertain	CS2
35	$N^*_{\dot{r}}$	Added mass yaw to sway	Hydro-dynamic Coefficient	-1	uncertain	CS2
36	X^*_u	1st ord. damping surge	Hydro-dynamic Coefficient	-0.6555	certain	LS
37	$X^*_{\|u\|u}$	2nd ord. damping surge	Hydro-dynamic Coefficient	0.3545	certain	LS
38	X^*_{uuu}	3rd ord. damping surge	Hydro-dynamic Coefficient	-3.787	certain	LS
39	X^*_{vv}	2nd ord. damping sway to surge	Hydro-dynamic Coefficient	-2.443	certain	LS
40	Y^*_v	1st ord. damping sway	Hydro-dynamic Coefficient	-1.33	certain	LS
41	$Y^*_{\|v\|v}$	2nd ord. damping sway	Hydro-dynamic Coefficient	-2.776	certain	LS
41	Y^*_{vvv}	2nd ord. damping sway	Hydro-dynamic Coefficient	-64.91	certain	LS
42	N^*_v	1st. ord. damping sway to yaw	Hydro-dynamic Coefficient	0	certain	LS
42	$N^*_{\|v\|v}$	2nd ord damping sway to yaw	Hydro-dynamic Coefficient	-0.2088	certain	LS

Table 4: Hydrodynamic coefficients used in in the dynamical model, these were tried to be estimated, the parameters commented with CS2 are similar to the values used and found for Cybership II (Skjetne et al., 2004).

No	Symbol	Name	Type	measures/ Model value	Precision	Comment
43	Y_r^*	1st. ord. damping yaw to sway	Hydro-dynamic Coefficient	-7.250	uncertain	CS2
44	$Y_{\|r\|r}^*$	2nd. ord. damping yaw to sway	Hydro-dynamic Coefficient	-3.450	uncertain	CS2
45	N_r^*	1st. ord. damping yaw	Hydro-dynamic Coefficient	-1.900	uncertain	CS2
46	$N_{\|r\|r}^*$	2nd. ord. damping yaw	Hydro-dynamic Coefficient	-0.750	uncertain	CS2
47	$Y_{\|v\|r}^*$	Damping sway coupled	Hydro-dynamic Coefficient	-0.845	uncertain	CS2
48	$Y_{\|r\|v}^*$	Damping sway coupled	Hydro-dynamic Coefficient	-0.805	uncertain	CS2
49	$N_{\|v\|r}^*$	Damping yaw coupled	Hydro-dynamic Coefficient	0.080	uncertain	CS2
50	$N_{\|r\|v}^*$	Damping yaw coupled	Hydro-dynamic Coefficient	0.130	uncertain	CS2

Table 5: Other possible states, internal signals, metrics and tuning parameters.

No	Symbol	Name	Type
51	z	Heave Position	state
52	θ	Pitch Angle	state
53	φ	Roll Angle	state
54	w	Heave Speed	state
55	q	Pitch Rate	state
56	p	Roll Rate	state
57	$\dot{w}$	Heave Acceleration	state
58	$\dot{q}$	Pitch Acceleration	state
59	$\dot{p}$	Roll Acceleration	state
60	e_{pos}	Position Error	measurement
61	e	Cross-Track Error	measurement
62	e^2	Squared Cross-Track Error	measurement
63	P	Power	measurement
64	$I\dot{\tau}$	Change of Force	measurement
65	ISE	Integrated Square Error	Norm
66	W	Work	Norm
67	I $\dot{\tau}$	Integrated change of Tau	Norm
68	$\boldsymbol{z_1}$	Pose Error	control signal
69	$\boldsymbol{z_2}$	Velocity Error	control signal
70	$\boldsymbol{\epsilon}$	Estimation Error	estimation signal
71	$\boldsymbol{K}_p$	Tuning Matrix Proportional	Tuning parameter
72	$\boldsymbol{K}_d$	Tuning Matrix Damping	Tuning parameter
73	$\boldsymbol{\Gamma}$	Estimation Gains	Tuning parameter
74	h	Time-step	parameter

List of Abbreviations

MVA	=	Multivariate Analysis
ISE	=	Integral of the Square Error
IAEW	=	Integral of the Absolute Error and Work
IAEWWT	=	Integral of the Absolute Error with Work and Wear and Tear
PCA	=	Principal Component Analysis
PLSR	=	Partial Least Squares Regression
VSP	=	Voith Schneider Propeller
BT	=	Bow Thruster
HIL	=	Hardware In the Loop
MC-lab	=	Marine Cybernetics Laboratory
CG	=	Center of Gravity
CM	=	Center of Moment
SVD	=	Singular Value Decomposition
CLF	=	Control Lyapunov Function
SVM	=	Singular Value Maximation
PE	=	Persistency of Excitation
ep	=	epsilon
cRIO	=	CompactRIO
NaN	=	Not a Number
W	=	Work
RBS	=	Regular Backstepping
BS	=	Backstepping
ABS	=	Adaptive Backstepping
CL	=	Concurrent Learning
CLWIN	=	Concurrent Learning with Window storage algorithm
CLSVD	=	Concurrent Learning with SVD/SVM storage algorithm
CLWZ	=	CLWIN with $\boldsymbol{z_2}$ as instantaneous adaptation error
CLWE	=	CLWIN with ϵ as instantaneous adaptation error
SysID	=	System Identification
PWM	=	Pulse Width Modulation
DOF	=	Degree-Of-Freedom
PlayStation 3	=	Ps3

Chapter 1

Introduction

1.1 Motivation for Adaptive Motion Control for Ships

Development of model-based controllers for ships has been a topic since the 20th century. As ships increase their automation, the need for more robust and flexible controllers has increased. When developing a control system for marine vessels, it has to be taken into account that the ocean is a highly nonlinear and unreliable environment. Being able to develop a robust controller that can handle changing dynamics is therefore essential for success in this field. Robust in handling disturbances and uncertainties, but also robust against oscillatory behaviour and measurement difficulties. A considerable challenge with surface vessels in general, is the uncertain nonlinear hydrodynamics and external disturbances that occur. The hydrodynamic forces are often modelled with hydrodynamical coefficients. Some can be found for certain scenarios as seen in (Skjetne et al., 2004), these include hydrodynamic coefficients for linear motions in surge and sway which can be found through towing experiments. Others can be harder to identify, like the hydrodynamic coefficients related to rotational motion, as can be seen in (Yoon and Rhee, 2003). In addition, not being able to measure all necessary states leads to the need of estimation. This can again introduce problems to the robustness of the controller, such as estimation spikes and biases. External disturbances, such as waves, wind and currents are also difficult if not impossible to measure. This rises the importance of having an adaptive and robust control that can deal with these uncertainties and disturbances in an efficient manner.

1.2 Adaptive Controllers

Robust and adaptive motion controllers have also been applied to other manoeuvring problems, such as aviation (Lavretsky and Wise, 2012) and quadcopter manoeuvring, and different nonlinear adaptive controllers are continuously being developed and improved. Several have been tried with surface vessels, with adaptive backstepping (ABS) as one of

them. The backstepping (BS) controller has been a popular controller design procedure for surface vessels, as its recursive method fits the structure of the vessel model. Two versions of adaptive backstepping (ABS) controllers are presented in (Skjetne et al., 2004) and (Breivik and Fossen, 2004). However these adaptive schemes requires restrictive persistence of excitation to adapt to the right parameters, which is often difficult to guarantee in a control scenario. A new adaptation scheme is presented in (Chowdhary and Johnson, 2010), concurrent learning (CL), which instead of relying on instantaneous measurements for adaptation, uses stored data and instantaneous data concurrently for adaptation. Further investigation on the adaptation and its convergence is discussed in (Chowdhary and Johnson, 2011a), and a demonstration of the CL adaptive control on a 3 degree-of-freedom helicopter model in (Chowdhary et al., 2012) and a rotor crafted unmanned aerial vehicle in Chowdhary and Johnson (2011b). In addition, the CL has been applied in a approximate optimal regulation in (Kamalapurkar et al., 2013). The results show that the CL has a potential for improved adaptation, although it is early in its development so its full potential is far from reached.

1.3 Main Contributions

- Several variants of the nonlinear adaptive controller concurrent learning (CL) was developed for Cybership Enterprise 1 (CSE1)
  - The difference in convergence and adaptation was discussed and investigated, and weaknesses and strength of different implementations of CL were identified
  - A comparative analysis was made, with performance metrics, of the CL controllers, was done with simulation. In addition a comparative analysis was made with selected CL controllers, against the ABS and BS controllers.
  - An demonstration of the correlation between the controllers ability to estimate the model-error of the ship and the controllers performance was made.
  - A CL implementation for CSE1 was done, although the results of the controller was not satisfactory.
- A full control system and a simulator was developed for CSE1.
  - This included developing the guidance and estimator for the ship.
  - The simulator was made in Simulink, to minimize the effort of transferring the controller from simulator to CSE1.
  - A Hardware-in-the-Loop (HIL) test on a CompactRIO was also preformed to evaluate the CL controllers on a real-time control system.
- A real life test was performed with CSE1 on the Marine Cybernetics Laboratory (MC-lab)
  - A comparative analysis of potential estimators was made. The derivative filter chosen in the end, was modified to handle signal freeze and wrap around error.

  - A tuning of the center of mass for the thruster allocation was preformed
  - 16 tests were in total preformed on CSE1, with 4 controllers, 2 trajectories and 2 set of speeds. This resulted in a comparative analysis of the CL, ABS and BS controllers working on a real-life system.

- An investigation of the field of multivariate analysis (MVA), and how it could contribute to the control field was made.
  - Several methods for making predictive models with MVA was investigated
  - A new way of organizing data for a partial least squares (PLS) regression is presented, continuous nominalisation.
  - A procedure for using principal component analysis PCA together with multivariate statistics to detect outliers and clean data was done
  - A procedure for analysing and predict model-errors for ships using PLS regression was demonstrated

- A post experimental analysis was done, were the model-error of the ship was identified, and modelling of this model-error was proposed
  - The modelling of the model error was done using System Identification (SysID) and MVA
  - Recouping of data and validation of the estimators was done with an open loop simulation with the experimental data.
  - Identification of weaknesses of the CL adaptation was identified, and an improvement was proposed.
  - The open loop simulator, with an improved CL adaptation proved to be working for SysID
  - A method for identifying thruster failure using SysID was also presented
  - Validation of the model-error model was done by implementing it on the simulator, and compare it with experimental data

- With improved knowledge of the model error, a modification of the CL controller was made, and it was tested on the simulator, with the newly found model-error.The improved CL-controller managed to handle the model error and satisfactory tracking was preformed.

1.4 Overview

This master thesis focuses on designing, simulating and experimental verifying the CL controller on CSE1, and through experiments investigate methods for identifying and create a model that predicts the model-error.

In Chapter 2, the dynamic model for CSE1 is presented, as well as a discussion on model-error. In Chapter 3, the theory behind MVA is presented. In Chapter 4, the motion

control system design is shown, including the controllers tested and estimators used in the experiments, and performance metrics for comparing controllers. In Chapter 5 the simulations with the controllers presented 4 on the dynamic system from Chapter 2 are shown, in addition to a discussion. In Chapter 6, the experimental platform is laid out, and the experimental results are shown, as well as a discussion. Chapter 7, contains a post experimental analysis, in addition to a proposed expansion of the CSE1 model together with a proposed controller improvement. Finally, in Chapter 8, the master thesis is concluded, and future work is proposed.

Chapter 2

Vessel Model

2.1 Vessel Dynamics

As stated in Chapter 1, the numerical vessel model for this project is based on the Cybership Enterprice 1 (CSE1) model ship, where the nonlinear model from (Fossen, 2011) has been used:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\nu} \tag{2.1}$$

$$\boldsymbol{M}^*\dot{\boldsymbol{\nu}} + \boldsymbol{C}^*(\boldsymbol{\nu}) + \boldsymbol{D}^*(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}^* + \boldsymbol{\omega}^*(t) \tag{2.2}$$

where $\boldsymbol{\eta} = [x_n, y_n, \psi]^\top \in \mathbb{R}^2 \times \mathbb{S}$ is the three-dimensional vector describing the vessel's position and heading, see Figure 2.1 . The x_n and y_n represents the position of the vessel in North and East in the local earth-fixed frame, and $\psi \in \mathbb{S} = [-\pi, \pi]$ represents the angle of the vessel's yaw in radians. The velocity vector $\boldsymbol{\nu} = [u, v, r]^\top \in \mathbb{R}^3$ is the body fixed velocities. u and v are velocities in surge and sway, and r is the vessel's yaw rate. The $\boldsymbol{\omega}^*(t)$ is the model-error and external disturbances.

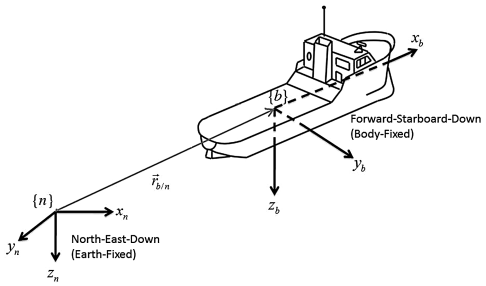
$$\boldsymbol{\omega}^*(t) = \boldsymbol{R}^\top(\psi)\boldsymbol{\omega}_n^*(t) + \boldsymbol{\omega}_b^*(t), \tag{2.3}$$

where $\boldsymbol{\omega}_n^*(t)$ is the earth fixed disturbance, and $\boldsymbol{\omega}_b^*(t)$ is the body-fixed model-error or disturbance. $\boldsymbol{M}^*, \boldsymbol{C}^*(\boldsymbol{\nu}), \boldsymbol{D}^*(\boldsymbol{\nu})$ and $\boldsymbol{R}(\psi)$ are respectively the inertia, Coriolis, damping and rotation matrix, where $\boldsymbol{M}^* = \boldsymbol{M}^{*\top}$, $\boldsymbol{C}^*(\boldsymbol{\nu}) = -\boldsymbol{C}^{*\top}(\boldsymbol{\nu})$, $\boldsymbol{D}^*(\boldsymbol{\nu}) > 0$ and $\boldsymbol{R}(\psi) \in SO(3)$. These are derived from the model ship described in (Skjetne et al., 2004),

$$\boldsymbol{M}^* \triangleq \boldsymbol{M}_{RB} + \boldsymbol{M}_A \tag{2.4}$$

$$\boldsymbol{M}_{RB} = \begin{bmatrix} m^* & 0 & 0 \\ 0 & m^* & m^* x_g^* \\ 0 & m^* x_g^* & I_z^* \end{bmatrix}, \boldsymbol{M}_A = \begin{bmatrix} -X_{\dot{u}}^* & 0 & 0 \\ 0 & -Y_{\dot{v}}^* & -Y_{\dot{r}}^* \\ 0 & -N_{\dot{v}}^* & -N_{\dot{r}}^* \end{bmatrix} \tag{2.5}$$

Figure 2.1 Local earth-fixed and body coordinates for the ship from (Fossen, 2011)



Here, m^* is the mass of the ship, ${I_z}^*$ is the inertia of the ship around its body-z axis z_b, see Figure 2.1 , and x_g^* is the distance from center of gravity (CG) to the body coordinate's origin O_b. $X_{\dot{u}}^*, Y_{\dot{r}}^*, Y_{\dot{r}}^*, N_{\dot{v}}^*, N_{\dot{r}}^*$ are hydrodynamic coefficients. The Coriolis matrix is

$$\boldsymbol{C}^*(\boldsymbol{\nu}) \triangleq \boldsymbol{C}_{RB}(\boldsymbol{\nu}) + \boldsymbol{C}_A(\boldsymbol{\nu}) \tag{2.6}$$

$$\boldsymbol{C}^*(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & -m^*(x_g^* + v) \\ 0 & 0 & m^* u \\ m^*(x_g^* r + v) & -m^* u & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & c_{13}^*(\boldsymbol{\nu}) \\ 0 & 0 & c_{23}^*(\boldsymbol{\nu}) \\ -c_{13}^*(\boldsymbol{\nu}) & -c_{23}^*(\boldsymbol{\nu}) & 0 \end{bmatrix} \tag{2.7}$$

Here, $c_{13}^*(\boldsymbol{\nu})$ and $c_{23}^*(\boldsymbol{\nu})$ are functions built up from other hydrodynamic coefficients. $c_{13}^*(\boldsymbol{\nu}) = Y_{\dot{v}}^* v + r$ and $c_{23}^*(\boldsymbol{\nu}) = -X_{\dot{u}}^* u$. The damping matrix is

$$\boldsymbol{D}^*(\boldsymbol{\nu}) \triangleq \boldsymbol{D_L} + \boldsymbol{D_{NL}}(\boldsymbol{\nu}) \tag{2.8}$$

$$\boldsymbol{D}^*(\boldsymbol{\nu}) = \begin{bmatrix} -X_u^* & 0 & 0 \\ 0 & -Y_v^* & -Y_r^* \\ 0 & -N_v^* & -N_r^* \end{bmatrix} + \begin{bmatrix} -d_{11}^*(\boldsymbol{\nu}) & 0 & 0 \\ 0 & -d_{22}^*(\boldsymbol{\nu}) & -d_{23}^*(\boldsymbol{\nu}) \\ 0 & -d_{32}^*(\boldsymbol{\nu}) & -d_{33}^*(\boldsymbol{\nu}) \end{bmatrix} \tag{2.9}$$

where d_{ii}^* is also a function of $\boldsymbol{\nu}$ and hydrodynamic coefficients: $d_{11}^*(\boldsymbol{\nu}) = X_{|u|u}^*|u| + X_{uuu}^* u^2$, $d_{22}^*(\boldsymbol{\nu}) = Y_{|v|v}^*|v| + Y_{|r|v|}^*|r|$, $d_{23}^*(\boldsymbol{\nu}) = Y_{|v|r}^*|v| + Y_{|r|r}^*|r|$, $d_{32}^*(\boldsymbol{\nu}) = N_{|v|v}^*|v| + N_{|r|r}^*|r|$. The hydrodynamic coefficients have been calculated for CSE1, and are listed in Table 2.1

Table 2.1: CSE1 hydrodynamic coefficients from (Sandved, 2015). The hydrodynamic coefficients that are related to the ships rotation and added mass has been taken from (Skjetne et al., 2004) and are marked with a $^{\#}$

Parameters for CSE1									
m^*	14.79	$X^*_{\dot{u}}$	-2$^{\#}$	X^*_u	-0.6555	Y^*_{vvv}	-64.91	$N^*_{\|r\|r}$	-0.750$^{\#}$
I^*_z	1.760$^{\#}$	$Y^*_{\dot{v}}$	-10$^{\#}$	$X^*_{\|u\|u}$	0.3545	N^*_v	0	$Y^*_{\|v\|r}$	-0.845$^{\#}$
x^*_g	0.375	$Y^*_{\dot{r}}$	0$^{\#}$	X^*_{uuu}	-3.787	$N^*_{\|v\|v}$	-0.2088	$Y^*_{\|r\|v}$	-0.805$^{\#}$
		$N^*_{\dot{v}}$	0$^{\#}$	$X^*_{\|v\|v}$	-2.443	Y^*_r	-7.250$^{\#}$	$N^*_{\|v\|r}$	0.080$^{\#}$
		$N^*_{\dot{v}}$	-1$^{\#}$	Y^*_v	-1.33	$Y^*_{\|r\|r}$	-3.450$^{\#}$	$N^*_{\|r\|v}$	0.130$^{\#}$
				$Y^*_{\|v\|v}$	-2.776	N^*_r	-1.900$^{\#}$		

2.1.1 Thruster Model and Allocation

The thrusters are positioned as in Figure 2.2, where two Voith Schneider propellers (VSP) are at the back, and can give forces in all directions. Hence they work like two fast responding azimuth thruster. The bow thruster is placed at the front, and ensures that the ship is fully actuated, at least for slow motions. The mapping from control signal to force from the thrusters is shown in (Sørensen, 2013), and are the following

$$\boldsymbol{f} = \boldsymbol{K}\boldsymbol{u} \tag{2.10}$$

where $\boldsymbol{f}$ is the vector with the force from every thruster, $\boldsymbol{K}$ is a diagonal matrix that maps the control signal to actual forces, and $\boldsymbol{u}$ are the control signals proportional to $\boldsymbol{f}$. For propeller the thrust is

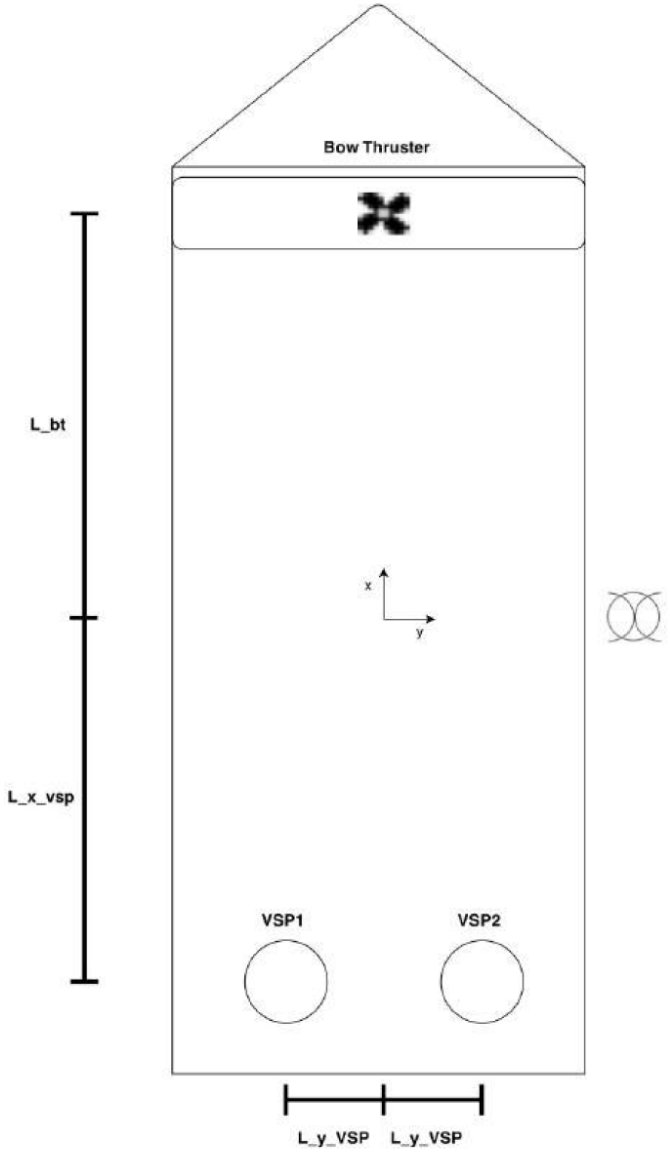
$$f_i = \text{sign}(n_i) K_{T0i} \rho D_i^4 n_i^2 \tag{2.11}$$

The f_i is the force magnitude from the given propeller, K_{T0i} is the thrust coefficient and is found by open water tests. The D_i is the diameter of the propeller, and n_i is its rotation speed. For the VSP, the mapping is a look-up table with pre-calculated values under open-water conditions (Koschorrek et al., 2015)

$$\begin{bmatrix} f_{ix} \\ f_{iy} \\ P_{iD} \end{bmatrix} = VSP(\boldsymbol{p}, n, \boldsymbol{v^b}) \tag{2.12}$$

were f_{ix} and f_{iy} are the forces in x- and y-direction at the propeller location, and P_{iD} is the power delivered power. As seen in Figure 2.3, the $\boldsymbol{p}$ is the pitch vector deciding the angle of the force, n is the revolution rate and $\boldsymbol{v^b}$ is the inflow vector. These are presented in body-fixed coordinates. Given a desired force, it is then the thruster controller's task to find the right power for the different thrusters. To see how the forces from the thrusters affect the ship, the thruster configuration is used (Sørensen, 2013), and can be seen in Figure 2.2. This leads to the mapping

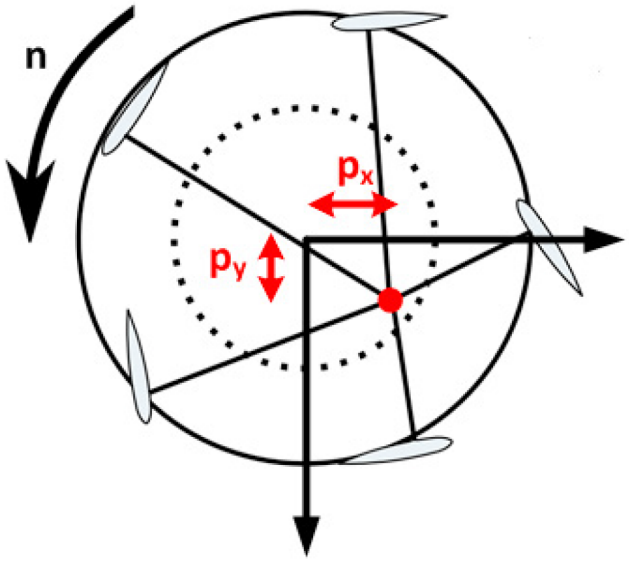
Figure 2.2 The figure shows the locations of the thrusters on CSE1, from (Sandved, 2015)



$$\boldsymbol{\tau} = \boldsymbol{T}(\boldsymbol{\alpha})\boldsymbol{f} = \boldsymbol{T}(\boldsymbol{\alpha})\boldsymbol{K}\boldsymbol{u} \tag{2.13}$$

where the $\boldsymbol{\alpha}$ vector is the angle of attack of the different thrusters. The thruster allocation of CSE1 is then just an inverse map of (2.13) were requested forces are received from the

Figure 2.3 The VSP seen from underneath, with the pitch definition, from (Koschorrek et al., 2015)



controller, and the control signals are sent out to the thruster controllers on the ship, as can be seen in Figure 2.4. The control signals from CSE1 thruster allocation is

$$\boldsymbol{u} = \begin{bmatrix} u_{VSPx} \\ u_{VSPy} \\ u_{BT} \end{bmatrix} \tag{2.14}$$

which are the control signal for the VSP force in the x and y body-fixed direction, and the bow thruster.

The thrust mapping in the CSE1 is

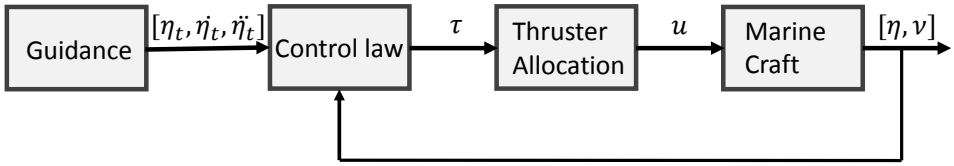
$$\boldsymbol{K} = \text{diag}(F_{VSP_{max}},\ F_{VSP_{max}},\ l_{BT}) \tag{2.15}$$

where $F_{VSP_{max}} = 1.165$ and $l_{BT} = 2.629$. The configuration mapping is

$$\boldsymbol{T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & l_{VSP} & l_{BT} \end{bmatrix} \tag{2.16}$$

where $l_{VSP} = -0.4575$ and $l_{BT} = 0.3875$.

Figure 2.4 The block diagram of the marine craft control system



Point of attack: The thruster allocation above assumes that the CG or center of mass (CM) of the ship is known, but this is not necessarily the case. And the result can be a thruster allocation that induces unwanted rotation on the ship. If a thruster allocation with flexible point of attack is used, the correct CM can be found through testing, as explained in Section 6.3.4 and the point of attack can be set so that no unwanted rotation is induced. The transformation of a force on a rigid body from a point to another can be done through the transformation matrix

$$\boldsymbol{\tau} = \boldsymbol{H}(\boldsymbol{p_o})\boldsymbol{\tau_o} = \begin{bmatrix} \boldsymbol{I}_{2\times 2} & \boldsymbol{0}_{2\times 1} \\ \boldsymbol{S}_{1\times 2}(\boldsymbol{p_o}) & 1 \end{bmatrix} \boldsymbol{\tau_o} \tag{2.17}$$

where $\boldsymbol{p_o} = [p_{ox},\ p_{oy}]$ is the distance between the point of attack of $\boldsymbol{\tau}$ and $\boldsymbol{\tau_o}$, and

$$\boldsymbol{S}_{1\times 2}(\boldsymbol{p_o}) = \begin{bmatrix} -p_{oy} & p_{ox} \end{bmatrix} \tag{2.18}$$

This relationship applies for all rigid bodies when the frames of $\boldsymbol{\tau}$ and $\boldsymbol{\tau_o}$ are not rotated against each other. In that case the relationship between wanted force, and thruster output is

$$\boldsymbol{\tau} = \boldsymbol{H}(\boldsymbol{p_o})\boldsymbol{T}(\boldsymbol{\alpha})\boldsymbol{K}\boldsymbol{u} \tag{2.19}$$

and the thruster allocation is therefore the inverse map of this.

2.2 Model-Error Parametrization

The nonlinear model (2.1) and (2.2) is a simplification of the real world. In addition, the hydrodynamic coefficients can be uncertain or unknown, and other parameters can be incorrect or unreliable. A robust controller must be able to handle all these uncertainties. The controllers will therefore be designed with an uncertainty and disturbance model in mind. Multivariate analysis methods will also be tried out to see if the disturbances are possible to predict and hence compensate for.

Model-Error When describing the dynamics of a real life system, the plant model is utilized. In this project the plant model will be (2.1) and (2.2). The control model is often a simplification of the plant model and is implemented when designing the controller. When designing a controller, the control model that is applied can be mistaken, or have wrong values. This is modelled as the difference between the control model and plant model, and will be referred to as the model-error. This will be given the symbol $\boldsymbol{\omega}^*$, and its estimate will be $\boldsymbol{\omega}$. The controllers estimate of this model-error will be denoted $\hat{\boldsymbol{\omega}}$.

2.2.1 Scaled Error

One possible simplification of model-error is described in (Sørensen and Breivik, 2015) and can be

$$\boldsymbol{M}^* = \delta_M \boldsymbol{M} \tag{2.20}$$
$$\boldsymbol{C}^*(\boldsymbol{\nu}) = \delta_M \boldsymbol{C}(\boldsymbol{\nu}) \tag{2.21}$$
$$\boldsymbol{D}^*(\boldsymbol{\nu}) = \delta_D \boldsymbol{D}(\boldsymbol{\nu}) \tag{2.22}$$
$$\boldsymbol{\tau}^* = \delta_\tau \boldsymbol{\tau} \tag{2.23}$$

In this case, the matrices with asterisk e.g. $\boldsymbol{M}^*$, are the matrices from the plant model, while the matrices without asterisk are matrices for the control model. The δ_i is then a scalar relationship between the different matrices in the plant and control model. This is probably too simplified, but gives robust and good adaptive schemes, and limits the number of variables that have to be estimated.

2.2.2 Error with Unknown Parameter Values

Another assumption is that all the parameters in Table 2.1 are unreliable, except the inertia matrix, which means that there are 19 unknown parameters that need to be estimated.

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} = \boldsymbol{\tau} + \boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* + \boldsymbol{R}^\top(\psi)\boldsymbol{\omega_n} \tag{2.24}$$

where all the dynamics from $\boldsymbol{D}^*(\boldsymbol{\nu})\boldsymbol{\nu}$ and $\boldsymbol{C}^*(\boldsymbol{\nu})\boldsymbol{\nu}$ are left in the $\boldsymbol{\Phi}(\boldsymbol{\nu})$. The unknown parameters are ordered in the vector $\boldsymbol{\varphi}^*$

$$\begin{aligned}\boldsymbol{\varphi}^* = [&c_1, c_2, c_3, X_u^*, Y_v^*, Y_r^*, N_v^*, N_r^*, X_{|u|u}^*, X_{uuu}^*, X_{|v|v}^*, Y_{|v|v}^* \cdots \\ &..., Y_{|r|v}^*, Y_{|v|r}^*, Y_{|r|r}^*, Y_{vvv}^*, N_{|v|v}^*, N_{|r|v}^*, N_{|r|v}^*, N_{|r|r}^*]\end{aligned} \tag{2.25}$$

where $c_1 = -m^* x_g^* + \frac{1}{2}(N_{\dot{v}}^* + Y_{\dot{r}}^*)$, $c_2 = -m + Y_{\dot{v}} v$ and $c_3 = m - X_{\dot{u}}$. The regressor matrix will be

$$
\begin{aligned}
&\boldsymbol{\Phi}(\boldsymbol{\nu}) = \left[\boldsymbol{\Phi}_C(\boldsymbol{\nu}) | \Phi_{DL}(\boldsymbol{\nu}) | \Phi_{NL}(\boldsymbol{\nu})\right] \\
&\boldsymbol{\Phi}_C(\boldsymbol{\nu}) = \begin{vmatrix} r^2 & vr & 0 \\ 0 & 0 & ur \\ -ru & -vu & -uv \end{vmatrix}, \qquad \boldsymbol{\Phi}_{DL}(\boldsymbol{\nu}) = \begin{vmatrix} -u & 0 & 0 & 0 & 0 \\ 0 & -v & -r & 0 & 0 \\ 0 & 0 & 0 & -v & -r \end{vmatrix} \\
&\boldsymbol{\Phi}_{NL}(\boldsymbol{\nu}) = \\
&\begin{vmatrix} -|u|u & -u^3 & -|v|v & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -|v|v & -|r|v & -|v|r & -|r|r & v^3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -|v|v & -|r|v & -|v|r & -|r|r \end{vmatrix}
\end{aligned}
\tag{2.26}
$$

2.2.3 Error with Known and Unknown Parameter Values

?? Usually some of the parameters are known, estimated or found through experiments. While others are found by a best guess, certainly different parameters will have different reliability in a model. When investigating how the model of CSE1 was found this seemed to be the case. The parameters on the CSE1 hydrodynamic model correlated with the ships rotation, were taken from the Cybership II model in (Skjetne et al., 2004), as a best guess. Which leaves these parameters unreliable, and the dynamic model can be represented as following

$$
\boldsymbol{M}\dot{\boldsymbol{\nu}} = \boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* + \boldsymbol{R}^\top(\psi)\boldsymbol{\omega_n} \tag{2.27}
$$

where $\boldsymbol{g}(\boldsymbol{\nu}) + \boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* = \boldsymbol{D}^*(\boldsymbol{\nu})\boldsymbol{\nu}$[1]. So $\boldsymbol{g}(\boldsymbol{\nu})$ is the part of the nonlinear damping known to the controller, and $\boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^*$ needs to be estimated. This would correspond to a model-error with disturbance

$$
\boldsymbol{\omega}^* = -\boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* + \boldsymbol{R}^\top(\psi)\boldsymbol{\omega_n} \tag{2.28}
$$

The unknown hydrodynamic coefficients are the ones related to the yaw rate of the ship. The other coefficients $X_u^*, Y_v^*, N_v^*, X_{|u|u}^*, X_{uuu}^*, X_{|v|v}^*, Y_{|v|v}^*, Y_{vvv}^*, N_{|v|v}^*$ were found through towing experiments, described in (Skåtun, 2011). The known part of the hydrodynamics will then be

$$
\boldsymbol{g}(\boldsymbol{\nu}) = \begin{bmatrix} -X_u^* u - X_{|u|u}^* |u|u - X_{uuu}^* u^3 - X_{|v|v}^* |v|v \\ -Y_v^* v - Y_{|v|v}^* |v|v - Y_{vvv}^* v^3 \\ -N_v^* v - N_{|v|v}^* |v|v \end{bmatrix} \tag{2.29}
$$

The true unknown coefficients are

[1] It should be noted that the $sign$ of $\Phi(\nu)$ and $g(\nu)$ is changed with regards to the model described in (Skjetne et al., 2004)

$$\boldsymbol{\varphi}^* = [Y^*_{|r|v},\ Y^*_r,\ Y^*_{|v|r},\ Y^*_{|r|r},\ N^*_{|r|v},\ N^*_r,\ N^*_{|v|r},\ N^*_{|r|r}] \tag{2.30}$$

and the controller will be estimating the coefficients in the vector

$$\hat{\boldsymbol{\varphi}} = [Y_{|r|v},\ Y_r,\ Y_{|v|r},\ Y_{|r|r},\ N_{|r|v},\ N_r,\ N_{|v|r},\ N_{|r|r}] \tag{2.31}$$

Then

$$\boldsymbol{\Phi}(\nu) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -|r|v & -r & -|v|r & -|r|r & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -|r|v & -r & -|v|r & -|r|r \end{bmatrix} \tag{2.32}$$

2.2.4 Disturbance

Another name for the model-error is a disturbance. Although disturbance often account for external effects, such as unknown forces, not accounted for in the model. These can be external forces as winds, currents, waves and operation related disturbances as in anchor handling and towing. The disturbing effects can also come from eg. unwanted roll or unknown hydrodynamics. The disturbance $\boldsymbol{\omega}^*(t)$ can be parametrised as (2.3). The earth-fixed disturbance can be split into two uncertainties like in (Fossen and Strand, 1999)

$$\boldsymbol{\omega^*_n(t)} = \boldsymbol{b^*} + \boldsymbol{\xi}^*(t) \tag{2.33}$$

where $\boldsymbol{b^*}$ accounts for a constant, or slowly-varying force like current and wind, and $\boldsymbol{\xi(t)}$ represents the time-varying disturbances such as waves.

In this project, we did not account for wave motions, but the controllers were designed to adapt to more slowly-varying and constant earth-fixed disturbances. Although this was not applied during experiments. It was kept out of simulations as well. The uncertainty model (2.27) was used in the first simulations , and it will be the model used when the controllers are designed in Chapter 4. Since there were no earth fixed disturbances, only body fixed, the model-error $\boldsymbol{\omega^*}$ and its estimates are

$$\boldsymbol{\omega}^* = -\boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* + \boldsymbol{\omega_b}(\boldsymbol{\nu}, \boldsymbol{\tau}) \tag{2.34}$$

which gives the controllers estimate of the model-error

$$\hat{\boldsymbol{\omega}} = -\boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi} \tag{2.35}$$

The rest of (2.27), will be the off-line estimate of the model error

$$\boldsymbol{\omega} = \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\tau} \tag{2.36}$$

The controllers adaptation error-signal $\boldsymbol{\epsilon}$ will then be

$$\boldsymbol{\epsilon} = \boldsymbol{\omega} - \hat{\boldsymbol{\omega}} \tag{2.37}$$

and will be used in the adaptation of the controllers designed in Chapter 4.

2.2.5 Thruster allocation Error

As explained in Section 2.1.1 the thruster allocation is based on look-up tables and thruster models. This implies that the force requested from the controller is not identical to the force produced by the thruster.

$$\boldsymbol{\tau}_{req} = \boldsymbol{\tau_{ac}} + \boldsymbol{\tau}_{\omega} \tag{2.38}$$

there $\boldsymbol{\tau}$ is the force requested by the controller, $\boldsymbol{\tau_{th}}$ is the thrust produced by the thrusters and $\boldsymbol{\tau}_{\omega}$ is the thrust error. The problem is that feed forwarding the thrust error, could again propagate new errors, so if $\boldsymbol{\tau}_{\omega}$ is identified, a new thruster map should be investigated. In Chapter 7, a linear thruster error is proposed and a system identification is tried out to identify this error.

$$\boldsymbol{\tau}_{req} = \boldsymbol{\tau}_{act} + \boldsymbol{\Delta}_{\tau}\boldsymbol{\tau}_{req} \tag{2.39}$$

where $\boldsymbol{\Delta}_{\tau} = \text{diag}([\delta\tau_u,\ \delta\tau_v,\ \delta\tau_r])$. As seen in Figure 2.4 we rely on two models, the thruster allocation model, and the marine craft model. The problem is, when an error is identified is it coming from the thruster allocation, or the ship model?

Chapter 3

Multivariate Analysis

3.1 Multivariate Analysis and Regression

With the increases of computational power , cheaper and more accurate sensors, and increasingly cheaper storage, the problem of handling quantitatively huge amount of data has exploded during the last decade. The buzzword related to this topic is big data.

The field of MVA is about handling multi dimensional data, or quantitative big data (Martens, 2015) and MVA techniques and knowledge has been developed for fields like chemometrics, genometrics, econometrics and psychometrics. Similar techniques have been used in control design in for example model reduction and balanced truncation, (Benner et al., 2013). Later these techniques have been tried in model-error analysis and prediction inspired by the work from (Martens et al., 2013b).

When faced with data that have multiple variables and samples, visualisation and understanding relationships in the data can be challenging. For smaller data sets, plotting variables against each other, or parallel in sub-plots can give this insight, but as the number of variables increase, it gets harder to find underlying relationships between the data. Additionally, it can be difficult to distinguish noise from casual non-linearities. The field of MVA addresses these issues with techniques fields for: Model reduction, data analysis and multi linear regression. The backbone of many of these techniques is the principal component analysis (PCA) (Martens et al., 2013a). This has already been applied in the control theory literature for model reduction, such as analysing a systems controllability and observability to find balanced truncation for a system. However it is often refereed to as singular value decomposition (SVD). In addition to the PCA, partial least square (PLS) regression will be used to analyse the experimental data in this project.

3.1.1 Calibration and Validation

MVA techniques, like PCA and PLS regression are empirical methods. PCA is used to analyse the data, to find if there are some greater structural correlations between the variables, that can explain the most important dynamics of the model. PLS regression on the

other hand, is a predictive technique. For building a model mathematical model is found from a set of data, to predict the response variables Y from predictor variables X. The finding of a model is divided into: calibration and validation (Martens and Næs, 1992). Calibration involves dividing the data, or finding a subset of the data from which you want to build a model. This also includes building the mathematical model from these data. This subset of the data is called the training set or calibration set. The process of assuring the quality of the model is called validation, and involves testing the models predictability against another subset of the data. These are called the validation set. In this project multivariate analysis was used to investigate the model-error of CSE1, and a model for this was presented. This was used to improve the simulator, and also a suggestion on how this can be implemented in the controller to improve the performance of the controllers is shown.

3.1.2 Principal Component Analysis/ Singular Value Decomposition

The PCA is an orthogonal decomposition of the data matrix, which as mentioned, is closely related to the SVD known from linear algebra. The strength of the PCA is shown when presented with a matrix of low rank, and this is often the case when the number of variables and states are grater than necessary i.e. not minimal. PCA is used to verify that the data has underlying structure so that the model of the data can be reduced. In this subsection the PCA will be reviewed through the SVD.

Consider a data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times m}$, where n is the number of samples or time steps, and m is the number of variables. This matrix has a SVD which is the following

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top \tag{3.1}$$

Here both $\boldsymbol{U} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{V} \in \mathbb{R}^{m \times m}$ are orthogonal matrices, while the $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times m}$ is a diagonal matrix, with its entries in acceding orders.

$$\boldsymbol{\Sigma} = diag([\sigma_1,\ \sigma_2,\ \ldots, \sigma_m]) \ \text{ where } \ \sigma_1, > \sigma_2, \ \ldots, > \sigma_m \tag{3.2}$$

and σ_i are called the singular values, which are the square roots of the eigenvalues of $\boldsymbol{X}\boldsymbol{X}^\top$ and $\boldsymbol{X}^\top\boldsymbol{X}$. The column vectors of $\boldsymbol{U}$ and $\boldsymbol{V}$ are the eigenvectors of $\boldsymbol{X}\boldsymbol{X}^\top$ and $\boldsymbol{X}^\top\boldsymbol{X}$ which are referred to as the left and right eigenvectors of $\boldsymbol{X}$.

$$\begin{aligned} \boldsymbol{U} &= [\boldsymbol{u}_1,\ \boldsymbol{u}_2,\ \ldots,\ \boldsymbol{u}_n] \\ \boldsymbol{V} &= [\boldsymbol{v}_1,\ \boldsymbol{v}_2,\ \ldots,\ \boldsymbol{v}_m] \end{aligned} \tag{3.3}$$

This decomposition gives several implications of the data matrix. The SVD gives an overview of the transformation through matrix multiplication of $\boldsymbol{X}$.

$$\begin{aligned} \boldsymbol{X}\boldsymbol{v}_i &= \sigma_i \boldsymbol{u}_i \\ \boldsymbol{X}^\top \boldsymbol{u}_i &= \sigma_i \boldsymbol{v}_i \end{aligned} \tag{3.4}$$

When vector $\boldsymbol{v}_1$ is multiplied by $\boldsymbol{X}$, it changes direction to $\boldsymbol{u}_1$ and is scaled with σ_1. Since σ_1 is the biggest singular value, it can be shown that $\boldsymbol{v}_1$ is the vector which is scaled the most when multiplied by $\boldsymbol{X}$.

$$\boldsymbol{v}_1 = \arg\max_{\boldsymbol{v}\in\mathbb{R}^m} \frac{||\boldsymbol{X}\boldsymbol{v}||_2}{||\boldsymbol{v}||_2} \tag{3.5}$$

and $\boldsymbol{v}_2$ is the $\arg\max$ in the subspace orthogonal to $\boldsymbol{v}_1$ etc. This also implies that the data will have biggest variance if it is projected onto $\boldsymbol{v}_1$. Another result of the SVD is that the matrix can be ordered sum of separable matrices. By separable, we mean that matrix $\boldsymbol{A}$ can be written as an outer product $\boldsymbol{A} = \boldsymbol{u}\boldsymbol{v}^\top \Leftrightarrow A_{ij} = u_i v_j$.

$$\boldsymbol{X} = \sum_{k=1}^{k=m} \boldsymbol{A}_k = \sum_{k=1}^{k=m} \sigma_k \boldsymbol{u}_k \boldsymbol{v}_k^\top \tag{3.6}$$

The matrix $\boldsymbol{A}_k$ will be called the principal components (PC). It can be seen that the first PC is the biggest part of $\boldsymbol{X}$ because the σ_1 is the biggest singular value. The matrix $\boldsymbol{X}$ can also be approximated by adding the first p principal components.

$$\boldsymbol{X} = \tilde{\boldsymbol{X}} + \boldsymbol{E}_p = \sum_{k=1}^{k=p} \boldsymbol{A}_k + \boldsymbol{E}_p \text{ where } p < m \tag{3.7}$$

The matrix $\boldsymbol{E}_p$ is the residual matrix containing the reminder of matrix $\boldsymbol{X}$. For matrices with low rank, or huge spread in singular values, an approximation where $p << m$ can give a fairly good approximation. The result is that $\boldsymbol{X}$ which have $m \times n$ data points can be approximated by p vectors $\boldsymbol{v}_i$, p vectors $\boldsymbol{u_i}$ and p singular values, which is substantial less if for instance $m = 100$ and $p = 5$. Then instead of storing the matrix of $100n$ elements, you just have to store the decomposition corresponding to $10n + 5$ elements, which takes the 10th of the storage capacity.

The PCA is alsmot identical to the SVD, only difference is that it is a decomposition of two matrices instead of three.

$$\boldsymbol{X} = \boldsymbol{Z}\boldsymbol{P}^\top \tag{3.8}$$

where $\boldsymbol{Z} = \boldsymbol{U}\boldsymbol{\Sigma}$ and is called the loadings, and $\boldsymbol{P} = \boldsymbol{V}$ is called the scores. The order is the same as in the singular value decomposition, which means that $\boldsymbol{z}_1$ has the largest magnitude, then comes $\boldsymbol{z}_2$ and so on.

When analysing the data through PCA the data is projected into a the score plane. As explained above, all the rows in the data matrix $\boldsymbol{X}$ are the different samples, so by multiplying $\boldsymbol{X}$ with the scores $\boldsymbol{p}_i$ a vector where the elements are the inner product between the samples and the score is created. This vector is the corresponding a loading to the data $\boldsymbol{X}$

$$\boldsymbol{z}_1 = \boldsymbol{X}\boldsymbol{p}_1 \tag{3.9}$$

By plotting the loadings against each other, the data comes out as projected onto the plane of the corresponding scores, with the score directions as the different axis, hence the scores are given the names principal directions.

3.1.3 Partial Least Squares Regression

PLS regression is a statistical projection method which finds correlating structure between two sets of data. It is therefore good at building models between two data sets $\boldsymbol{X}$ and $\boldsymbol{Y}$, especially when there are many correlated latent variables. Given the data set $\boldsymbol{X} \in \mathbb{R}^{n \times m}$ where each row is an observation/sample with m variables, and $\boldsymbol{Y} \in \mathbb{R}^{n \times p}$, where each row is an observation with p variables which we want to predict. The equations for the PLS regression model will then be

$$\begin{aligned} \boldsymbol{X} &= \boldsymbol{Z}\boldsymbol{P}_x^\top + \boldsymbol{E}_x \\ \boldsymbol{Y} &= \boldsymbol{Z}\boldsymbol{P}_y^\top + \boldsymbol{E}_y \\ \boldsymbol{Z} &= \boldsymbol{X}\boldsymbol{W}^* = \boldsymbol{X}\boldsymbol{W}(\boldsymbol{P}_x\boldsymbol{W})^{-1} \end{aligned} \tag{3.10}$$

The matrices $\boldsymbol{W}$, $\boldsymbol{Z}$ and $\boldsymbol{P}$ matrices are found through an iterative scheme, where $\boldsymbol{Z}$ contains variation in $\boldsymbol{X}$ and is correlated with the variation in $\boldsymbol{Y}$. The estimate of an observation $\boldsymbol{Y}$ can then be made

$$\begin{aligned} \tilde{\boldsymbol{y}} &= \boldsymbol{z}\boldsymbol{P}_y^\top \\ \tilde{\boldsymbol{y}} &= \boldsymbol{x}\boldsymbol{W}(\boldsymbol{P}_x\boldsymbol{W})^{-1}\boldsymbol{P}_y^\top = \boldsymbol{x}\boldsymbol{B}_x \end{aligned} \tag{3.11}$$

where $\tilde{\boldsymbol{Y}}$ and $\boldsymbol{X}$ are row vectors, and $\boldsymbol{B}_x \in \mathbb{R}^{m \times p}$

3.1.4 Validation and Statistics

As MVA deal with large amount of variables and samples, it can be difficult to see if a model represent the data in a good way, therefore some multivariate statistics and validation techniques have been created. To measure how well a PCA model approximates a data matrices the square Frobenius norm is used, which is defined as

$$||\boldsymbol{X}||_F^2 = \sum_{i=1}^{n}\sum_{j=1}^{m}|x_{ij}|^2 = \text{trace}(\boldsymbol{X}^\top\boldsymbol{X}) = \sum_{i=1}^{m}\sigma_i^2 \tag{3.12}$$

R^2, Calibration statistic: To see how well a model explains its own data, the goodness of fit R^2 is used, and is defined as

$$R^2 = \frac{||\boldsymbol{X}||_F^2 - ||\boldsymbol{X}_{\boldsymbol{\epsilon}}||_F^2}{||\boldsymbol{X}||_F^2} = \% \text{ of explained variance} \tag{3.13}$$

where $|\boldsymbol{X}$ is the data matrix, and $\boldsymbol{X}_{\boldsymbol{\epsilon}}$ is the residual matrix after the PCA model. We see that for a residual matrix $||\boldsymbol{X}_{\boldsymbol{\epsilon}}||_F^2 = 0$, $R^2 = 1$ and the opposite, where $\boldsymbol{X}_{\boldsymbol{\epsilon}} = |\boldsymbol{X}$ the $R^2 = 0$. This shows that R^2 is a measure of how well the PCA model approximates the data.

The R^2 is used on a calibration set, were you test how well the model predicts its own data, it is therefore safe to say that if enough factors or principal components are used, the R^2 will eventually go to 0.

Q^2, Validation statistic: When testing the model against new data, far validation, Q^2 is used. It is approximately the same as the R^2 statistic only that the validation sett $\boldsymbol{X}^v$ is used instead of $\boldsymbol{X}$. The $\tilde{\mathbf{E}}$ is the residual matrix, and the prediction error sum of squares (PRESS) is

$$PRESS = ||\boldsymbol{X}_{\epsilon}^{v}||_F^2 \tag{3.14}$$

and the Q^2 is then

$$Q^2 = 1 - \frac{PRESS}{SS_{tot}} = \frac{SS_{tot} - PRESS}{SS_{tot}} = \% \text{ of explained variance} \tag{3.15}$$

and as R^2 this also goes from 0 to 1, $0 < Q^2 < R^2 < 1$

3.1.5 Outlier Detection and Validation

To see how well the different samples fit into the created model, different statistics have been made to find outliers.

Q residual: The squared prediction error (SPE), or Q residual, is one of these. It is defined as

$$SPE_j = \sum_{i=1}^{n} e_{j,1}^2 \tag{3.16}$$

This can be interpreted as the orthogonal distance from a data point X to a latent variable space. If this distance is significantly higher for a data point than the rest of the data, it is a good indication that it is an outlier.

Hotelling's T^2: Hotelling's T^2 is a statistic for all the latent variable, and is the sum of the balanced distance from a point to the mean center of the data. Were the space is balanced down in the principal directions by the corresponding singular values. So a value being far away from the mean, in a direction were the singular value is very low, would get a high Hotelling's T^2. The Hotelling's T^2 is calculated as

$$T_i^2 = \frac{z_{i1}^2}{\sigma_1^2} + \frac{z_{i2}^2}{\sigma_2^2} + \frac{z_{i3}^2}{\sigma_3^2} + ... + \frac{z_{im}^2}{\sigma_m^2} \tag{3.17}$$

where $z_{i,j}$ is the loading element for the sample i , and σ_j is the singular value related to loading j. A sample having a significantly high T^2, is an indication that the sample is an outlier.

Validation: Cross validation is a technique for validating a predictive model, and it gives a good indication on how the prediction will work in practice. The data is divided into complementary subsets. On one set the analysis is done, and on the other subset the validation is done. The results of the cross validation is the average of all validations. The cross validation can also give a good indication of which variables are consistent for the model. In PLS regression a regression matrix is made, then by cross validation, several regression matrices will be made, and the elements of this matrix can be compared. If one element jumps around, especially changes sign for every regression, it can be deemed inconsistent, or statistically insignificant. And one can consider removing the variable related to this element for further analysis.

In this project the PCA was used to calibrate the data set by getting rid of spikes and dirty samples. The Q residual and Hotellin's T^2 were used to get rid of outliers, which mostly included spikes in the estimates. Then a PLS regression was done to analyse the model-error, and cross validation helped to find which variables that was important for the the prediction of the model-error.

3.2 Organization of data

There are several extensions of the PLS regression, and hey all require that the data is organized in a certain way. For the PLS regression method above, the data can be organized in two matrices. If you have 6 states $x_t \in \mathbb{R}^6$ and two measurement $y_t \in \mathbb{R}^2$ the matrices could look like in Table 3.1 and 3.2

Table 3.1: The data of $\boldsymbol{x}$ organized in a matrix $\boldsymbol{X}$

	← Variables →					
	x_1	x_2	x_3	x_4	x_5	x_6
t	⋮	⋮	⋮	⋮	⋮	⋮
↓	·	·	·	·	·	·

The object will then be to map the data from $\boldsymbol{X}$ to $\boldsymbol{Y}$, hence $\boldsymbol{x}$ to $\boldsymbol{y}$. As described in previous section, the PLS regression is able to produce a linear map from $\boldsymbol{x}$ to $\boldsymbol{y}$ noted matrix $\boldsymbol{B}$. However when the relation is nonlinear, two possible extensions of the PLS regression exist. The nonlinear extended PLS regression , and the balanced nominal-level PLS regression.

Table 3.2: The data of $\boldsymbol{y}$ organized in a matrix $\boldsymbol{Y}$

	← Variables →	
	y_1	y_2
t	⋮	⋮
↓	·	·

3.2.1 Nonlinear Partial Least Squares

The nonlinear extended PLS regression is quite common from other regressions, such as polynomial fitting. In addition to having the different states in the columns, the nonlinear combinations are also added. A quadratic extension would then be

Table 3.3: The data of $\boldsymbol{x}$ organized in a matrix $\boldsymbol{X}$ as an nonlinear quadratic extension

	← Variables →													
	x_1	x_2	x_3	x_4	x_5	x_6	x_1^2	x_2^2	x_3^2	x_4^2	x_5^2	x_6^2	x_1x_2	$\cdots$
t	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	$\ldots$
↓	·	·	·	·	·	·	·	·	·	·	·	·	·	$\cdots$

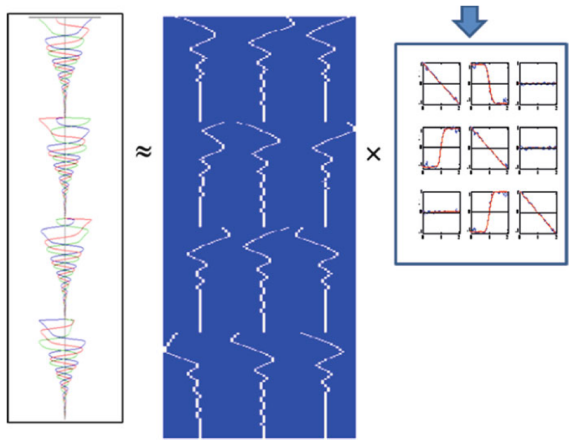
3.2.2 Nominal-level Partial Least Squares

The balanced nominal-level PLS regression works by dividing the states into vectors indicating the states value. If x_1 has values between $x_1 \in [0, 10]$, and the resolution of x_1 is decided to be 10. then the balanced nominal-level vector of x_1 would be

$$\boldsymbol{N}^{10}_{[0,10]}(x_1) = \begin{cases} [1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0] & 0 < x_1 < 1 \\ [0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0] & 1 < x_1 < 2 \\ [0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0] & 2 < x_1 < 3 \\ [0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0] & 3 < x_1 < 4 \\ [0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0] & 4 < x_1 < 5 \\ [0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0] & 5 < x_1 < 6 \\ [0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0] & 6 < x_1 < 7 \\ [0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0] & 7 < x_1 < 8 \\ [0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0] & 8 < x_1 < 9 \\ [0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1] & 9 < x_1 < 10 \end{cases} \tag{3.18}$$

This frees the PLS regression from being limited by proposed functions, and the resulting $\boldsymbol{B}$ matrix will plot out the function, as seen in Figure 3.1.

Figure 3.1 The figure shows the PLS regression of a dynamic system explained in (Martens et al., 2013a) with three states $[x_1,\ x_2,\ x_3]$. The box to the left, are the Y variables, which are the time derivatives of the states $[\dot{x}_1,\ \dot{x}_2,\ \dot{x}_3]$. The blue box in the middle with white lines are the three state variables nominalized as in (3.18), in which blue is 0 and white is 1, and the box to the left is the resulted $\boldsymbol{B} \in \mathbb{R}^{3\times 3n}$ matrix, where the functions represent the resulted vector that maps the nominal vector $N(x_i)$ to y_j.



3.2.3 Continuous Nominal Partial Least Squares

A further extension can be made by using the triangular function as basis for the nominal function, or other higher order basis functions as well.

$$\text{tri}(x)_{c\ h} = \max(-|c - x|/h + 1, 0) \tag{3.19}$$

the function can be seen in Figure 3.2, where c decides the position of the triangular's top, and h decides the breadth of the triangular. The triangles are ordered so that the they have value 1 in the middle of their section, and 0 at the edges. This leaves $h = Domain\_length/resolution$, and the result can be seen in Figure 3.3. To represent $x_1 = k$ where $k \in [1, 10]$

$$\begin{aligned}\boldsymbol{CN}^{10}_{[0,10]}(k) = [&\text{tri}(k)_{0\ 1},\ \text{tri}(k)_{1\ 1},\ \text{tri}(k)_{2\ 1},\ \text{tri}(k)_{3\ 1},\ ...\\ &\text{tri}(k)_{4\ 1},\ \text{tri}(k)_{5\ 1},\ \text{tri}(k)_{6\ 1},\ \text{tri}(k)_{7\ 1},\ \text{tri}(k)_{8\ 1},\ \text{tri}(k)_{9\ 1},\ \text{tri}(k)_{10\ 1}]\end{aligned} \tag{3.20}$$

3.2.4 Resulting Nominalisation

The nominalisation was done for both simulated and experimental data from chapters 5 and 6, which can be seen in Figure 3.4 and 3.5. The variables can be seen as plots, although they represent the value of the elements on the matrix. In the discrete plot, the red pixels correspond to a 1, while blue pixels correspond to a 0. For the continuous nomination

Figure 3.2 The figure shows the triangular function $\text{tri}(x)_{c\ h}$ described by (3.19). In this case it is $\text{tri}(x, 3, 1)$, where c decides the position of the triangular's top, and h decides the breadth of the triangular

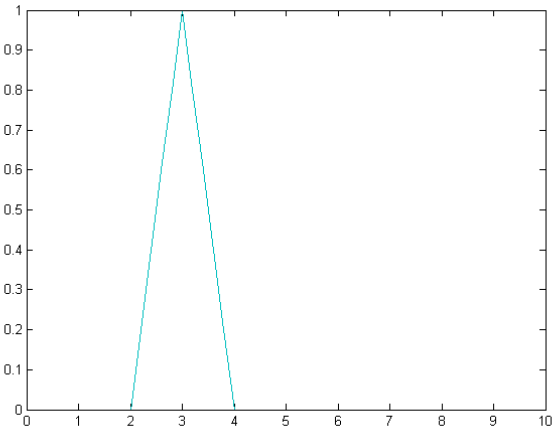
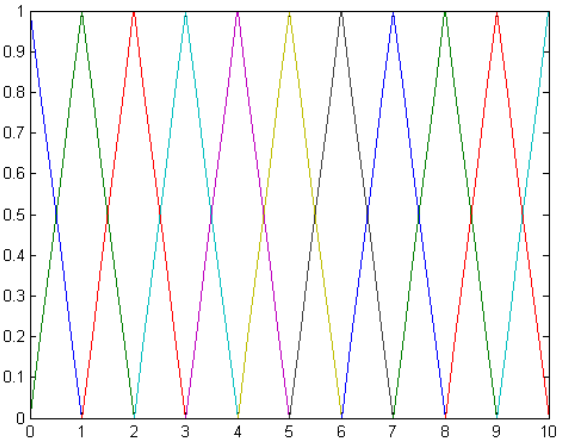


Figure 3.3 The figure shows the triangular function $\text{tri}(x)_{c\ h}$ described by (3.19). In this case it is $\text{tri}(x)_{3\ 1}$, where c decides the position of the triangular's top, and h decides the breadth of the triangular



the same rule applies, only that the colors between blue and red, correspond to a value between 0 and 1.

Figure 3.4 The Figure shows the discrete nominalisation of experimental data from an elliptic run. The value of the states can be seen evolving from left to right, looking like plots. Although this is a picture of the matrix seen from above, where a pixel represent value of the element in the matrix. The pixels with colour red, represents a 1, the pixels that are blue, represent a 0. The three lowest variables, which can be seen as the three lowest plots, are the states η. The three next variables, are the states ν. These are then followed by $\dot{\nu}$, τ, φ and the cross terms of ν, $\dot{\nu}$ and τ at the top

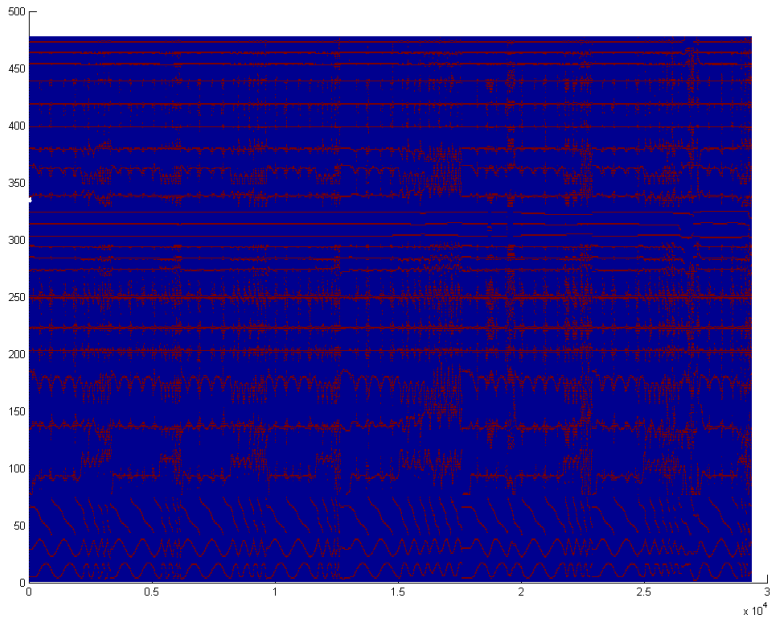
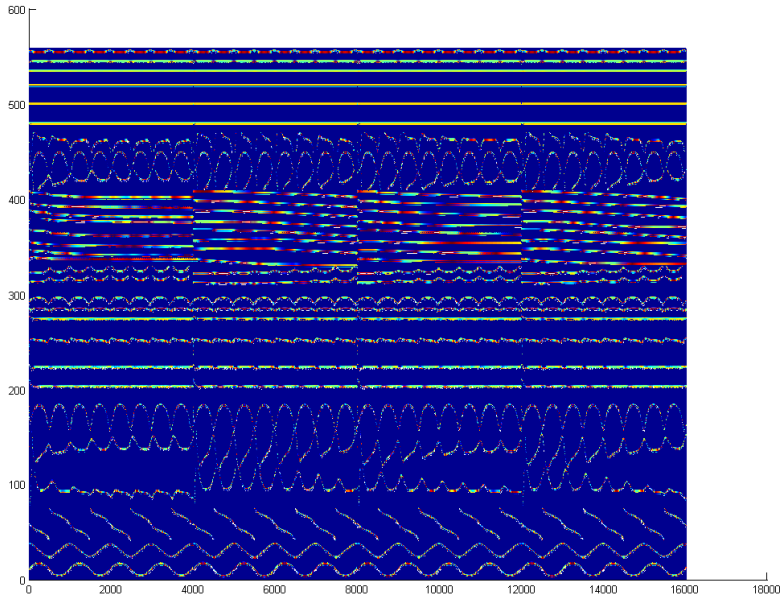


Figure 3.5 The Figure shows the continuous nominalisation of simulated data from an elliptic run. The value of the states can be seen evolving from left to right, looking like plots. Although this is a picture of the matrix seen from above, where a pixel represent value of the element in the matrix. The pixels with colour red, represents a 1, the pixels that are blue, represent a 0. Every pixel with a colour in between represent a number from 0 to 1. The three lowest variables, which can be seen as the three lowest plots, are the states η. The three next variables, are the states ν. These are then followed by $\dot{\nu}$, τ, φ and the cross terms of ν, $\dot{\nu}$ and τ at the top



Chapter 4

Motion Control System Design and Evaluation

4.1 Control Design

The controllers is designed for the the model of Cybership Enterprise 1 (CSE1) described in Section 2.1. The Adaptive backstepping (ABS) controller is used as the base controller for this project, and is designed with the control model (2.27). ABS controller with a concurrent learning (CL) adaptation law to estimate the uncertainties and disturbances are also tested, where different versions of CL is presented. The control model (2.27) was also used for the design the latter controllers. The controllers will be designed for the plant model, (2.1) and (2.2). The stability of the system will be proven through Lyapunov theory from appendix A to ensure that the controllers are robust and that their estimates does not diverge.

4.1.1 Adaptive Backstepping

Designing a Backstepping (BS) controller requires a certain procedure. One layer of the system is stabilized for every step by asserting a Lyapunov function for the sub system, and using Theorem A.1.1. Our system has two "layers" so we need two control signals that we want to stabilize to zero.

$$\begin{aligned} \boldsymbol{z_1} &\triangleq \boldsymbol{R}^\top(\psi)(\boldsymbol{\eta} - \boldsymbol{\eta_t}) \\ \boldsymbol{z_2} &\triangleq \boldsymbol{\nu} - \boldsymbol{\alpha} \end{aligned} \tag{4.1}$$

where $\boldsymbol{\eta_t}$ is the target of the ship, and $\boldsymbol{\alpha}$ is a stabilizing function found by using Theorem A.1.1. The goal is therefore to design a controller such that $\lim_{t\to\infty} \boldsymbol{z_1} = \boldsymbol{0}$, so that also $\boldsymbol{\eta} - \boldsymbol{\eta}_t$ becomes zero.

Step 1 Since we want the pose of the ship to converge towards $\boldsymbol{\eta}_t$, which means $\boldsymbol{\eta} = \boldsymbol{\eta}_t$, our first control Lyapunov function (CLF) will be

$$V_1 = \frac{1}{2}\boldsymbol{z_1}^\top \boldsymbol{z_1}, \tag{4.2}$$

which is certainly positive definite. We then find its time derivative

$$\dot{V}_1 = \boldsymbol{z_1}^\top \dot{\boldsymbol{z_1}} \tag{4.3}$$

which by the product rule and differential kinematics is

$$\begin{aligned} \dot{V}_1 &= \boldsymbol{z_1}^\top (\boldsymbol{S}(r)^\top \boldsymbol{R}^\top(\psi)(\boldsymbol{\eta} - \boldsymbol{\eta}_t) + \boldsymbol{R}^\top(\psi)(\dot{\boldsymbol{\eta}} - \dot{\boldsymbol{\eta}}_t)) \\ &= \boldsymbol{z_1}^\top (\boldsymbol{S}(r)^\top \boldsymbol{z_1} + \boldsymbol{R}^\top(\psi)(\dot{\boldsymbol{\eta}} - \dot{\boldsymbol{\eta}}_t))) \\ &= \boldsymbol{z_1}^\top (\boldsymbol{\nu} - \boldsymbol{R}^\top(\psi)\dot{\boldsymbol{\eta}}_t) \end{aligned} \tag{4.4}$$

where we in the last line has used the screw-symmetric property $\boldsymbol{z_1}^\top \boldsymbol{S}(r)\boldsymbol{z_1} = \boldsymbol{0}\ \forall \boldsymbol{z_1}$. We then see by introducing the control signal (4.1) that we get

$$\dot{V}_1 = \boldsymbol{z_1}^\top (\boldsymbol{z_2} + \boldsymbol{\alpha} - \boldsymbol{R}^\top(\psi)\dot{\boldsymbol{\eta}}_t). \tag{4.5}$$

By Theorem A.1.1 we want $\dot{V}_1 < 0$. We therefore choose

$$\boldsymbol{\alpha} = \boldsymbol{R}^\top(\psi)\dot{\boldsymbol{\eta}}_t - \boldsymbol{K_p}\boldsymbol{z_1} \tag{4.6}$$

and $\boldsymbol{K_p} > \boldsymbol{0}$, so we get the first step CLF derivative

$$\dot{V}_1 = -\boldsymbol{z_1}^\top \boldsymbol{K_p}\boldsymbol{z_1} + \boldsymbol{z_2}^\top \boldsymbol{z_1}. \tag{4.7}$$

The term $\boldsymbol{z_2}^\top \boldsymbol{z_1}$ will be used next to couple the $\boldsymbol{z_2}$ dynamics to V_1.

Step 2 The second CLF is chosen to be

$$V_2 = V_1 + \frac{1}{2}\boldsymbol{z_2}^\top \boldsymbol{M^*}\boldsymbol{z_2} = \frac{1}{2}\boldsymbol{z_1}^\top \boldsymbol{z_1} + \frac{1}{2}\boldsymbol{z_2}^\top \boldsymbol{M^*}\boldsymbol{z_2}, \tag{4.8}$$

which also is positive definite since $\boldsymbol{M}^* > \boldsymbol{0}$. The time derivative will then be

$$\dot{V}_2 = \dot{V}_1 + \boldsymbol{z_2}^\top \boldsymbol{M}^* \dot{\boldsymbol{z_2}} \tag{4.9}$$

For this controller we will use the uncertainty model described by (2.27). We use this to find $\boldsymbol{M}^*\dot{\boldsymbol{z_2}}$

$$\begin{aligned} \boldsymbol{M}^*\dot{\boldsymbol{z_2}} &= \boldsymbol{M}^*(\dot{\boldsymbol{\nu}} - \dot{\boldsymbol{\alpha}}) \\ &= \boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* + \boldsymbol{R}^\top(\psi)\boldsymbol{\omega_n} - \boldsymbol{M}^*\dot{\boldsymbol{\alpha}}. \end{aligned} \tag{4.10}$$

If we then substitude equation (4.10) into (4.9) we get the CLF

$$\begin{aligned}\dot{V}_2 = &- \boldsymbol{z_1}^\top \boldsymbol{K_p} \boldsymbol{z_1} \\ &+ \boldsymbol{z_2}^\top (\boldsymbol{z_1} + \boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* + \boldsymbol{R}^\top(\psi)\boldsymbol{\omega_n} - \boldsymbol{M}^*\dot{\boldsymbol{\alpha}}),\end{aligned} \tag{4.11}$$

which means that the controller has to be

$$\begin{aligned}\boldsymbol{\tau} = &- \boldsymbol{z_1} - \boldsymbol{K_d}\boldsymbol{z_2} + \boldsymbol{M}^*\dot{\boldsymbol{\alpha}} \\ &+ \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) + \boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* - \boldsymbol{R}^\top(\psi)\boldsymbol{\omega_n},\end{aligned} \tag{4.12}$$

for the Lyapunov function to satisfy Theorem A.1.1. And will get the derivative of the CLF to be

$$\dot{V}_2 = -\boldsymbol{z_1}^\top \boldsymbol{K_p} \boldsymbol{z_1} - \boldsymbol{z_2}^\top \boldsymbol{K_d} \boldsymbol{z_2}. \tag{4.13}$$

Step 3 The parameters in $\boldsymbol{\varphi}^*$ are unknown, hence they must be estimated. We choose the CLF

$$V_3 = \tilde{\boldsymbol{\varphi}}^\top \boldsymbol{\Gamma}_{\boldsymbol{\varphi}}^{-1} \tilde{\boldsymbol{\varphi}} + \tilde{\boldsymbol{\omega}}_n^\top \boldsymbol{\Gamma}_{\boldsymbol{\omega_n}}^{-1} \tilde{\boldsymbol{\omega_n}} + V_2 \tag{4.14}$$

where $\tilde{\boldsymbol{\varphi}} = \boldsymbol{\varphi}^* - \hat{\boldsymbol{\varphi}}$ and $\hat{\boldsymbol{\varphi}}$ is the estimate of $\boldsymbol{\varphi}^*$. Including the error from using the estimated values $\boldsymbol{\Phi}(\boldsymbol{\nu})\hat{\boldsymbol{\varphi}}$ and $\boldsymbol{R}^\top(\psi)\hat{\boldsymbol{\omega}}_n$ in $\boldsymbol{\tau}$ instead of the real values, into equation (4.11) gives

$$\dot{V}_2 = -\boldsymbol{z_1}^\top \boldsymbol{K_p} \boldsymbol{z_1} - \boldsymbol{z_2}^\top \boldsymbol{K_d} \boldsymbol{z_2} - \tilde{\boldsymbol{\varphi}}^\top \boldsymbol{\Phi}(\boldsymbol{\nu})^\top \boldsymbol{z_2} + \tilde{\boldsymbol{\omega}}_n^\top \boldsymbol{R}(\psi) \boldsymbol{z_2} \tag{4.15}$$

When differentiating V_3 bear in mind the assumption that the derivatives $\dot{\boldsymbol{\varphi}}^* = \dot{\boldsymbol{\varphi}}^* = \mathbf{0}$, or at least neglectable. This gives $\dot{\tilde{\boldsymbol{\varphi}}} = -\dot{\hat{\boldsymbol{\varphi}}}$ and $\dot{\tilde{\boldsymbol{\omega}}}_{\boldsymbol{n}} = -\dot{\hat{\boldsymbol{\omega}}}_n$ the last Lyapunov function is therefore

$$\begin{aligned}\dot{V}_3 = -\tilde{\boldsymbol{\varphi}}^\top \Gamma_{\boldsymbol{\varphi}}^{-1} \dot{\hat{\boldsymbol{\varphi}}} \quad &- \tilde{\boldsymbol{\omega_n}}^\top \Gamma_{\boldsymbol{\omega_n}}^{-1} \dot{\hat{\boldsymbol{\omega}}}_n \quad - \tilde{\boldsymbol{\varphi}}^\top \boldsymbol{\Phi}(\boldsymbol{\nu})^\top \boldsymbol{z_2} + \tilde{\boldsymbol{\omega}}_n^\top \boldsymbol{R}(\psi) \boldsymbol{z_2} \\ &- \boldsymbol{z_1}^\top \boldsymbol{K_p} \boldsymbol{z_1} - \boldsymbol{z_2}^\top \boldsymbol{K_d} \boldsymbol{z_2},\end{aligned} \tag{4.16}$$

reorganized look like

$$\begin{aligned}\dot{V}_3 = \tilde{\boldsymbol{\varphi}}^\top (-\boldsymbol{\Gamma}_{\boldsymbol{\varphi}}^{-1} \dot{\hat{\boldsymbol{\varphi}}} \quad &- \boldsymbol{\Phi}(\boldsymbol{\nu})^\top \boldsymbol{z_2}) + \\ \tilde{\boldsymbol{\omega}}_n^\top (-\boldsymbol{\Gamma}_{\boldsymbol{\omega_n}}^{-1} \dot{\hat{\boldsymbol{\omega}}}_n \quad &+ \boldsymbol{R}(\psi) \boldsymbol{z_2}) \\ &- \boldsymbol{z_1}^\top \boldsymbol{K_p} \boldsymbol{z_1} - \boldsymbol{z_2}^\top \boldsymbol{K_d} \boldsymbol{z_2}\end{aligned} \tag{4.17}$$

To eliminate the uncertainties of $\tilde{\boldsymbol{\varphi}}$ and $\tilde{\boldsymbol{\omega_n}}$ we choose the adaptation laws

$$\begin{aligned}\dot{\hat{\boldsymbol{\varphi}}} &= -\boldsymbol{\Gamma}_{\boldsymbol{\varphi}} \boldsymbol{\Phi}^\top(\boldsymbol{\nu}) \boldsymbol{z_2} \\ \dot{\hat{\boldsymbol{\omega}}}_n &= \boldsymbol{\Gamma}_{\boldsymbol{\omega_n}} \boldsymbol{R}(\psi) \boldsymbol{z_2}\end{aligned} \tag{4.18}$$

The controller is summarized in Table 4.2.

4.1.2 Concurrent Learning

CL is an adaptive law based on the intuition that if the recorded data is sufficiently rich, i.e. there is a linear independence in the data, CL adaptation can be used to estimate true values without the need of persistency of excitation (Chowdhary and Johnson, 2010). For this algorithm to work, certain conditions have to be fulfilled.

The adaptation law is as follows

$$\dot{\hat{\boldsymbol{\varphi}}} = -\boldsymbol{\Gamma}_{\boldsymbol{\varphi}} \boldsymbol{\Phi}^{\top}(\boldsymbol{\nu}_n)\boldsymbol{\epsilon}_n - \sum_{j=1}^{n-1} \boldsymbol{\Gamma}_{\boldsymbol{\varphi}} \boldsymbol{\Phi}^{\top}(\boldsymbol{\nu}_j)\boldsymbol{\epsilon}_{\boldsymbol{j}} \tag{4.19}$$

where $\boldsymbol{\nu}$ are the states, $\hat{\boldsymbol{\varphi}}$ is the parameter estimates of $\boldsymbol{\varphi}^*$, $\boldsymbol{\Gamma}_{\boldsymbol{\varphi}}$ is the adaptive gain matrix and $\boldsymbol{\Phi}(\boldsymbol{\nu}_j)$ is the regressor matrix. $\boldsymbol{\epsilon}$ is the approximation error, denoted $\boldsymbol{\epsilon} = \boldsymbol{y} - \hat{\boldsymbol{y}}$ where

$$\begin{aligned} \boldsymbol{y}(t) &= \boldsymbol{\Phi}(\boldsymbol{\nu}(t))\boldsymbol{\varphi}^* \\ \hat{\boldsymbol{y}}(t) &= \boldsymbol{\Phi}(\boldsymbol{\nu}(t))\hat{\boldsymbol{\varphi}} \end{aligned} \tag{4.20}$$

Which leads to

$$\boldsymbol{\epsilon} = \Phi(\boldsymbol{\nu}(t))\tilde{\boldsymbol{\varphi}} \tag{4.21}$$

Then, by setting $\Gamma_{\phi} = I_{n\times n}$ and assuming that $\boldsymbol{\varphi}^*$, the error dynamics become

$$\dot{\tilde{\boldsymbol{\varphi}}} = -\boldsymbol{\Phi}^{\top}(\boldsymbol{\nu}_n)\boldsymbol{\epsilon}_n - \sum_{j=1}^{n-1} \boldsymbol{\Phi}^{\top}(\boldsymbol{\nu}_j)\boldsymbol{\epsilon}_{\boldsymbol{j}} \tag{4.22}$$

then substituting for $\boldsymbol{\epsilon}$

$$\dot{\tilde{\boldsymbol{\varphi}}} = -\boldsymbol{\Phi}^{\top}(\boldsymbol{\nu}_n)\Phi(\boldsymbol{\nu}(t))\tilde{\boldsymbol{\varphi}} - \sum_{j=1}^{n-1} \boldsymbol{\Phi}^{\top}(\boldsymbol{\nu}_j)\Phi(\boldsymbol{\nu}_j)\tilde{\boldsymbol{\varphi}} \tag{4.23}$$

which has some linear structure. Then we present a condition of the richness of the data, that is required for the convergence analysis.

Condition 1 The recorded data has as many linearly independent elements as the dimension of $\boldsymbol{\Phi}(x(t)) \in \mathbb{R}^{n\times m}$. That is if $\boldsymbol{Z} = [\boldsymbol{\Phi}(x_1)^{\top},\ \boldsymbol{\Phi}(x_2)^{\top},\ ...,\ \boldsymbol{\Phi}(x_p)^{\top}]$, then $rank(\boldsymbol{Z}) = m$.

Then the theory says

Theorem 4.1.1. *If the stored data points satisfy condition 1, then $\tilde{\boldsymbol{\varphi}}$ is globally exponentially stable when using the CL gradient descent weight adaptation law of equation* (4.19)*.*

Proof. Let

$$V = \frac{1}{2}\tilde{\boldsymbol{\varphi}}^\top \tilde{\boldsymbol{\varphi}} \tag{4.24}$$

be a CLF of the estimation error. We then differentiate V

$$\begin{aligned} \dot{V} &= \tilde{\boldsymbol{\varphi}}^\top \dot{\boldsymbol{\varphi}} \\ \dot{V} &= -\tilde{\boldsymbol{\varphi}}^\top (\boldsymbol{\Phi}^\top(\boldsymbol{\nu}_n)\Phi(\boldsymbol{\nu}(t)) + \sum_{j=1}^{n-1} \boldsymbol{\Phi}^\top(\boldsymbol{\nu}_j)\Phi(\boldsymbol{\nu}_j))\tilde{\boldsymbol{\varphi}} \\ \dot{V} &= -\tilde{\boldsymbol{\varphi}}^\top \boldsymbol{Q}\tilde{\boldsymbol{\varphi}} \end{aligned} \tag{4.25}$$

where $\boldsymbol{Q} = \boldsymbol{Q}_c + \boldsymbol{Q}_d$ is positive definite due to Condition 1. The $\boldsymbol{Q}_d$ is the data matrix defined in Condition 1. We also note that

$$-\tilde{\boldsymbol{\varphi}}^\top \boldsymbol{Q}\tilde{\boldsymbol{\varphi}} >= \lambda_{\min}||\tilde{\boldsymbol{\varphi}}|| \tag{4.26}$$

where $\lambda_{\min}$ is the smallest singular value of $\boldsymbol{Q}$

$$\dot{V} < -\lambda_{\min}||\tilde{\boldsymbol{\varphi}}|| \tag{4.27}$$

□

We also note that a bigger $\boldsymbol{Q}$ will probably also lead to a bigger convergence. Thus the product of the singular values will be tried out.

In the proof it is required that $\sum_{j=1}^{n-1} \boldsymbol{\Phi}(\boldsymbol{\nu_j})^\top \boldsymbol{\Phi}(\boldsymbol{\nu_j}) > 0$ is positive definite. We will try out two methods to ensure that this property is hold.

4.1.3 Concurrent Learning Choosing Algorithm

From (4.26) the convergence rate is related to the matrix $\boldsymbol{Q}$. And the CL gives the option of choosing witch data do build up $\boldsymbol{Q}_d$. Here two algorithms will be presented for how the data is to be chosen.

Concurrent Learning With Data Window The data window algorithm works like a tube with a constant number of matrices. So that if a new measurement is sufficiently different from the previous one then the regression matrix is stored, and the oldest regression matrix is rejected.

Algorithm 1 pseudocode for data window choosing algorithm

$end \leftarrow n$
for $i = 1$ to N **do**
  $\boldsymbol{\Phi}_{temp} \leftarrow \boldsymbol{\Phi}(\boldsymbol{\nu}_i)$
  $\boldsymbol{\Phi}_p \leftarrow \boldsymbol{\Phi_M}[1]$
  if norm$(\boldsymbol{\Phi}_{temp} - \boldsymbol{\Phi}_p) < \delta$ **then**
    $\boldsymbol{\Phi_M}[2 : end] \leftarrow \boldsymbol{\Phi_M}[1 : end - 1]$
    $\boldsymbol{\Phi_M}[1] \leftarrow \boldsymbol{\Phi}_{temp}$ { % Queuing the regression matrices}
    $\boldsymbol{\Sigma_M}[2 : end] \leftarrow \boldsymbol{\Sigma_M}[1 : end - 1]$
    $\boldsymbol{\Sigma_M}[1] \leftarrow \boldsymbol{\epsilon_i}$ { % Queuing the ϵ error vectors}
    $\boldsymbol{Q_d} = \Phi_M * \Sigma_M$ { % Multiplication like in Σ (4.19)}
  end if
end for

Concurrent Learning With Singular Value Maximization The singular value maximization (SVM) algorithm works by checking if the new regression matrix will increase the minimum singular value. It tries to change all its data matrices with the new data, and if the new data point increases the minimal singular value of the data matrix, switch in the new regression matrix.

Algorithm 2 pseudocode for maximizing SVD choosing algorithm

$end \leftarrow n$
for $i = 1$ to N **do**
  $\boldsymbol{\Phi}_{temp} \leftarrow \boldsymbol{\Phi}(\boldsymbol{\nu}_i)$
  $\boldsymbol{\Phi}_p \leftarrow \boldsymbol{\Phi_M}[1]$
  if norm$(\boldsymbol{\Phi}_{temp} - \boldsymbol{\Phi}_p) < \delta$ **then**
    for $j = 1$ to end **do**
      $\boldsymbol{\Phi}_{\boldsymbol{M_{temp}}} \leftarrow \boldsymbol{\Phi_M}$
      $\boldsymbol{\Phi}_{\boldsymbol{M_{temp}}}[j] \leftarrow \boldsymbol{\Phi}_{temp}$
      $SingValues \leftarrow$ svd$(\boldsymbol{\Phi}_{\boldsymbol{M_{temp}}})$
      $MinSingValues[j] \leftarrow \min(SingValues)$
    end for
    $[MinSing, argMinSing] = \max(MinSingValues)$
    if $MinSing \geq MinSingOld$ **then**
      $\boldsymbol{\Phi_M}[argMinSing] \leftarrow \boldsymbol{\Phi}_{temp}$
      $\boldsymbol{\Sigma_M}[argMinSing] \leftarrow \boldsymbol{\epsilon_i}$
      $\boldsymbol{Q_d} = \Phi_M * \Sigma_M$ { % Multiplication like in Σ (4.19)}
    end if
  end if
end for

4.1.4 Model-Error Parametrization and Estimation

For model (2.27) the regression matrix will be

$$\boldsymbol{\Phi}_{reg}(\boldsymbol{\nu}) = [\boldsymbol{\Phi}(\boldsymbol{\nu}),\ \boldsymbol{R}(\psi)] \tag{4.28}$$

and the parameter vector is

$$\boldsymbol{\varphi}^*_{reg} = \begin{bmatrix} \boldsymbol{\varphi}^* \\ \boldsymbol{\omega}_n \end{bmatrix} \tag{4.29}$$

Instead of having one regression matrix, they are split, because the changes in the hydrodynamic regression matrix $\boldsymbol{\Phi}(\boldsymbol{\nu})$ are independent of the rotation matrix $\boldsymbol{R}(\psi)$, and when storing the the data for the CL, the data chosen for the hydrodynamic estimation will be separated from the data chosen for disturbance estimation.

$$\dot{\hat{\boldsymbol{\varphi}}} = -\boldsymbol{\Gamma}_{\boldsymbol{\varphi}}\boldsymbol{\Phi}(\boldsymbol{\nu})^\top\boldsymbol{\epsilon} - \sum_{j=1}^{k-1}\boldsymbol{\Gamma}_{\boldsymbol{\phi}}\boldsymbol{\Phi}^\top(\boldsymbol{\nu}_j)\boldsymbol{\epsilon_j}, \tag{4.30}$$

the $\boldsymbol{\epsilon}$ is found by rearranging model (2.27), we have

$$\begin{aligned} \boldsymbol{\omega}^* = \boldsymbol{R}^\top(\psi)\boldsymbol{\omega}^*_n - \boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* = \boldsymbol{M}^*\dot{\boldsymbol{\nu}} + \boldsymbol{C}^*(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\tau} \\ \boldsymbol{y} = \boldsymbol{M}^*\dot{\boldsymbol{\nu}} + \boldsymbol{C}^*(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\tau}, \end{aligned} \tag{4.31}$$

and our estimate of the uncertainty is

$$\boldsymbol{\omega} = \hat{\boldsymbol{y}} = \boldsymbol{R}^\top(\psi)\hat{\boldsymbol{\omega}}_n - \boldsymbol{\Phi}(\boldsymbol{\nu})\hat{\boldsymbol{\varphi}}. \tag{4.32}$$

We then get

$$\begin{aligned} &\boldsymbol{\epsilon} = \boldsymbol{y} - \hat{\boldsymbol{y}} \\ &\boldsymbol{\epsilon} = \boldsymbol{M}^*\dot{\boldsymbol{\nu}} + \boldsymbol{C}^*(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\tau} - (\boldsymbol{R}^\top(\psi)\hat{\boldsymbol{\omega}}_n - \boldsymbol{\Phi}(\boldsymbol{\nu})\hat{\boldsymbol{\varphi}}) \end{aligned} \tag{4.33}$$

For the earth-fixed disturbance, the regression matrix is the rotation matrix, and by definition $\boldsymbol{R}(\psi)^\top\boldsymbol{R}(\psi) = \boldsymbol{I}$ and the positive definite condition is satisfied. We then get the adaptation law

$$\dot{\hat{\boldsymbol{\omega}}}_{nk} = \boldsymbol{\Gamma}_{\boldsymbol{\omega_k}}\boldsymbol{R}(\psi_k)\boldsymbol{\epsilon}_k - \sum_{j=1}^{k-1}\boldsymbol{\Gamma}_{\boldsymbol{\omega_n}}\boldsymbol{R}(\boldsymbol{\psi}_j)\boldsymbol{\epsilon_j}. \tag{4.34}$$

In the project thesis, a choosing algorithm wasn't used, but rather all the data was taken into account. The problem was then that the algorithm didn't take time into consideration. With the result of different adaptive behaviour depending on the time between the measurements, or step length in the simulations. The summation part was therefore transformed into an integral

$$\begin{aligned}\dot{\hat{\boldsymbol{\omega}}}_{nk} =& \boldsymbol{\Gamma}_{\boldsymbol{\omega_n}} \boldsymbol{R}(\psi_k)\boldsymbol{\epsilon}_k - \sum_{j=1}^{k-1} \boldsymbol{\Gamma}_{\boldsymbol{\omega_n}} \boldsymbol{R}(\boldsymbol{\psi}_j)\boldsymbol{\epsilon_j} h \\ \dot{\hat{\boldsymbol{\omega}}}_n(t) =& \boldsymbol{\Gamma}_{\boldsymbol{\omega_n}} \boldsymbol{R}(\psi(t))\boldsymbol{\epsilon}(t) - \rho \int_{\tau=t_0}^{t} \boldsymbol{\Gamma}_{\boldsymbol{\omega_n}} \boldsymbol{R}(\psi(\tau))\boldsymbol{\epsilon}(\tau) d\tau\end{aligned} \tag{4.35}$$

where h is the time between measurement, or step length in the simulations. The ρ is a tuning parameter to ensure stable adaptation[1].

$$\boldsymbol{\epsilon}(t) = \boldsymbol{M}\dot{\boldsymbol{\nu}}(t) + \boldsymbol{C}(\boldsymbol{\nu}(t))\boldsymbol{\nu}(t) + \boldsymbol{g}(\boldsymbol{\nu}(t)) - \boldsymbol{\tau}(t) - (\boldsymbol{R}^\top(\psi(t))\hat{\boldsymbol{\omega}}_n(t) - \boldsymbol{\Phi}(\boldsymbol{\nu}(t))\hat{\boldsymbol{\varphi}}(t)) \tag{4.36}$$

In this project, the h was dependent on how many data points that was stored, and both the CL with window algorithm and singular value maximization was simulated and tested in a hardware-in-the-loop (HIL) test. The CL widow algorithm was in addition tested on the CSE1. The CL singular value maximization algorithm proved to have problems that are discussed in Chapter 7.

4.1.5 Modifications of Concurrent Learning

As can be seen from previous subsection, CL can be implemented in different ways, by which data is stored. In addition the instantaneous adaptation error $\boldsymbol{\epsilon}$ can be changed with tracking error $e = x_{rm}x$, which is done for a model refrence adaptive control in (Chowdhary and Johnson, 2011a), this can also be applied to the BS controller, only using z_2 in the adaptation as can be seen from appendix A.2

$$\begin{aligned}\dot{\hat{\boldsymbol{\varphi}}} &= \boldsymbol{\Gamma}_\varphi \boldsymbol{\Phi}^\top \mathbf{z}_2 + \sum_{j=1}^{p} \boldsymbol{\Gamma}_\phi \boldsymbol{\Phi}_j^\top \boldsymbol{\epsilon}_j \\ \dot{\hat{\boldsymbol{w}}}_n &= \boldsymbol{\Gamma}_w \mathbf{R} \mathbf{z}_2 + \sum_{j=1}^{p} \boldsymbol{\Gamma}_w \mathbf{R}_j \boldsymbol{\epsilon}_j\end{aligned} \tag{4.37}$$

This leaves four possible implementations for the CL controllers, as can be seen in Table 4.1. Where the SVD is claimed to have the best convergence, but as implemented it can potentially store an error and this can lead to instability. The epsilon has the possibility of faster convergence than $\boldsymbol{z_2}$, but as also discussed in Chapter 6, getting an good instantaneous estimate of $\boldsymbol{\epsilon}$ proved to be difficult. In Chapter 5 the four versions of the CL controllers are simulated, and their performance is compared, and the adaptation laws and controllers are summarized in Table 4.2.

[1]The $\boldsymbol{\Gamma}_{\omega_n}$ in the integral term of the CL adaptation was reduced by factor of $\rho = 0.1$ to make the adaptation stable for all the simulated tests.

Table 4.1: The different variants of the CL controller

Adaptation Error ⧹ Storage Algorithm	Window	Singular Value Maximation
$\boldsymbol{\epsilon}$	CL-WIN-ep	CL-SVD-ep
$\boldsymbol{z_2}$	CL-WIN-z2	CL-SVD-z2

4.1.6 Summary

In Table 4.2, the summary of the BS controller and the different adaptations are found. Since the CL controllers can change between having two storage algorithms and two instantaneous adaptation errors, this leaves us with four possible CL controllers.

Table 4.2: Summary table of the controller and adaptations presented in this section. The XXX means either WIN or SVD, and similarly X stands for W or S

Backstepping Controller
Internal Signal:
$\boldsymbol{z_1} = \boldsymbol{R}^\top(\psi)(\boldsymbol{\eta} - \boldsymbol{\eta_t})$ $\boldsymbol{z_2} = \boldsymbol{\nu} - \boldsymbol{\alpha}$ $\boldsymbol{\alpha} = \boldsymbol{R}^\top(\psi)\dot{\boldsymbol{\eta}}_t - \boldsymbol{K_p z_1}$ $\dot{\boldsymbol{\alpha}} = \boldsymbol{S}^\top(r)\boldsymbol{R}^\top(\psi)\dot{\boldsymbol{\eta}}_t + \boldsymbol{R}^\top(\psi)\ddot{\boldsymbol{\eta}}_t - \boldsymbol{K_p}\dot{\boldsymbol{z_1}}$ $\dot{\boldsymbol{z_1}} = \boldsymbol{S}^\top(r)\boldsymbol{z_1} + \boldsymbol{\nu} - \boldsymbol{R}^\top(\psi)\dot{\boldsymbol{\eta}}_t$
Control Law:
$\boldsymbol{\tau} = -\boldsymbol{z_1} - \boldsymbol{K_d z_2} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) + \boldsymbol{\Phi}(\boldsymbol{\nu})\hat{\boldsymbol{\varphi}} - \boldsymbol{R}^\top(\psi)\hat{\boldsymbol{\omega}}_n + \boldsymbol{M}^*\dot{\boldsymbol{\alpha}}$

Adaptation Laws	
Backstepping	Adaptive Backstepping
Abbreviations: BS-NORMAL, BSP	Abbreviations: BS-ADAPT, ABS
Internal signals: NONE	Internal signals: $\boldsymbol{z_2} = \boldsymbol{\nu} - \boldsymbol{\alpha}$
Adaptive Law: $\dot{\hat{\boldsymbol{\varphi}}} = 0$ $\dot{\hat{\boldsymbol{\omega}}}_n = 0$	 $\dot{\hat{\boldsymbol{\varphi}}} = -\boldsymbol{\Gamma_\varphi}\boldsymbol{\Phi}^\top(\boldsymbol{\nu})\boldsymbol{z_2}$ $\dot{\hat{\boldsymbol{\omega}}}_n = \boldsymbol{\Gamma}_{\boldsymbol{\omega}_n}\boldsymbol{R}(\psi)\boldsymbol{z_2}$

Concurrrent Learning with z_2	Concurrrent Learning with ϵ
Abbreviations: CL-XXX-z2, CXZ	Abbreviations: CL-XXX-ep, CXZ
Internal signals: $\boldsymbol{z_2} = \boldsymbol{\nu} - \boldsymbol{\alpha}$ $\boldsymbol{\epsilon} = \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\tau}$ $-(\boldsymbol{R}^\top(\psi)\hat{\boldsymbol{\omega}}_n - \boldsymbol{\Phi}(\boldsymbol{\nu})\hat{\boldsymbol{\varphi}})$	Internal signals: $\boldsymbol{\epsilon} = \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\tau}$ $-(\boldsymbol{R}^\top(\psi)\hat{\boldsymbol{\omega}}_n - \boldsymbol{\Phi}(\boldsymbol{\nu})\hat{\boldsymbol{\varphi}})$
Adaptive Law: $\dot{\hat{\boldsymbol{\varphi}}} = -\boldsymbol{\Gamma_\varphi}\boldsymbol{\Phi}(\boldsymbol{\nu})^\top\boldsymbol{z_2} - \sum_{j=1}^{k-1}\boldsymbol{\Gamma_\phi}\boldsymbol{\Phi}^\top(\boldsymbol{\nu}_j)\boldsymbol{\epsilon_j}$ $\dot{\hat{\boldsymbol{\omega}}}_{nk} = \boldsymbol{\Gamma}_{\boldsymbol{\omega_k}}\boldsymbol{R}(\psi_k)\boldsymbol{z_2} - \sum_{j=1}^{k-1}\boldsymbol{\Gamma}_{\boldsymbol{\omega_n}}\boldsymbol{R}(\boldsymbol{\psi}_j)\boldsymbol{\epsilon_j}$	Adaptive Law: $\dot{\hat{\boldsymbol{\varphi}}} = -\boldsymbol{\Gamma_\varphi}\boldsymbol{\Phi}(\boldsymbol{\nu})^\top\boldsymbol{\epsilon} - \sum_{j=1}^{k-1}\boldsymbol{\Gamma_\phi}\boldsymbol{\Phi}^\top(\boldsymbol{\nu}_j)\boldsymbol{\epsilon_j}$ $\dot{\hat{\boldsymbol{\omega}}}_{nk} = \boldsymbol{\Gamma}_{\boldsymbol{\omega_k}}\boldsymbol{R}(\psi_k)\boldsymbol{\epsilon} - \sum_{j=1}^{k-1}\boldsymbol{\Gamma}_{\boldsymbol{\omega_n}}\boldsymbol{R}(\boldsymbol{\psi}_j)\boldsymbol{\epsilon_j}$

Data Storage Algorithm	
Window: WIN Algorithm: 3	*Singular Value Maximation:* SVD Algorithm: 2

4.2 Guidance Design

The guidance system produces the target states $\boldsymbol{\eta}_t$ that the ship is supposed to follow. The guidance system in this project is made from parametrization.

$$\begin{aligned} \boldsymbol{\eta}_t &= g(\theta) \\ \dot{\theta} &= f(\theta) \end{aligned} \tag{4.38}$$

There were two trajectories chosen for the simulations and testing. One elliptic and one Figure-Eight trajectory. The elliptic trajectory had the parametrization

$$\boldsymbol{\eta}_t = \begin{bmatrix} 5 + \sin((\frac{\pi}{180})\theta) \\ 0.5 + 1.5\cos((\frac{\pi}{180})\theta) \\ \text{atan2}(-1.5\sin(\frac{\pi}{180}\theta), \cos(\frac{\pi}{180}\theta)) \end{bmatrix} \tag{4.39}$$

and

$$\dot{\theta} = \frac{u_t}{\sqrt{(1.5\frac{\pi}{180}\sin(\frac{\pi}{180}\theta))^2 + (\frac{\pi}{180}\cos(\frac{\pi}{180}\theta))^2}} \tag{4.40}$$

where u_t is the target speed witch will be held constant in the elliptic trajectory. The eight figure trajectory is parametrized as

$$\boldsymbol{\eta}_t = \begin{bmatrix} 5 + \cos((\frac{\pi}{180})\theta) \\ 0.5 - 2\sin((\frac{\pi}{180})\theta)\cos((\frac{\pi}{180})\theta) \\ \text{atan2}(-2\cos(\frac{\pi}{90}\theta), -\sin(\frac{\pi}{180}\theta)) \end{bmatrix} \tag{4.41}$$

and

$$\dot{\theta} = 5.33u_t \tag{4.42}$$

the $\dot{\boldsymbol{\eta}}_t$ and $\ddot{\boldsymbol{\eta}}$ where found through finite differences method, is a numerical method for differentiating a signal. One can increase the accuracy of the differentiation by choosing an differentiation of higher order, although it will lead to more noise if there are any inaccuracies. The differentiations in different order of accuracy and derivative can be seen in Table 4.3

Depending on the frequency and accuracy of the function that is to be differentiated may determine which of the methods to use. When differentiating the guidance function, the frequency was so high that first order forward differentiation gave sufficiently good $\dot{\boldsymbol{\eta}}_t$ and $\ddot{\boldsymbol{\eta}}_t$.

$$\hat{\dot{\boldsymbol{\eta}}}_{\boldsymbol{t}} = \frac{{\boldsymbol{\eta}_{\boldsymbol{t}}}^n - {\boldsymbol{\eta}_{\boldsymbol{t}}}^{n-1}}{h} \tag{4.43}$$

$$\hat{\ddot{\boldsymbol{\eta}}}_{\boldsymbol{t}} = \frac{{\boldsymbol{\eta}_{\boldsymbol{t}}}^n - 2{\boldsymbol{\eta}_{\boldsymbol{t}}}^{n-1} + {\boldsymbol{\eta}_{\boldsymbol{t}}}^{n-2}}{h^2} \tag{4.44}$$

Table 4.3: Table of the coeficents for the finite difference forward methods. These functions depend on how many measurements that are planed

Derivative	Accuracy	0	-1	-2	-3	-4	-5	-6	-7	-8
1	1	-1	1							
	2	-3/2	2	-1/2						
	3	-11/6	3	-3/2	1/3					
2	2	1	-2	1						
	4	2	-5	4	-1					
	6	35/12	-26/3	19/2	-14/3	11/12				

4.3 State Estimation

During the tests, only the position and orientation $\boldsymbol{\eta}$ of the vessel is measured, hence the $\boldsymbol{\nu}$ and $\dot{\boldsymbol{\nu}}$ had to be estimated. This can be done with an nonlinear observers described in (Sørensen, 2013), or a derivation filter. Three observers were tested, and the results can be seen in Section 6.3.3.

Dealing with real life measurements can provide a few challenges for the estimators and the controllers

4.3.1 Measurement Challenges

When using measurements in a real time control setting, the measurements usually has several faults. Such faults can be identified as

Noise: Noise is a short term error in determining the value of the measurement, and can be a result of physical or computational effects. It can often be characterized as a probability distribution, but usually it is described as white Gaussian noise with a certain variance.

Bias: A bias is a systematically error in the measurement. An example can be that the measurement point on a ship is wrong, so that this fault is propagated to the position measurements and there is an error in the position measurement. These errors, when found can often be compensated for so that they are cancelled out.

Signal freeze: If the measurement system is unable to give out an updated measurement, the measurement system will continue to give out a frozen measurement, until the system is able to find a new update of the measurement. Then the measurement will jump to the newly updated measurement. This can both affect the controller by falsely providing an extensive error into the controller, but also the jump can set of oscillations if the controller isn't well tuned.

Signal drop: The signal drop, is the same as the signal freeze, but instead of giving out a constant measurement, they are either set to zero, or nan. which is why they often have to be handled in another way

Wrap around: The problem with the angle measurements is that they are finite $\psi \in [-\pi, \pi]$. This means that the measurements will jump from $-\pi$ to π and visa versa. In a control setting this can cause problems, and make spikes in $\tilde{\psi}$.

Signal spikes: Signal spikes can come from measurements or estimates that for some reasons are way off in a temporarily moment. They can often induce osculations on controlled systems.

The pose measurements in the MC-lab had very little noise, but signal freeze occurred occasionally. To handle this, the estimators were modified so that they identified if there was a signal freeze, and if that was the case, tried to estimate $\hat{\boldsymbol{\eta}}$ as seen in (4.46), (4.49) and (4.52). In addition, the measurements were used for estimating the ships velocities and accelerations, resulting in a spike in these estimation every time the signal was recovered.

4.3.2 Modified Derivative Filter

The derivative filter works by low pass filtering the the measurement signal, then differentiating the filtered signal, and then differentiating again, as can be seen below.

$$\begin{aligned} \boldsymbol{\eta}_{f,1} =& a\boldsymbol{\eta}_m + (1-a)\boldsymbol{\eta}_{f,1,k-1} \\ \dot{\boldsymbol{\eta}} =& \frac{\boldsymbol{\eta}_{f,1} - \boldsymbol{\eta}_{f,1,k-1}}{h} \\ \dot{\boldsymbol{\eta}}_{f,2} =& b\dot{\boldsymbol{\eta}} + (1-b)\dot{\boldsymbol{\eta}}_{f,2,k-1} \\ \hat{\boldsymbol{\nu}} =& \boldsymbol{R}^\top(\psi)\dot{\boldsymbol{\eta}}_{f,2} \end{aligned} \tag{4.45}$$

where $\boldsymbol{\eta}_m$ is the measured pose and $\boldsymbol{\eta}_{f,i}$ the ith filtered pose. If measurements is lost

$$\begin{aligned} \dot{\hat{\boldsymbol{\eta}}} &= \boldsymbol{R}(\psi)\hat{\boldsymbol{\nu}} \\ \dot{\hat{\boldsymbol{\nu}}} &= [0,0,0] \end{aligned} \tag{4.46}$$

The measurement was not lost in the experiments, but signal freeze (see Section 4.3.1) occurred. It was therefore checked if there was a signal freeze by comparing $\boldsymbol{\eta}_n$ and $\boldsymbol{\eta}_{n-1}$, and if so, (4.46) was used to update the estimates.

In addition there was made a continuous conversion of the ψ measurement to continuous $\hat{\psi}$ measurements. This was don by checking if there was a jump in the ψ measurement, then the $\hat{\psi}$ was either added or subtracted 2π.

Algorithm 3 pseudocode for Data Window choosing algorithm

```
input ← ψ_m
persistent Rounds
if jumpIn(ψ)==upwards then
    Rounds = Rounds + 2π
else if jumpIn(ψ)==downwards then
    Rounds = Rounds − 2π
end if
ψ_o = Rounds + ψ_m
output ← ψ_o
```

4.3.3 Nonlinear Observer

The kinematic model used for the nonlinear observer is

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{\eta}} \\ \dot{\boldsymbol{\nu}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}(\psi)\boldsymbol{\nu} \\ 0 \end{bmatrix} \tag{4.47}$$

The nonlinear observer is then

$$\dot{\hat{\boldsymbol{x}}} = \begin{bmatrix} \dot{\hat{\boldsymbol{\eta}}} \\ \dot{\hat{\boldsymbol{\nu}}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}(\psi)\hat{\boldsymbol{\nu}} \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{K}_1 \\ \boldsymbol{K}_2\boldsymbol{R}^\top(\psi) \end{bmatrix} (\boldsymbol{\eta} - \hat{\boldsymbol{\eta}}) \tag{4.48}$$

if loss of measurements

$$\dot{\hat{\boldsymbol{x}}} = \begin{bmatrix} \dot{\hat{\boldsymbol{\eta}}} \\ \dot{\hat{\boldsymbol{\nu}}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}(\psi)\hat{\boldsymbol{\nu}} \\ 0 \end{bmatrix} \tag{4.49}$$

4.3.4 Luenberger Observer

The kinematic equation for CSE1 is

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{\eta}} \\ \dot{\boldsymbol{\nu}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}(\psi)\boldsymbol{\nu} \\ \boldsymbol{M}^{*-1}(-\boldsymbol{C}^*(\boldsymbol{\nu}) - \boldsymbol{D}^*(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{\tau}^* + \boldsymbol{\omega}^*(t)) \end{bmatrix} \tag{4.50}$$

And is used when creating the Luenberg filter

$$\dot{\hat{\boldsymbol{x}}} = \begin{bmatrix} \dot{\hat{\boldsymbol{\eta}}} \\ \dot{\hat{\boldsymbol{\nu}}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}(\psi)\hat{\boldsymbol{\nu}} \\ \boldsymbol{M}^{*-1}(-\boldsymbol{C}^*(\hat{\boldsymbol{\nu}})\boldsymbol{\nu} - \boldsymbol{D}^*(\hat{\boldsymbol{\nu}})\hat{\boldsymbol{\nu}} + \boldsymbol{\tau}^* + \boldsymbol{\omega}^*(t)) \end{bmatrix} + \begin{bmatrix} \boldsymbol{K}_1 \\ \boldsymbol{K}_2\boldsymbol{R}^\top(\psi) \end{bmatrix} (\boldsymbol{\eta} - \hat{\boldsymbol{\eta}}) \tag{4.51}$$

with loss of measurements

$$\dot{\hat{\boldsymbol{x}}} = \begin{bmatrix} \dot{\hat{\boldsymbol{\eta}}} \\ \dot{\hat{\boldsymbol{\nu}}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}(\psi)\hat{\boldsymbol{\nu}} \\ \boldsymbol{M}^{*-1}(-\boldsymbol{C}^*(\hat{\boldsymbol{\nu}})\boldsymbol{\nu} - \boldsymbol{D}^*(\hat{\boldsymbol{\nu}})\hat{\boldsymbol{\nu}} + \boldsymbol{\tau}^* + \boldsymbol{\omega}^*(t)) \end{bmatrix} \tag{4.52}$$

where the $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ will be found through tuning.

4.4 Performance Metrics

To judge the performance of the different controllers, performance metrics can help. A good controller needs to satisfy several features. The most important, is the control objective

$$\lim_{t\to\infty} \boldsymbol{\eta}(t) = \boldsymbol{\eta}_t(t), \tag{4.53}$$

which is to bring the ship to a given target. The $\boldsymbol{\eta}_t$ is the desired target for the ship, and this can be a stationary or moving target. How close the controlled object is to the target can be measured by the position error

$$e_{pos} = ||\boldsymbol{\eta_t} - \boldsymbol{\eta}||_{pos} = \sqrt{(x_t - x_n)^2 + (y_t - y_n)^2} \tag{4.54}$$

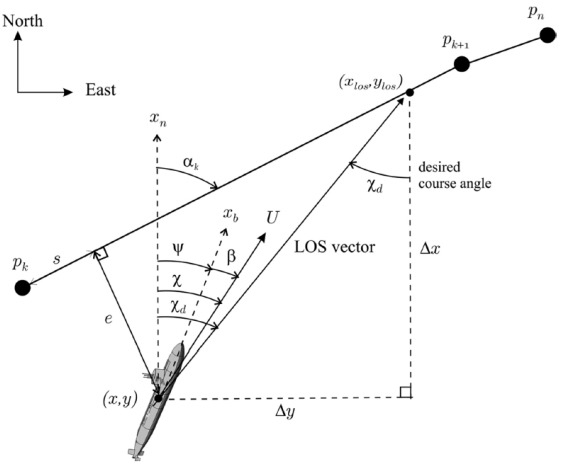
The error can also be measured by the cross-track error, which is the projection of the distance from $\boldsymbol{\eta}_t - \boldsymbol{\eta}$ on the orthogonal line of the path

$$\boldsymbol{\Psi}_t^{\perp} = [\cos(\psi_t), \sin(\psi_t)), 0]^{\top} \times [0, 0, -1]^{\top} \tag{4.55}$$

$$\tilde{\boldsymbol{p}} = [x_n, y_n]^{\top} - [x_t, y_t]^{\top} \tag{4.56}$$

$$e = \tilde{\boldsymbol{p}}^{\top} \boldsymbol{\Psi}_t^{\perp} \tag{4.57}$$

Figure 4.1 Guidance for a ship on a path from p_k to p_{k+1} from (Fossen, 2011), here $\boldsymbol{\eta}_t$ would be a parametrization from p_k to p_{k+1}, as explained in Section 4.2. The cross track error can be seen her as $\boldsymbol{e}$ and the angle ψ_t is α_k in this figure.



where ψ_t is the angle of η_t. The latter error has been used for the plots and metrics in this project. To measure the total error, the square-error is integrated over time which gives the Integrated Square Error (ISE)

$$ISE_{pos} = \int_0^{t_{end}} e(t)^2 dt. \tag{4.58}$$

The square error combines giving a huge penalty for being far off track, and neglecting the error when it is sufficiently small. In simulations an error can easily come down to $e = 10^{-5}$ or even smaller, but in real life, for a ship, the difference between a controller that has an error of $e = 10^{-5}$ or $e = 10^{-6}$ in simulations, will be insignificant compared other disturbances and uncertainties.

The controllers can also be differentiated by their energy use (Sørensen and Breivik, 2015). This is included in the metric Integral Absolute Error and Work (IAEW) where the energy is multiplied with the absolute error, thus giving a metric on how well the controller tracks the target and how energy efficient it is. Absolute error is chosen so the difference in tracking does not dominate the metric too much.

$$IAEW = \int_0^{t_{end}} |e(t)|dt \int_0^{t_{end}} P(t)dt \tag{4.59}$$

where $P = |\boldsymbol{\tau}^\top \boldsymbol{\nu}|$. Another property of the controller is how smoothly it is working, in essence how fast $\boldsymbol{\tau}$ is changing, thus $\dot{\boldsymbol{\tau}}$. The smoother the controller output is, the more realistic it is for the thrusters to produce the desired force. It also leads to less wear for the actuators and thrusters. Multiplying all these effects together gives the metric Integrated Absolute Error with Work and Wear and Tear. (IAEWWT)

$$IAEWWT = \int_0^{t_{end}} |e(t)|dt \int_0^{t_{end}} P(t)dt \int_0^{t_{end}} ||\dot{\boldsymbol{\tau}}||_2 dt \tag{4.60}$$

where $||\cdot||_2$ is the Euclidean norm. It can be difficult to extract the actual energy use, and wear and tear from these metrics. An increase in the metrics IAEW and IAEWWT can be due to increased IAE, W or $I\dot{\boldsymbol{\tau}}$. To see how the energy use and $\dot{\boldsymbol{\tau}}$ was for the different controllers, these were also calculated.

$$W = \int_0^{t_{end}} P(t)dt \tag{4.61}$$

$$I\dot{\tau} = \int_0^{t_{end}} ||\dot{\boldsymbol{\tau}}||_2 dt \tag{4.62}$$

$$IWWT = \int_0^{t_{end}} Pdt \int_0^{t_{end}} ||\dot{\boldsymbol{\tau}}||_2 dt \tag{4.63}$$

Chapter 5

Simulations

The simulations has several useful tasks. They were used to find interesting trajectories for the experiment, and verify the motion control system and see that the different controllers worked as intended. In addition it is a good start for coarse tuning of the adaptation and controllers before the experiments.

The simulations were also made for a comparative analysis of the controllers presented in Chapter 4. To compare them, the ship was simulated against the same trajectories as they encountered in the Mc-lab. There were non external disturbances imposed on the system, the only uncertainties where the rotational hydro dynamics as explained in Section **??**, which includes the uncertainty model (2.27). The process plant of the simulations were (2.1) and (2.2), and the controllers are summarized in Section 4.1.6.

5.1 Simulation Plots

In this section, the results of all the different simulations are presented in plots. For each scenario there are 5 plots:

- Plot 1(a) shows the trajectory of the controllers in the given scenario
- Plot 1(b) shows the offset angle and cross track error explained in Section 4.4
- Plot 2(a) shows the control metric also explained in Section 4.4
- Plot 2(b) shows the output forces from the controllers, linear, nonlinear and total force.
- Plot 3 shows the controllers body fixed estimate for the model-error ω together with the controllers estimate of this error $\hat{\omega}$

In Plot 2(b), the linear forces are the force from the feedback part of the controller, while the nonlinear part is the part that tries to zero out estimated and known nonlinear dynamics.

The total force is then these two forces added together. For the ABS controller this is

$$\begin{aligned} \boldsymbol{\tau}_{Linear} &= -\boldsymbol{z_1} - \boldsymbol{K_d}\boldsymbol{z_2} + \boldsymbol{M}^*\dot{\boldsymbol{\alpha}} \\ \boldsymbol{\tau}_{Nonlinear} &= \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) + \boldsymbol{\Phi}(\boldsymbol{\nu})\hat{\boldsymbol{\varphi}} - \boldsymbol{R}^\top(\psi)\hat{\boldsymbol{\omega}}_n \\ \boldsymbol{\tau}_{total} &= \boldsymbol{\tau}_{Linear} + \boldsymbol{\tau}_{Nonlinear} \end{aligned} \tag{5.1}$$

5.2 Tuning Parameters

The tuning parameters used for the simulations can be viewed in Table 5.1. The control parameters where found through trial and error, based on the control and adaptation parameters values in (Skjetne et al., 2004), with the condition that $||\boldsymbol{\tau}|| < 3$, since this was the limit of CSE1.

Table 5.1: The table shows the control parameters for the simulations.

Control Parameters	
$\boldsymbol{K_p}$	$diag([0.4,\ 1,\ 0.2])$
$\boldsymbol{K_d}$	$diag([1,\ 5,\ 1]$
$\boldsymbol{\Gamma}_\varphi$	$diag([8,\ 4,\ 8,\ 8,\ 8,\ 4,\ 8,\ 8])$

5.3 Comparison of Concurrent Learning Controllers

The first set of plots is a comparison of the four different version of the CL controller presented in Table 4.1.6. And their differences are presented in Section 4.1.5

- **CL-SVD-ep:** CL with SVD algorithm, using $\boldsymbol{\epsilon}$ as instantaneous adaptation error
- **CL-SVD-z2:** CL with SVD algorithm, using $\boldsymbol{z_2}$ as instantaneous adaptation error
- **CL-WIN-ep:** CL with WIN algorithm, using $\boldsymbol{\epsilon}$ the instantaneous adaptation error
- **CL-WIN-z2:** CL with WIN algorithm, using $\boldsymbol{z_2}$ the instantaneous adaptation error

they have been given the colours for the plots, and are as follows:

- **CL-SVD-ep:** is plotted with a blue line
- **CL-SVD-z2:** is plotted with a green dashed line
- **CL-WIN-ep:** is plotted with red dashed line
- **CL-WIN-z2:** is plotted with dashed and dotted line in magenta

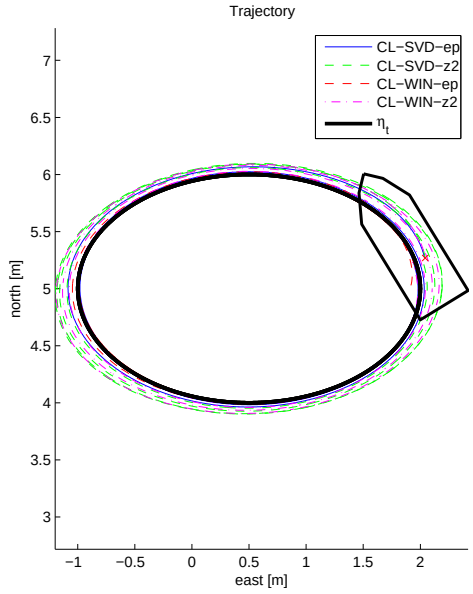
5.3.1 Elliptic Trajectory

The CSE1 model was simulated against the elliptic trajectory, explained in Section 4.2, and was set to $u_t = 0.8m/s$. The start pose was

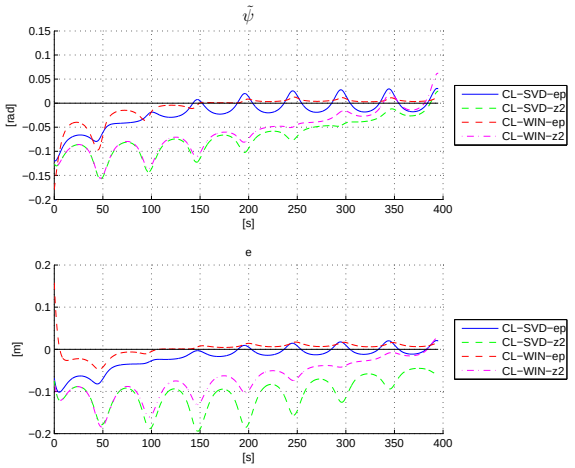
$$\begin{aligned} \boldsymbol{\eta_0} &= [5,\ 2,\ 0]^\top \\ \boldsymbol{\eta_{t0}} &= [5,\ 2,\ 0]^\top \end{aligned} \tag{5.2}$$

and the controllers where tested for two rounds. We see that all the controllers are able to track the elliptic trajectory, and are able to adapt to, and estimate the main part of the model-error, as can be seen in Figure 5.1 and 5.2. The CL controllers using $\boldsymbol{\epsilon}$ have the best preformance, were CL-WIN-ep seems to be a bit better, although it looks like it has a bias in its cross track error, while the CL-SVD-ep's cross track error is centred around zero, As can be seen in Figure 5.1b. The CL controllers also have the fastest estimation of $\boldsymbol{\omega}$, which can be seen in Figure 5.3. It would be interesting for the reader to examine this scenario, as it shows great evidence that better model-error adaptation, leads to better performance. By inspection, it can be seen how the estimation of $\boldsymbol{\omega}$ affects the τ_Nonlinear, and also reduces the τ_Linear, hence the need for feedback. Also by comparing these to $\tilde{\psi}$ and e, in Figure 5.1b. This strengthen the notion of that reducing the model error, improves the performance of the controllers. The controllers using $\mathbf{z_2}$ have substantial weaker performance, especially CL-SVD-z2.

Figure 5.1 The Figure shows the simulation of the CL controllers following an elliptic trajectory, explained in Section 5.3.1. In Figure (a) the trajectory of the different controllers and the target position $\boldsymbol{\eta}_t$ are seen, in Figure(b) the error in yaw can be seen, together with the cross track error, explained in Section 4.4

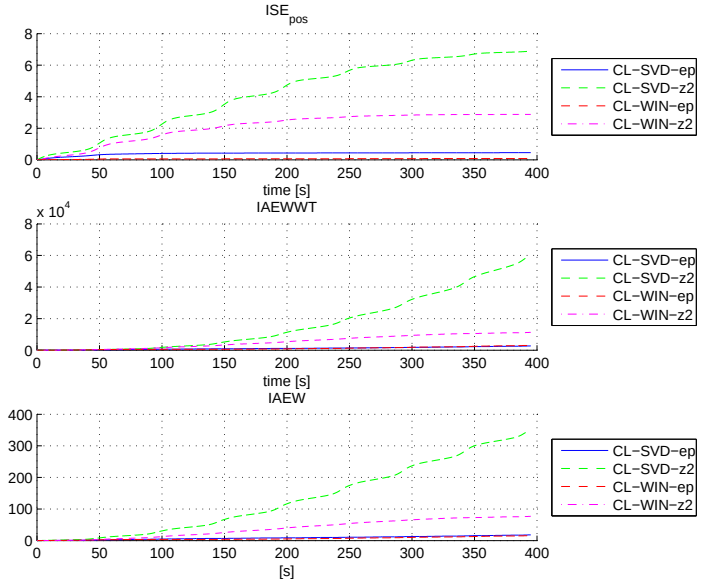


(a)

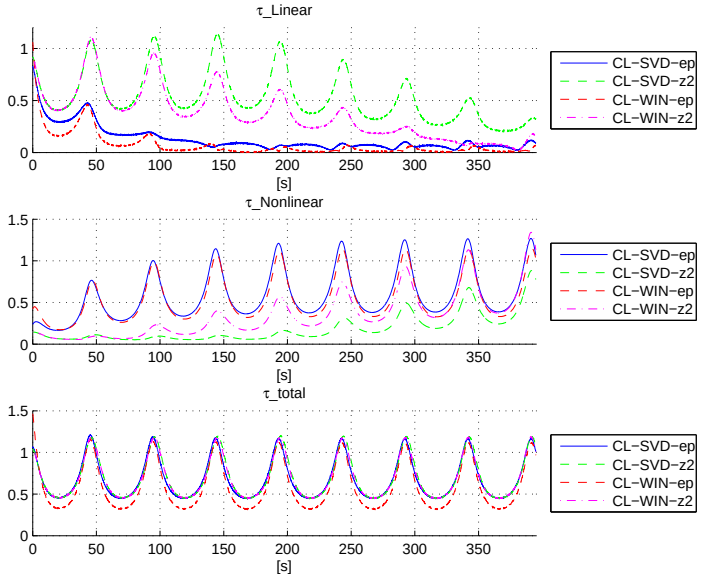


(b)

Figure 5.2 This figure shows the simulation results of the CL controllers following an elliptic trajectory, explained in Section 5.3.1. In Figure (a) the performance metrics, described in Section 4.4 can be seen. In Figure (b) the plots of the normed forces from the controller are shown, they are explained in Section 5.1

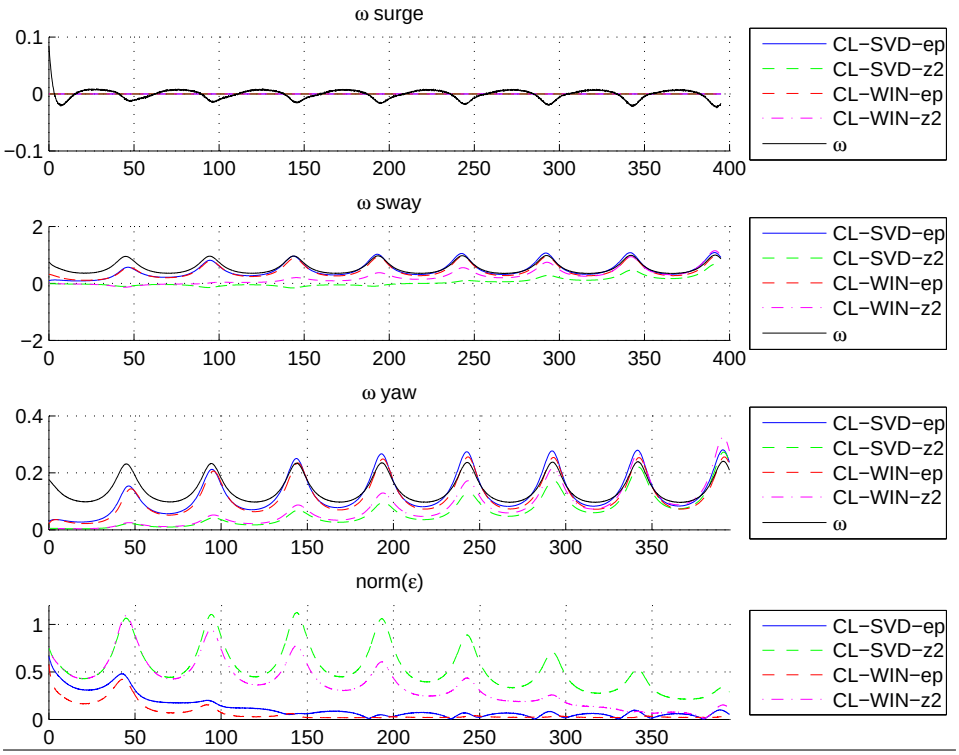


(a)



(b)

Figure 5.3 This figure shows the simulation results of the CL controllers following an elliptic trajectory, explained in Section 5.3.1. The model-error $\boldsymbol{\omega}$ is shown as the black line, and the lines of the controllers show their estimate of $\hat{\boldsymbol{\omega}}$ using (2.35).



5.3.2 Figure-Eight Trajectory

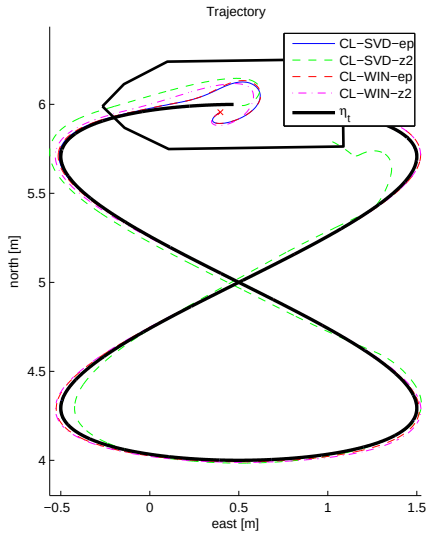
The CSE1 model was simulated against the figure-eight trajectory, explained in Section 4.2, and $u_t = 0.16$, witch equals $||\dot{\eta}_t|| \in [0.015, 0.05]$. The start pose was

$$\begin{aligned} \boldsymbol{\eta_0} &= [6,\ 0,\ 0]^\top \\ \boldsymbol{\eta_{t0}} &= [6,\ 0,\ -\pi]^\top \end{aligned} \tag{5.3}$$

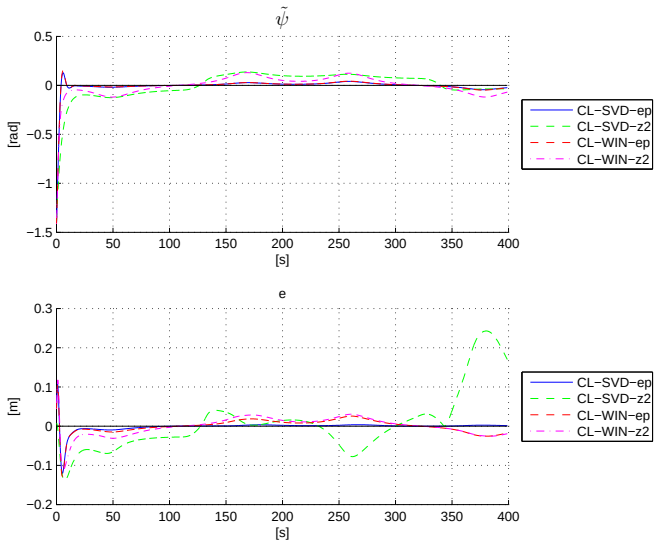
and the controllers where tested for a whole round. The transient start comes from the ship starting with their heading pointed north, which can be confirmed be examining the $\tilde{\psi}$ in Figure 5.4b.

We see that all the controllers seem to track the figure-eight the whole round, the only controller getting trouble is the CL-SVD-z2 which at the end gets off the trajectory. By looking at both the τ_Nonlinear in of Figure 5.5b and Figure 5.6 we see that the estimates has drifted of a bit in sway, causing bad control output. By examining the τ_Nonlinear this could be oscillatory behaviour, which would cause the control to collapse. Looking at the other controllers, and Figure 5.5a, we see that CL-SVD-ep has the best tracking, followed by CL-WIN-ep. They are also preforming best in regards of the IAEWWT and IAEW metrics. By looking at the $\boldsymbol{\omega}$ estimates in Figure 5.3 we see how this can be confirmed by looking at how they adapt to the model-error. In the CL-WIN-z2 has problems with the $\boldsymbol{\omega}$ yaw, which is peculiar, since it adapts right in the elliptic trajectory, but here it seems to estimate the opposite, which affects it performance, as can be seen in the $\tilde{\psi}$ in 5.4b. To summarize,CL-SVD-ep and CL-WIN-ep comes best out of this scenario, while CL-SVD-z2 show some troubling behaviour, and CL-WIN-z2 adapts wrongly in yaw.

Figure 5.4 The Figure shows the simulation of the CL controllers following an figure-eight trajectory, explained in Section 5.3.2. In subfigure (a) the trajectory of the different controllers and the target position $\boldsymbol{\eta}_t$, in subfigure(b) the error in yaw can be seen, together with the cross track error, explained in Section 4.4

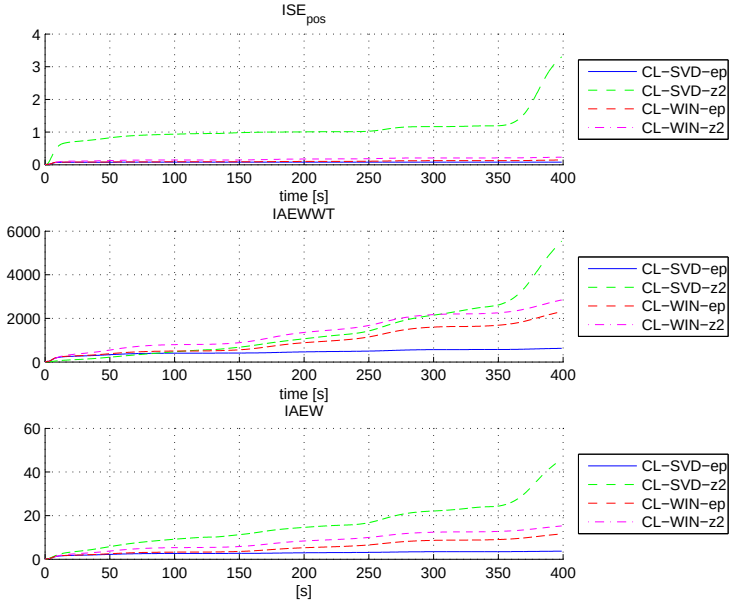


(a)

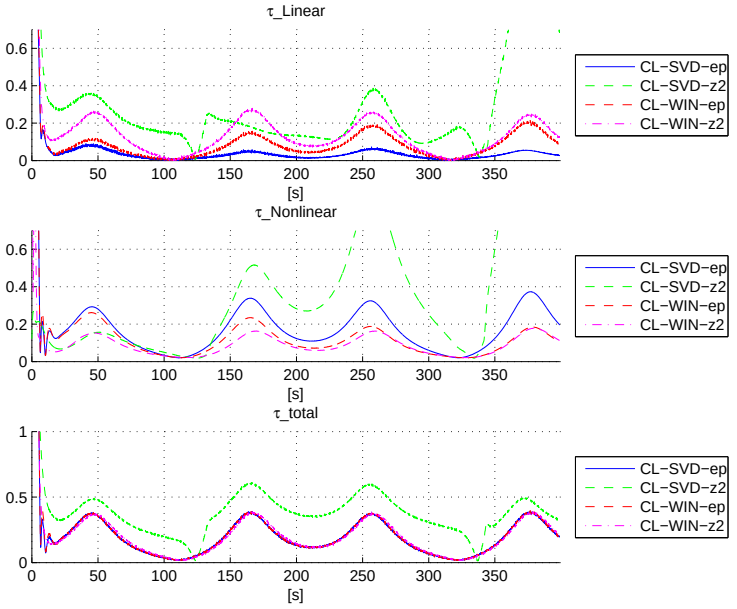


(b)

Figure 5.5 This figure shows the simulation results of the CL controllers following an figure-eight trajectory, explained in Section 5.3.2. In Figure (a) the performance metrics, described in Section 4.4 can be seen. In Figure (b) the plots of the normed forces from the controller are shown, they are explained in Section 5.1



(a)



(b)

Figure 5.6 This figure shows the simulation results of the CL controllers following an figure-eight trajectory, explained in Section 5.3.2. The model-error ω is shown as the black line, and the lines of the controllers show their estimate of $\hat{\omega}$ using (2.35).

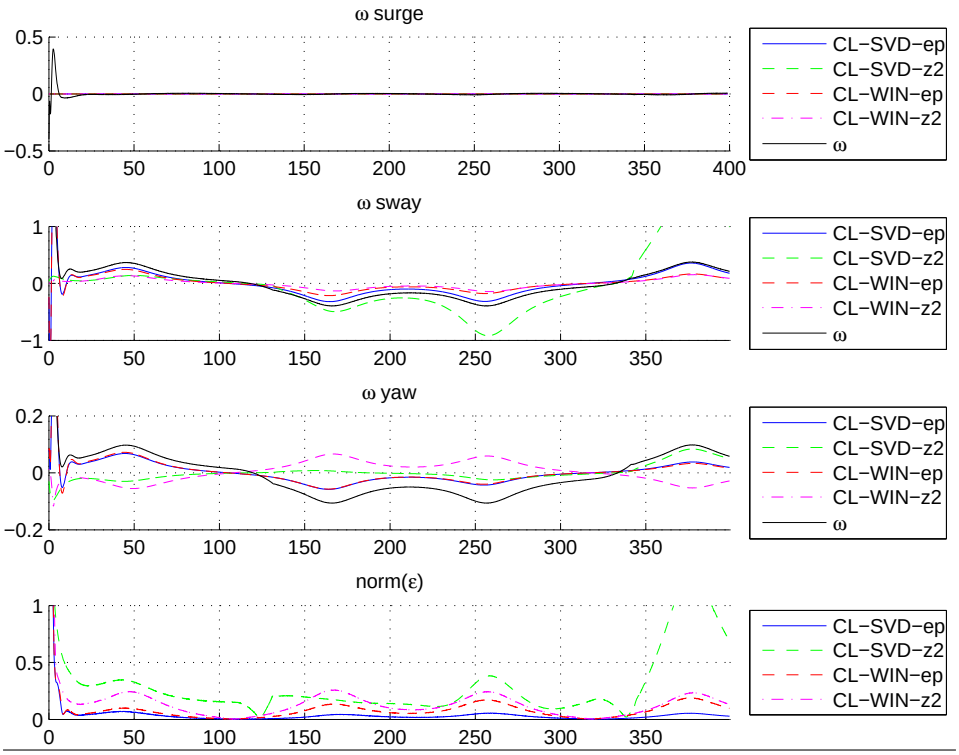
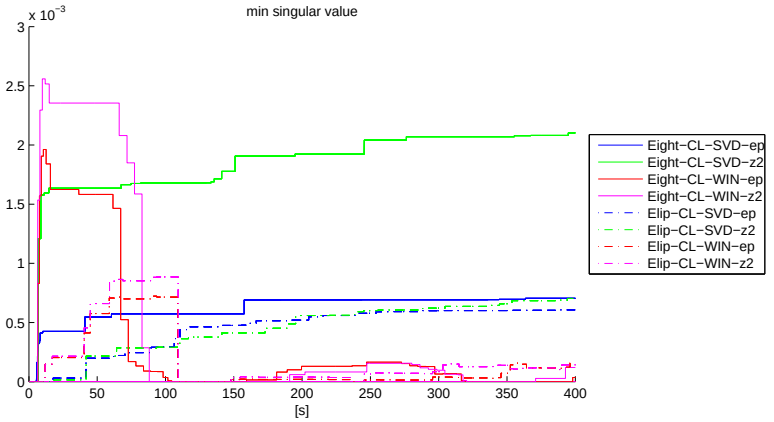


Figure 5.7 The singular values for the runs can be seen, the line plots are for the simulations using figure-eight trajectory, while the dot dashed are for the elliptic trajectory. The red and magenta plots are from the WIN storage algorithm, while the blue and green are for the SVD algorithm.



5.3.3 summary

When it comes to the convergence of the hydrodynamic parameter, it was experienced that only a few parameters converged to the correct value. By examining the min singular values of the different scenarios in Figure 5.7, it is clear that these were too small to guarantee any fast convergence, although they were the best found during trial, hence they were used for the experiments. We also witness that the figure-eight trajectories produces richer data, and that the SVD algorithm worked as intended. Although this would also be its bane, because during the experiments, spikes occurred, and the SVD algorithm made the CL controller store all the spikes, leading to instability. This was a problem for the WIN algorithm as well, although it replaced its data continuously, which resulted in more robust behaviour.

After comparing the simulations of the CL controllers, some factors were apparent. The CL-SVD-z2 is a bad combination and produced the worst and least robust control. The CL controllers using ϵ had the by far best adaptation, the CL-WIN-ep proved to have the best performance in regular elliptic scenarios, while CL-SVD-ep was best against more irregular model-error in the figure-eight trajectory. What was not tested in these comparisons, was how robust the controllers were against noisy measurements and spikes. Intuitively CL-WIN-z2 would be the most robust controller in this fashion, since it relies on z_2 which is a cleaner signal than ϵ, and the Window storage algorithm replaces its stored data, so its less sensitive to spikes. The pros and cons of the different CL controllers are summarized in Table 5.2. Also it would be more interesting to have one CL controller using ϵ and one CL controller using z_2. It was therefore decided to test CL-SVD-ep and CL-WIN-z2 in a comparative analysis against BS and ABS controllers.

Table 5.2: Summary of the pros and cons with the CL controller implementations

	Window		Singular Value Maximation	
ϵ	CL-WIN-ep:		CL-SVD-ep	
	Pros: Best Adaptation	Cons: Dificult to implement, because of ϵ	Pros: Best Adaptation, smoothest output	Cons: Dificult to implement, because of ϵ SVD is fragile to error in ϵ
z_2	CL-WIN-z2		CL-SVD-z2	
	Pros: Easiest to implement in a robust fassion	Cons: Worse adaptation than the CL controllers using ϵ	Pros: More realistic implementation, theoretically faster convergance than CL-WIN-z2	Cons: Badest control and adaptation SVD is fragile to error in ϵ

5.4 Comparison of Concurrent Learning and Adaptive Backstepping

To see if the CL controllers have the promising convergence as stated, they have to be compared to what already is the state-of-the-art adaptive controllers. The two controllers to be tested against the CL controllers are the adaptive backstepping (ABS) controller , and the regular backstepping (RBS) controller. This way the CL controllers can be compared to both a regular and an adaptive controller. In addition, by comparing the performance of the ABS stepping controller to the RBS controller, the benefits of having adaptation can be confirmed. It can also grant a test as insignificant if there is no significant performance difference between the RBS, and the ABS, thus rejecting that good adaptation is essential for improved performance. The controllers that are compared in this section are

- **BS-ADAP:** Adaptive backstepping controller also called ABS
- **BS-NORMAL:** Normal backstepping controller also called RBS or just BS
- **CL-SVD-ep:** CL with SVD algorithm, using $\boldsymbol{\epsilon}$ the instantaneous adaptation error
- **CL-WIN-z2:** CL with WIN algorithm, using $\boldsymbol{z_2}$ the instantaneous adaptation error

These are also summarized in section 4.1.6, hey have been given the colours for the plots, and are as follows:

- **BS-ADAP:** is plotted with a blue line
- **BS-NORMAL:** is plotted with a green dashed line
- **CL-SVD-ep:** is plotted with red dashed line
- **CL-WIN-z2:** is plotted with dashed and dotted line in magenta

and the tuning parameters are the same as in Table 5.1.

5.4.1 Elliptic Trajectory

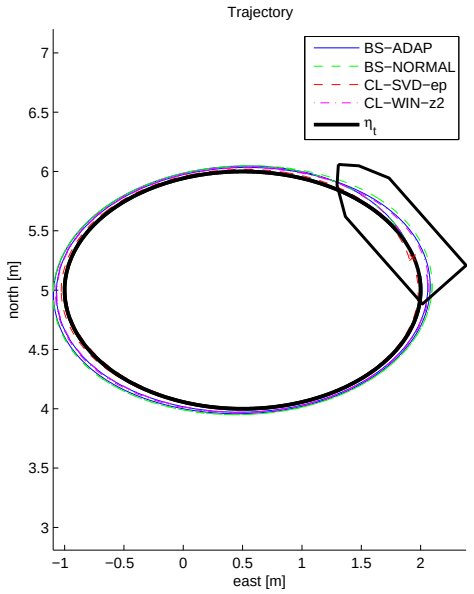
The CSE1 model was simulated against the elliptic trajectory, explained in Section 4.2, and was set to $u_t = 0.8m/s$. The start pose was

$$\begin{aligned} \boldsymbol{\eta_0} &= [5,\ 2,\ 0]^\top \\ \boldsymbol{\eta_{t0}} &= [5,\ 2,\ 0]^\top \end{aligned} \tag{5.4}$$

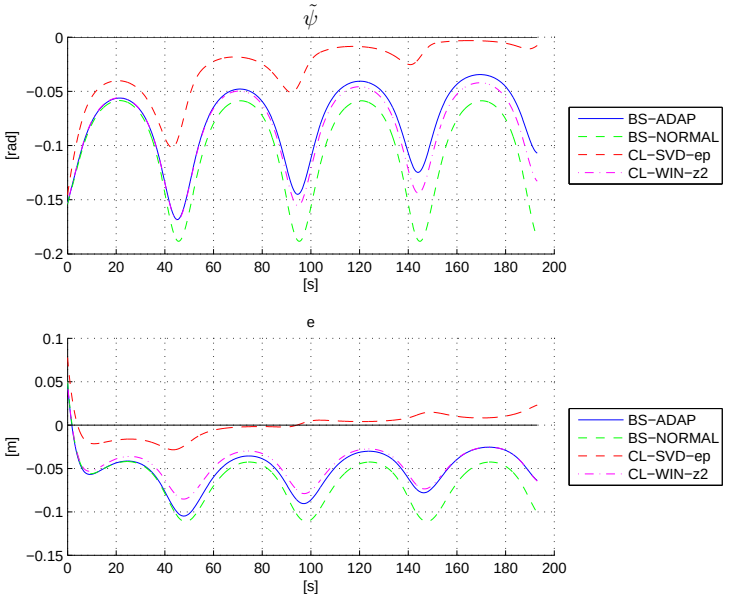
and the controllers where tested for two rounds. We see that all the controllers are able to track the elliptic trajectory, and are able to adapt to, and estimate most of the model-error, as can be seen in Figure 5.9 and 5.8. As expected the CL-SVD-ep has by far the best adaptation and tracking. As can be seen in Figure 5.8b, it has the fastest estimation of $\boldsymbol{\omega}$. This also leads to highest τ_Nonlinear, and smallest τ_Linear which means that it has to rely the least on feedback. The CL-WIN-z2 is also a satisfactory number two, as

it has a good early adaptation of ω. Although the difference to the ABS controller is not impressive. Also by examining the RBS controller it can be witnessed how the controllers would perform without adaptation.

Figure 5.8 The figure shows the simulation of the CL controllers together with the ABS and BS controller following an elliptic trajectory, explained in Section 5.4.1. In Figure (a) the trajectory of the different controllers and the target position $\boldsymbol{\eta}_t$ are seen, in Figure(b) the error in yaw can be seen, together with the cross track error, explained in Section 4.4

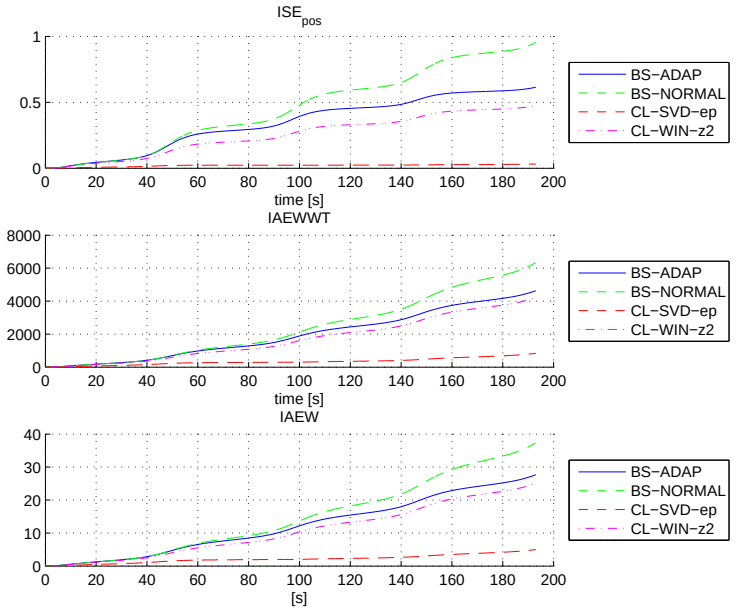


(a)

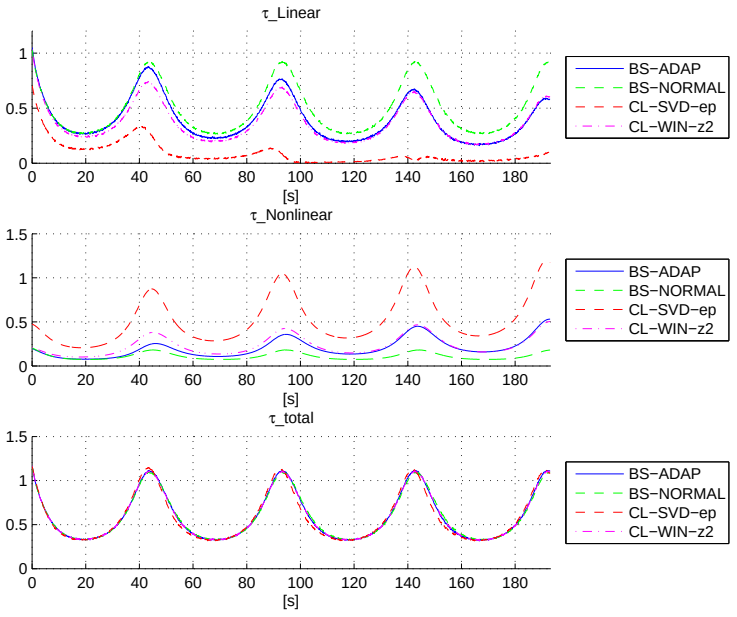


(b)

Figure 5.9 This figure shows the simulation results of the CL controllers following an elliptic trajectory, explained in Section 5.4.1. In Figure (a) the performance metrics, described in Section 4.4 can be seen. In Figure (b) the plots of the normed forces from the controller are shown, they are explained in Section 5.1

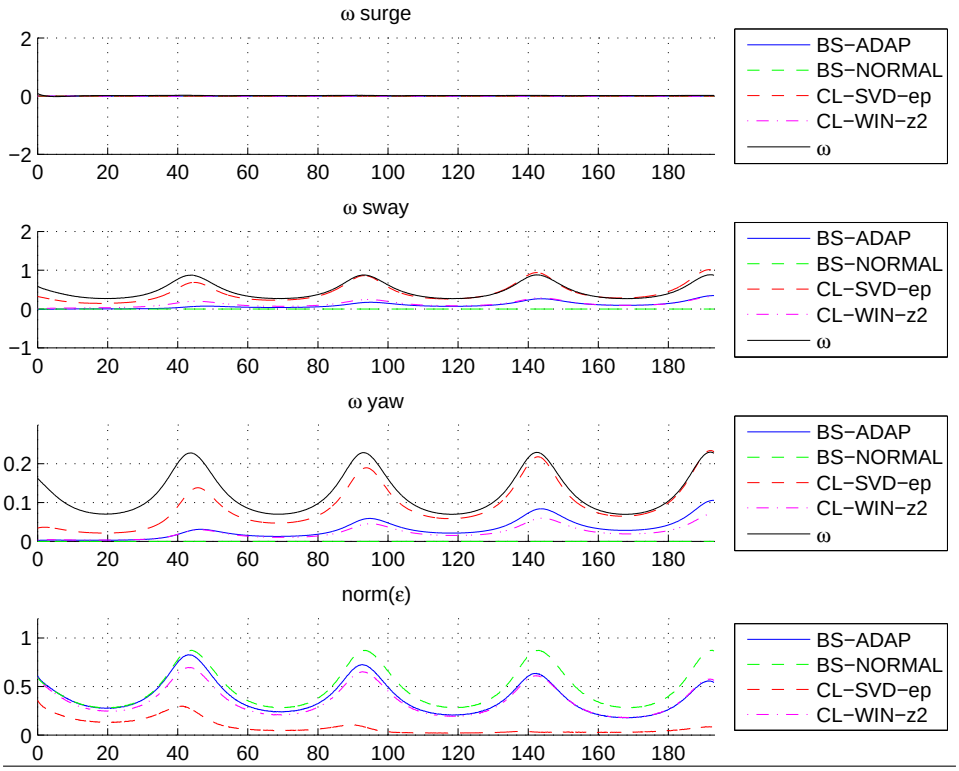


(a)



(b)

Figure 5.10 This figure shows the simulation results of the CL controllers following an elliptic trajectory, explained in Section 5.4.1. The model-error $\boldsymbol{\omega}$ is shown as the black line, and the lines of the controllers show their estimate of $\hat{\boldsymbol{\omega}}$ using (2.35).



5.4.2 Figure-Eight Trajectory

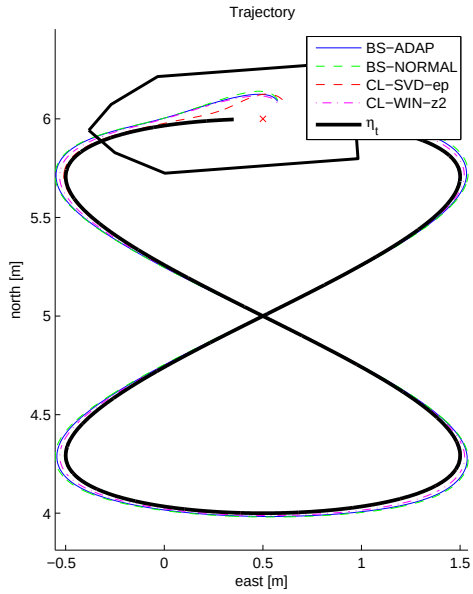
The CSE1 model was simulated against the figure-eight trajectory, explained in Section 4.2, and $u_t = 0.16$, witch equals $||\dot{\boldsymbol{\eta}}_t|| \in [0.012, 0.03]$. The start pose was

$$\begin{aligned} \boldsymbol{\eta_0} &= [6,\ 0,\ 0]^\top \\ \boldsymbol{\eta_{t0}} &= [6,\ 0,\ -\pi]^\top \end{aligned} \tag{5.5}$$

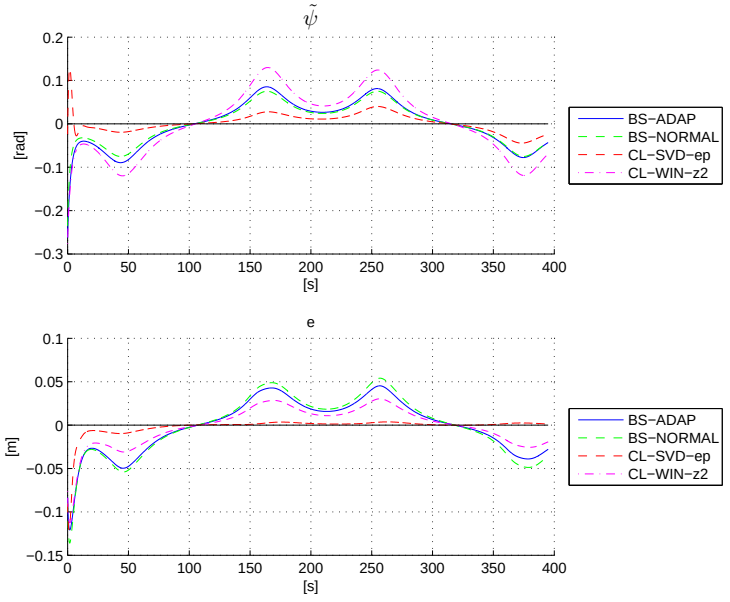
and the controllers where tested for a whole round. The transient start comes from the ship starting with their heading pointed north, which can be confirmed be examining the $\tilde{\psi}$ in Figure 5.11b.

We see that all the controllers seem to track the figure-eight trajectory the whole round. The CL clearly have the best tracking, where the CL-SVD-ep comes best out. We also see how the CL controllers are best at estimating $\boldsymbol{\omega}$, at least in sway. It is curious to see how CL-WIN-z2 adapts differently in $\boldsymbol{\omega}$ yaw. This happened to both CL controllers using z_2 as instantaneous adaptation error, but then again the bad adaptation does not happen to ABS. It is probably the data stored combined with a z_2 which is too small to help in the adaptation. We also see how the ABS works better than RBS, so adaptation does work in this setting. As CL-SVD-ep closely eliminates the model-error, it is interesting to see how the rotational forces affect the manoeuvring of the ship. In this setting the hydro forces gives an outward push on the boat, as can also be seen in the elliptic simulation which will not be the case for the real system, as can be seen in Chapter 6.

Figure 5.11 The figure shows the simulation of the CL controllers together with the ABS and BS controller following an elliptic trajectory, explained in Section 5.4.2. In Figure (a) the trajectory of the different controllers and the target position $\boldsymbol{\eta}_t$, in Figure(b) the error in yaw can be seen, together with the cross track error, explained in Section 4.4

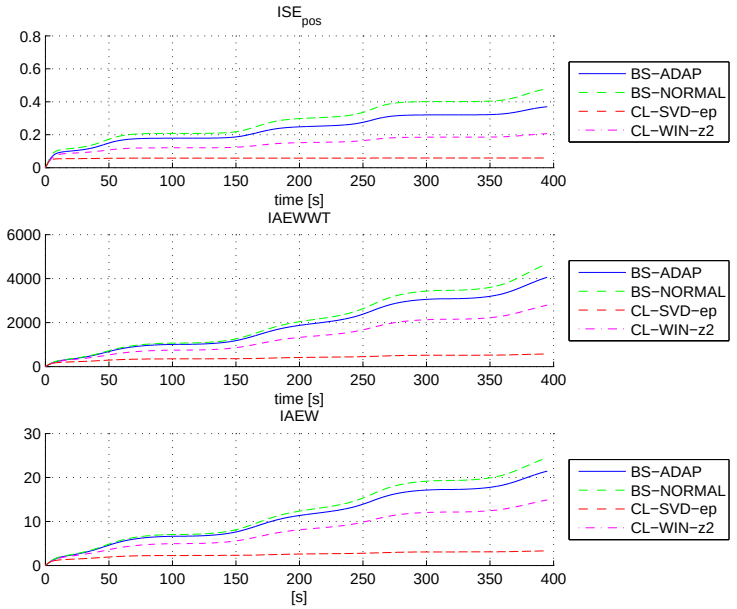


(a)

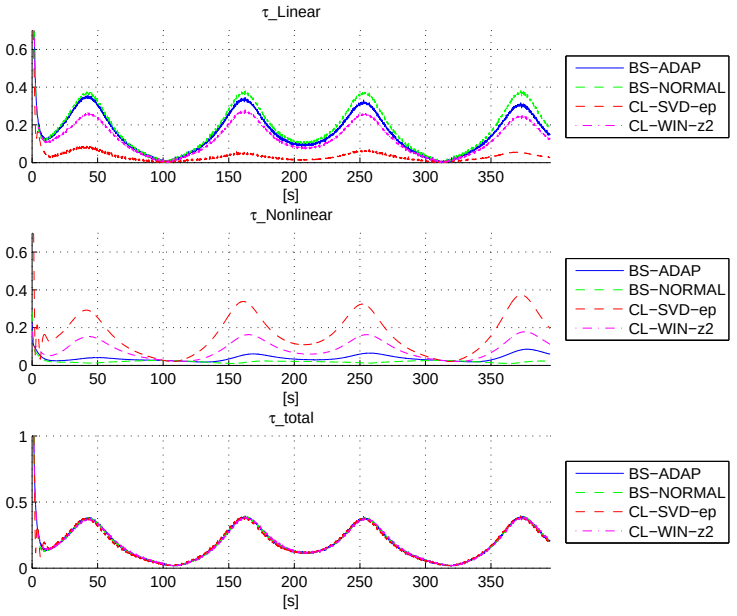


(b)

Figure 5.12 The figure shows the simulation of the CL controllers together with the ABS and BS controller following an figure-eight trajectory, explained in Section 5.4.2. In subfigure (a) the trajectory of the different controllers and the target position η_t, in subfigure (b) the error in yaw can be seen, together with the cross track error, explained in Section 4.4

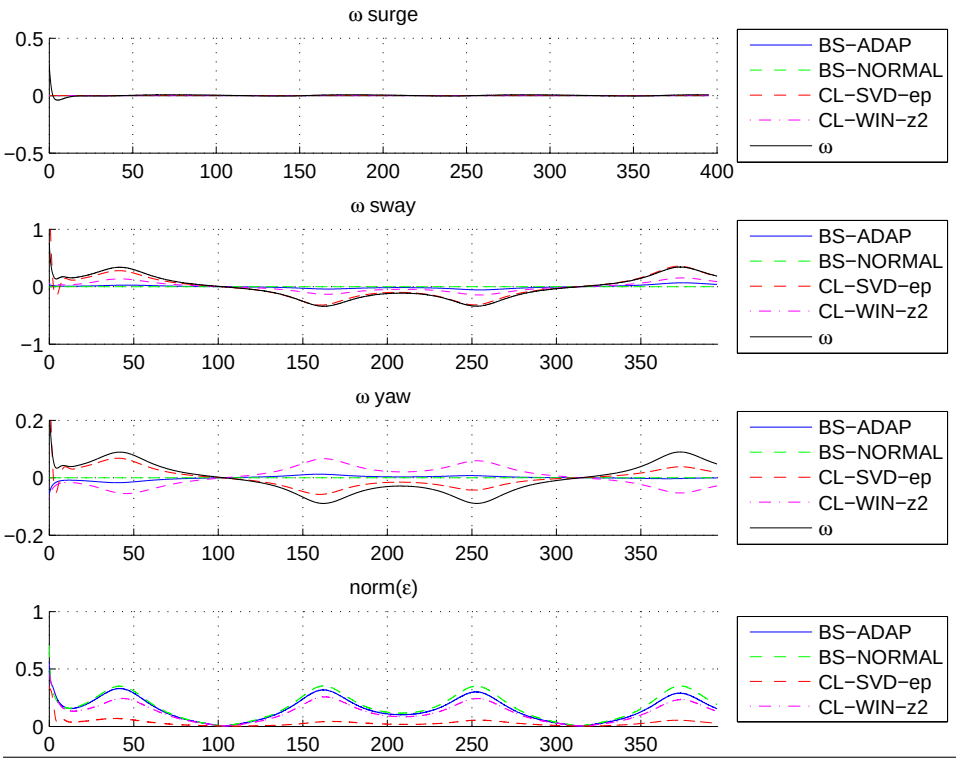


(a)



(b)

Figure 5.13 This figure shows the simulation results of the CL controllers following an figure-eight trajectory, explained in Section 5.4.2. The model-error $\boldsymbol{\omega}$ is shown as the black line, and the lines of the controllers show their estimate of $\hat{\boldsymbol{\omega}}$ using (2.35).



5.4.3 Summary

It can be seen from the simulations that a few conclusion can be made

- There is clearly a correlation between the controllers ability to adapt to the model-error ω, and its tracking performance
- The BS-ADAPT outperformed the BS-NORMAL adapter, leading to the conclusion that the trajectories seems valid for testing adaptive controllers.
- CL has good potential in achieving greater adaptation and thus control than the state-of-the-art adaptive controllers.
- Using an Eight figure trajectory captures a more variety of the hydrodynamic forces, and can therefore be a better tool for testing adaptive controllers, this was also confirmed in Figure 5.7
- In simulations the CL using ϵ had best adaptation and performance, although the robustness of these controllers was not impressive as will be demonstrated in Chapter 6.
- The elliptic and figure-eight trajectory seems like good indicators of the adaptation abilities of the, although it was not demonstrated that the parameters of the controllers converged to the correct value. And by examining the min singular values of the scenarios in Figure 5.7 these were too small to guarantee any fast convergence. But these were the best results which was realistic to preform in real life testing, so these were chosen for the CSE1 experiment.

Chapter 6

Experimental Results

To see if the promising result of CL could work for a real system, there was set up an experiment with CSE1 on the MC-lab, were the controllers designed in Chapter 4 were to be compared. In this chapter the experimental platform is presented in Section 6.1, the Section 6.1.3 presents the goal and set-up of the experiments. In Section 6.3, the chronological steps done as preparation for the experiment are gone through, and the result of the experiments are presented and discussed in Section 6.4. At the end of this chapter, a lessons learned section summarizes the results of the Experiments and what lessons can be taken from them, in Section 6.5.

6.1 Experimental Platform

The experiments were preformed on the MC-lab at Marine Technology Centre at NTNU, on CSE1. The description of the lab can be found in (Mc-lab and NTNU, 2015). The CSE1 can be seen in Figure 6.1

6.1.1 Vessel Body and Design

The CSE1 is model scaled ship designed as a supply vessel used in offshore operations with parameters seen in Table 6.1. Further details regarding the body and design of CSE1 can be found in (Skåtun, 2011).

Table 6.1: Weight and dimension parameters for CSEI from (Sandved, 2015)

Parameter	Symbol	Value
Mass	m	14.79 Kg
Length	L	1.105 m
Width	B	0.248
Scale	λ	1:50

Figure 6.1 Picture of CSE1 from (Sandved, 2015)

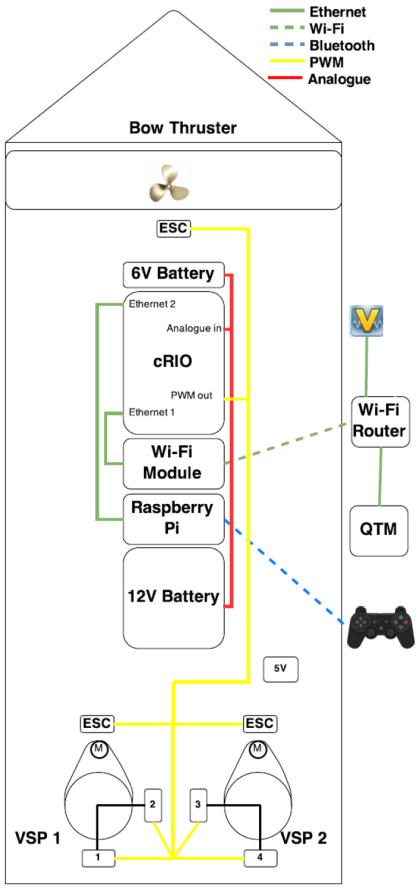
6.1.2 Sensors and Communication

The communication on CSE1 is done through Wi-Fi and bluetooth, Ethernet and PWM, as can be seen in Figure 6.2, which connects the CompactRIO (cRIO), thrusters and servos, Wi-Fi module, Respberry Pi, Qualisys motion capture system and the computer.

CompactRIO The cRIO is the controller on the ship. This means that it is the central computer, handling everything from sending PWM thruster commands, to acquiring measurements from Qualisys motion capture system, and taking commands from the remote Play Station 3 (Ps3) controller, and the Veristand program on the computer, In addition, the cRIO sent logged data to the computer through Wi-Fi during tests. The code is deployed from a computer to the cRIO through the National Instruments program Veristand. With the Veristand program the user can also monitor and send in commands and change parameters in the controller. How this is done is also described in (Mc-lab and NTNU, 2015) and (Sandved, 2015). The controller was designed in Simulink, and a hardware-in-the-loop (HIL) test was preformed, by running the control system against a process model on the cRIO. This was done before the controllers were tested on CSE1.

Qualisys motion capture system Qualisys motion capture system works by three cameras calculate taking pictures of the model ship. The set-up can be seen in Figure 6.3. By locating and triangulating the white spheres placed on CSE1, the pose of the ship is found. The spheres can be seen seen in Figure 6.1. The measurements are very accurate, but a problem came when the spheres occasionally were blocked, or blocked on another from one of the cameras, the system lost the track of the boat, and measurements froze until the ship was found again. These freezes occurred regularly on the same place when CSE1 followed the elliptic and figure-eight trajectory, and the freeze could happen for as long as a second.

Figure 6.2 Picture of the communication system on CSE1 (Mc-lab and NTNU, 2015)



Raspberry Pi and six axis control The Raspberry Pi communicates with a Ps3 controller, see Figure 6.4, and the cRIO, so that commands can be given from the controller to CSE1. For a more thorough explanation see (Mc-lab and NTNU, 2015). If the CSE1 project from (GitHub, 2016) is deployed to the cRIO, the ship can be controlled from the Ps3 controller. The figure buttons will switch between different modes of controlling the CSE1, and the joysticks will optionally give commands to the thrusters. The different modes are

- Cross: ctrl_student, this mode lets the students run a self built controller on CSE1 with the interfaces pose measurement $\boldsymbol{\eta}$ as input, and thruster commands $\boldsymbol{u}$ as output.

- Circle: ctrl_DP_basic, this mode get the model ship to navigate to a set point which can be moved with the joystick

Figure 6.3 Picture of the Qualisys motion capture system from (Mc-lab and NTNU, 2015)

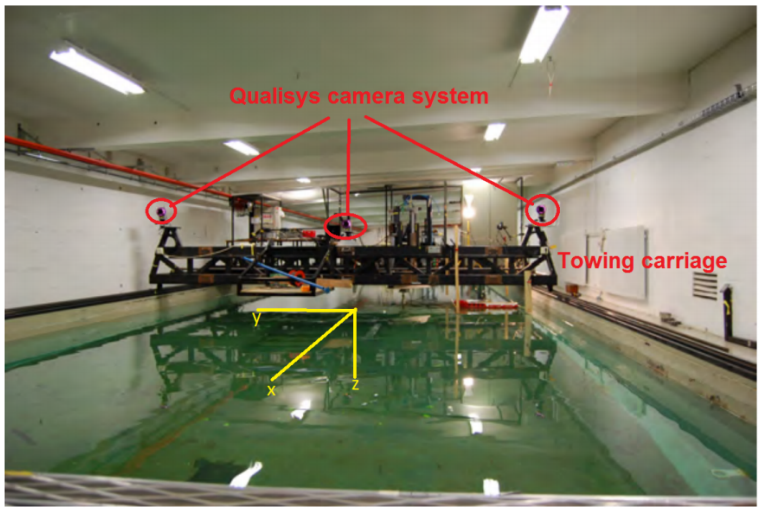


Figure 6.4 Picture of the controller used to send commands to CSE1

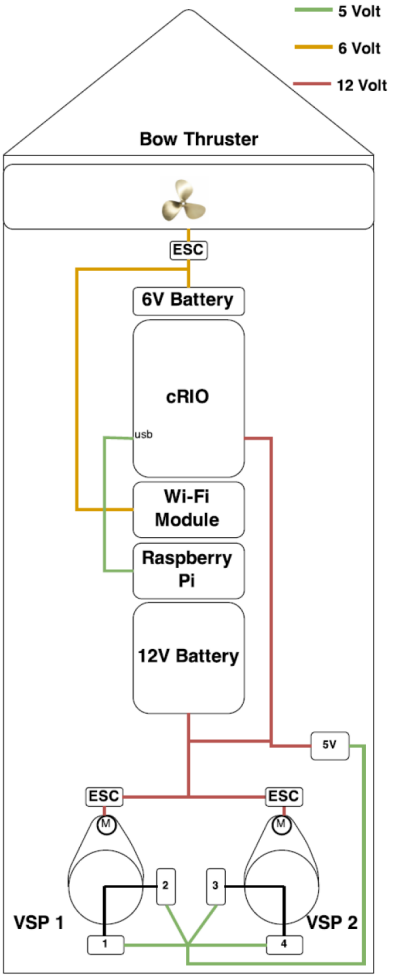
- Triangle: ctrl_sixaxis2thruster, this mode lets the model ship take commands directly from the joysticks to the thrusters

- Square: ctrl_sixaxis2force, this mode lets the model ship take commands as a desired force from the joystick.

6.1.3 Propulsion and Power System

The Propulsion system is composed of two Voith Schneider propellers and one bow thruster propeller, that are also described in Section 2.1.1. The VSP are actuated by two servos for each VSP, and are connected to the main 12V battery, while the servos use an adapter to run on 5V, as can be seen on Figure 6.5. The servos and motors are controlled with PWM signals. The BT is connected to its own 6V battery, and different from the figure, the Wi-Fi is connected to its own 6V battery, although the batteries were connected. A possible reason for the fall out of some of the data delivered through Wi-Fi, was the combination of low battery and a heavy working BT.

Figure 6.5 Picture of the power set-up for CSE1



6.2 Experimental Setup and Goals

The tests ran in th MC-lab had several purpose. One was to verify and compare different nonlinear adaptive controllers. Another was to present multivariate methods for identifying model-error, and improve the ship model. After discussing the CSE1 with students and stipendiates whom worked with the CSE1, and examining the system identification of CSE1 a few issues were raised towards the dynamic model og CSE1.

Problems with the current model:

- The parameters related to rotational hydro dynamics forces seemed uncertain, and was taken from a Cybership II experiment in (Skjetne et al., 2004),
- the thruster allocations had issues like the absence of superposition property for the thrust, and inaccurate thruster allocation.

To identify the problem with the rotational hydro dynamic forces, the CSE1 ran on an elliptic and figure-eight trajectory. The plan was to gather as many variables as possible during the test, to see if some structure in the model-error could be found using system identification (SysID), or MVA. In Chapter 3, the structuring of the data before a multivariate analysis is presented, and the post-experimental analysis can be seen in Chapter 7. For this experiments, the signals and states stored can be seen in Table 6.2 and 6.3.

Table 6.2: All the variables measured

	← Variables →											
	τ_u	τ_v	τ_r	x	y	ψ	u	v	r	$\dot{u}$	$\dot{v}$	$\dot{r}$
t	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
↓	·	·	·	·	·	·	·	·	·	·	·	·

The goal is too map these variables to a model-error. The model-error is defined as in section 2.2.4. The off-line estimate of the model-error is

$$\boldsymbol{\omega} = \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\tau} \tag{6.1}$$

And through analysis, the goal was to make a model of $\boldsymbol{\omega}$. How this was investigated can be seen in Chapter 7. It was also of interest to see if there could be any relationship between the $\boldsymbol{\omega}$ and the tracking performance, thus the tracking error z_1 was stored as in Table 6.3.

6.3 Experimental Preparations

Before experiments could be preformed, several steps were made. After the control system had been designed, the system was tested through simulations. The next step was to test the control system in a HIL-test, and at the end open and closed loop test were done on SCE1. In this section these procedures are given in chronological order.

Table 6.3: The internal signals logged, in addition estimates like ϕ and ω_n were logged, but they were not used for MVA.

	← Variables →					
	ω_u	ω_v	ω_v	z_{1u}	z_{1v}	z_{1r}
t	⋮	⋮	⋮	⋮	⋮	⋮
↓	·	·	·	·	·	·

6.3.1 Simulations

Testing how the control system designed worked, and the planing of trajectories were done through simulations. The different controllers were tested with different trajectories and observers, and with and without noise in the measurement. What could have been added in addition to make the simulation more realistic was the signal freeze and NaN measurements, so that this could have been taken to account earlier in the design of the estimators and controllers. The final simulations done after the basin experiments can be seen in Chapter 5. When planing trajectories, there were certain considerations that had to be made. The goal was to see how rotational hydro dynamic forces affected the control. Therefore, having trajectories where the rotational states were sufficiently rich, was essential to see if any model could be built up from the model-error. From Section 4.19 the convergence of the estimates were dependent on the the size of matrix $\mathbf{Q}$, and intuitively this increased with both larger values of v and r, and the spread of the stored data. This was tested by several trajectories, also by recording the $\lambda_{\min}$, and seeing how the different parameters converged, as can be seen in Figure 5.7. The resulting simulations and analysis of the convergence can be seen in Chapter 5. In multivariate analysis, the models are often easier to build up when more states are visited, especially when using the nominal techniques described in Section 3.2, it was therefore important to examine trajectories who gave more variate states. Hence the trajectories chosen at the end were an elliptic and figure-eight trajectory. The plots of the simulations are plotted in Section 5.

6.3.2 Hardware-In-The-Loop Tests

Before testing the control system on CSE1, the control system was tested against a process model of CSE1 on the cRIO that is presented in Section 6.1.2. This was to ensure that the signal flow was correct. This included that the coordinates chosen were right, that the signals went were they was supposed to and that non of the signals diverged. It was also important to see if cRIO was fast enough to run the control systems. During the HIL test, it was experienced how sensitive the controller was of reaching NaN signals. This was due to the implementation of the controller, that propagated NaN signals when they occurred. A solution was to use saturated signals, and checking for NaN states. In addition redundant memories were deleted. Doing the HIL test also gave great training on how the cRIO worked, how the designed controller had to be deployed and how data could be logged and monitored during a test so that the routines were better when the actual basin tests started.

6.3.3 Estimator Tests

Before being able to to test the controllers, we had to be sure that the estimators worked. This was ensured by first testing the estimators in the simulations, where both the nonlinear estimator, and the derivative filter were tested. Then the estimators were tested open loop on CSE1 while another controller worked on the ship, or the CSE1 was controlled by a Ps3 controller as explained in Section 6.1.2. The result of one of the tests can be seen in Figure 6.6c. The Derivative filter was lagging compared to the other controllers, but produced less noisy estimates. In addition, the derivation filter was more robust than the other observers at handling the jump in measurements happening after a signal freeze. The implementation of the nonlinear and lungenberg observer had estimates stored, so if a NaN estimate was produced due to high spikes, the observers were stuck in a NaN state. Because of time pressure the derivative filter seemed most robust, and was chosen as the observer. The derivative filter was also modified to handle signal freeze, as shown in (4.46). The estimator also produced continuous angle estimates, so that the measured jump in ψ did not occur.

6.3.4 Tuning Thruster Allocation

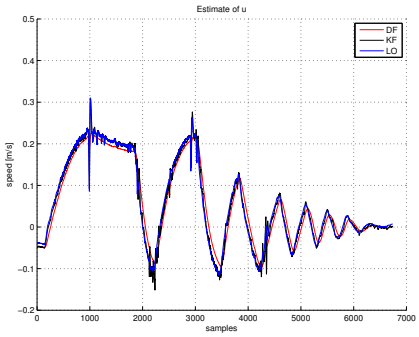
As explained at the end of Section 2.2, When requesting only forces from the thruster allocation, moments can also be induced by having the point of attack at a distance from center of moment CM. To minimize this, the point of attack can be changed in the thruster allocation, as can be seen in Figure 6.7. The result is that the thruster allocation will produce a moment that is equal to the moment induced on the ship, when there is a certain distance between the point of attack, and CM. The thruster allocation was tuned by requesting a force in sway motion, if the ship happened to rotate as the ship to the left in Figure 6.7, the point of attack was moved backwards on the boat. If the rotation was like the ship on the right in Figure 6.7, the point of attack was moved forward. This was done until the induced moments were minimized. This was at

$$\boldsymbol{P}_0 = [0.18,\ 0,\ 0]^\top \tag{6.2}$$

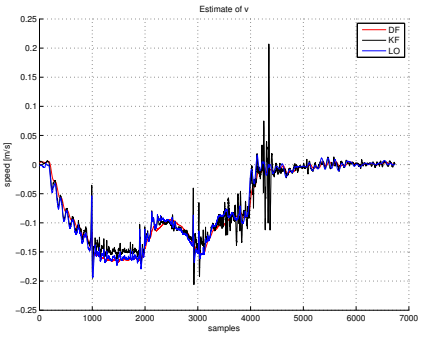
6.3.5 Basin Tests

When the estimator was chosen, and thruster allocation was tuned, the whole controller could be tested on CSE1. First the controllers were tested towards a set point, and changing of set point. Then the controller was tested in a zig-zag manoeuvre back and forth, and at the end, controllers were tested on the elliptic and figure-eight trajectory. The guidance was designed to give first a desired set point at the start of the trajectory to be run. When the CSE1 was close enough and the test was ready, the guidance was set to start, and CSE1 started to follow the trajectory . For the elliptic trajectory, the CSE1 went two rounds, for every controller, and every speed. For the figure-eight trajectory, the controller were tested for one round on every speed. The desired velocities was set to $u_t = 0.2$, $u_t = 0.4$, but because of an error in the implementation, the guidance were called 4 times as fast as desired, so the resulting speed were $u_t = 0.8$, $u_t = 0.16$. In addition, a round were u_t

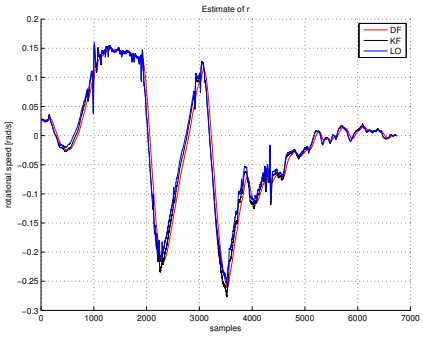
Figure 6.6 The figure shows the comparison of the different observers proposed in section 4.3, compared with the derivative filter previously implemented on the ship. They estimates for u, v and r are shown in (a), (b) and (c). The derivative filter is denoted DF, the nonlinear observer KF and lungenberg observer LO. It is clear that the lungenberg observer and nonlinear observer have somewhat faster responce, but creates more noise than the derivation filter.



(a)

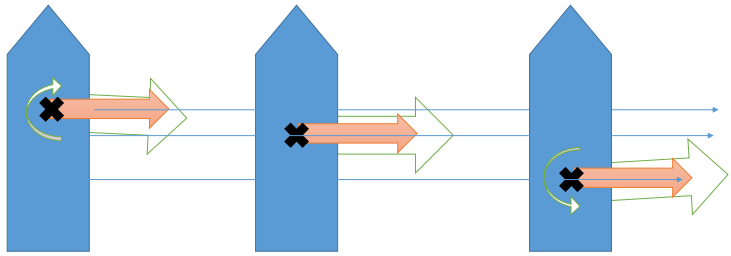


(b)



(c)

Figure 6.7 The figure shows the effect of changing the point of attack on the thruster allocation, and how it effects the moment of the ship, when tuning the thruster allocation.



changed from $u_t = 0$ to $u_t = 0.6$ was run, with pulse period of 15 seconds and pulse width of 70%. This was to get richer data for the multivariate analysis. The results of the tests can be seen in Chapter 6, were the two first rounds are plotted.

6.3.6 Preparations Outcome and Results

During these Experimental preparations, a few issues and children diseases were identified and had to be addressed, such as

Problems identified under pretesting

- The controllers did not handle wraparound problems
- The SVD controller had stability issues due to slow replacement of stored data
- Freezing signals effect on the controllers
- Oscillations when the controllers were badly tuned

The measurement problems were solved as explained in Section 4.3.2. The measurement problems also introduced rapid change for the controller, this introduced oscillations in roll, and the controller reached marginal stability. An example can be seen in Figure 6.23a. Another problem identified in this process was the big weakness of the CL SVD storage algorithm described in Section 4.1.3. Since the SVD algorithm always tries to maximize the minimal singular value of data matrix, it often keeps wrong with estimation errors but high deviation from the rest of the data, because changing them out would result

in lower minimal singular value. Seeing as the SVD algorithm does not have a "forgetting time" on its data, the wrong data are kept, which leads to instability. This was also confirmed in the post-experimental simulations and a possible solution was presented in Algorithm 4 in Section 7.2.2. By having stronger regulations on which data is allowed for the data matrix, and by introducing a maximum time for a data to be stored, it would lead to a significant increase in the robustness of the controller.

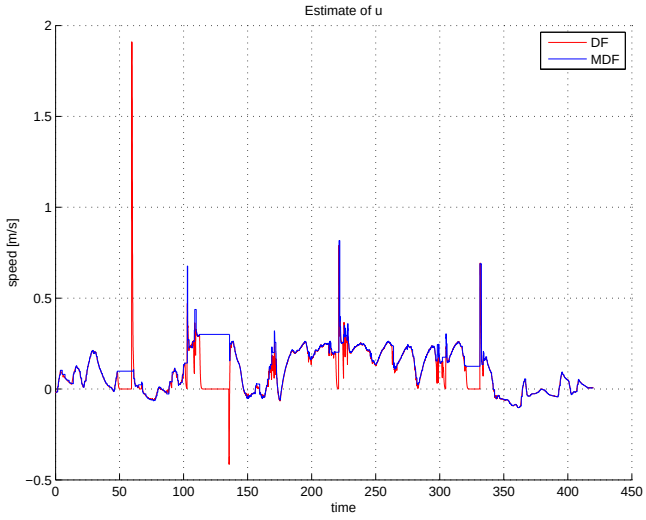
6.3.7 Estimator Test Results

Before running the the full control system, the estimators were tested separately, and in open loop ass explained in Section 6.3.3. When the DF was chosen, it had to be modified. The wraparound problem was solved by making a continuous ψ estimate as described in Section 4.3, the solution to the signal freeze is also described in this section. These measures decreased the problems with the measurement substantially. The result of the modified derivative filter compared to the old one can be see seen in Figure 6.8 and 6.9. For the experiments the parameters were $a = 0.08$ and $b = 0.05$.

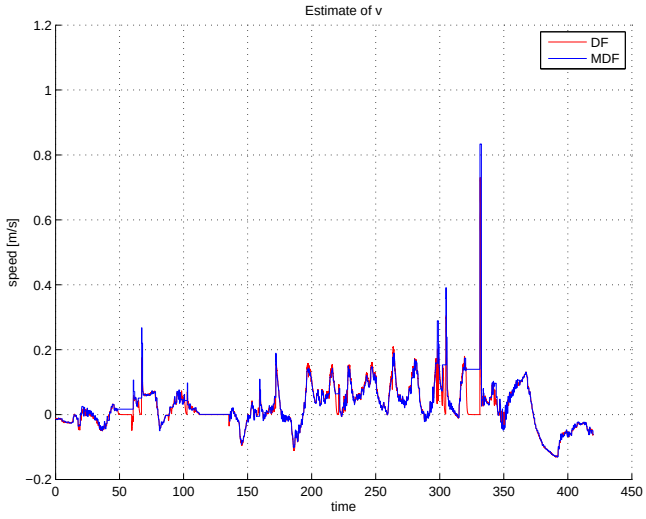
Tuning Thruster Allocation The thruster allocation was tested as explained in Section 6.3.4. Some moment was induced, and tuned to have $p_o = [0.18,\ 0,\ 0]^\top$.

Guidance Problems When the experiment date came closer, the guidance of the control system was changed into taking η as an input. This was to ensure a smoother set-point output from the Guidance. For some reason, this resulted in the guidance quadruple its frequency, resulting in the θ integrating from $\dot{\theta}$ four times as often as intended, which resulted in CSE1 driving four times as fast as requested and leaving the domain where the dynamic model was accurate. The dynamic model was said to work for slow velocities (Mc-lab and NTNU, 2015), and in (Skåtun, 2011) the towing experiment is done from $u \in [-0.6,\ 0.6]m/s$ to find the linear damping. During the tests this became especially apparent, as the surge error was significantly high, which was due to the controllers inability of adapt to uncertainties in surge. This was a result of using of the hydrodynamic parametrization from (Skjetne et al., 2004), which was chosen since the rotation related forces were in focus. On the bright side, CSE1 high speed, which resulted in a substantial high model-error, which made the analysis easier as can be seen in Chapter 7.

Figure 6.8 The figures show the estimates of the chosen MDF, compared with the DF previously implemented on CSE1. In Figure (a) the estimates of the surge velocity u are shown, and in (b) the sway velocity v are shown. It can be seen that the MDF keeps the velocity estimates constant during signal freeze, while the DF takes the estimated velocities to zero. This also affects the spikes happening after the signal freeze, and it should be noted that the blue spikes for MDF also are read spikes for DF.

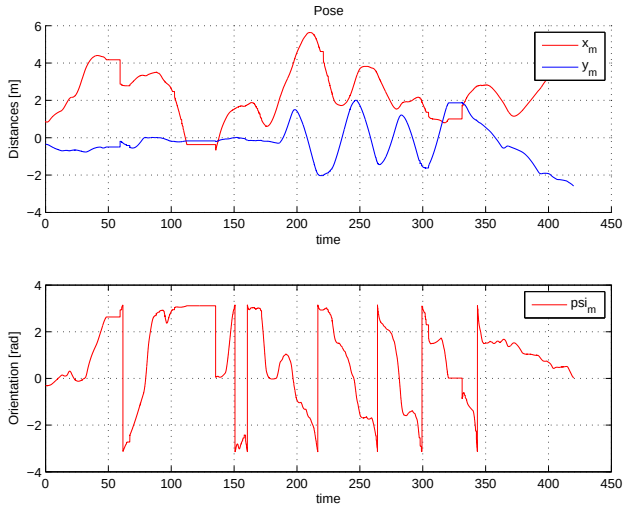


(a)

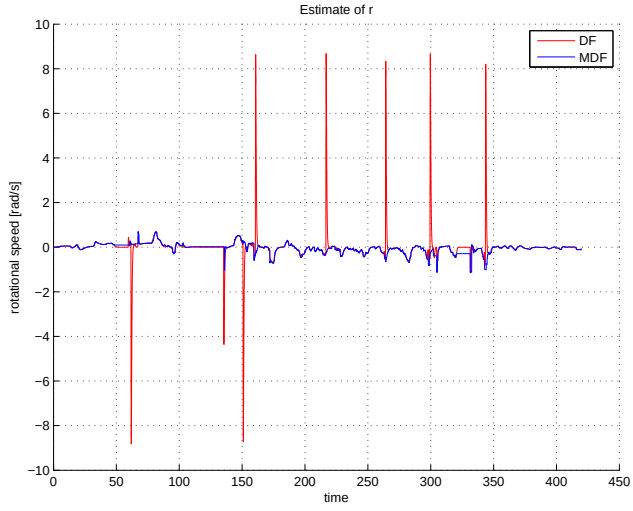


(b)

Figure 6.9 The figure show the chosen modified derivative filter, compared with the derivative filter previously implemented on the ship, together with the position and heading measurements. The measurements are in (a), and the rotation rate estimates are in (b). The previous derivative filter did not handle wraparound, witch clearly can be seen by comparing the heading measurements with the spikes in the DF estimates of the rotation rates.



(a)



(b)

6.4 MC-lab Plots

The tests were ran with CSE1 as explained in Section 6.3.5. First ran the experiment with the control gains found through simulations. These were not satisfactory, especially because there was a significant lag in surge, so the controller were tuned harder to get better tracking. The tuning was done while the ship moved in an ellipse at speed set to $u_t = 0.02$ which in reality was a $u_t = 0.08$, since the guidance was four times faster than intended. The resulting gains can be seen in Table 6.4. The tuning worked somewhat, but lead to a less stable system. The two first set of plots are of the experiments performed at the lowest speed, first in an elliptic trajectory, and then in a figure-eight trajectory. Subsequent are the plots in higher speeds, in the same order, these results are somewhat bad, but they are presented to give a comprehensive picture of the experiment done, and help in the discussion of the controllers.

Table 6.4: The control gains used after tuning CSE1, which was done while it followed the elliptic trajectory with a velocity of $u = 0.08[m/s]$

Tuning matrix	Tuning Gains	Values
$\boldsymbol{K}_p$	diag($[k_{pu}, k_{pv}, k_{pr}]$)	diag([0.2, 0.6, 0.2])
$\boldsymbol{K}d$	diag($[k_{du}, k_{dv}, k_{dr}]$)	diag([3, 6, 3])
$\boldsymbol{\Gamma}_\varphi$	diag($[\gamma_{\phi_1}, \cdots, \gamma_{\phi_8}]$)	diag([8, 4, 8, 8, 8, 4, 8, 8])

6.4.1 Result Plots

In this section, the results of all the different simulations are presented in plots. For each scenario there are 6 plots:

- Plot 1 shows the trajectory of the controllers in the given scenario
- Plot 2 (a) shows the offset angle and cross track error explained in Section 4.4
- Plot 2 (b) shows the $\boldsymbol{z_1}$ control signal, or body fixed pose error
- Plot 3 (a) shows the control metric also explained in Section 4.4
- Plot 3 (b) shows the output forces from the controllers, linear, nonlinear and total force.
- Plot 4 shows the estimate of $\boldsymbol{\omega}$, and the controllers prediction of $\boldsymbol{\omega}$.

The $\boldsymbol{\tau}$ plot 3 (b) is explained in section 5.1.

The different controllers have gotten their abbreviation and colours for the plots, and are as follows:

- Concurrent learning window storage algorithm with $\boldsymbol{z_2}$ as instantaneous error (CLZ)
- Concurrent learning window storage algorithm with $\boldsymbol{\epsilon}$ as instantaneous error (CLE)

- Regularl backstepping controller (RBS)
- Adaptive backstepping controller (ABS)

The controllers are summarized in Table 4.1.6

6.4.2 Elliptic Trajectory $u_t = 0.08$

The results of the ship following the ellipse trajectory, can be seen in Figure 6.10. The first thing to notice which is different from the simulations, is that the trajectories are inside the desired trajectory, opposed to the simulations in Figure 5.8a where they lay outside. This is due to the difference in model-error in surge and sway. This can be seen by comparing Figure 6.13 and 5.10, where surge is way greater in real life scenario than for the simulation model. The result is that the ship lags behind the tracking, and tries to take the inner turn to catch up. What is peculiar is that the RBS and ABS have equal performance, which paints the experiment as invalid for a comparative analysis. It can be seen that the ABS controller does not adapt in sway which leaves little room for improvement by the small adaptation in yaw. There can also be seen oscillations in Figure 6.10 and 6.12a, which is significantly stronger for the CL controllers. The amusing part is that these oscillations were not caught up by the IAEWWT metric, however, it may seem from looking at Figure 6.12b that CLZ has the superior tracking for this test. When looking at Figure 6.13 one can see how the different controllers adapt to the model-error, and it is curious to notice how CLZ and CLE adapt differently, while it may make sense to do like ABS and keep the estimates at zero. Later in the post analysis of the data it will be shown that although it is seemingly no error in sway, this can be the case, as is seen in Figure 7.15. Seemingly the CLZ estimates are the most correct, which can be confirmed by comparing the cross track error of CLZ and CLE in the figures 6.12a and 6.11b. In the rotation error estimates the controllers are trying to adapt to the epsilon estimates, but are slower than the simulations, which can be seen in Figure 5.10. However, the biggest problem with the controllers can be seen in the surge sub plot, where one can see that the controllers does not adapt in surge direction at all. This will be a repeating problem in the rest of the experiments.

Figure 6.10 The figure shows the tests of CSE1 following an elliptic trajectory, with a desired speed of $u_t = 0.08m/s$, using the controllers mentioned in Section 6.4.We see the desired elliptic trajectory in black, with the paths of CSE1 with the different controllers

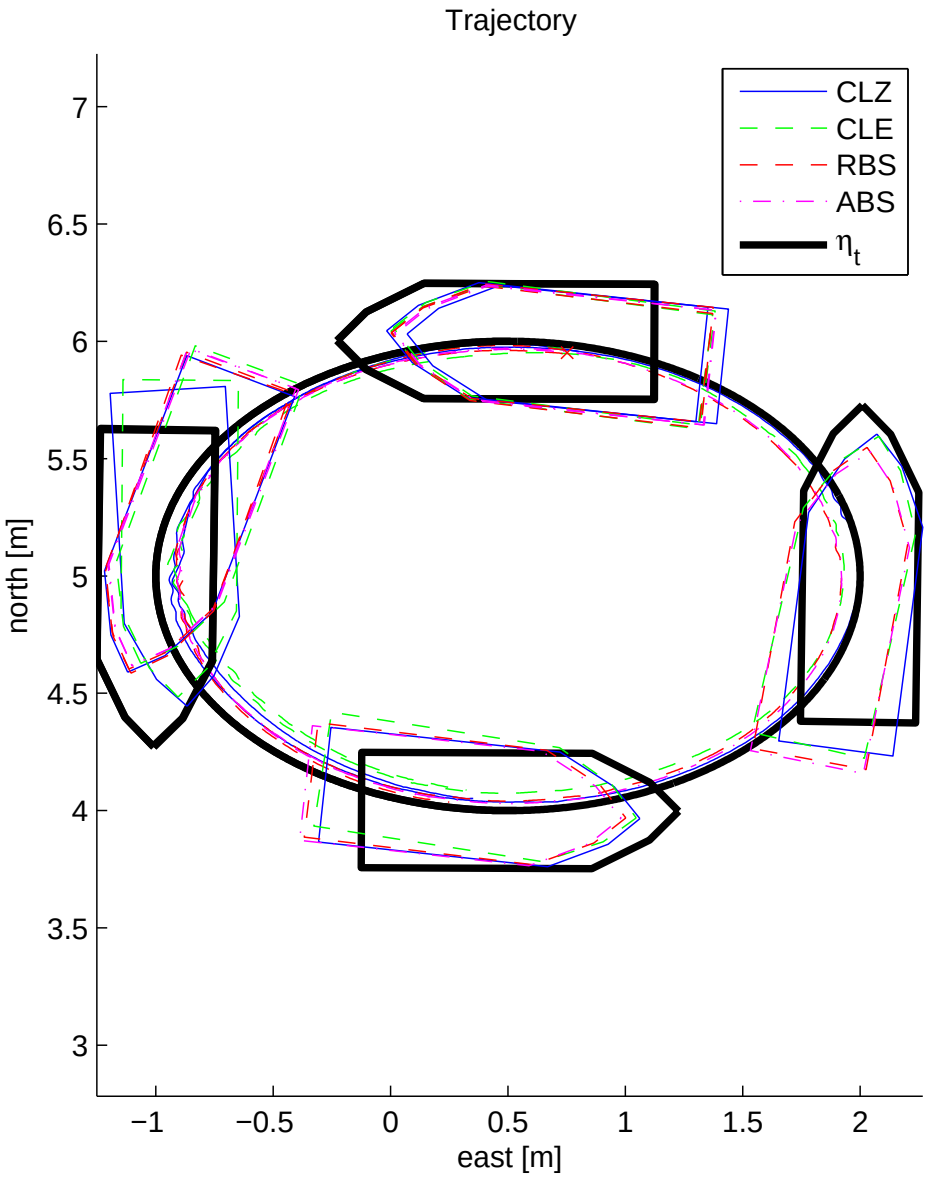
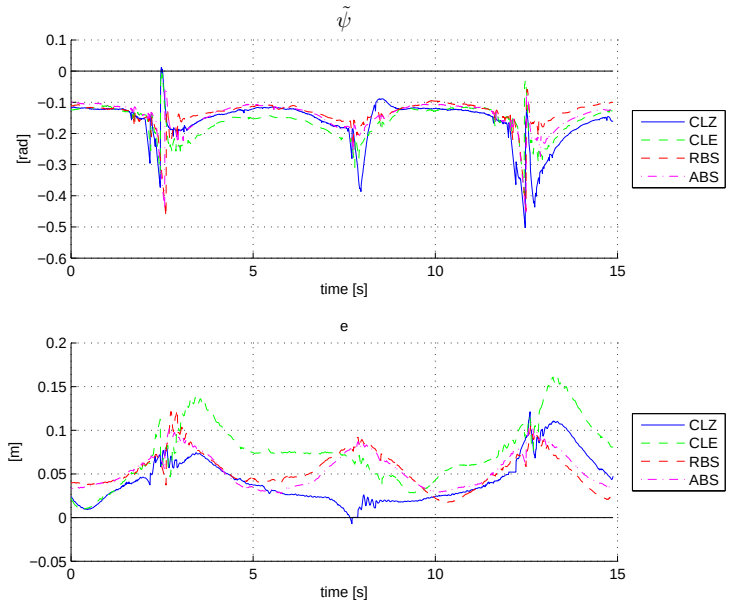
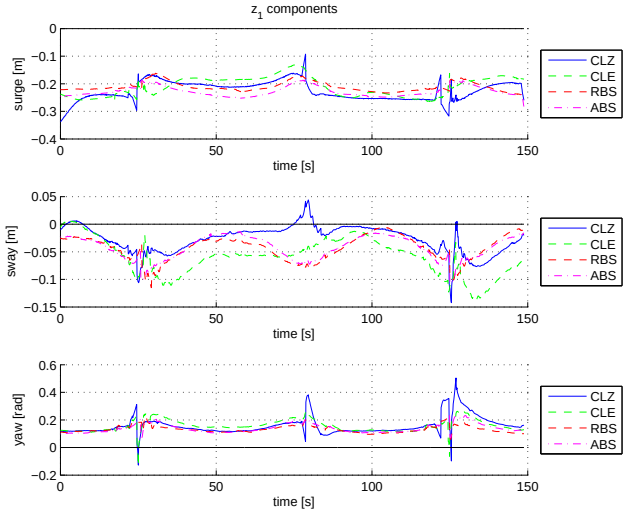


Figure 6.11 The figure shows the tests of CSE1 following an elliptic trajectory, with a desired speed of $u_t = 0.08m/s$, using the controllers mentioned in Section 6.4. In Figure (a) the cross track error described in Section 4.4 is plotted together with $\tilde{\psi}$. In Figure (b) the $\boldsymbol{z_1}$ used in the control is plotted. In essence the body fixed pose error.

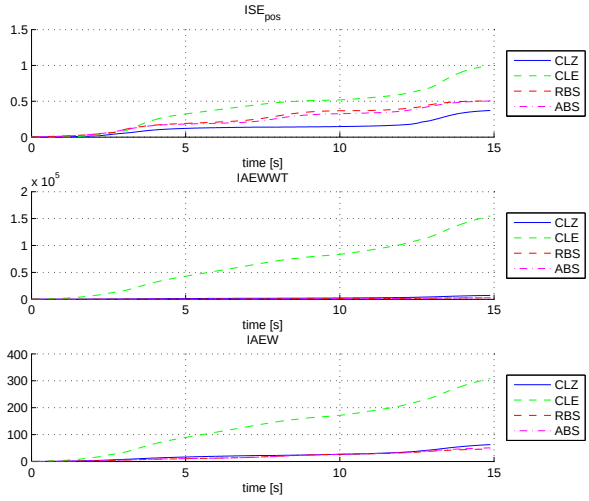


(a)

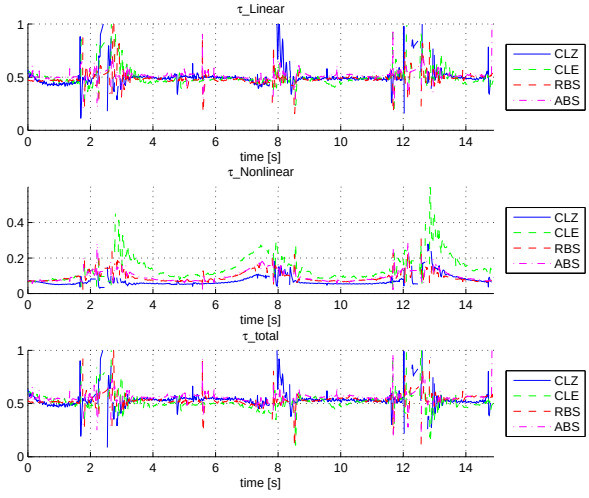


(b)

Figure 6.12 The figure shows the tests of CSE1 following an elliptic trajectory, with a desired speed of $u_t = 0.08m/s$, using the controllers mentioned in Section 6.4. In Figure (a) the performance metrics described in Section 4.4 are shown. In (b) the norm of the thrust was shown, where the τ_Linear accounts for the thrust part related to feedback, while the τ_Nonlinear part accounts for the feed forward part of the control including both the non linearities in the model, and the estimated part of the controller.

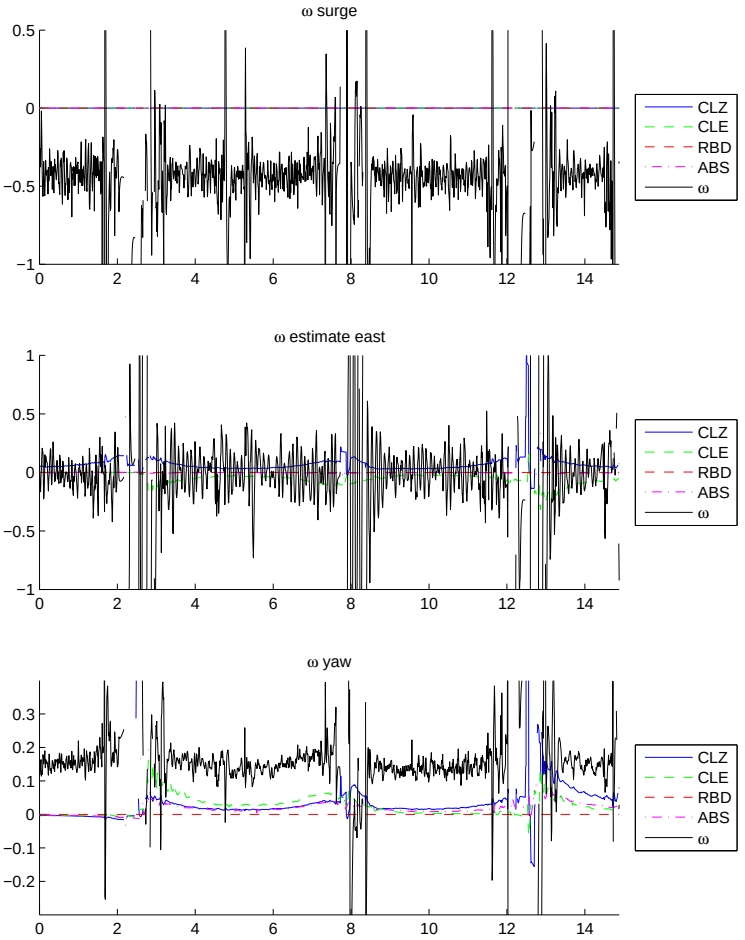


(a)



(b)

Figure 6.13 The figure shows the model-error estimates The model-error ω that is shown as the black line, and the lines of the controllers show their estimate of $\hat{\omega}$ using (2.35).



6.4.3 Figure-Eight Trajectory $u_t \in [0.015, 0.5]$

The results of CSE1 following the figure-eight path can be seen in Figure 6.15 and 6.16. The controllers are able to follow the figure-eight. The effect of the frozen measurements can also be seen in the lower right corner of Figure 6.22. Here again there is little difference between the BS and ABS, and it can be seen that the ABS and RBS manage to get closer to the desired trajectory, while the CLZ and CLE manage to have a constant error in sway, although this is not reflected in the $\boldsymbol{z_1}$ as can be seen in figure 7.1, which is peculiar. The RBS and ABS comes best out of the metrics in 6.16b, and ABS seams a bit more energy efficient than RBS, while CLE is by far the worst. By examining the model-error in Figure 6.17 one can see that this error is not as systematic as the one from the ellipse, and the adaptive controllers struggle with estimating. This could be because the error is mostly related to other parametrizations, or the propulsion vector, or that the noisy estimates damage the adaptation. It can also be that the adaptive gains are badly tuned, or that there are some faults in the implementation. This was further investigated during the system identification and post processing simulations explained in Section 7.2.1.

Figure 6.14 The figure shows the tests of CSE1 following an figure-eight trajectory, explained in Section 6.4.3 using the controllers mentioned in Section 6.4. We see the desired elliptic trajectory in black, with the paths of CSE1 with the different controllers

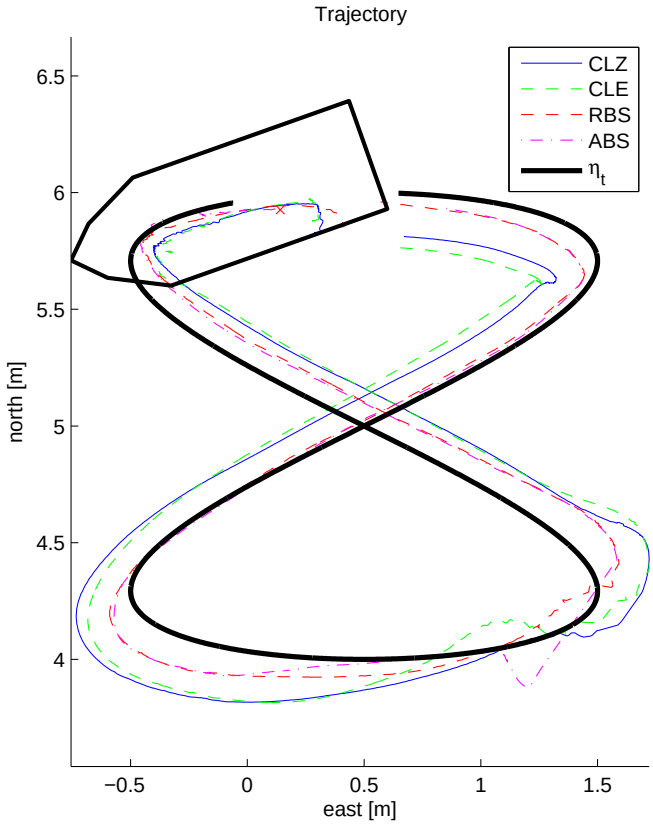
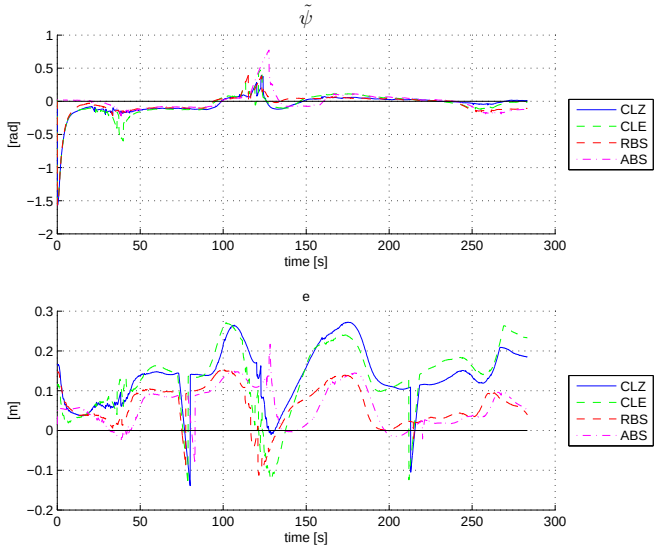
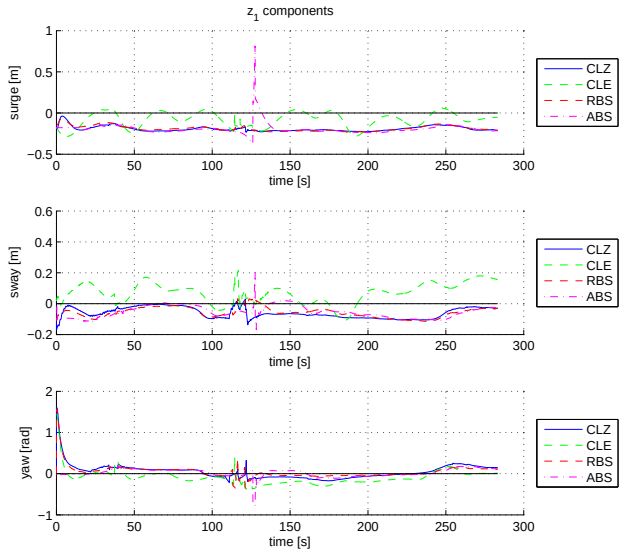


Figure 6.15 The figure shows the tests of CSE1 following an figure-eight trajectory, explained in Section 6.4.3, using the controllers mentioned in Section 6.4. In Figure (a) the cross track error described in Section 4.4 is plotted together with $\tilde{\psi}$. In Figure (b) the $\boldsymbol{z_1}$ used in the control is plotted. In essence the body fixed pose error.

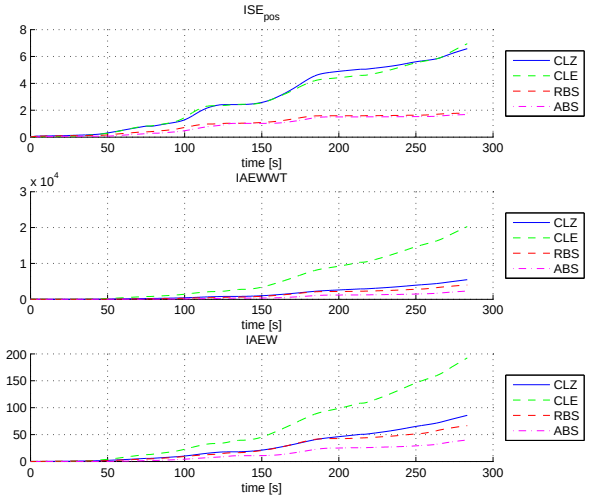


(a)

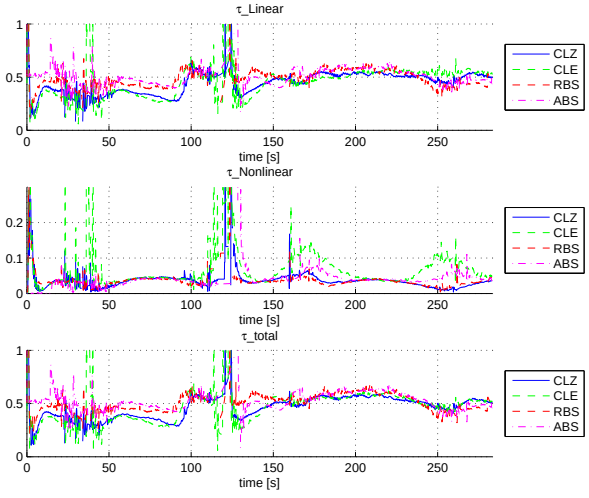


(b)

Figure 6.16 The figure shows the tests of CSE1 following an figure-eight trajectory, explained in Section 6.4.3, using the controllers mentioned in Section 6.4. In Figure (a) the performance metrics described in Section 4.4 are shown. In (b) the norm of the thrust was shown, where the τ_Linear accounts for the thrust part related to feedback, while the τ_Nonlinear part accounts for the feed forward part of the control including both the non linearities in the model, and the estimated part of the controller.

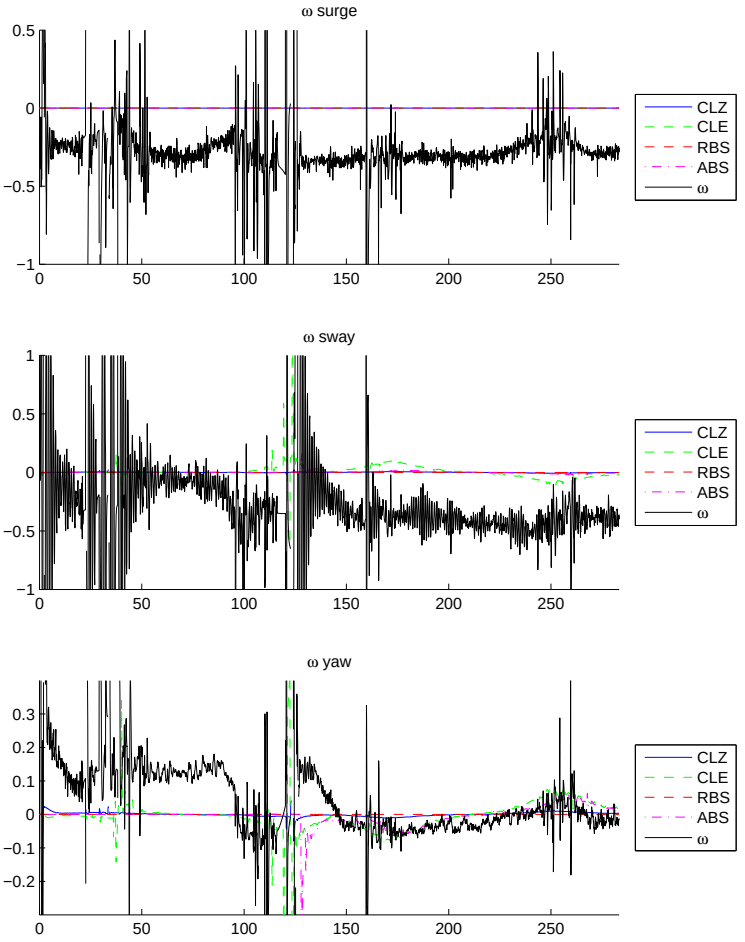


(a)



(b)

Figure 6.17 The figure shows the model-error ω that is shown as the black line, and the lines of the controllers show their estimate of $\hat{\omega}$ using (2.35).



6.4.4 Elliptic Trajectory $u_t = 0.16$

The results of CSE1 following the elliptic path can be seen in Figure 6.18. It should be noticed that the tracking abilities are substantially decreased, and these experiments are more valid in testing the controllers robustness than their actual tracking performance. The oscillations has increased, and one can see more clearly where the measurement is lost especially in Figure 6.20a. Here the CLZ and CLE controllers fails compared with the regular RBS and ABS. And as implemented in this project, the CL controllers are less robust than the ABS and RBS controllers. It is also interesting to see that the ABS still manages to show improved performance compared to RBS. Also by looking at Figure 6.21 one can see that the increased error in surge hasn't increased that much compared to Figure 6.13, but the sway error has drifted to the side. A possible explanation of this is that the bow thruster has trouble with working with speeds exceeding $u = 0.15m/s$ (Skjetne et al., 2004) which will affect the sway thrust on the ship.

Figure 6.18 The figure shows the tests of CSE1 following an elliptic trajectory, with a desired speed of $u_t = 0.08m/s$, using the controllers mentioned in Section 6.4. We see the desired elliptic trajectory in black, with the paths of CSE1 with the different controllers

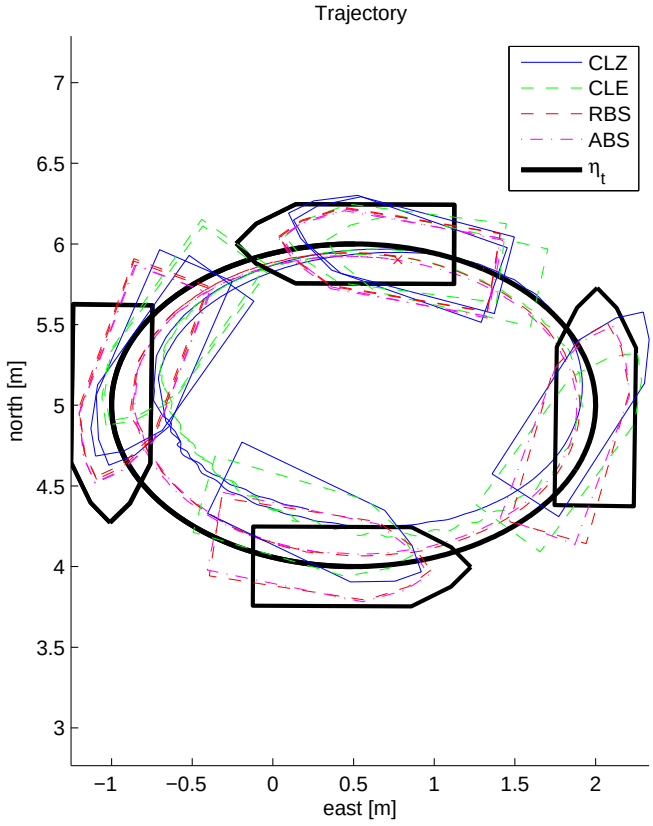
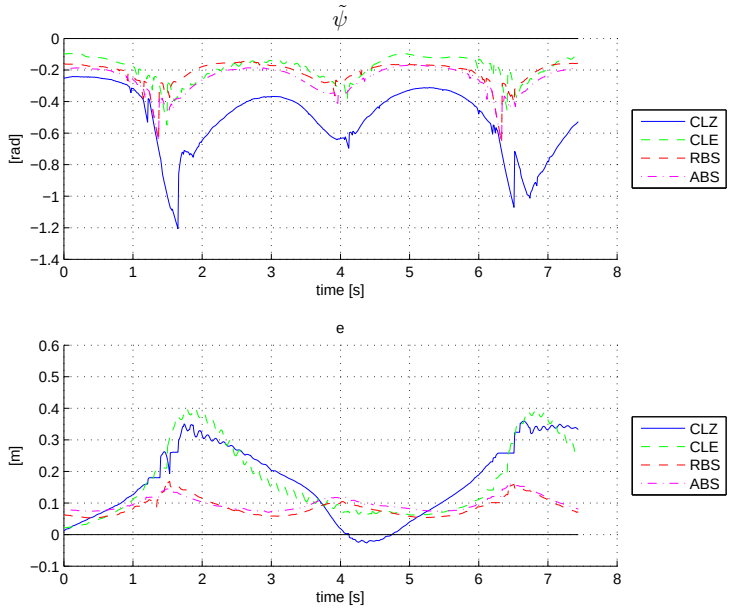
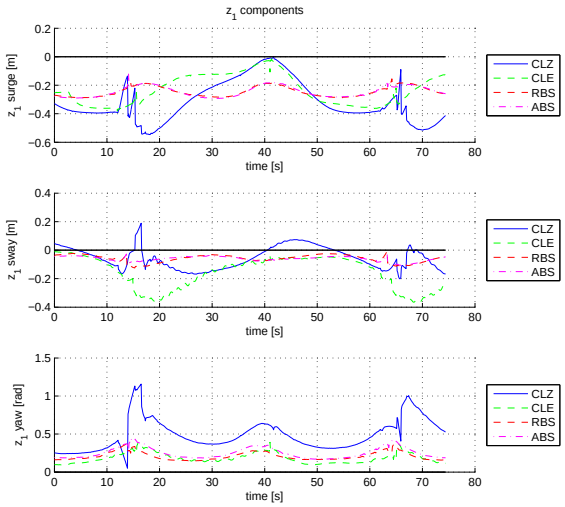


Figure 6.19 The figure shows the tests of CSE1 following an elliptic trajectory, with a desired speed of $u_t = 0.16m/s$, using the controllers mentioned in Section 6.4. In Figure (a) the cross track error described in Section 4.4 is plotted together with $\tilde{\psi}$. In Figure (b) the $\boldsymbol{z_1}$ used in the control is plotted. In essence the body fixed pose error.

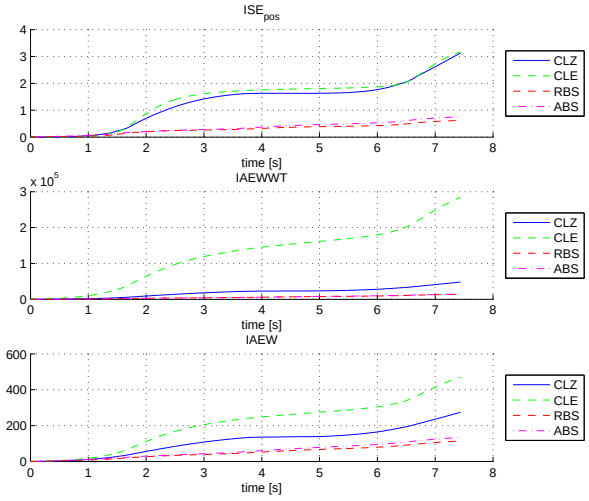


(a)

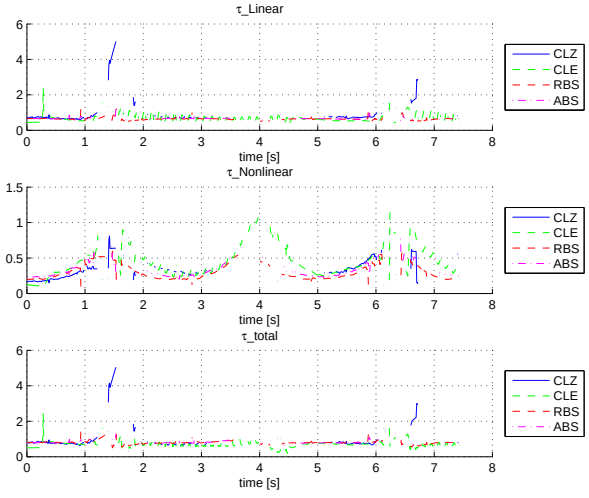


(b)

Figure 6.20 The figure shows the tests of CSE1 following an elliptic trajectory, with a desired speed of $u_t = 0.16m/s$, using the controllers mentioned in Section 6.4. In Figure (a) the performance metrics described in Section 4.4 are shown. In (b) the norm of the thrust was shown, where the τ_Linear accounts for the thrust part related to feedback, while the τ_Nonlinear part accounts for the feed forward part of the control including both the non linearities in the model, and the estimated part of the controller.

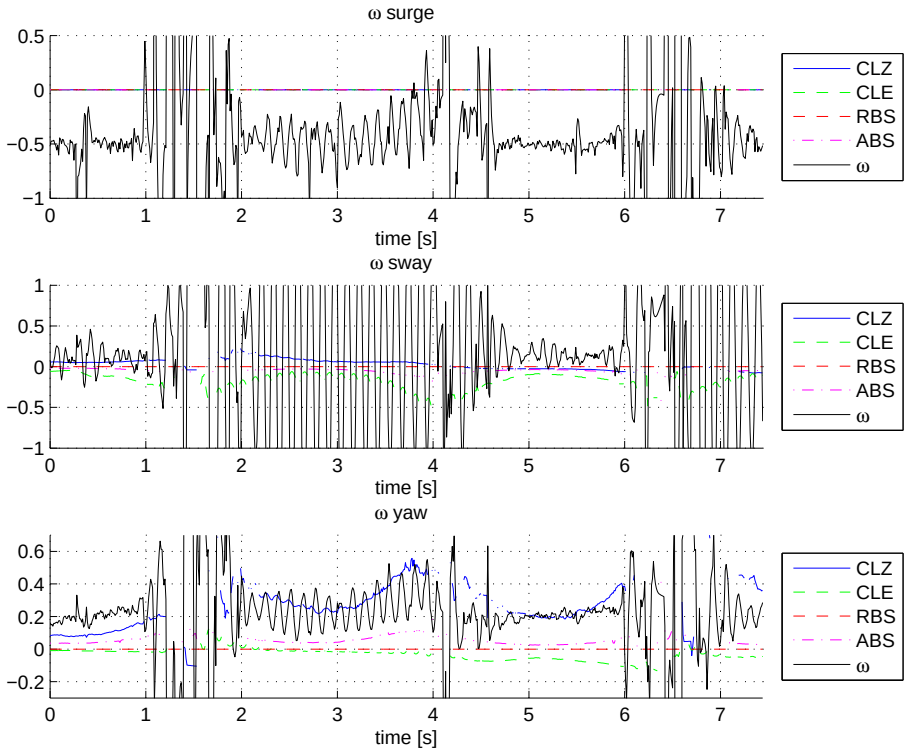


(a)



(b)

Figure 6.21 The figure shows the model-error ω that is shown as the black line, and the lines of the controllers show their estimate of $\hat{\omega}$ using (2.35).



6.4.5 Figure-Eight Trajectory $u_t \in [0.02, 0.67]$

The results of CSE1 following the figure-eight path can be seen in Figure 6.22. The controllers are able to follow the figure-eight, however in this case the CLZ are set into a marginal stability together with ABS, which can be seen as oscillations in the sway motion. The effect of this can be seen in the IAEWWT as CLZ surpasses the CLE, and ABS surpasses RBS. Although the metric should have caught the oscillations better. Though it should be noted that the adaptation parameters does not adabt, they probably come from hydrodynamic estimate in both CLZ and ABS that has made the system under damped. This combined with under damped roll and some time delay in the controller, or thrust system, especially the BT gives the oscillations. The period of the oscillations was close to $T = 0.1s$. Although its curious that the ABS controller also had oscillations, as it was belied it would be more robust against oscillations than the CL controllers, this also confirms that the ships were tuned aggressively.

Figure 6.22 The figure shows the tests of CSE1 following an figure-eight trajectory, explained in Section 6.4.5 using the controllers mentioned in Section 6.4. We see the desired elliptic trajectory in black, with the paths of CSE1 with the different controllers

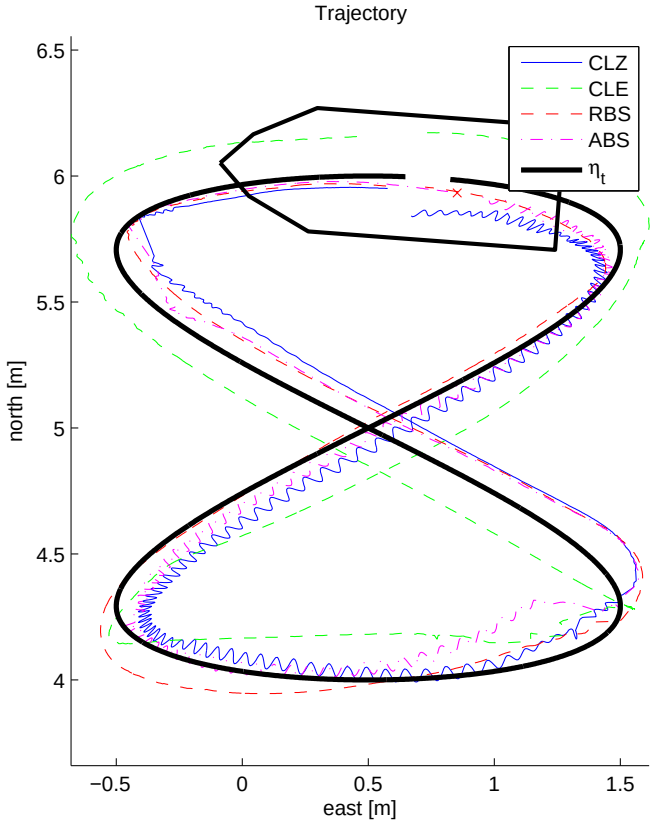
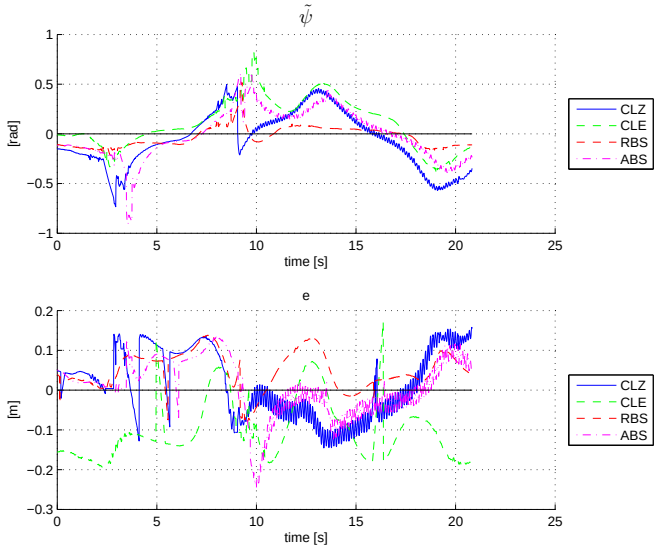
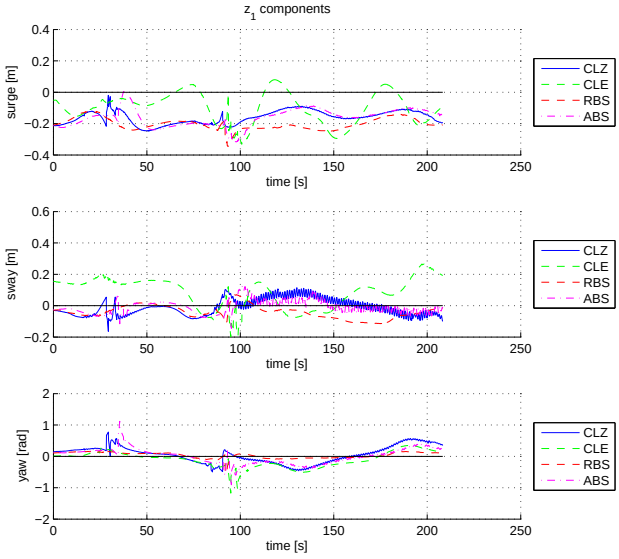


Figure 6.23 The figure shows the tests of CSE1 following an figure-eight trajectory, explained in Section 6.4.5 using the controllers mentioned in Section 6.4. In Figure (a) the cross track error described in Section 4.4 is plotted together with $\tilde{\psi}$. In Figure (b) the $\boldsymbol{z_1}$ used in the control is plotted. In essence the body fixed pose error.

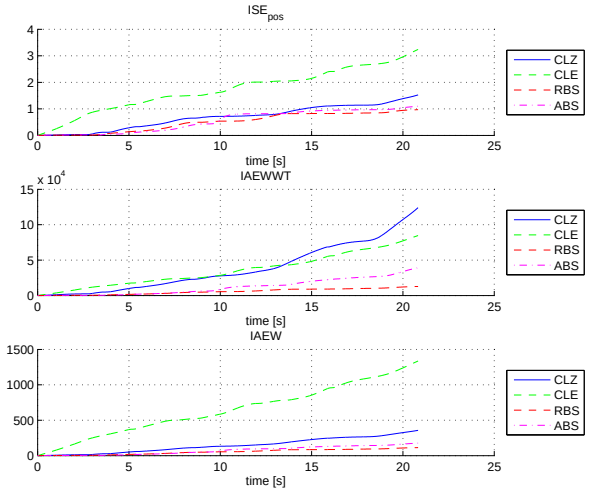


(a)

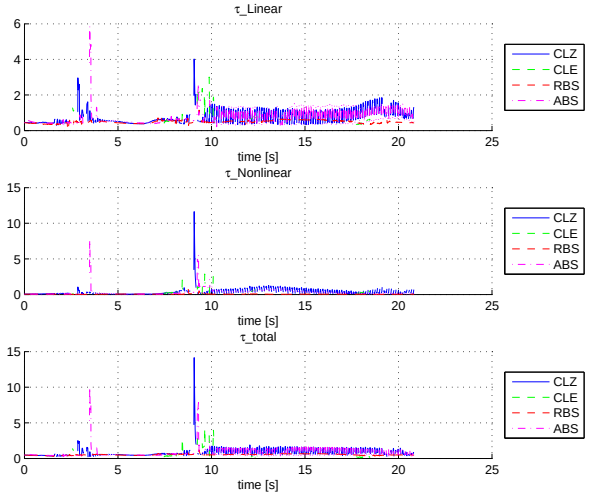


(b)

Figure 6.24 The figure shows the tests of CSE1 following an figure-eight trajectory, explained in Section 6.4.5 using the controllers mentioned in Section 6.4. In Figure (a) the performance metrics described in Section 4.4 are shown. In (b) the norm of the thrust was shown, where the τ_Linear accounts for the thrust part related to feedback, while the τ_Nonlinear part accounts for the feed forward part of the control including both the non linearities in the model, and the estimated part of the controller.

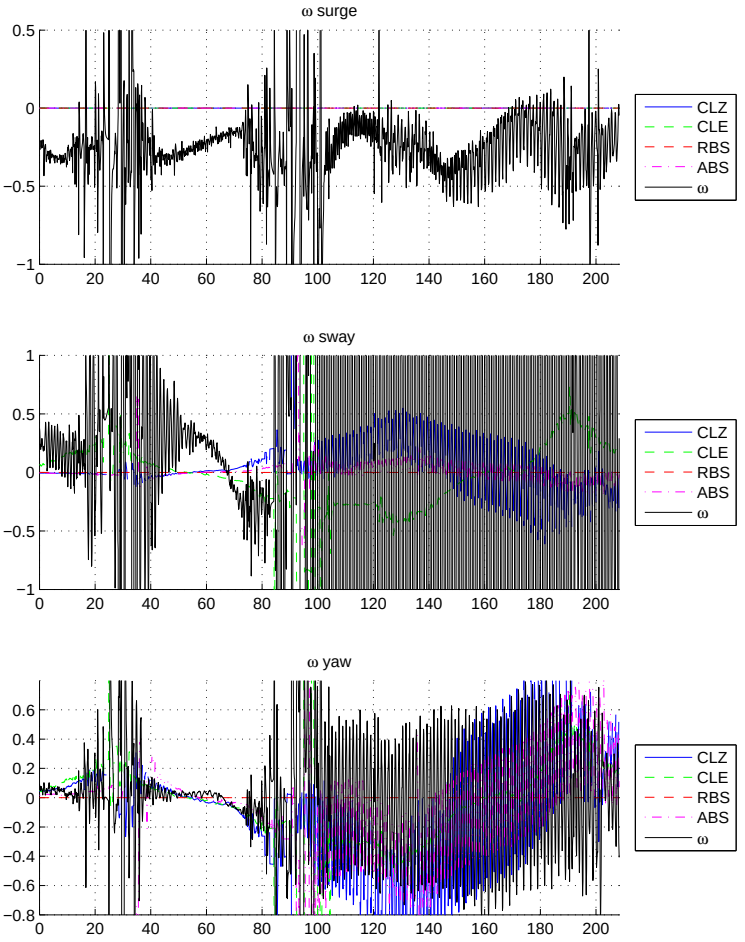


(a)



(b)

Figure 6.25 The figure shows the model-error ω that is shown as the black line, and the lines of the controllers show their estimate of $\hat{\omega}$ using (2.35).



6.5 Lessons Learned

As one can see from the results, not everything in the experiments worked as intended. To summarize:

- Guidance was four times as fast as desired
- CSE1 manoeuvred outside the domain of the dynamical model
- The performance difference of BS and ABS controllers were not significant, disqualifying the experiments for comparative analysis.
- Large model-error, especially in the surge motion
- The parametrization of the hydrodynamic model-error lead to the controllers inability of adapting in surge direction.
- CL storage algorithms stored spikes from the ϵ estimates.
- The nonlinear estimators had insufficient robustness
- Aggressive tuning lead to an underdamped control system.

Some of these problems could have been caught by having better routines. The guidance problem for instance could easily have been detected if the speed of CSE1 had been monitored online. The ship was watched in the basin, but distinguishing a surge speed of $0.08[m/s]$ with $0.02[m/s]$ seemed rather difficult in retrospective.

Another big error was to trust the dynamical model too blindly, and it is easy in hindsight to overthrow the parametrization as reckless, although the plan was to move CSE1 such that only the hydrodynamics related to the rotation rate of the ship were unknown. This could have been possible if the experiments had been performed at slower speeds or the dynamical model had been valid for higher speeds, as can be seen in (Skjetne et al., 2004)[1]. It is an important eye-opener to see how bad results can become if the foundation you are working on is not strong enough. This parametrization is modified in Chapter 7, and a simulation is shown where it tackles the newly acquired model-error. In this chapter, the unknown disturbance was introduced in the kinetic model of the simulator. This could have been introduced earlier, even as a constant, and the obvious weakness of the model-error parametrization would have been apparent.

As of the robustness of the CL controllers, and nonlinear estimators, these problems could have been addressed better and earlier if the simulator was built with noise and signal freeze. The problem with CL controllers was that the storage algorithms stored regression matrices and ϵ with spikes, and in addition adapted further to the spikes in ϵ, this became apparent when the adaptations were post-experimental stimulated in Chapter 7. This problem is also addressed in this chapter and an adaptation blocker was made to hinder adaptation or storage of data during spikes. This was again used when the CL algorithm with ϵ was used for system identification in the post-experimental simulations. Another

[1] Although Cybership II does have higher speed than in this experiment, the dynamic model was tested, and thus valid for the speed it was tested in.

solution to this problem could also be addressed by having a more accurate estimate of $\boldsymbol{\nu}$ and $\dot{\boldsymbol{\nu}}$, which is possible in retrospective. The data stored does not have to be calculated instantaneously, which gives room to other estimators like optimal fixed point smoother (Chowdhary et al., 2012). This way, more accurate estimates of the states $\boldsymbol{\nu}$,$\dot{\boldsymbol{\nu}}$ and the error signal $\boldsymbol{\epsilon}$ could be acquired for the stored data in the CL controller.

Chapter 7

Post-Experimental Analysis and Improvements

In this chapter, we go through the procedures done after the experiments. From post process simulations to recuperation of lost data, and clarification of the adaptation, to the building the ω model through system identification (SysID) and MVA. The validation of the ω model was also tested through simulations, and at the end a modification of the CL controller was proposed, and simulated to handle the proposed ω.

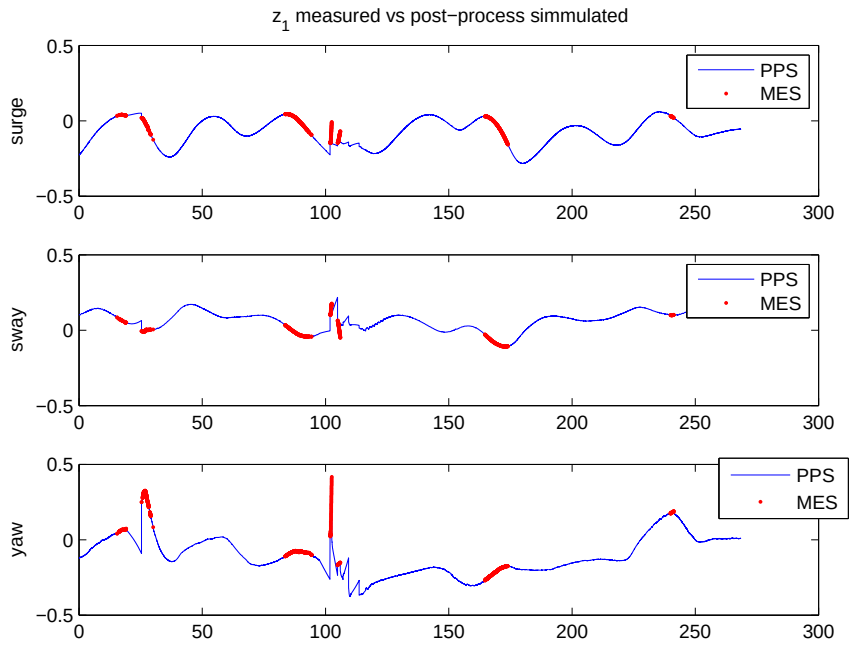
7.1 Data Post Processing

During the experiment, some of the states measured had failure in their delivery through the Wi-Fi. This resulted in the data of for instance τ and z_1 having NaN measurements in them. A solution was to modify the simulator, so it could run open loop simulation, where instead of taking measurement from simulator model, the measurement, guidance and thrust was fed from the measurements. By doing this, the adaptation of the different controllers could be reproduced and verified. The results of the post-processing simulation can be seen in Figure 7.1.

7.1.1 Adaptation Verification

After the experiments, there was an uncertainty about the adaptation of the controllers. Some seemed to have produced estimates of ω but they were not as those from the simulations. Doing the open loop simulation confirmed this, and it seemed like a combination of noisy estimates and spikes slowed down the estimations. For CLZ and CLE, the storage algorithm seemed to store up the spikes, which eliminated each other, thus gave bad estimations. In addition trying to eliminate the the stored data for the adaptation resulted in the instantaneous adaptation were mostly effected by the spikes, thus not being able to converge. This also accounted for the ABS controller, and obviously the controllers

Figure 7.1 The figure shows the z_1 signal for the eight figure trajectory of CSE1 with CLE controller. The red points are what was measured, where the rest of the measurements were NaN. The blue line was z_1 found during the post process simulation of the data. PPS stands for post process simulation



did not adapt to $\boldsymbol{\omega}_u$ in surge, in the simulations as well. In addition, as stated before, the parametrization was made so that the controllers was enable to adapt in surge, this would probably be the easiest and most effective fix for the adaptations.

Also with small changes the simulator with the CL controller using ϵ could be used for SysID.

7.2 Analysis of Model-Error

When the adaptation had been reproduced from the tests, the work on improving the weaknesses of the CL adaptation schemes started. This included being able to reproduce the model-error, but also changing the parametrization of the model-error $\boldsymbol{\omega}$. By looking at the model-error from the experiments in figures 6.13, 6.21 and 6.17 there seemed likely that other effects than the hydrodynamics were part of the $\boldsymbol{\Omega}$. Especially the error on surge during the elliptic trajectories, which only went from $0.4[N]$ to $0.5[N]$ as the surge speed doubled from $0.08[m/s]$ to $0.16[m/s]$. To investigate the model-error further, the data was analysed using MVA and SysID in a off-line estimation.

7.2.1 Post Processing Simulations and System Identification

After verifying the adaptation, it was certain that there had to be improvements if the CL algorithm with $\boldsymbol{\epsilon}$ was to be used for SysID. In which the adaptation of the controller was used to find parametrization that approximated the model-error. Since the post processing was open loop, the $\boldsymbol{z_2}$ was pre determined, so CLE was the controller chosen for SysID. Although the weaknesses addressed in Section 7.1.1 had to be improved. This also resulted in a new parametrization of the model-error for the SysID

7.2.2 Improved Concurrent Learning Adaptation

The weaknesses of the CL algorithm which was tested in the MC-lab was:

- The CL adaptation stored the spikes in the data matrix, and spikes dominated adaptation.
- The parametrization of the hydrodynamic uncertainties, made it impossible for the controller to adapt for disturbance and model-error in surge direction.
- It only adapts to hydrodynamic forces, and do not adapt to thruster loss or other other disturbances

Three measures were therefore done; one was to change the parametrization of the hydrodynamics to account for surge forces. The regression matrix was then

$$\boldsymbol{\Phi}(\boldsymbol{\nu}) = \begin{bmatrix} -|u|u & -u & -|u|r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -|r|v & -r & -|v|r & -|r|r & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -|r|v & -r & -|v|r & -|r|r \end{bmatrix} \tag{7.1}$$

The parameters to be estimated are then

$$\hat{\boldsymbol{\varphi}} = [X^{\epsilon}_{|u|u},\ X^{\epsilon}_{u},\ X^{\epsilon}_{|u|r},\ Y_{|r|v},\ Y_{r},\ Y_{|v|r},\ Y_{|r|r},\ N_{|r|v},\ N_{r},\ N_{|v|r},\ N_{|r|r}] \tag{7.2}$$

For the thruster error, we assume that the thruster error can be written as

$$\boldsymbol{\tau}_{req} = \boldsymbol{\tau}_{act} + \boldsymbol{\Delta}_{\tau}\boldsymbol{\tau}_{req} \tag{7.3}$$

where $\boldsymbol{\Delta}_{\tau} = \text{diag}([\delta\tau_u,\ \delta\tau_v,\ \delta\tau_r])$, but since we are trying to estimate the $[\delta\tau_u,\ \delta\tau_v,\ \delta\tau_r]$ we instead parametrize it as

$$\boldsymbol{\Delta}_{\tau}\boldsymbol{\tau} = \boldsymbol{R}_{\boldsymbol{\tau}_{req}}(\boldsymbol{\tau})\boldsymbol{\omega}_{\boldsymbol{\tau}} = \text{diag}([\tau_u,\ \tau_v,\ \tau_r])\boldsymbol{\omega}_{\boldsymbol{\tau}} \tag{7.4}$$

where $\boldsymbol{R}_{\boldsymbol{\tau}_{req}}(\boldsymbol{\tau})$ is the regression matrix, and the parameters to be estimated are

$$\boldsymbol{\omega}_{\boldsymbol{\tau}} = [\delta\tau_u,\ \delta\tau_v,\ \delta\tau_r]^{\top} \tag{7.5}$$

The thrust is therefore modelled as

$$\boldsymbol{\tau}_{act} = \boldsymbol{\tau}_{req} - \boldsymbol{R}_{\boldsymbol{\tau}}(\boldsymbol{\tau}_{req})\boldsymbol{\omega}_{\boldsymbol{\tau}} \tag{7.6}$$

To avoid the problem with spikes in $\boldsymbol{\tau}$, $\boldsymbol{\nu}$ and $\dot{\boldsymbol{\nu}}$, a variable was introduced, to block adaptation and data storage if the value of $\boldsymbol{\tau}$, $\boldsymbol{\nu}$ or $\dot{\boldsymbol{\nu}}$ exceeded certain values.

Algorithm 4 pseudocode for the adoption block variable, that is set to 0 and kept 0 for a time if there occur spikes in $\boldsymbol{\tau}$ or $\dot{\boldsymbol{\nu}}$.

```
input ← ν, ν̇, τ, β
timer ← persistent
if β == 1 then
    if ||ν̇|| > ε1 or ||ν|| > ε2 or ||τ|| > ε3 then
        β ← 0
        timer ← 0
    end if
else if β == 0 then
    if ||ν̇|| > ε1 or ||ν|| > ε2 or ||τ|| > ε3 then
        timer ← 0
    else if timer > N then
        adaptation ← 1
    else
        timer + +
    end if
end if
output ← β
```

so if a spike appear in $\boldsymbol{\nu}$, $\dot{\boldsymbol{\nu}}$ or $\boldsymbol{\tau}$, the variable β is set to zero, and hold there for a time, else will be $\beta = 1$.

7.2.3 Adaptation Law

The adaptation that is derived from the adaptations in Section 4.1.4 , and (7.6) is substituted for $\boldsymbol{\tau}$ and setting $\boldsymbol{\omega}_n = \mathbf{0}$

$$\begin{aligned} -\boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* &= \boldsymbol{M}^*\dot{\boldsymbol{\nu}} + \boldsymbol{C}^*(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - (\boldsymbol{\tau}_{req} - \boldsymbol{R}_{\boldsymbol{\tau}}(\boldsymbol{\tau}_{req})\boldsymbol{\omega}_{\boldsymbol{\tau}}{}^*) \\ -\boldsymbol{R}_{\boldsymbol{\tau}}(\boldsymbol{\tau}_{req})\boldsymbol{\omega}_{\boldsymbol{\tau}}{}^* - \boldsymbol{\Phi}(\boldsymbol{\nu})\boldsymbol{\varphi}^* &= \boldsymbol{M}^*\dot{\boldsymbol{\nu}} + \boldsymbol{C}^*(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\tau}_{req} \\ \boldsymbol{y} = \boldsymbol{\omega} &= \boldsymbol{M}^*\dot{\boldsymbol{\nu}} + \boldsymbol{C}^*(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) - \boldsymbol{\tau}_{req}, \end{aligned} \tag{7.7}$$

and our estimate of the uncertainty is

$$\hat{\boldsymbol{y}} = \hat{\boldsymbol{\omega}} = -R_{\boldsymbol{\tau}}(\boldsymbol{\tau}_{req})\hat{\boldsymbol{\omega}}_{\tau} - \boldsymbol{\Phi}(\boldsymbol{\nu})\hat{\boldsymbol{\varphi}}. \tag{7.8}$$

then the estimation error of $\boldsymbol{\omega}$ is

$$\boldsymbol{\epsilon} = \boldsymbol{\omega} - \hat{\boldsymbol{\omega}} = \boldsymbol{y} - \hat{\boldsymbol{y}} \tag{7.9}$$

and the adaptations laws are then

$$\dot{\hat{\boldsymbol{\omega}}}_{\tau k} = (-\boldsymbol{\Gamma}_{\boldsymbol{\omega}_{\boldsymbol{\tau}\boldsymbol{k}}}\boldsymbol{R}_{\boldsymbol{\tau}}(\boldsymbol{\tau}_{req})\boldsymbol{\epsilon}_k - \sum_{j=1}^{k-1}\boldsymbol{\Gamma}_{\boldsymbol{\omega}_{\boldsymbol{\tau}}}\boldsymbol{R}_{\boldsymbol{\tau}}(\boldsymbol{\tau}_{req})\boldsymbol{\epsilon}_{\boldsymbol{j}})\beta \tag{7.10}$$

$$\dot{\hat{\boldsymbol{\varphi}}} = (-\boldsymbol{\Gamma}_{\boldsymbol{\varphi}}\boldsymbol{\Phi}(\boldsymbol{\nu})^{\top}\boldsymbol{\epsilon} - \sum_{j=1}^{k-1}\boldsymbol{\Gamma}_{\boldsymbol{\phi}}\boldsymbol{\Phi}^{\top}(\boldsymbol{\nu}_j)\boldsymbol{\epsilon}_{\boldsymbol{j}})\beta, \tag{7.11}$$

where the concurrent learning window algorithm from Section 4.1.3 with 20 data points was used for the SysID. We see here that the adaptation stops if $\beta = 0$;

7.2.4 Results from System Identification

The SysID was ran with the data from the experiments, after a run, the starting value of $\boldsymbol{\varphi}$ and $\boldsymbol{\omega}_{\tau}$ was updated, and the simulation ran again with the same data. This was repeated until all the parameters had reached stationary, or nearly stationary values. Then the same model was tested on other experiment data to validate the result. The result of the error modelled can be seen in Figure 7.2, where one can see that the proposed model captures the main dynamic of the model. The parameters for different SysID's can be seen in Table 7.1

7.2.5 Thruster Error Identification

In one of the tests, the bow thruster suddenly stopped working, due to low voltage in the battery. A SysID was done on this to see if the $\boldsymbol{\omega}_{\tau}$ estimates could capture this fault. The

Figure 7.2 The figure shows the model-error of CSE1 for an elliptic run in $u_t = 0.8m/s$, with four attempts of modelling the error, with the parameters seen in Table 7.1

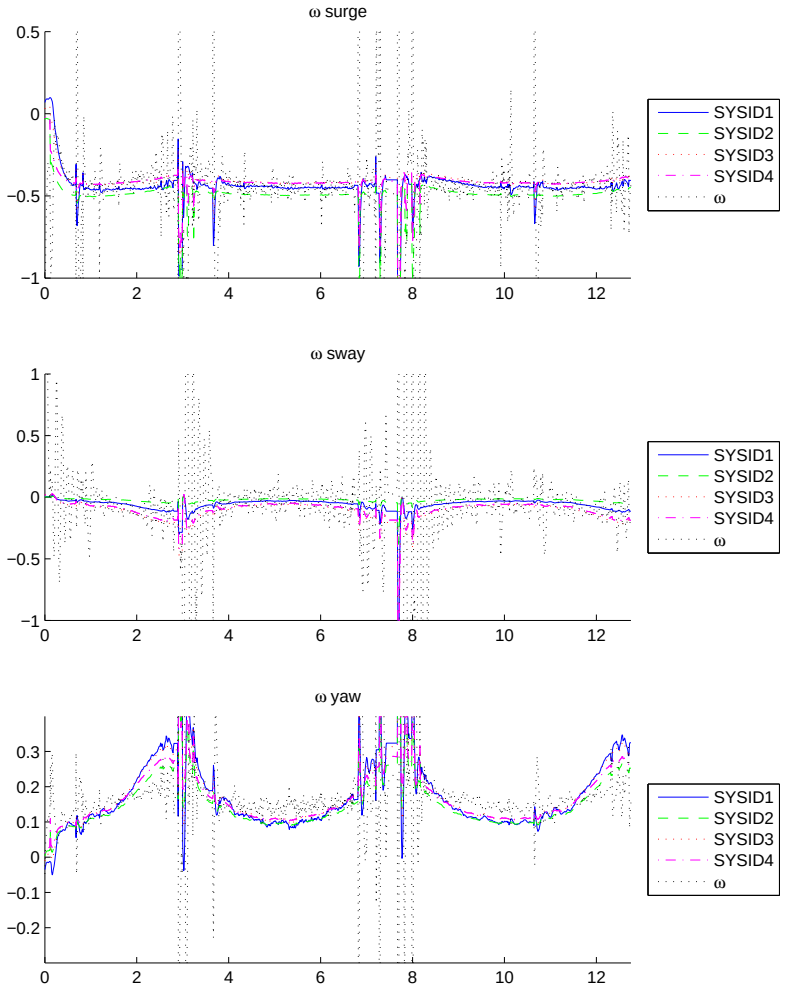
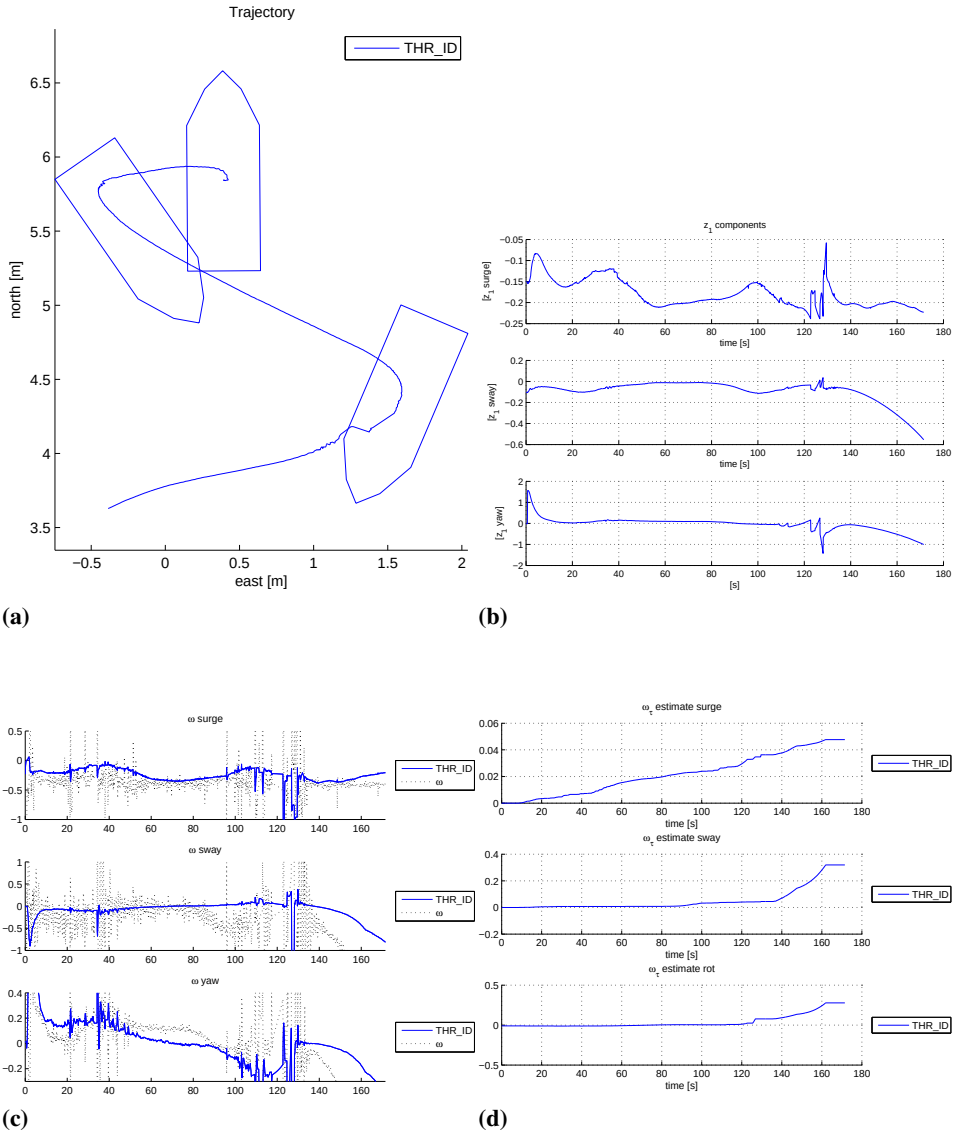


Table 7.1: The table shows the result of diffrent SysID of CSE1 moving in an elliptic trajectory. Although there are some differences in the result, it is apparent that there are some similarities, such as the domination of linear parameters.

Scenario	Phi parameters	ω_τ parameter
1: Elipse_BS	[-0.3, -6.5,-0.025, 0 0 0 0 -1.44, -2.7, 1.4, 8.2]	[0, 0, 0]
2: Elipse_CLE	[-0.30,-4.0,0.225,0,0.38,0,0.067,0.02,-1.7,-0.02,-0.23]	[0.36, 0, 0.23]
3: Elipse_BS	[-0.33,-4.6,0.27,-0.015,1.7,0.02,0.2,0.02,-2.3,-0.026,-0.26]	[0.11, 0, 0.18]
4: Elipse_BS	[-0.26,-3.5,0.2,-0.014,1.55,0.017,0.2,0.017,-1.8,-0.02,-0.2]	[0.3, 0, 0.31]

results can be seen in Figure 7.3. And it is clear that the ω_τ estimates in surge and sway, are affected by this fault, and this could be used in a fault detection. The idea could be expanded further for more complex thruster set-ups as having several azimuth thrusters, where these errors are more difficult to identify.

Figure 7.3 In this figure, the scenario of a failed bow thruster under an experiment is shown. The bow thruster failed due to low voltage in its battery, which happened after there was was a jump in the measurement, seen in figure (a). In figure (b) the z_1 error is seen, and we can see the ship starting to drift of η_t after 140 seconds has gone. The model-error can also be seen drifting. This is because as the ship drifts from z_1, an equal increase in τ is requested in sway and yaw, which results in the increase in model-error by the lack of the bow thrust. We see that this makes the estimates of ω_τ increase from close to 0 to $30 - 40\%$ in seconds.



7.3 Multivariate Analysis

As for the MVA, we want do predict $\boldsymbol{\omega}$, and as described in Section 3.2, the data was organized in different ways. The data for this analysis were the elliptic trajectory from the the basin tests seen in Chapter 6. Both PCA and PLS were used to analyse the data, and see which variables correlated with the error $\boldsymbol{\omega}$. What was discovered was that since $\dot{\boldsymbol{\nu}}$ had such high ratio between its noise and actual values, and in addition was multiplied with $\boldsymbol{M}$, the result was that $\dot{\boldsymbol{\nu}}$ overrode the other variables in the regression, which resulted in $\dot{\boldsymbol{\nu}}$ having to be kept outof the regression. In the end, the values that seemed relevant for the $\boldsymbol{\omega}$ prediction was $\boldsymbol{\nu}$ and $\boldsymbol{\tau}$, which gave the regression

$$\boldsymbol{Y} = [\omega_u,\ \omega_v,\ \omega_r] \tag{7.12}$$

while the $\boldsymbol{X}$ is dependent on how the PLS is moddeled

$$\boldsymbol{X} = [u,\ v,\ r,\ \tau_u,\ \tau_v,\ \tau_r,\ ] \tag{7.13}$$

or another proposed term was

$$\boldsymbol{X} = [u,\ v,\ r,\ \tau_u,\ \tau_v,\ \tau_r,\ |u|u,\ |u|v,\ |u|r,\ |v|v,\ |v|r,\ |r|v,\ |r|r] \tag{7.14}$$

At the end the shape of X depended on trial and error, and observing the validation of the regression coefficients. First the calibration had to be done. As the data gathered from the experiment, includes spikes, and irregularities, these had to be disqualified from the calibration set. First, the data is checked, by plotting the variables, in line and against each other, to see if there are some abnormalities. As can be seen from the plots of $\boldsymbol{\nu}$, seen in Figure 7.5a, there are several spikes in the data, that will have bad effect on the PLS regressio The variables were also plotted against each other, which can be seen in Figure 7.4, but no big abnormalities were detected. To get rid of data that seemed invalid, a PCA was done with the variables $\boldsymbol{\epsilon},\ \boldsymbol{\nu}$ and $\boldsymbol{\tau}$.

The samples detected as outliers by the hotelling's T^2 statistics and Q residuals were removed from the calibration set. The result can be seen by comparing figures 7.5a and 7.5b. We see that the PCA outlier removal get rid of the worst spikes, but also removes the most ugly part of the data. The calibration set is then found, and it is time to build a model from the calibration set using PLS regression. At first a PLS was done with all the variables, and cross terms. And it was seen that the most relevant variables were the $\boldsymbol{\nu}$ and $\boldsymbol{\tau}$ as well as the cross terms of these. The PLS regression of these variables is PLS1. The next regression, the cross terms of $\boldsymbol{\tau}$ were kept out, and is denoted PLS2. The validation of the regression factors are seen in Figure 7.7. From the regression validation, one can see which variables are statistically significant and important for the regression, and from this, the couplings $|u|v,\ |v|v,\ |v|r,\ |r|v$ were kept out of the further PLS regressions, this can also be verified by looking at the loadings of the PLS, as these variables are kept near the center of the correlation loading plot, which means that they are uncorrelated with the important latent variables in the model. In addition, one can see from the explained variance plot in Figure 7.6a, that there is only need for 3 factors to build the PLS model.

Figure 7.4 Plot of scatter plots of ω and ν against each other, from CSE1 with the four elliptic runs from Chapter 6, described in Section 6.3.5, beside each other. No apparent abnormalities as clusterings or faraway samples were detected.

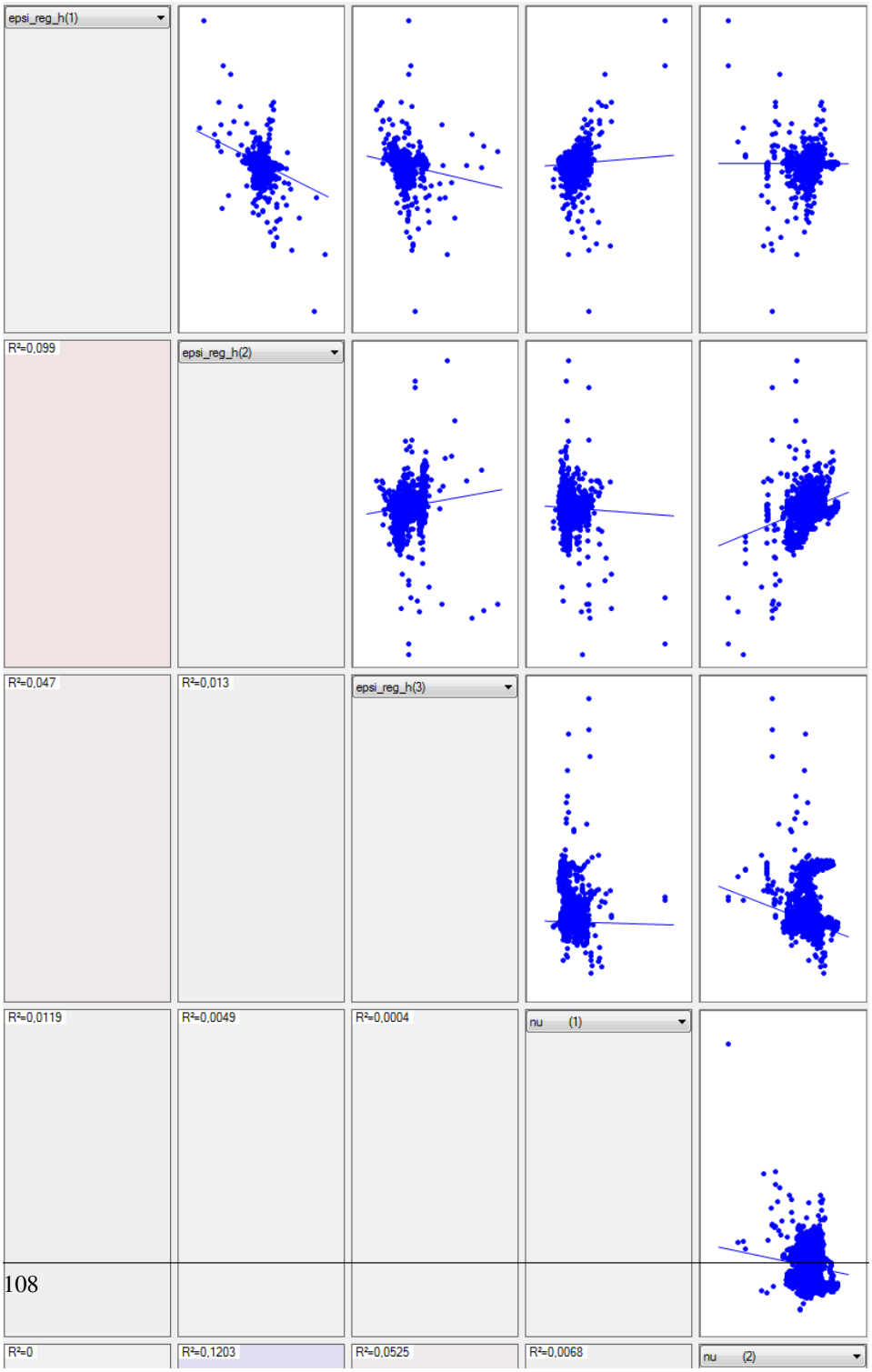
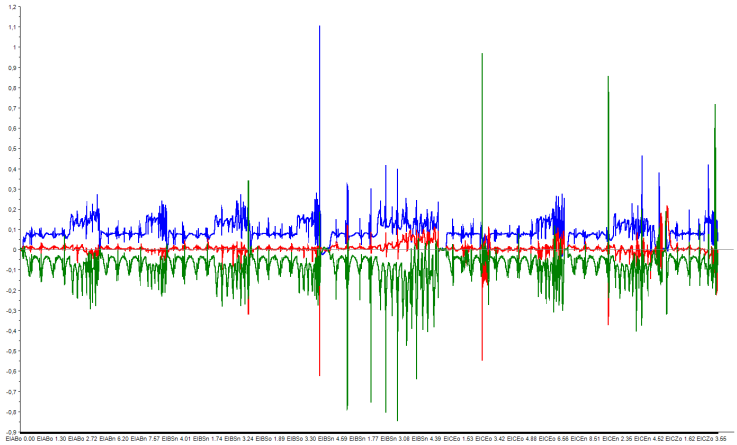
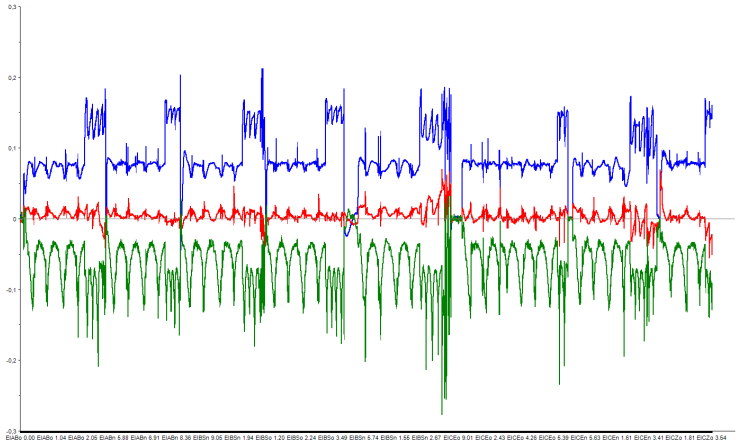


Figure 7.5 The plots show the subsequent ν samples after the four elliptic runs on CSE1 described in chapter 6, described in Section 6.3.5. The PCA was used to detect and remove outliers, where the results can be seen by comparing subfigure (a) and (b), where (a) is the ν samples before the PCA outlier removal, and (b) is the ν samples after the PCA outlier removal.



(a) Plot of the raw data of ν from CSE1 before the PCA was used to detect and remove outliers

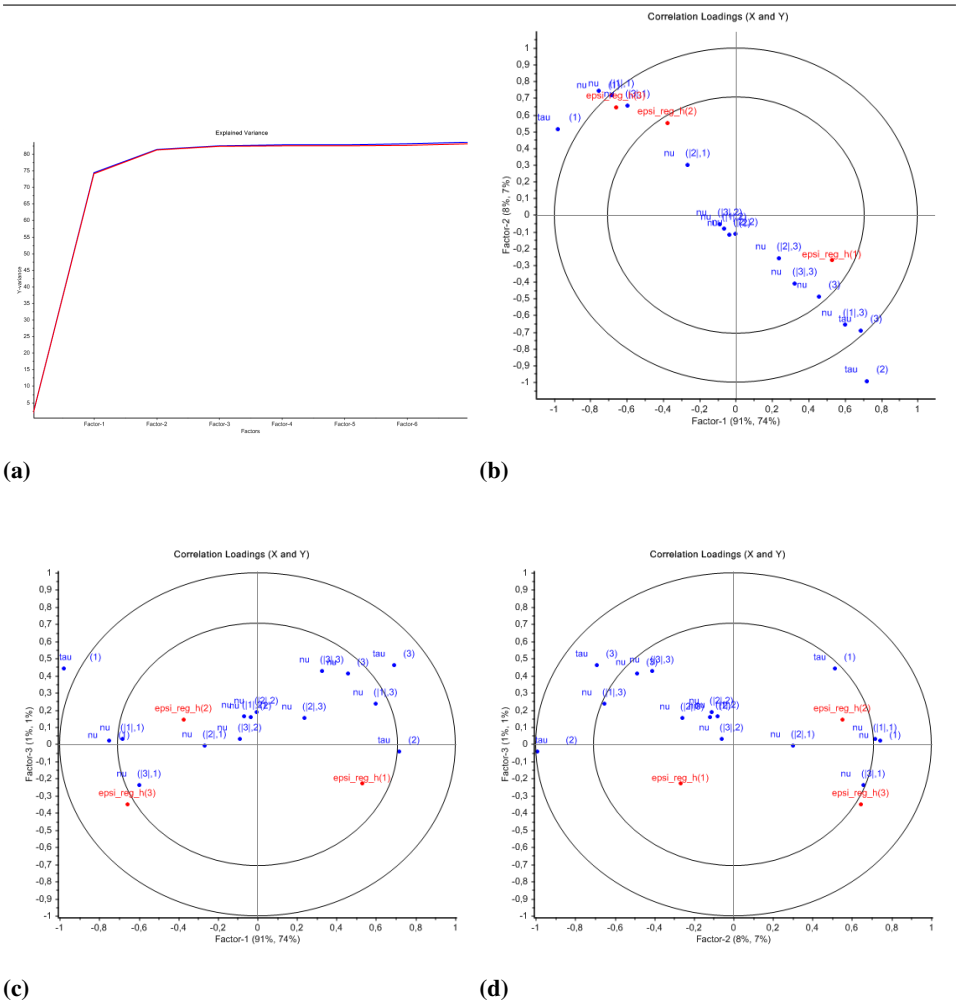


(b) Plot of the remaining data of ν from CSE1 after PCA was used to detect and remove outliers

The resulting PLS can be seen in Figure 7.8, and is denoted PLS3. It was also tried a PLS, where the different regressions were calculated separately, in addition, τ was kept out of this regression, to see if the error could be modelled by only hydrodynamic coeffi-

cients. The result can be seen in Figure 7.9. In this PLS regression, it can be seen that the model-error is mostly systematic in surge and yaw, and that the error in surge is closely uncorrelated with potential hydrodynamic forces.

Figure 7.6 The figure shows the PLS-regression of ω using the variables ν and τ and cross correlation of ν. This will be denoted PLS2. The plot (a) shows how well the model predicts the data, while (c)-(d) shows the correlation loadings for the three first factors.



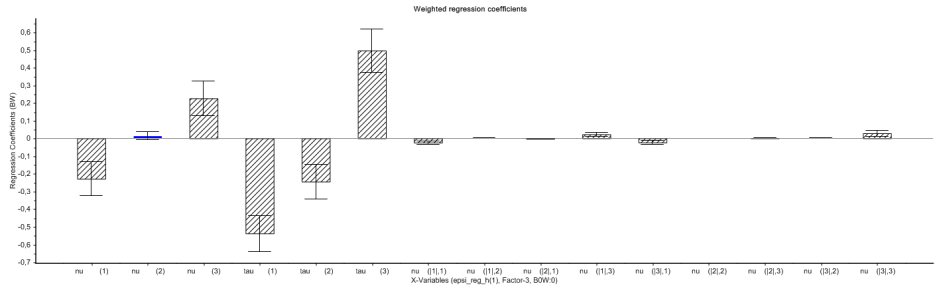
(a) **(b)**

(c) **(d)**

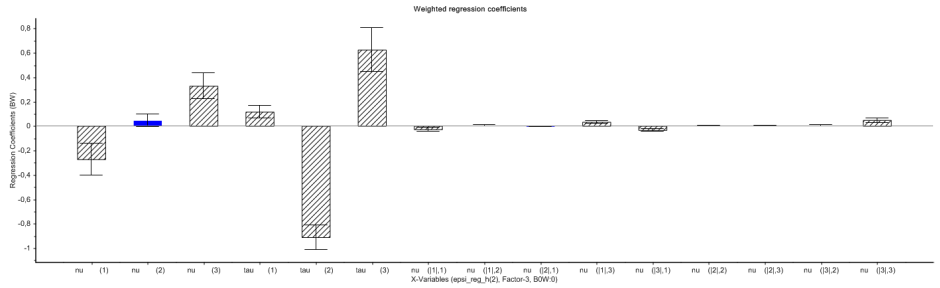
7.3.1 Partial Least Squares Validation

In this section, several models were presented. But the models also had to be validated. The validation of the PLS regression models were done by cross validation, as explained in Section 3.1.5, by dividing the data set eight subsequent parts. The results can bee seen

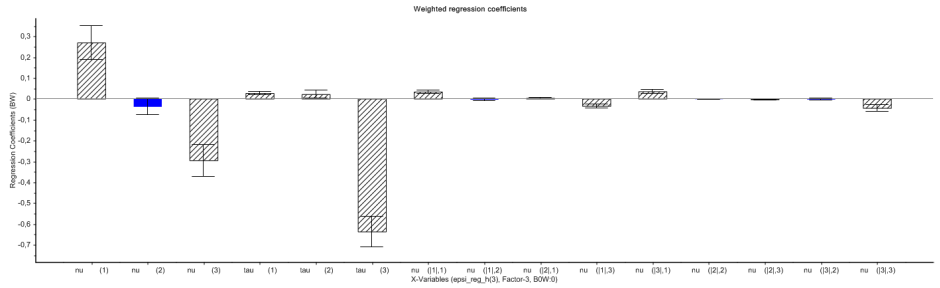
Figure 7.7 The figure shows the validation of the PLS-regression coeficents of $\boldsymbol{\omega}$ where (a)-(c) is the regression coefficients for $\epsilon(1) - \epsilon(3)$. So every bar represents the element of the $\boldsymbol{B}$ matrix produced by the PLS. There was done a cross validation with the data parted in 8 subsequent parts, where the standard deviation of the different coefficients are shown



(a)



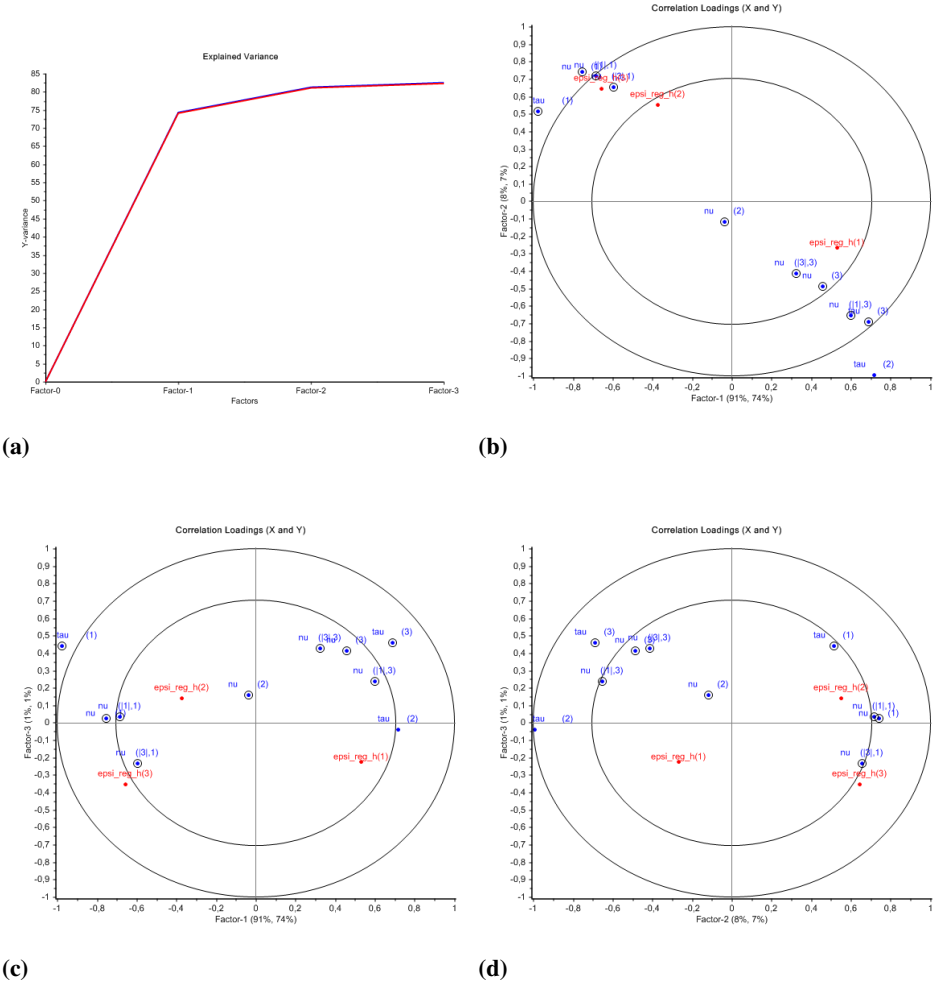
(b)



(c)

by examining the red line in the figures 7.6a, 7.8a and 7.9. The red line shows the Q^2 statistic, which is done during the cross validation, while the blue line is the R^2 statistic, both are explained in Section 3.1.4. The difference is most apparent in Figure 7.9b. Also by looking at Figure 7.7 the results of the cross validation of the regression coefficients

Figure 7.8 The figure shows the PLS-regression of $\boldsymbol{\omega}$ using the variables $\boldsymbol{\nu}$ and $\boldsymbol{\tau}$ and cross terms of $\boldsymbol{\nu}$, were the cross terms related to ν_v (nu(2)) is kept out of the PLS . This will be denoted PLS3. The plot (a) shows how well the model predicts the data, while (c)-(d) shows the correlation loadings for the three first factors.



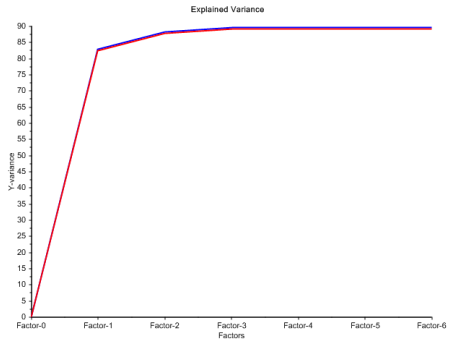
(a) **(b)**

(c) **(d)**

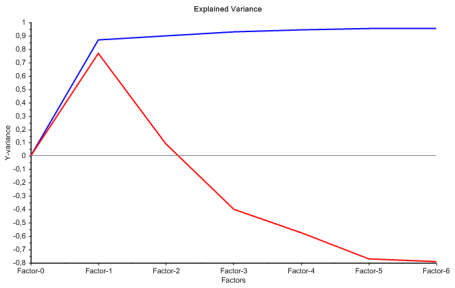
can be seen. We can see how the coefficients related to v (nu (2)), seemed statistically insignificant as their value were so close to zero, and some even changed sign. The cross terms related to v was therefore taken out in the subsequent PLS regressions.

The models were also tested against the experimental data, in the post proses simulator as in Section 7.2.4, and the result of this can be seen in Figure 7.10. The difference of the PLS models are summarized in Table 7.2. We can see that the models are able to predict the most important part of $\boldsymbol{\omega}$, although they predicts the model-error differently. Form PLS1 that has caught up the spike dynamics in sway, to PLS4 who neglect most of the

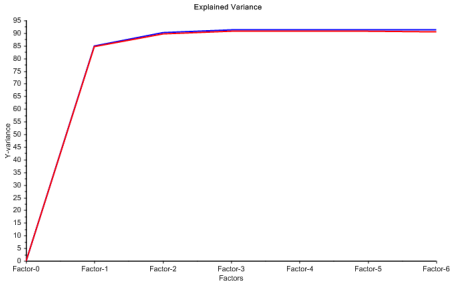
Figure 7.9 The figure shows the PLS-regression of the elements in $\boldsymbol{\omega}$ done separately, using the elements and cross terms of $\boldsymbol{\nu}$ related to the given direction. This will be denoted PLS4. The subfigure (a) is the explained variance of the PLS regression of $\boldsymbol{\omega}$ surge. The subfigure (b) is the explained variance of the PLS regression of $\boldsymbol{\omega}$ sway and subfigure (c) is the explained variance of the PLS regression of $\boldsymbol{\omega}$ yaw. The subfigure (d) is the loadings of the PLS-regression of $\epsilon(2)$, and it confirms that the $\epsilon(2)$ is uncorrelated with the hydrodynamic coefficients. A cross validation was also done, where the data was partitioned into eight sequential data sets.



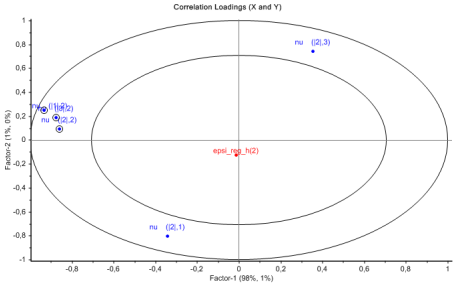
(a)



(b)



(c)



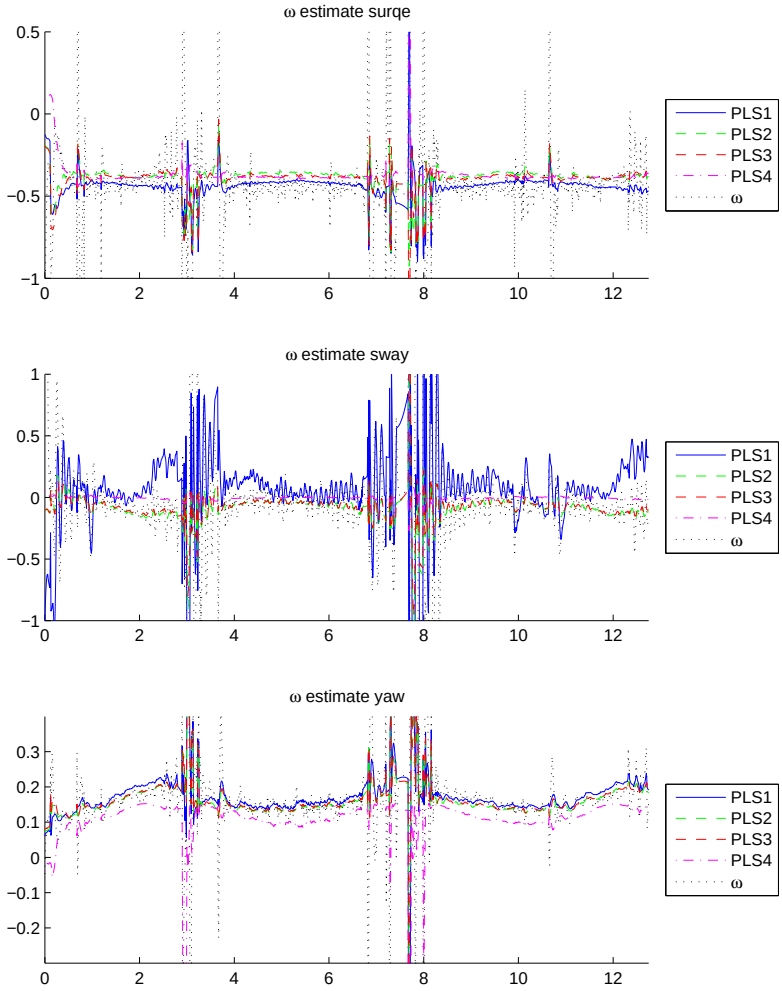
(d)

dynamics in sway motion. We also see that PLS4 has problems with reaching up to the model-error in yaw.

Table 7.2: The table summarize the properties of the different PLS regressions. The QE() represents the quadratic extension, and is described in Section 3.2

Name	X	factors	Figures	comments
PLS1	$\boldsymbol{\tau},\ \boldsymbol{\nu}, \mathrm{QE}(\boldsymbol{\tau},\ \boldsymbol{\nu})$	7	-	
PLS2	$\boldsymbol{\tau},\ \boldsymbol{\nu}, \mathrm{QE}(\boldsymbol{\nu})$	7	7.6	
PLS3	$\boldsymbol{\tau},\ \boldsymbol{\nu}, \mathrm{QE}(\boldsymbol{\nu})$	3	7.8	the cross terms related to v were kept out of the regression
PLS4	$\boldsymbol{\nu}, \mathrm{QE}(\boldsymbol{\nu})$	2-3	7.9	The PLS was done sepperatly for every direction in $\boldsymbol{\omega}$

Figure 7.10 The figure shows the model-error of CSE1 for an elliptic run in $u_t = 0.8m/s$, with four attempts of modelling the error with a model derived from the PLS method. The models are summarized in Table 7.2



7.4 Model-Error Simulations

To validate the models of $\boldsymbol{\omega}$ further, they were introduced into the simulators kinematics

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{\eta}} \\ \dot{\boldsymbol{\nu}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}(\psi)\boldsymbol{\nu} + \boldsymbol{\xi}^*(t) \\ \boldsymbol{M}^{*-1}(-\boldsymbol{C}^*(\boldsymbol{\nu}) - \boldsymbol{D}^*(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{\tau} + \boldsymbol{\omega}^*(\boldsymbol{\nu}, \boldsymbol{\tau})) \end{bmatrix} \tag{7.15}$$

where the $\boldsymbol{\xi}^*(t)$ also was added to simulate wave noise, and was implemented as in (Sørensen, 2013). The $\boldsymbol{\omega}$ was tested against the BS controller in an elliptic trajectory, to see if this produced the same position error and trajectory as seen in the experiments. BS was chosen because it lacks adaptation, thus the results will more likely be comparable with the experimental results. The comparison can be seen in Figure 7.11 and 7.12, clearly there are significant similarities in both the trajectories, position error $\mathbf{z_1}$ and the model-error $\boldsymbol{\omega}$. This is a good indication that the $\boldsymbol{\omega}$ from the test can be represented by the models of $\boldsymbol{\omega}$ produced in this chapter. Although. This was only demonstrated for this trajectory, and for the $\boldsymbol{\omega}$ model to be relevant for other trajectories, a similar SysID would have to be made. This could be done for the figure-eight trajectory as well.

Further we wanted to validate the models, and see how the CL controller would handle the $\boldsymbol{\omega}$ if adaptation worked properly. This was also to see how the models of $\boldsymbol{\omega}$ reacted to a simulation setting. The models of $\boldsymbol{\omega}$ was therefore implemented as in 7.15. The controller CLZ was used for these scenarios as it was deemed the most robust CL controller, and the $\boldsymbol{\omega}$ was SYSID1, SYSID4, PLS3 and PLS4. The results can be seen in figures 7.13, 7.14 and 7.15. We see that the CL controller is able to minimize the cross track error, but still is unable to eliminate the deviation in surge. This can be confirmed by looking at the error signal $\boldsymbol{z_1}$ in Figure 7.14a. The model-errors seams to react quite similarly in to the simulations. The SYSID1 was modelled with only hydrodynamics, while SYSID4 was modelled with an $\delta_{tau} = 0.3$ in both surge and yaw. The PLS4 was also only modelled with only hydrodynamics, but it has notable difference compared to the two other models, especially in yaw, but also in sway and surge. It should also be noted that the SYSMV3 was tested, and failed as can be seen in Figure 7.16. By examining the regression coefficient in Figure 7.7, the coefficients related to $\boldsymbol{\tau}$ are very high, so its reasonable that this could have an affect on the stability of the system. This raises important questions about MVA methods when they are implemented in control theory. How can stability easily be analysed when the PLS regression model is introduced to the system, and is it possible to modify the PLS so that it produces stable $\boldsymbol{\omega}$ models? It is also interesting to see from Figure 7.14 and 7.15, how the error in sway appeared random in $\boldsymbol{\omega}$, but the controllers are still able to adapt to $\boldsymbol{\omega}^*$ in an efficient manner.

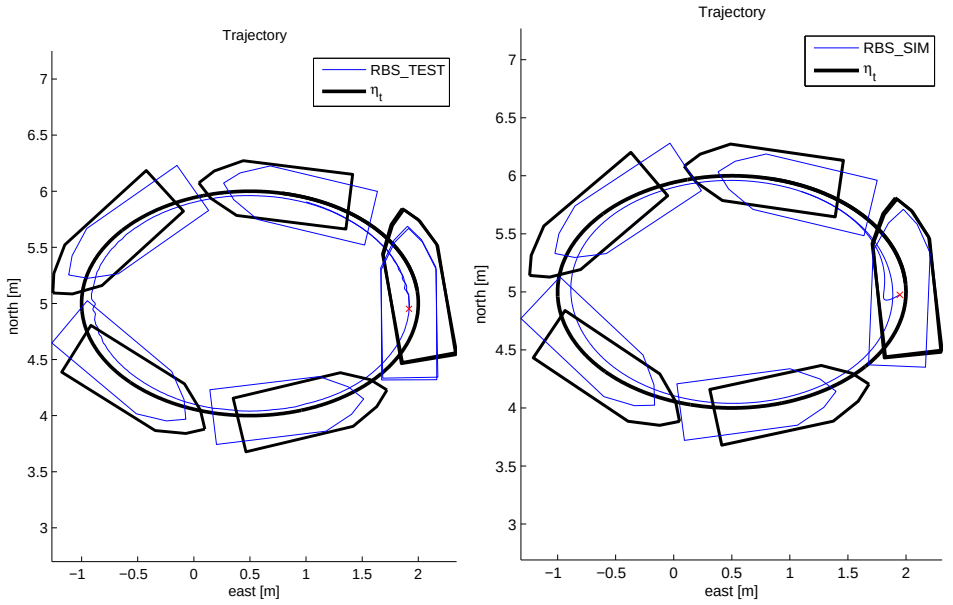
7.5 Improved Concurrent Learning Controller

It was apparent that the CL controller had to be modified to be able to handle the model-error $\boldsymbol{\omega}$ in surge. A natural choice was to modify the parametrization of the hydrodynamic regression matrix $\boldsymbol{\Phi}(\boldsymbol{\nu})$, so the regression matrix chosen was (7.1). The CL-WIN-ep controller was chosen for this last simulation, and the SYSID4 with δ_τ was the implemented model-error. The results can be seen in Figure 7.17. As expected, the modified controller

is able to adapt to the uncertainties, and as can be seen in Figure 7.18a the error signals goes towards zero. Only a little cross track error is kept in sway, but as can be seen in Figure 7.18b, the adaptation in sway is a bit off.

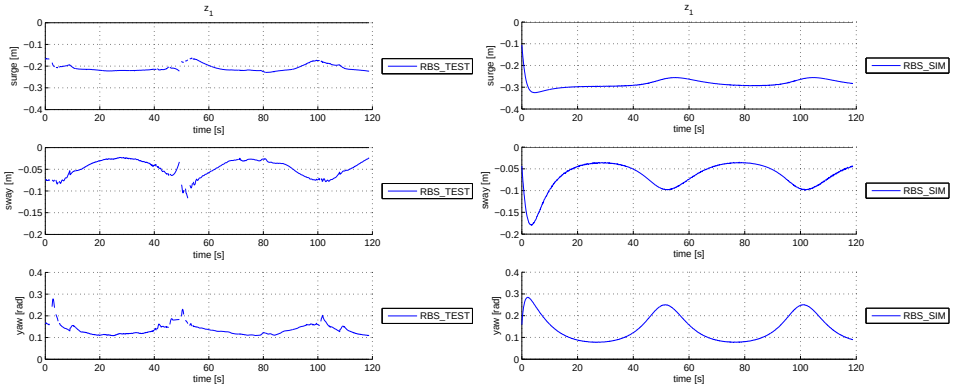
But to conclude, the most important lesson taken from this master thesis is that your controller can be as fancy and promising as you want, but if the basics has faults, the controller will probably fail or give disappointing results.

Figure 7.11 The figure shows the comparison of the Mc-lab test with the RBS controller moving in an elliptic trajectory, with the same scenario simulated with ω in the simulator, on the right.

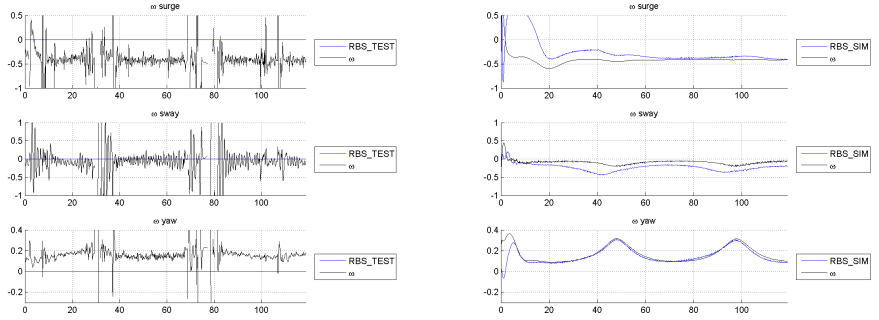


(a) **(b)**

Figure 7.12 The figure shows the comparison of the Mc-Lab test with the RBS controller moving in a elliptic trajectory, with the same scenario simulated with ω in the simulator, on the right. The sub-figures (a) and (b) are the z_1 components of the signals in the two scenarios, and the sub-figures (c) and (d) are the ω



(a) **(b)**



(c) **(d)**

Figure 7.13 The figure shows the simulation of CSE1 flowing an elliptic trajectory, with a desired speed of $u_t = 0.08m/s$. The different models of ω are implemented in the simulator for each scenario. The CLE controller from Section 6.4 was used.

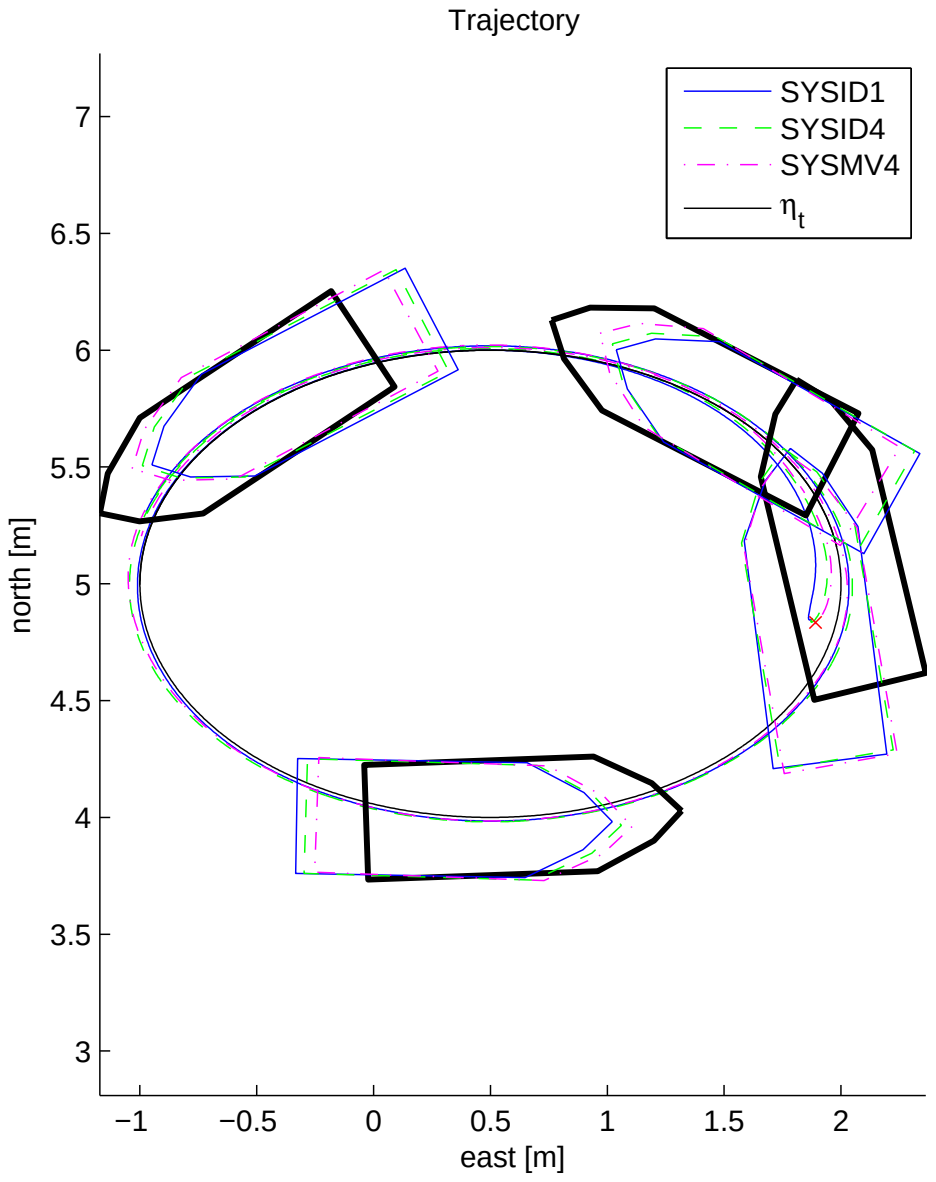
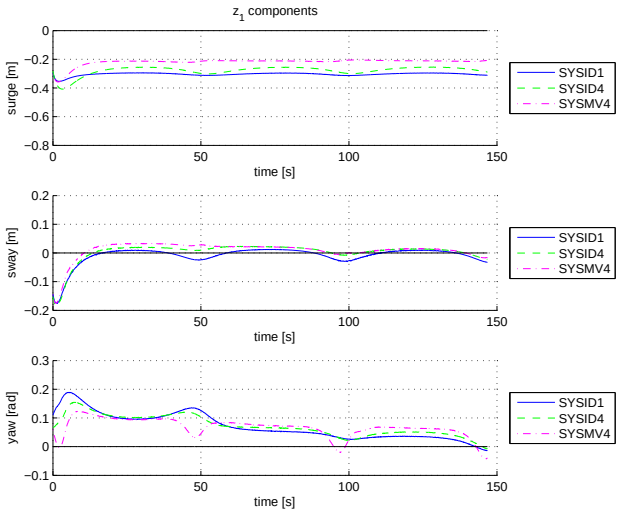
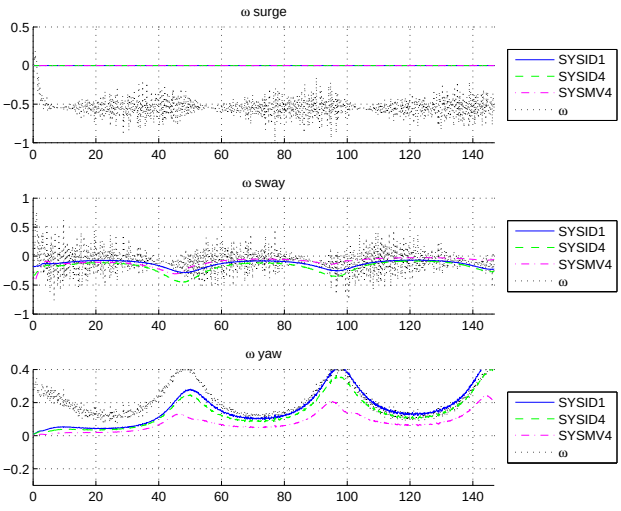


Figure 7.14 The figure shows the simulation of CSE1 flowing an elliptic trajectory, with a desired speed of $u_t = 0.08m/s$. The different models of $\boldsymbol{\omega}$ are implemented in the simulator for each scenario. The CLE controller from Section 6.4 was used. In subfigure (a) the components of the error-signal $\boldsymbol{z_1}$ is shown. In subfigure (b) the $\hat{\boldsymbol{\omega}}$ from the different controllers are shown.

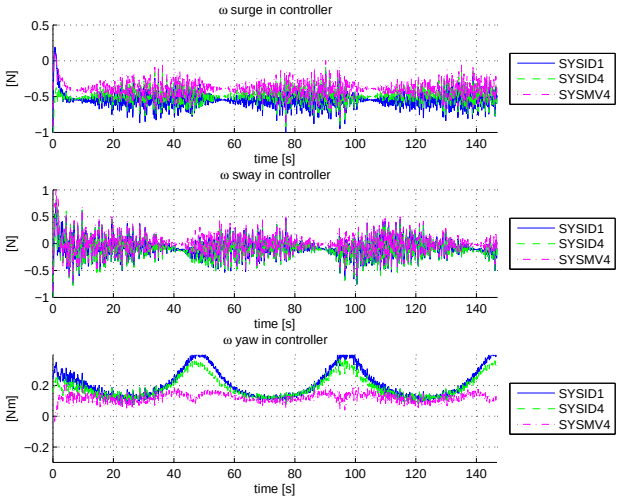


(a)

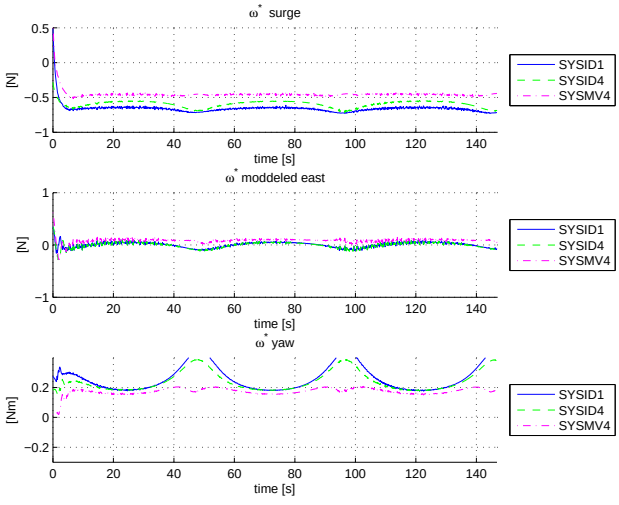


(b)

Figure 7.15 The figure shows the simulation of CSE1 flowing an elliptic trajectory, with a desired speed of $u_t = 0.08m/s$. The different models of $\boldsymbol{\omega}$ are implemented in the simulator for each scenario. The CLE controller from Section 6.4 was used. In subfigure (a) the $\boldsymbol{\omega}$ estimates made in the controllers for the different model-errors are shown. In subfigure (b) the real $\boldsymbol{\omega}^*$ from the different model-errors can be seen.

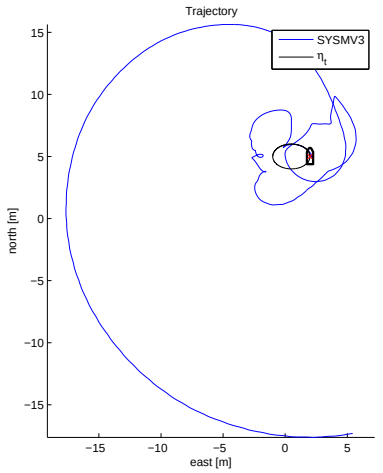


(a)

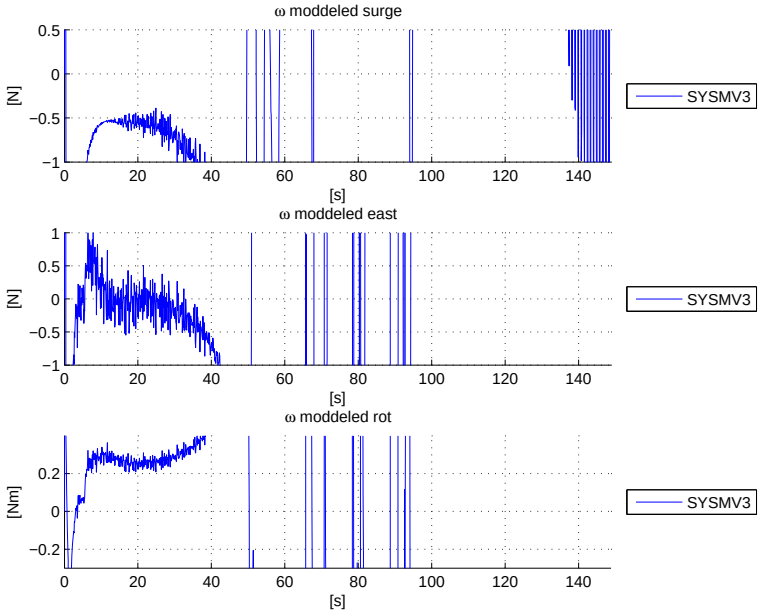


(b)

Figure 7.16 The figure shows the Simulation of CSE1, trying to follow an elliptic trajectory. The ω model SYSMV4 was implemented in the simulator, and it clearly had some stability issues.

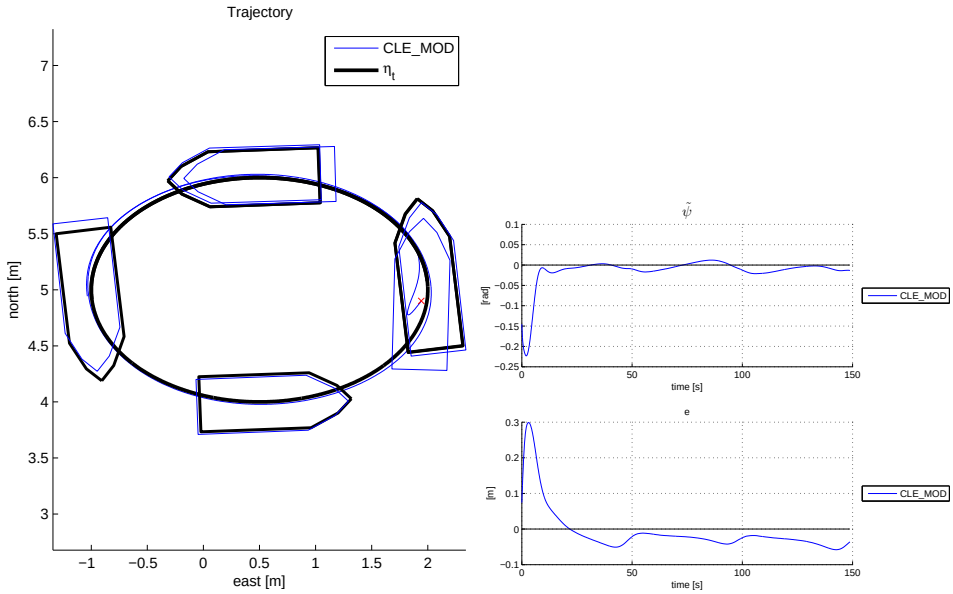


(a)

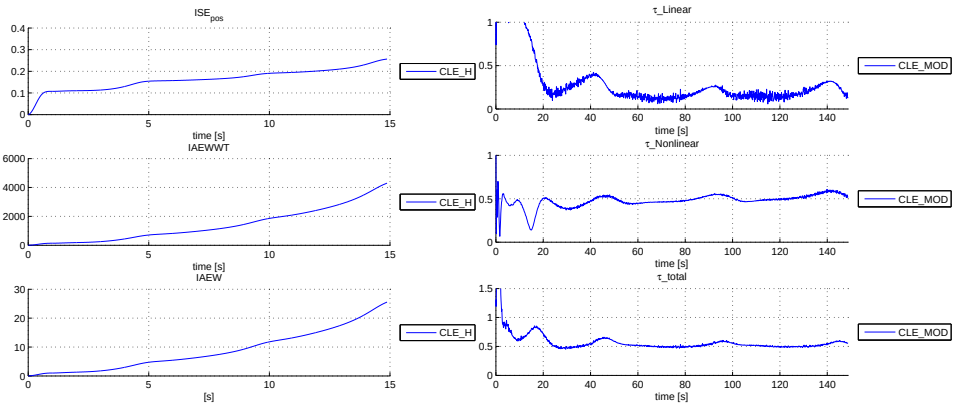


(b)

Figure 7.17 The figure shows the tests of CSE1 folowing an elliptic trajectory, with a desired speed of $u_t = 0.08m/s$, using the improved CL controllers from Section 7.5. In Figure (a) the trajectory of the ship can be seen, together with the target $\boldsymbol{\eta}_t$. In (b) the error in ψ is shown, together with the cross-track error described in Section 4.4 . In (c) the performance metrics presented in Section 4.4 of the controller is shown and in (d) the norm of the thrusters can be seen explained in Section 6.4.1

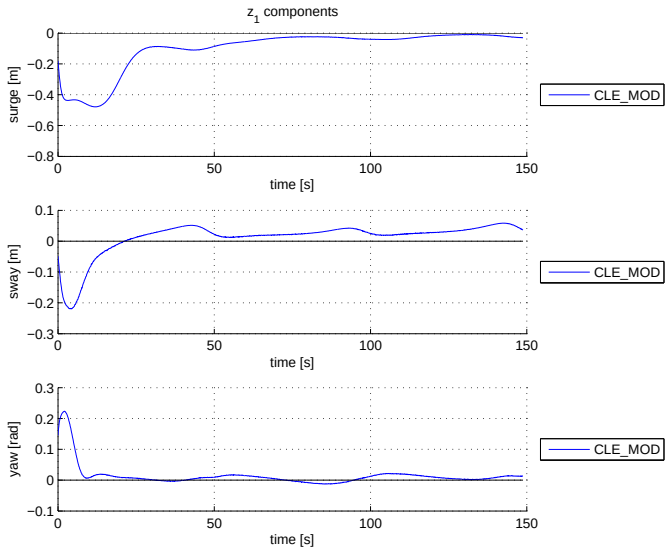


(a) **(b)**

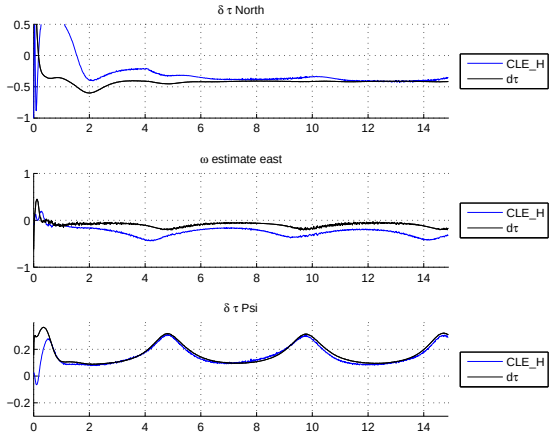


(c) **(d)**

Figure 7.18 The figure shows the tests of CSE1 folowing an elliptic trajectory, with a desired speed of $u_t = 0.08m/s$, using the improved CL controllers from Section 7.5. In Figure (a) the error-signal $\boldsymbol{z_1}$ is shown. In (b) $\boldsymbol{\omega}$ estimate of the model-error together with the controllers own estimate $\hat{\boldsymbol{\omega}}$



(a)



(b)

Chapter 8

Conclusion and Future Work

In this report, several variants of the nonlinear adaptive controller, named concurrent learning (CL) were developed for the model-scale ship Cybership Enterprise 1 (CSE1). Two storage algorithms were presented for the adaptation, namely the window (WIN) and singular value maximization (SVD), and two error-signals, z_2 and ϵ. This resulted in four alternative combinations for the CL controllers. The difference in the convergence and performance of the CL controllers was discussed and investigated. It was also demonstrated through simulations how the controllers' ability to estimate the model-error ω affected their performance. How this lead to smaller tracking error, and less need for linear feedback from the controllers, was also shown. The CL controllers using ϵ as error-signal seemed to be able estimate the model-error ω fastest, resulting in the best performance. Although, it should be noted that this was shown only during simulations, and ϵ proved to be a very difficult signal to estimate during the experiments.

The evaluations of the controllers were done through a comparative analysis where performance metrics were used. A comparative analysis was also done between the CL controllers and the nonlinear controllers adaptive backstepping (ABS) and backstepping (BS). The CL controllers proved to have the best performance, and it was clear that their performance in adapting the model-error ω was superior.

A control system and a simulator for CSE1 was also developed. The simulator was built in Simulink, to minimize the effort of transferring the control system from the simulator to CSE1. A HIL test was performed on the CompactRIO controller, similar to the one used on CSE1.

As preparation for the real life basin experiments with CSE1, in the Marine Cybernetics Lab (MC-lab), a comparison of potential estimators was also performed. These tests were performed on CSE1, where the estimators were tested in open loop. The derivative filter was chosen in the end, and a modification of the derivative filter was made so it could handle signal freeze and yaw wraparound. In addition, the thruster allocation on CSE1 was tuned by changing the point of attack for the thruster allocation to the center of mass. A guidance system was designed, although an error in the late implementation of the system made it produce a target η_t that was four times as fast as intended, which flawed the ex-

periments. In total 16 tests were performed on CSE1, with 4 controllers, 2 trajectories and 2 sets of speed. This resulted in a comparative analysis of the controllers, although they were difficult to compare because of the poor tracking. The CSE1 went too fast during the experiments, so the dynamic model was no longer accurate. The result was a substantial model-error ω, especially in surge. This combined with the controllers inability to adapt to disturbances in surge, resulted in the bad tracking. With the results of not being able to distinguish the performance of ABS and BS controller. Although a few weaknesses of the CL controllers were identified, and as implemented in this project, it was clear that the CL controllers were less robust than the BS and ABS controllers.

In parallel, methods from the field of multivariate analysis (MVA) had been investigated. Some of the MVA methods and techniques are presented in this project, along with multivariate statistics. Also a new way of organizing data for multivariate regression was shown, called continuous nominalization. The MVA was used during the post-experimental analysis, to clean the data, analyze the model-error and propose models of ω, using partial least squares (PLS) regression. The models were compared to the experimental data, and the results seemed promising. Further, the models were implemented on the simulator, with variable success. Some of the simulations produced similar results as the experiments when the BS controller was used, however others seemed to have problems with stability. A method for analyzing the stability of the models produced by the PLS regression will therefore be crucial for implementing them on a controller.

In the post-experimental analysis, regular methods were also used. An open loop simulation with the data from the experiments was performed. This was done to recoup lost data during the experiments, but also to further analyse the problems with the adaptations. A few weaknesses of the CL adaptation was identified, such as its sensitiveness to spikes. In addition, the parametrization of hydrodynamic uncertainties, resulting in the controllers' inability to adapt in surge. These problems were addressed, and solutions provided. The CL controller with ϵ as error signal, as well as a spike adaptation blocker, proved to be useful for system identification (SysID), and a system identification scheme for identifying ω was shown. In addition to model ω using hydrodynamic disturbances, the effects of inefficient thrusters were also introduced to the ω modelling. Several models using SysID were found. A scenario where SysID was used to identify thruster failure was also demonstrated. These models were also implemented in the simulator. It was shown that the CL controller would still have produced unsatisfactory tracking even with perfect adaptation, since it still was unable to adapt in surge. This was resolved by modifying the regression matrix Φ to also account for hydrodynamics in surge, and a simulation where the CL controller was able to tackle the new disturbances ω was presented.

I would like to emphasize that this project has been a great learning experience, and the main lesson to take from this master thesis is that the foundation of the controller, such as the parametrization of the model error, has to be satisfying, before more complex solutions can be introduced to help improve the controller.

8.1 Future Work

Future work would include the improvement of the CL controller from Chapter 7, and testing its performance in a new experiment. Also implementing an estimator providing satisfactory estimates of $\boldsymbol{\nu}$, $\dot{\boldsymbol{\nu}}$ and $\boldsymbol{\epsilon}$, especially for the stored data, would be a high priority for the further development of the CL controller. In addition, an implementation and stability analysis of a CL controller with thruster error adaptation would have been interesting. Further in the model-error analysis, a model-error of the eight-figured trajectory would have been interesting, especially comparing the MVA to SysID. Also doing an analysis, or finding a criteria for the disturbance model produced by the MVA methods, guaranteeing a stable system would be interesting. How the PLS could be modified so that the models produced would have stable properties is also an curious problem, and is essential if these PLS models are going to be applied to controllers in the future.

Bibliography

Benner, P., Gugercin, S., Willcox, K., Aug. 2013. A survey of model reduction methods for parametric systems. Preprint MPIMD/13-14, Max Planck Institute Magdeburg, available from `http://www.mpi-magdeburg.mpg.de/preprints/`.

Breivik, M., Fossen, T. I., 2004. Path following of straight lines and circles for marine surface vessels. In: In: Proceedings of the OCEANS '04. Kobe, Japan.

Chowdhary, G., Johnson, E., 2010. Concurrent learning for convergence in adaptive control without persistency of excitation. In: Proceedings of the 49th IEEE Conference on Decision and Control. Atlanta, GA, USA.

Chowdhary, G., Johnson, E., 2011a. A singular value maximizing data recording algorithm for concurrent learning. In: Proceedings of the 2011 American Control Conference. San Francisco, CA, USA.

Chowdhary, G., Johnson, E., 2011b. Theory and flight-test validation of a concurrent-learning adaptive controller. Journal of Guidance, Control, and Dynamics 34 (2), 592–607.

Chowdhary, G., Yucelen, T., Muhlegg, M., Johnson, E. N., 2012. Concurrent learning adaptive control of linear systems with exponentially convergent bounds. International Journal of Adaptive Control and Signal Processing 27 (4), 280301.

Fossen, T., Strand, J. P., 1999. Passive nonlinear observer design for ships using lyapunov methods: full-scale experiments with a supply vessel. Automatica 35 (1), 3–16.

Fossen, T. I., 2011. Handbook of marine craft hydrodynamics and motion control, John Wiley & Sons Ltd.

GitHub, 2016. CS_EnterpriseI github project. https://github.com/NTNU-MCS/CS_EnterpriseI_cRIO.

Kamalapurkar, R., Walters, P., Dixon, W., 2013. Concurrent learning-based approximate optimal regulation. In: Proceedings of 52nd IEEE Conference on Decision and Control. Firenze, Italia.

Koschorrek, P., Siebert, C., Haghani, A., Jeinsch, T., 2015. Dynamic positioning with active roll reduction using Voith Schneider. In: Proceedings of 10th IFAC Conference on Manoeuvring and Control of Marine Craft. Copenhagen, Denmark.

Lavretsky, E., Wise, K. A., 2012. Robust and adaptive control with aerospace applications, Springer.

Martens, H., 2015. Quantitative big data: where chemometrics can contribute. Journal of Chemometrics 29 (11), 563–581.

Martens, H., Næs, T., 1992. Multivariate Calibration, John Wiley & Sons.

Martens, H., Tndel, K., Tafintseva, V., Kohler, A., Plahte, E., Vik, J. O., Gjuvsland, A. B., Omholt, S. W., 2013a. Pls-based multivariate metamodeling of dynamic systems. In: Springer Proceedings in Mathematics and Statistics 56. New York, USA.

Martens, H., Tøndel, K., Tafintseva, V., Kohler, A., Plahte, E., Vik, J. O., Gjuvsland, A. B., Omholt, S. W., 2013b. Pls-based multivariate metamodeling of dynamic systems. In: Springer Proceedings in Mathematics & Statistics 56 56 (1), 3–30.

Mc-lab, NTNU, 2015. Handbook of marine hil simulation laboratory and marine cybernetics laboratory.
URL `http://folk.ntnu.no/andrdahl/handbook.pdf`

Sandved, F., 2015. Remote Control and Automatic Path-following for C/S Enterprise I and ROV Neptunus. Master thesis, Department of Marine technology, NTNU.

Skåtun, H. N., 2011. Development of a DP system for CS Enterprice I with Voith Schneider thrusters. Master thesis, Department of marine technology, NTNU.

Skjetne, R., Smogeli, Ø., Fossen, T. I., 2004. Modeling, identification, and adaptive maneuvering of Cybership II: A complete design with experiments. Modeling, Identifcation and Control 25 (1), 3–27.

Sørensen, A. J., 2013. Marine control systems, lecture notes.
URL `http://folk.ntnu.no/assor/publications/marcyb.pdf`

Sørensen, M., Breivik, M., 2015. Comparing nonlinear adaptive motion controllers for marine surface vessels. In: Proceedings of the 10th IFAC Conference on Manoeuvring and Control of Marine Craft. Copenhagen, Denmark.

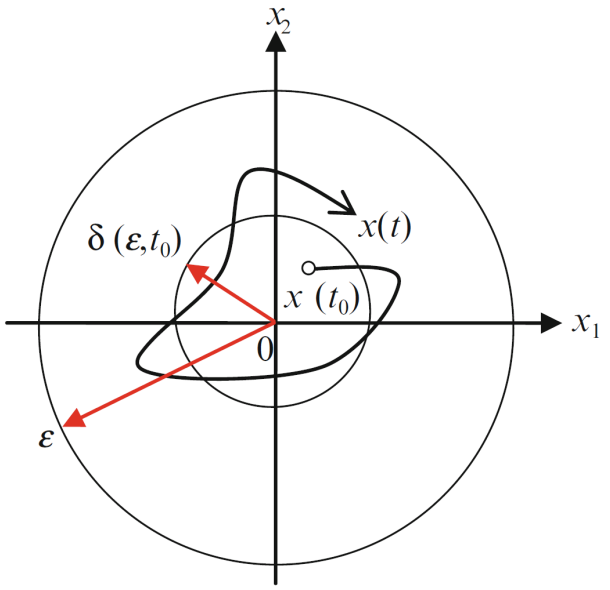
Appendix A

Stability and Stability Analysis

When talking about stabillity, we often talk about stability in the Lyapunov sense. And there are different definitions when it comes to stability, the most relevant for this project, that are taken from (Lavretsky and Wise, 2012) chapetr 8 are

$$\dot{\boldsymbol{x}} = f(t, \boldsymbol{x}) \tag{A.1}$$

Figure A.1 Geometric interpretation of Lyapunov stability for two-dimensional dynamics from (Lavretsky and Wise, 2012)



A.1 Definitions

Definition A.1.1 (Stability of Equilibrium in the Sense of Lyapunov)**.** *The equilibrium point $x^* = 0$ of the nonautonomous unforced dynamics* (A.1) *is stable if for any $e > 0$ and $t_0 \geq 0$ there exists $\delta(\varepsilon, t_0) \geq 0$ such that for all initial conditions $||x(t_0)|| < \delta$ and for all $t \geq t_0 \geq 0$, the corresponding system trajectories are bounded, as in $||x(t)|| < \varepsilon$. The equilibrium is uniformly stable if it is stable and δ does not depend on t_0. Finally, the equilibrium is unstable if it is not stable.*

Definition A.1.2 (Global Stability)**.** *The origin is globally stable if it is stable and* $\lim_{\varepsilon \to \infty} \delta(\varepsilon, t_0) = \infty$.

Definition A.1.3 (Asymptotic Stability)**.** *The equilibrium point $x^* = 0$ is asymptotically stable if it is stable and there exists a positive constant $c = c(t_0)$ such that $x(t) \to 0$ as $t \to \infty$, for all $||x(t_0)|| < c$.*

Definition A.1.4 (Uniform Asymptotic Stability)**.** *The equilibrium point $x^* = 0$ is uniformly asymptotically stable if it is uniformly stable and there exists a positive constant c independent of t_0 such that $x(t) \to 0$ as $t \to \infty$, for all $||x(t_0)|| < c$, uniformly in t_0*

Definition A.1.5 (Uniform Global Asymptotic Stability (UGAS))**.** *The origin is uniformly globally asymptotically stable if it is uniformly asymptotically stable and* $\lim_{\varepsilon \to \infty} \delta(\varepsilon, t_0) = \infty$.

It is important that the controllers are able to stabilize the system they are controlling, and brings it to desired states. To have a robust controller, the stabilization of the system should be as general as possible, and the goal is therefore to show that a controller makes the system UGAS or UAS and worst case stable. This stability is usually proven through Lyapunov stability, which will be used when the controllers are designed.

Theorem A.1.1 (Lyapunov's Direct Method for Assessing Uniform Stability of Nonautonomous Systems)**.** *Let $x^* = 0 \in \mathbb{R}^n$ be an equilibrium point for the nonautonomous dynamics* (A.1)*, whose initial conditions are drawn from a domain $D \subset \mathbb{R}^n$, with $x^* \in D$ and $t_0 = 0$. Suppose that on the domain D there exists a continuously differentiable locally positive-definite function $V(x) : D \to \mathbb{R}$, whose time derivative along the system trajectories is locally negative semidefinite:*

$$\dot{V}(x) = \nabla V(x) f(t, x) \leq 0 \tag{A.2}$$

for all $t \geq 0$ and for all $x \in D$. Then, the system equilibrium $x^* = 0$ is locally uniformly stable in the sense of Lyapunov. If in equation (A.2) *$\dot{V} < 0$ for all nonzero x and for all $t \geq 0$ (the time derivative along the system trajectories is locally negative definite), then the origin is locally uniformly asymptotically stable.* □

Theorem A.1.2. *Let $x^* = 0$ be an equilibrium point for equation* (A.1)*. Let $V(x) : \mathbb{R}^n \to \mathbb{R}$ be a radially unbounded Lyapunov function of the system. Then, the system equilibrium is globally uniformly asymptotically stable.* □

A usual example of the unbounded Lyapunov function is on the form $V(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{P} \boldsymbol{x}$ where $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{P} \geq 0$.

Also used to conclude convergence is Barbalat's lemma and LaSalle-Yoshizawa.

Lemma A.1.1 (Barbalat)**.** *Let $f : \mathcal{R} \to \mathcal{R}$ be a uniformly continuous function on $[0, \infty)$. Suppose that $\lim_{t\to\infty} \int_0^t f(\tau)d\tau$ exists and is finite. Then, $\lim_{t\to\infty} = 0$.* □

Theorem A.1.3 (LaSalle Yoshizawa)**.** *Starting anywhere in a domain D, all trajectories of the nonautonomous dynamics*

$$\dot{x} = f(t, x), x(t_0) = x_0, f(t, 0) = 0 \tag{A.3}$$

with a Lyapunov function satisfying

$$\dot{V}(x) = \nabla V(x) f(t, x) \leq -W(x) \leq 0 \tag{A.4}$$

will uniformly asymptotically approach the set E

$$E = x \in D : W(x) = 0 \tag{A.5}$$

□

A.2 Stability of Concurrent Learning Backstepping

Concurrent learning is an adaptive law based on the intuition that if the recorded data is sufficiently rich, i.e. there is a linear independence in the data, concurrent learning adaptation can be used to estimate true values without the need of persistency of excitation. However, Condition 1 from Chowdhary and Johnson (2010) needs to be fulfilled.

Condition 1: The recorded data has as many linearly independent elements as the dimension of regressor matrix $\mathbf{\Omega}(\boldsymbol{x}(t)) \in \mathbb{R}^{l\times m}$. That is if

$$\mathbf{Z} = [\mathbf{\Omega}(\boldsymbol{x}(t_1))^\top, \mathbf{\Omega}(\boldsymbol{x}(t_2))^\top, ..., \mathbf{\Omega}(\boldsymbol{x}(t_p))^\top] \tag{A.6}$$

then $rank(\mathbf{Z}) = m$.

If this condition is satisfied for $\mathbf{\Phi}$ and $\mathbf{R}^\top$, the adaptation laws are

$$\dot{\hat{\boldsymbol{\varphi}}} = \mathbf{\Gamma}_\varphi \mathbf{\Phi}^\top \mathbf{z}_2 + \sum_{j=1}^{p} \mathbf{\Gamma}_\phi \mathbf{\Phi}_j^\top \boldsymbol{\epsilon}_j \tag{A.7}$$

$$\dot{\hat{\boldsymbol{w}}}_n = \mathbf{\Gamma}_w \mathbf{R} \mathbf{z}_2 + \sum_{j=1}^{p} \mathbf{\Gamma}_w \mathbf{R}_j \boldsymbol{\epsilon}_j, \tag{A.8}$$

where $j \in \{1, 2, ...p\}$ denotes the index of a recorded data point $\boldsymbol{x}_j = [\boldsymbol{\eta}_j^\top, \boldsymbol{\nu}_j^\top]^\top$, $\mathbf{\Phi}_j$ and $\mathbf{R}_j$ are the regressor matrices evaluated at point $\boldsymbol{x}_j$, $\boldsymbol{\epsilon}$ is the approximation error, denoted $\boldsymbol{\epsilon} \triangleq \boldsymbol{y} - \hat{\boldsymbol{y}}$ and

$$\begin{aligned} \boldsymbol{y} &= \mathbf{\Phi}\boldsymbol{\varphi}^* + \mathbf{R}^\top \boldsymbol{w}_n^* \\ &= \boldsymbol{M}\dot{\boldsymbol{\nu}} - \boldsymbol{\tau} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{g}(\boldsymbol{\nu}) \end{aligned} \tag{A.9}$$

$$\hat{\boldsymbol{y}} = \mathbf{\Phi}\hat{\boldsymbol{\varphi}} + \mathbf{R}^\top \hat{\boldsymbol{w}}_n. \tag{A.10}$$

By applying the control law found in (4.1.6), that stabilizes V_2 in section 4.1.1, and combing it with the concurrent learning adaptation laws (A.7) and (A.8), the derivative of

$$V_3 = \tilde{\boldsymbol{\varphi}}^\top \mathbf{\Gamma}_{\boldsymbol{\varphi}}^{-1} \tilde{\boldsymbol{\varphi}} + \tilde{\boldsymbol{\omega}}_n^\top \mathbf{\Gamma}_{\boldsymbol{\omega_n}}^{-1} \tilde{\boldsymbol{\omega_n}} + V_2 \tag{A.11}$$

it becomes

$$\begin{aligned} \dot{V}_3 =& -\mathbf{z}_1^\top \mathbf{K}_1 \mathbf{z}_1 - \mathbf{z}_2^\top \mathbf{K}_2 \mathbf{z}_2 - \tilde{\boldsymbol{\varphi}}^\top \sum_{j=1}^{p} \mathbf{\Phi}_j^\top \boldsymbol{\epsilon}_j - \tilde{\boldsymbol{w}}_n^\top \sum_{j=1}^{p} \mathbf{R}_j \boldsymbol{\epsilon}_j \\ =& -\mathbf{z}_1^\top \mathbf{K}_1 \mathbf{z}_1 - \mathbf{z}_2^\top \mathbf{K}_2 \mathbf{z}_2 - \tilde{\boldsymbol{\varphi}}^\top \sum_{j=1}^{p} \mathbf{\Phi}_j^\top (\mathbf{\Phi}_j \tilde{\boldsymbol{\varphi}} + \mathbf{R}_j^\top \tilde{\boldsymbol{w}}_n) \\ & - \tilde{\boldsymbol{w}}_n^\top \sum_{j=1}^{p} \mathbf{R}_j (\mathbf{\Phi}_j \tilde{\boldsymbol{\varphi}} + \mathbf{R}_j^\top \tilde{\boldsymbol{w}}_n) \\ \leq& -\mathbf{z}_1^\top \mathbf{K}_1 \mathbf{z}_1 - \mathbf{z}_2^\top \mathbf{K}_2 \mathbf{z}_2 \leq 0 \quad \forall\, \mathbf{z}_1, \mathbf{z}_2, \end{aligned} \tag{A.12}$$

which can be shown to be UGAS by utilising Theorem A.1.2.