



**NTNU – Trondheim**  
Norwegian University of  
Science and Technology

# Optimal recursive thrust allocation applied for ROV Minerva

**Robert Kajanus**

Marine Technology

Submission date: December 2014

Supervisor: Roger Skjetne, IMT

Co-supervisor: Øyvind K. Kjerstad, IMT  
Mauro Candeloro, IMT

Norwegian University of Science and Technology  
Department of Marine Technology





## PROJECT DESCRIPTION SHEET

<b>Name of the candidate:</b>	Robert Kajanus
<b>Field of study:</b>	Marine control engineering
<b>Thesis title (Norwegian):</b>	Optimal rekursiv nullrom-basert thrust-allokering for ROV Minerva.
<b>Thesis title (English):</b>	Optimal recursive nullspace-based thrust allocation for ROV Minerva.

### Background

For dynamic positioning (DP) of marine vessels, the limiting capacity of stationkeeping in heavy environmental conditions is the thruster configuration and the maximum resultant forces and moment that can be produced for surge, sway, and yaw motions. The DP control law (or joystick control) provides a commanded net force/moment to be produced by the propulsion system to compensate the environmental loads. A thrust allocation algorithm distributes this net force/moment in an efficient manner to a commanded force and direction for each individual thruster based on their rated power, locations in the hull, thruster types, and individual constraints.

The Remotely Operated Vehicle (ROV) named Minerva is operated by the NTNU Applied Underwater Robotics Laboratory (AUR-Lab). It contains a 3D DP system, developed in-house by NTNU, that can be used for underwater stationkeeping and low-speed path-following. In sideways current there have been some issues in experiments, in situations when the thrusters saturate, the yaw moment is affected too early. Hence, to prioritize the yaw moment in the thrust allocation algorithm is essential. There are also some issues in regard to the moving center-of-gravity (CG) during operations with Minerva that affect the lever arms for producing yawing moment (the yawing moment from the front transversal thruster can switch sign depending on the CG moving in front of or behind this thruster). This is primarily a problem in the control law, but it should be considered in this project.

This project aims to combine the recursive nullspace-based thrust allocation with constrained optimal thrust allocation, to verify this by HIL simulation, and to experimentally test the allocation algorithm for ROV Minerva.

### Work description

- 1) Perform a literature review to provide background and relevant references on:
  - Optimal constrained thrust allocation and control allocation.
  - The Minerva dynamics, control system, and relevant DP/tracking/path-following control laws.
  - Recursive nullspace-based thrust allocation.Write a list with abbreviations and definitions of terms and concepts, explaining relevant concepts related to constrained thrust allocation and ROV control systems.
- 2) Consider Minerva as a design case and discuss its configuration, dynamic behavior, limitations and constraints, and other practical/operational issues. Write down its relevant dynamic control model and explain this in terms of presenting relevant coordinate frames, kinematics, and kinetics for the ROV. Present and discuss the simulation system for this ROV, both HIL and non-HIL. Present its thrust configuration and formulate the constrained thrust allocation problem.
- 3) Present a constrained optimal thrust allocation algorithm. Synthesize and implement it for ROV Minerva. Verify its effectiveness through relevant simulation cases.
- 4) Develop a constrained optimal recursive nullspace-based thrust allocation algorithm for Minerva. Simulate and test its effectiveness with comparison to the non-recursive (Item 3) and the original thrust allocation method for Minerva. Present and discuss the results.

- 5) Implement the constrained optimal recursive nullspace-based thrust allocation algorithm on the HIL simulation model for Minerva and demonstrate its effectiveness through relevant simulation cases.

**Tentatively:**

- 6) Implement the recursive nullspace-based thrust allocation algorithm on the real ROV Minerva and demonstrate its effectiveness on a test cruise.  
7) Discuss the control design problem with a time-varying CG for Minerva. Propose a strategy to compensate this problem.

**Guidelines**

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

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

The report shall be organized in a rational manner to give a clear exposition of results, assessments, and conclusions. The text should be brief and to the point, with a clear language. The report shall be written in English (preferably US) and contain the following elements: Abstract, acknowledgements, table of contents, main body, conclusions with recommendations for further work, list of symbols and acronyms, references, and (optionally) appendices. All figures, tables, and equations shall be numerated. The original contribution of the candidate and material taken from other sources shall be clearly identified. Work from other sources shall be properly acknowledged using quotations and a Harvard citation style (e.g. *natbib* Latex package). The work is expected to be conducted in an honest and ethical manner, without any sort of plagiarism and misconduct. Such practice is taken very seriously by the university and will have consequences. NTNU can use the results freely in research and teaching by proper referencing, unless otherwise agreed upon.

The thesis shall be submitted with two printed and electronic copies, to 1) the main supervisor and 2) the external examiner, each copy signed by the candidate. The final revised version of this thesis description must be included. The report must appear in a bound volume or a binder according to the NTNU standard template. Computer code and a PDF version of the report shall be included electronically.

**Start date:** 7 August, 2014                      **Due date:** As specified by the administration.

**Supervisor:** Prof. Roger Skjetne  
**Co-advisor(s):** PhD cand. Mauro Candeloro  
PhD cand. Øivind K. Kjerstad



*Roger Skjetne*

Digitally signed by Roger Skjetne  
Date: 2014.12.16 13:49:50 +01'00'

---

**Roger Skjetne**  
Supervisor



# Summary

This master thesis considers the thrust allocation problem of the Remotely Operated Vehicle Minerva, operated by the NTNU Applied Underwater Robotics Laboratory. The Minerva is reported to have some issues with the yaw capability when subject to current. Thus, motivating the testing of thrust allocation algorithms prioritizing yaw capability.

To get familiarized with the Minerva, and the Minerva control and simulation systems, the dynamical model of Minerva is presented. In order to present the dynamical model, relevant background information is given. Then, the dynamical behavior and limitations of the Minerva are discussed. Further, the thruster configuration, and current thrust allocation scheme is presented.

Three constrained thrust allocation algorithms are proposed; Recursive nullspace-based thrust allocation, optimal thrust allocation, and optimal recursive thrust allocation. The optimal recursive thrust allocation are the combination of the first two proposed algorithms. The recursive algorithms are motivated by the desire to ensure enough thruster capacity is available to produce the required yaw moment. After satisfying the yaw allocation, the remaining thrust capacity can be utilized to allocate the degrees of freedom in subsequent steps. The algorithms are then tested in both open-loop, and closed-loop condition, to ensure functionality.

A simulation case study to test the effectiveness, and compare the performance of the three proposed thrust allocation algorithms were carried out. From open-loop tests it was found that the thruster weighting matrix of the  $2^{nd}$  step of the recursive 3-step nullspace-based thrust allocation algorithm, are of great importance for the overall performance of the algorithm when applied for the Minerva. Leaving the  $2^{nd}$  step thruster weighting matrix as the identity matrix, caused the recursive 3-step algorithm to perform poorer than the pseudoinverse thrust allocation. From closed-loop tests, the proposed implementation of the optimal recursive thrust allocation algorithm is seen to be flawed. The optimal recursive algorithm allocated the thrusters such that the rotational direction of the thrusters were constantly changed back and forth, inducing instability in an otherwise stable system. Such utilization will drastically increase wear and tear of the thrusters. Further, it was found from testing that the optimal thrust allocation algorithm could achieve yaw priority by means of modifying the slack variables. By making the cost on the slack variable corresponding to yaw higher, in order 1000 times higher, than the other slack variables, yaw priority are achieved.

Finally, concluding remarks are given; The proposed optimal recursive thrust allocation algorithm as implemented for the Minerva is seen to be flawed. Therefore there is no point in HIL-testing of the algorithm, as proposed in this thesis. Lastly, recommendations for further work are presented.



# *Sammendrag*

Denne diplomoppgaven ser på thrust-allokeringsproblemet med den fjernstyrte undervannsroboten Minerva, som drives av NTNU "Applied Underwater Robotics Laboratory". For Minerva er det rapportert å være noen problemer med girevne når den er påvirket av strømkrefter. Dette motiverte testing av thrust-allokeringsalgoritmer med girevne som prioritet.

For å bli kjent med Minerva, og Minervas kontroll- og simuleringssystemer, er den dynamiske modellen av Minerva presentert. For å kunne presentere den dynamiske modellen er relevant bakgrunnsinformasjon gitt. Deretter blir den dynamiske oppførsel og begrensninger for Minerva diskutert, og thruster-konfigurasjonen, og den nåværende thrust-allokeringen, presentert.

Tre thrust-allokeringsalgoritmer med begrensning på allokert thrustkraft er foreslått: Rekursiv nullspacebasert thrustallokering, optimal thrustallokering, og optimalrekursiv thrustallokering. Den optimalrekursive thrustallokering er en kombinasjonen av de to første foreslåtte algoritmene. Valget av de rekursive algoritmene er motivert av ønsket om å sikre at nok thrusterkapasitet er tilgjengelig for å produsere det nødvendige giringsmomentet. Etter tilfredsstillende girallokering kan den gjenværende thrustkapasiteten utnyttes til å allokere kraft til de gjenværende frihetsgradene i påfølgende trinn. Algoritmene blir deretter testet både i åpen sløyfe- og lukket sløyfetilstand, for å sikre funksjonalitet.

Det ble utført en simuleringstudie for å teste effektiviteten og sammenligne resultatene fra de tre foreslåtte thrust-allokeringsalgoritmene. Fra åpen-sløyfetest ble det funnet at thrustervektmatrisen fra trinn 2 i den rekursive tre-trinns nullrombaserte thrust-allokeringsalgoritmen er av stor betydning for den totale ytelsen av algoritmen når det er anvendt for Minerva. Ved å la thrustervektingsmatrisen i trinn 2 forbli identitetsmatrisen, produserer den rekursive tre-trinnsalgoritmen dårligere resultat enn pseudoinvers thrustallokering. Fra lukket-sløyfetest viste det seg at den foreslåtte implementeringen av den optimalrekursive thrust-allokeringsalgoritmen var feil. Den optimalerekursive algoritmen allokerte thrusterene, slik at rotasjonsretningen for thrusterne stadig ble skiftet frem og tilbake, og dermed induserte ustabilitet i et ellers stabilt system. Slik utnyttelse vil drastisk øke slitasje av thrusterne. Videre ble det fra testingen funnet at den optimale thrust-allokeringsalgoritme kan oppnå girprioritet ved å modifisere slack-variablene. Ved å øke kostnaden på slack-variabelene tilhørende gir med en størrelse orden 1000 ganger større enn de andre slakk-variablene, oppnås girprioritet.

Til slutt er avsluttende merknader gitt: Den foreslåtte optimalrekursive thrust-allokeringsalgoritmen som ble implementert for Minerva viste seg å være feil. Derfor er det ikke noe poeng i videre HIL-testing av algoritmen, som foreslått i denne oppgaven. Endelig blir anbefalinger for videre arbeid presentert.



# *Acknowledgements*

This thesis is written during the fall of 2014, and marks the finalization of my studies for the degree of Master of Science in Marine Technology, from the Norwegian University of Science and Technology (NTNU).

My supervisor, Professor Roger Skjetne, presented the topic of this thesis to me. The work with the thesis, under his guidance, has been highly motivating and rewarding. His willingness to share his time and knowledge, and the ability to set new focus when needed, is greatly appreciated. For those reasons, I would like to give my first, and biggest thank you, to Professor Roger Skjetne.

My co-advisors, PhD-candidates Mauro Candeloro, and Øyvind K. Kjerstad, deserves a big thanks for their assistance and guidance during my work with this thesis. In particular; I would like to tank Mauro Candeloro for his help with the LabVIEW software, and sharing his detailed knowledge about the ROV Minerva. And Øyvind K. Kjerstad, for better understanding the recursive nullspace-based thrust allocation algorithm.

PhD-candidate Svenn Are Tuttoren deserves special thanks for providing invaluable feedback on my thesis, and suggestions for improvements. Further, for helping me gain a better understanding of the optimization problem. His help has been greatly appreciated.

Also PhD-candidate Daniel de A. Fernandes deserves big thanks for many enlightening discussions, and for his willingness to share his time and advice, whenever asked. His good mood and spirit are indeed infectious.

Further, I would like to thank my fellow students in office C1.062 @ MTS for the collaborative effort of keeping the spirit high during pressure periods, countless good laughs, and last but not least, our many good games of pool.

Last, but not least I would like to partially dedicate this thesis to my parents for always supporting me.

Trondheim, December 15, 2014



---

Robert Kajanus



# Contents

Summary	v
Sammendrag	vii
Acknowledgements	ix
List of Figures	xiii
Nomenclature	xix
<b>1 Introduction</b>	<b>1</b>
1.1 Background . . . . .	1
1.2 Objective . . . . .	1
1.3 Contributions and outline of the thesis . . . . .	1
1.4 Notation . . . . .	2
<b>2 Literature review</b>	<b>3</b>
2.1 The Minerva dynamic modeling, and control system . . . . .	3
2.2 Control allocation . . . . .	4
<b>3 Minerva design case</b>	<b>5</b>
3.1 Background . . . . .	5
3.2 Dynamical modeling of ROV Minerva . . . . .	8
3.3 Configuration, dynamical behavior and limitations . . . . .	10
3.4 ROV Minerva simulation systems . . . . .	13
3.5 Thruster configuration, and current thrust allocation scheme . . . . .	19
<b>4 Constrained recursive pseudoinverse-based thrust allocation</b>	<b>23</b>
4.1 Problem formulation . . . . .	23
4.2 Design of recursive nullspace-based thrust allocation for Minerva . . . . .	24
4.3 Simulation to show performance of algorithm . . . . .	28
<b>5 Constrained optimal thrust allocation</b>	<b>35</b>
5.1 Problem formulation . . . . .	35
5.2 Design of optimal thrust allocation for Minerva . . . . .	36
5.3 Simulation to show performance of algorithm . . . . .	37
<b>6 Constrained optimal recursive thrust allocation</b>	<b>45</b>
6.1 Problem formulation . . . . .	45
6.2 Design of optimal recursive thrust allocation for Minerva . . . . .	45
6.3 Simulation to show performance of algorithm . . . . .	48
<b>7 Simulation case study</b>	<b>55</b>

7.1	Comparison of thrust allocation algorithms . . . . .	55
7.2	Discussion . . . . .	65
<b>8</b>	<b>Conclusion and further work</b>	<b>69</b>
8.1	Conclusion . . . . .	69
8.2	Recommendations for further work . . . . .	70
<b>Bibliography</b>		<b>I</b>
<b>Appendices:</b>		<b>III</b>
A	Minerva software CD . . . . .	III
B	Simulation case 1; Pseudoinverse thrust allocation . . . . .	V
C	Simulation case 2; . . . . .	IX
C.1	Pseudoinverse thrust allocation . . . . .	X
C.2	Recursive 3-step nullspace-based thrust allocation . . . . .	XII
C.3	Weighted optimal thrust allocation . . . . .	XIV
C.4	Optimal recursive thrust allocation . . . . .	XVI
D	Recursive 1-step nullspace-based thrust allocation . . . . .	XXI



# List of Figures

3.1	Illustration of body and NED reference frames. Courtesy ROV sketch: Wordpress. . . . .	5
3.2	Illustration of 6 DOF motion in body-frame, {b}. Courtesy ROV sketch: Wordpress. . . . .	7
3.3	Illustration showing ROV Minerva’s sensor equipment. Picture courtesies: <b>RV Gunnerus</b> , Fredrik Skoglund; <b>IMU</b> , Xsens; <b>DVL</b> , Seatronics; <b>cRIO + LabVIEW logo</b> , NI; <b>ROV console</b> , iN-DepthSystems; <b>Switch</b> , WiringProducts; <b>PC</b> , Puzzledworld. . . . .	12
3.4	Illustration of the hardware structure for the ROV Minerva control system. Pictures with courtesy: <b>PC</b> , Puzzledworld; <b>cRIO + Logo</b> , NI; <b>Minerva</b> , TU. . . . .	13
3.5	Illustration of the hardware structure for the software simulator of ROV Minerva. Pictures with courtesy: <b>PC</b> , Puzzledworld; <b>Logo</b> , MATLAB®. . . . .	13
3.6	Illustration of the hardware structure for the HIL simulator for ROV Minerva. Pictures with courtesy: <b>PC</b> , Puzzledworld; <b>cRIO + Logo</b> , NI; <b>ROV sketch</b> , Wordpress. . . . .	14
3.7	Block diagram in SimuLink showing the top level for the software simulator of the ROV Minerva. . . . .	14
3.8	Illustrating the file structure of the LabVIEW project ”AURLab ROV Control System.lvproj”, showing the main VI, ”Njord Control System.vi” and the connection to the cRIO named ”Minerva-HILSim”. . . . .	17
3.9	Illustrating the file structure of the LabVIEW project ”Graphical User Interface for Control System.lvproj”, showing the main VI, ”Graphical User Interface - Frigg.vi”. . . . .	17
3.10	Illustrating the file structure of the LabVIEW project ”HILpro.lvproj”, showing the main VI, ”HIL sim sys.vi”. . . . .	17
3.11	Screenshot of a working LW project, before <b>Run</b> is pressed, indicated by solid white arrow. . . . .	18
3.12	Screenshot of a working LW project, after <b>Run</b> is pressed, indicated by solid black arrow. . . . .	18
3.13	Screenshot of the running GUI Frigg from the HIL simulation system. The map view in center of the GUI will show the ROV position, indicated by the solid green symbol, along with the desired position (red ROV trace) and the measured position (blue ROV trace). . . . .	18
3.14	Illustration of thruster layout, with positive force direction, for ROV Minerva . . . . .	19
3.15	Illustration of script testing set-up . . . . .	21
3.16	Illustration of MATLAB®/Simulink testing set-up . . . . .	21

4.1	1 <sup>st</sup> level of testing; Testing the 3-step recursive nullspace-based thrust allocation algorithm with a $\tau_{ramp-up}$ control vector. Showing the resulting body forces and moment. . . . .	29
4.2	1 <sup>st</sup> level of testing; Testing the 3-step recursive nullspace-based thrust allocation algorithm with a $\tau_{ramp-up}$ control vector. Showing the thruster responses in rpm. . . . .	30
4.3	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the weighted recursive 3-step nullspace-based thrust allocation algorithm in closed-loop. Showing the ROV position in the North-East plane. . . . .	31
4.4	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the weighted recursive 3-step nullspace-based thrust allocation algorithm in closed-loop. Showing the 6 DOF response of the ROV. . . . .	32
4.5	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the weighted recursive 3-step nullspace-based thrust allocation algorithm in closed-loop. Showing the resulting body forces and moment. . . . .	33
4.6	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the weighted recursive 3-step nullspace-based thrust allocation algorithm in closed-loop. Showing the allocated thruster revolutions. . . . .	33
5.1	1 <sup>st</sup> level of testing; Testing the optimal thrust allocation algorithm with a $\tau_{ramp-up}$ control vector. Showing the resulting body forces and moment. . . . .	38
5.2	1 <sup>st</sup> level of testing; Testing the optimal thrust allocation algorithm with a $\tau_{ramp-up}$ control vector. Showing the thrusters rpm's. . . . .	39
5.3	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the optimal weighted thrust allocation algorithm in closed-loop. Showing the ROV position in the North-East plane. . . . .	40
5.4	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the optimal weighted thrust allocation algorithm in closed-loop. Showing the 6 DOF response of the ROV. . . . .	41
5.5	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the optimal weighted thrust allocation algorithm in closed-loop. Showing the resulting body forces and moment. . . . .	42
5.6	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the optimal weighted thrust allocation algorithm in closed-loop. Showing the allocated thruster revolutions. . . . .	42
6.1	1 <sup>st</sup> level of testing; Testing the optimal recursive thrust allocation algorithm with a $\tau_c$ ramp-up control vector. Showing the resulting body forces and moment. . . . .	49
6.2	1 <sup>st</sup> level of testing; Testing the optimal recursive thrust allocation algorithm with a $\tau_c$ ramp-up control vector. Showing the thrusters rpm's. . . . .	50
6.3	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the optimal recursive thrust allocation algorithm in closed-loop. Showing the ROV position in the North-East plane. . . . .	51

6.4	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the optimal recursive thrust allocation algorithm in closed-loop. Showing the 6 DOF response of the ROV. . . . .	52
6.5	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the optimal recursive thrust allocation algorithm in closed-loop. Showing the resulting body forces and moment. . . . .	53
6.6	2 <sup>nd</sup> level of testing; Running the MATLAB <sup>®</sup> /Simulink simulator with the optimal recursive thrust allocation algorithm in closed-loop. Showing the allocated thruster revolutions. . . . .	53
7.1	Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Open-loop testing with a $\tau_{ramp-up}$ . Showing the resulting body forces and moment. . . . .	56
7.2	Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Open-loop testing with a $\tau_{ramp-up}$ . Showing the allocated thruster revolution. . . . .	56
7.3	Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Case 1, DP in set-point. Comparing the instantaneous forces errors, $\tau_{error}$ . . . . .	58
7.4	Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Case 1, DP in set-point. Comparing cumulative position errors, $\eta_{error}$ . . . . .	59
7.5	Case study; Case 2, change of pose. Showing the resulting body forces and moment, running the system with the pseudoinverse thrust allocation. . . . .	61
7.6	Case study; Case 2, change of pose. Showing the resulting body forces and moment, running the system with the 3-step recursive thrust allocation. . . . .	61
7.7	Case study; Case 2, change of pose. Showing the resulting body forces and moment, running the system with the weighted optimal thrust allocation. . . . .	62
7.8	Case study; Case 2, change of pose. Showing the resulting body forces and moment, running the system with the optimal recursive thrust allocation. . . . .	62
7.9	Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Case 2, change of pose. Comparing the instantaneous forces errors, $\tau_{error}$ . . . . .	63
7.10	Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Case 2, change of pose. Comparing cumulative position errors, $\eta_{error}$ . . . . .	64
B.1	Case 1, DP in set-point. Running the system with the pseudoinverse thrust allocation. Showing the ROV position in the North-Easth plane. . . . .	VI
B.2	Case 1, DP in set-point. Running the system with the pseudoinverse thrust allocation. Showing the 6 DOF response of the ROV. . . . .	VI
B.3	Case 1, DP in set-point. Running the system with the pseudoinverse thrust allocation. Showing the resulting body forces and moment. . . . .	VII

B.4	Case 1, DP in set-point. Running the system with the pseudoinverse thrust allocation. Showing allocated thruster revolutions. . . . .	VII
C.1	Case 2, change in pose; Running the system with the pseudoinverse thrust allocation. Showing the ROV position in the North-East plane. . . . .	X
C.2	Case 2, change in pose; Running the system with the pseudoinverse thrust allocation. Showing the 6 DOF response of the ROV. . . . .	X
C.3	Case 2, change in pose; Running the system with the pseudoinverse thrust allocation. Showing the resulting body forces and moment. . . . .	XI
C.4	Case 2, change in pose; Running the system with the pseudoinverse thrust allocation. Showing the allocated thruster revolution. . . . .	XI
C.5	Case 2, change in pose; Running the system with the recursive 3-step thrust allocation. Showing the ROV position in the North-East plane. . . . .	XII
C.6	Case 2, change in pose; Running the system with the recursive 3-step thrust allocation. Showing the 6 DOF response of the ROV. . . . .	XII
C.7	Case 2, change in pose; Running the system with the recursive 3-step thrust allocation. Showing the resulting body forces and moment. . . . .	XIII
C.8	Case 2, change in pose; Running the system with the recursive 3-step thrust allocation. Showing the allocated thruster revolution. . . . .	XIII
C.9	Case 2, change in pose; Running the system with the weighted optimal thrust allocation. Showing the ROV position in the North-East plane. . . . .	XIV
C.10	Case 2, change in pose; Running the system with the optimal thrust allocation. Showing the 6 DOF response of the ROV. . . . .	XIV
C.11	Case 2, change in pose; Running the system with the optimal thrust allocation. Showing the resulting body forces and moment. . . . .	XV
C.12	Case 2, change in pose; Running the system with the optimal thrust allocation. Showing the allocated thruster revolution. . . . .	XV
C.13	Case 2, change in pose; Running the system with the optimal recursive thrust allocation. Showing the ROV position in the North-East plane. . . . .	XVI
C.14	Case 2, change in pose; Running the system with the optimal recursive thrust allocation. Showing the 6 DOF response of the ROV. . . . .	XVI
C.15	Case 2, change in pose; Running the system with the Optimal recursive thrust allocation. Showing the resulting body forces and moment. . . . .	XVII
C.16	Case 2, change in pose; Running the system with the Optimal recursive thrust allocation. Showing the allocated thruster revolution. . . . .	XVII
C.17	Case 2, change in pose, reduces current velocity; Running the system with the optimal recursive thrust allocation. Showing the ROV position in the North-East plane. . . . .	XVIII
C.18	Case 2, change in pose, reduces current velocity; Running the system with the optimal recursive thrust allocation. Showing the 6 DOF response of the ROV. . . . .	XVIII
C.19	Case 2, change in pose, reduces current velocity; Running the system with the Optimal recursive thrust allocation. Showing the resulting body forces and moment. . . . .	XIX

C.20 Case 2, change in pose, reduces current velocity; Running the system with the Optimal recursive thrust allocation. Showing the allocated thruster revolution. . . . .XIX



# *Nomenclature*

## **Acronyms & Abbreviations:**

CG	Center of Gravity.
CO	Center of Origin.
cRIO	CompactRIO (Compact Reconfigurable Input-Output module) from National Instruments.
CS	Configuration space.
CM	Control Model.
ctrl	Ctrl-key on the keyboard.
DOF	Degrees of Freedom.
DVL	Dopler Velocity Log.
DP	Dynamic Positioning.
FPGA	Field Programmable Gate Array.
GUI	Graphical User Interface.
HIL	Hardware In-the-Loop.
HiPAP	High Precision Acoustic Positioning.
IMU	Inertial Measurement Unit.
NaviPac	Integrated Navigation and Data Acquisition software package.
IP	Internett Protokoll.
LW	LabVIEW.
LQ	Linear Quadratic.
MSS	Marine Systems Simulator.
NI	National Instruments.
NED	North East Down reference frame, also denoted $\{n\}$ .
PM	Process Model.
PID	Proportional Integral Derivative.
QP	Quadratic Programming; an optimization problem with quadratic objective function (J), and linear constraints.

ROV	Remotly Operated Vehicle.
RPM	Revolutions Per Minute.
RB	Rigid-Body.
SVD	Single Value Decomposition. In this context a method for inverting a badly scaled matrix.
TA	Thrust Allocation.
VI	Virtual Instrument; used whitin the NI LabVIEW software.
WP	Way Point.
WS	Workspace.
ZOH	Zero-Order-Hold.

### Terms/Names:

AUR-lab	Applied Underwater Robotics Laboratory.
Body-frame	Body-fixed reference frame, also denoted {b}.
Njord	The Control System for ROV Minerva.
Frigg	The Graphical User Interface (GUI) for ROV Minerva.
Altitude	The height above ground of an object.
Attitude	The orientation of an object.
Pose	The position and attitude of an object.

### Greek Symbols:

$\alpha$	Azimuth angle; The fixed angles the thrusters of the ROV, rotated relative the body-frame.
$\Phi$	Combined weighting matrix for slack variables and thruster usage.
$\Theta_{nb}$	Euler angels.
$\tau$	Generalized forces and moments.
$\nu$	Generalized velocities.
$\psi$	Heading angle.
$\eta$	Pose of the ROV.

### Lower Case Symbols:

$b$	Bias vector.
$r$	Moment arm; gives the thruster location relative the ROV's CO.



$g$	Restoring forces vector.
$s$	Slack variables.
$f$	Thruster forces.
$n$	Thruster revolutions.
$t$	Time.

**Upper Case Symbols:**

$C$	Coriolis and centripetal rotation matrix.
$D$	Damping matrix.
$Z$	Heave force.
$M$	Inertia matrix.
$J$	Objective function; used in the QP problem.
$X$	Surge force.
$Y$	Sway force.
$B$	Thruster configuration matrix.
$T_{\Theta}()$	Transformation matrix for angular velocities.
$R_b^n()$	Transformation matrix for linear velocities
$Q$	Weighting matrix for slack variables.
$W$	Weighting matrix for thruster usage.
$N$	Yaw moment.



# *Introduction*

## 1.1 Background

As stated in the thesis assignment, *”for the dynamic positioning (DP) of marine vessels, the limiting capacity of stationkeeping in heavy environmental conditions is the thruster configuration and the maximum resultant force and moment that can be produced for surge, sway and yaw motions. The DP control law provides a commanded net force/moment to be produced by the propulsion system to compensate the environmental loads. A thrust allocation algorithm distributes this net force/moment in an efficient manner to a commanded force and direction for each individual thruster based on their rated power, location in the hull, thruster type, and individual constraints”*.

Further, *”the Remotely Operated Vehicle (ROV) named Minerva is operated by the NTNU Applied Underwater Robotics Laboratory (AUR-lab). It contains a 3D DP system, developed in-house by NTNU, that can be used for underwater stationkeeping and low-speed path-following. In sideways current there have been some issues in experiments, in situations when the thrusters saturate, the yaw moment are affected too early”*. Thus, a constrained thrust allocation algorithm, prioritizing the yaw moment, is thought to be beneficial to the ROV Minerva.

## 1.2 Objective

The objective of this thesis will be to propose a constrained thrust allocation algorithm for the ROV Minerva, prioritizing the yaw moment. This is thought achieved by combining the recursive nullspace-based thrust allocation, with an optimal (QP) thrust allocation, into an optimal constrained recursive thrust allocation algorithm. The proposed thrust allocation algorithm is then to be tested. For each successful level of testing, the level of testing will be increased. Tentatively, performing full-scale tests on the ROV Minerva.

## 1.3 Contributions and outline of the thesis

**Chapter 2** presents references on the dynamic modeling of the ROV Minerva, and the developed control systems running it. Further, references on DP, tracking, and path-following control laws are presented. Lastly references on control allocation

are presented, with particular focus on thrust allocation. Of particular interest to this thesis are the recursive thrust allocation, and the optimal thrust allocation.

**Chapter 3** gives relevant background information in order to present the dynamical control model for the ROV Minerva. Then, the configuration, dynamical behavior, and limitations of the Minerva are discussed. Further, presenting the Minerva simulation and control systems, both non-HIL and HIL. Lastly, presenting the thruster configuration and current thrust allocation scheme, for which the aim of this thesis is to improve. Chapter 3 are in great extent taken from the work carried out in Kajanus (2014, spring).

**Chapter 4** presents the constrained recursive nullspace-based thrust allocation algorithm applied to the Minerva design case. The algorithm is tested in both open-loop, and closed-loop, with the results from the tests are presented and discussed.

**Chapter 5** presents the constrained optimal (QP) thrust allocation algorithm applied to the Minerva design case. The algorithm is tested in both open-loop, and closed-loop, with the results from the tests are presented discussed.

**Chapter 6** presents the constrained optimal recursive thrust allocation algorithm applied to the Minerva design case. The optimal recursive algorithm is the combination of the algorithms presented in the two previous chapters. The algorithm is tested in both open-loop, and closed-loop, with the results from the tests presented are discussed.

**Chapter 7** compares the performance of the thrust allocation algorithms presented in the three previous chapters, with the pseudoinverse thrust allocation utilized in the Minerva simulation system.

**Chapter 8** gives the concluding remarks, and recommendations for further work.

## 1.4 Notation

In this thesis lower case boldface letters/symbols are used to represent vectors, while upper case boldface letters/symbols are used to represent matrices.

## *Literature review*

This chapter aims to give an overview over relevant references related to the ROV Minerva control system, and the dynamical modeling it is based on. Further, references for the recursive nullspace-based thrust allocation, and optimal thrust allocation, are presented.

### **2.1 The Minerva dynamic modeling, and control system**

The dynamical model of the ROV Minerva is presented in amongst other Kirkeby (2010); Candeloro (2011); Sørensen et al. (2012); In Kirkeby (2010) the focus are on testing of DP, and tracking controllers for the ROV Minerva. While in Candeloro (2011) the development and testing of observer for the ROV Minerva are presented. Sørensen et al. (2012) presents results from full scale testing of the dynamic positioning (DP) and tracking system, developed for the Minerva by MSc students, PhD-candidates and NTNU researchers in the period 2010-2011. The software simulator resulting from the before mentioned sources, where implemented into the LabVIEW software from National Instruments as part of the work carried out in Tolpinrud (2011, 2012), and is what formed the basis for the current control system for Minerva.

The Minerva simulation and control systems utilizes the "Fossen's Robot-Like Vectorial Model for Marine Craft" found in Fossen (2011); This is a compact and elegant way to arrange the equation needed to describe the 6 DOF motion of a marine vessel. The Fossen model agrees with the SNAME equations and notation, but on an vectorial form rather than the SNAME component form. The SNAME component form can be found in amongst others in Faltinsen (1990).

For references on DP, tracking, and path-following control laws the textbooks from Fossen (2011); Sørensen (2011); Fossen (1991), and references therein, are a good source for the interested reader.

In Skjetne (2013) a parameter analysis of the ROV Minerva, looking into the cause and effect of the time-varying center of gravity (CG), are presented. Further references of dynamical modeling of marine vessels can be found in Sørensen (2011), Fossen (2011, 1991), Lewandowski (2004), and Do and Pan (2009).

## 2.2 Control allocation

For thrust and control allocation there are a variety of approach and references, ranging from the simple pseudo inverse Fossen (2011); Sørensen (2011), till optimizing problems Fossen (2011, 1991) for maneuvering, tracking, path following etc. The *Control Allocation - A survey* by Johansen and Fossen (2012), and references therein, gives a good overview over various control allocation schemes and references.

### 2.2.1 Recursive nullspace-based thrust allocation

When developing DP-system for ice condition, the ability to ensure the heading of the vessel into the ice forces is of utmost importance. This is due to the fact that the bow of the vessel typically gives the least projected area for which the ice force can act on. Thus, strictly prioritizing the DOF's when allocating the thruster forces, such that the effect of the ice force is minimized. This is what motivated the papers Kjerstad et al. (2013); Skjetne and Kjerstad (2013). The *Recursive Nullspace-Based Thrust Allocation* (Skjetne and Kjerstad, 2013), forms the basis for the proposed 3-step recursive thrust allocation algorithm in this thesis.

### 2.2.2 Optimal thrust allocation

The linear quadratic (QP) constrained formulation that the optimal thrust allocation proposed in this thesis, is based on Fossen (2011). The textbook (Fossen, 2011) is found to be a good source, and starting point, for any problem I have been asked to solve. In Ruth (2008) a number on references on QP are found, covering the unconstrained quadratic thrust allocation problem, aswell as the linearly quadratic constrained thrust allocation problem. Further sources where the QP problem are proposed, and applied to various applications are found in Diehl (2011); Wold (2013); Veksler et al. (2014).

For implementation of QP solvers in the Minerva control and simulation system, see National Instruments (fall 214); MathWorks (fall 214), respectively.

## Chapter 3

# *Minerva design case*

This chapter will present key-concepts and necessary background in order to present the dynamical control model for the ROV Minerva. Further Minerva's configuration, dynamical behavior and limitations will be discussed. Then the current simulation system, both HIL and Non-HIL, will be presented.

## 3.1 Background

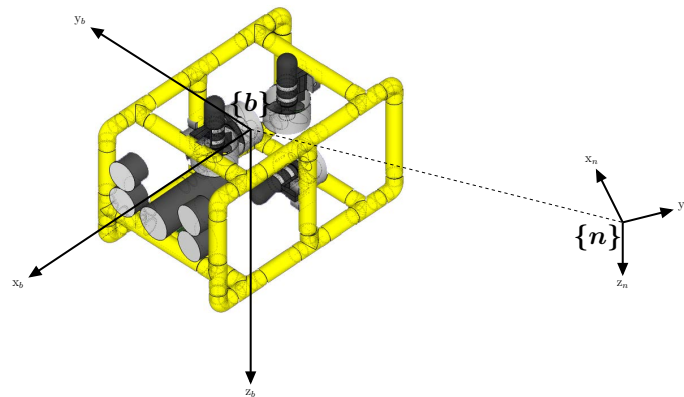
### Reference frames

For the ROV Minerva two reference frames are needed; the **NED** frame and the **body-frame**, also denoted as  $\{\mathbf{n}\}$  and  $\{\mathbf{b}\}$ , respectively. The following descriptions are based on the descriptions found in Fossen (2011), and illustrated in Figure (3.1).

The **North-East-Down (NED)**: Is fixed to the earth's surface as a tangent plane, defined by the longitudinal ( $\lambda$ ) and latitudinal ( $\mu$ ) coordinates. The x-, y-, z-axis of the NED-frame points in the directions North (true), East and Down (normal the earth's surface).

The ROV is utilized for local applications, so flat Earth navigation is assumed, and thus  $\{\mathbf{n}\}$ -frame is assumed inertial. This assumption is necessary in order for Newton's law to apply.

**Body-frame:** Is a local frame, fixed to the body of the ROV. This is typically centered in the center of origin (CO) of the object, with axis pointing in the direction coinciding with the primary axes of inertia.



**Figure 3.1:** Illustration of body and NED reference frames. Courtesy ROV sketch: Wordpress.

To describe the relationship between the {n} and the {b} frame, a rotation matrix is needed. Based on the Euler angles ( $\Theta_{nb}$ ), then according to Fossen (2011), the transformation matrix can be given according to

$$\mathbf{J}_{\Theta}(\boldsymbol{\eta}) = \begin{bmatrix} \mathbf{R}_b^n(\Theta_{nb}) & 0 \\ 0 & \mathbf{T}_{\Theta}(\Theta_{nb}) \end{bmatrix} \in \mathbb{R}^{6 \times 6}, \quad \text{and} \quad \Theta_{nb} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \in \mathbb{S}^3. \quad (3.1)$$

The two components of the transformation matrix in (3.1) are the rotation matrix for linear velocity ( $\mathbf{R}_b^n(\Theta_{nb})$ ), and angular velocity ( $\mathbf{T}_{\Theta}(\Theta_{nb})$ ), which can be given according to (3.2).

$$\begin{aligned} \mathbf{R}_b^n(\Theta_{nb}) &:= \begin{bmatrix} c\psi c\theta & -s\psi c\theta + c\psi s\theta s\phi & s\psi s\theta + c\psi c\theta s\phi \\ s\psi c\theta & c\psi c\theta + s\phi s\theta s\psi & -c\psi s\theta + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \in \mathbb{R}^{3 \times 3}, \\ \mathbf{T}_{\Theta}(\Theta_{nb}) &= \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & \frac{s\phi}{c\theta} & \frac{c\phi}{c\theta} \end{bmatrix} \in \mathbb{R}^{3 \times 3}. \end{aligned} \quad (3.2)$$

An important limitation with the Euler angle representation is the singularity for  $\theta = \pm 90^\circ$ . However, this is not a problem for Minerva. More details and alternative altitude representations can be found in Fossen (2011).

## Degrees of Freedom

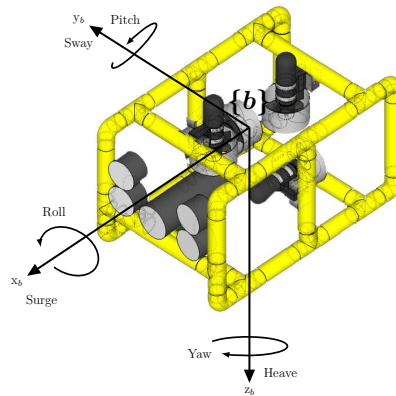
In order to fully describe the motion of an object in space, 6 Degrees of Freedom (DOF) are needed. These are three translational and three rotational motions. With basis in the {b}-frame, they are denoted {Surge, Sway, Heave} and {Roll, Pitch, Yaw}, respectively. The positive translational motion is in the direction of the {b}-axis, whereas the rotational motion is the rotation around these respective axis as illustrated in the figure below.

**Table 3.1:** Summarization of SNAME notation for marine vessel; DOF {1,2,3} are for linear motions. DOF {4,5,6} are for rotations.

DOF #	Motion / Rotation	Forces / moments	Linear velocities / angular velocities	Positions / Euler angles
1	Surge	X	u	x
2	Sway	Y	v	y
3	Heave	Z	w	z
4	Roll	K	p	$\phi$
5	Pitch	M	q	$\theta$
6	Yaw	N	r	$\psi$

Noteworthy is that the order of the state space is given as, Order =  $2 \times \text{DOF}$ , and thus the order of a system operating in 6 DOF will be 12.





**Figure 3.2:** Illustration of 6 DOF motion in body-frame,  $\{b\}$ . Courtesy ROV sketch: Wordpress.

## Process Model and Control Model

Further the terms Process Model (PM) and Control Model (CM) must be established.

According to Sørensen (2011, p.177) they can be defined as

- **Process Model:** “a comprehensive description of the actual process and should be as detailed as needed using high fidelity models”.
- **Control Model:** “a simplified mathematical description containing only the necessary an important physical properties of the process”.

In Section 3.2 the Minerva control model will be presented.

## Configuration Space and Workspace

Yet another set to terms to be defined is the configuration space (CS) and the workspace (WS). From Definition 9.2 and Definition 9.5 in Fossen (2011, p.236-237), they can be stated respectively as

- **Configuration Space:** “The n-dimensional configuration space is the space of possible positions and orientation that a craft may attain, possibly subject to external constraints”.
- **The workspace:** The m-dimensional workspace “is a reduced space of dimension  $m < n$  in which the control objective is defined”.

where their dimensions can be found

$$n := \dim(\boldsymbol{\eta}_{cs}) \quad , \quad m := \dim(\boldsymbol{\eta}_{ws}) < n. \quad (3.3)$$

The ROV Minerva is operated in 6 DOF, and thus the dimension of the CS will be  $n = 6$ . As Minerva is only actuated in 4 DOF, {Surge, Sway, Heave, Yaw }, the dimension of the WS will be  $m = 4$ . The two unactuated states { Pitch, Roll } are passively stable.

## Kinematics and Kinetics

In classical mechanics the study of dynamics can be divided into two parts, namely kinematics and kinetics. This subdivision of the dynamical problem is convenient, and is utilized in the *Fossen's Robot-Like Vectorial Model for Marine Craft*. A short description of this subdivision follows.

**The Kinematics** - from the Greek word *Kinemat*, meaning movement. The kinematic sub problem deals only with the geometrical aspects of motion, and is evaluated in the inertial frame.

**The Kinetics** - from the Greek word *Kinetikos* meaning moving. The kinetic sub problem deals with the forces causing motion, and is evaluated in the body-frame .

### 3.2 Dynamical modeling of ROV Minerva

The 4DOF control model for ROV Minerva, using the "*Fossen's Robot-Like Vectorial Model for Marine Craft*" (?), can be given according to (3.4).

$$\begin{aligned} i) \quad & \dot{\boldsymbol{\eta}} = \mathbf{J}_{\Theta}(\boldsymbol{\eta}) \boldsymbol{\nu} \\ ii) \quad & \mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} + \boldsymbol{\tau}_{\text{other}} + \mathbf{J}_{\Theta}^{-1}(\boldsymbol{\eta}) \mathbf{b}(t) \end{aligned} \quad (3.4)$$

The kinematic equation (i) relates the velocities ( $\dot{\boldsymbol{\eta}}$ ) of the vessel in the {n}-frame, through a transformation matrix ( $\mathbf{J}_{\Theta}(\boldsymbol{\eta})$ ), to translational and rotational velocities ( $\boldsymbol{\nu}$ ) of the vessel in the {b}-frame. While the kinetic equation (ii) describes all rigid-body (rb), hydrodynamic, environmental, and actuator forces, expressed in the {b}-frame. The {n}-frame is assumed to be inertial.

Representing the attitude with Euler angles ( $\Theta$ ), the 4 DOF displacement vector ( $\boldsymbol{\eta}$ ) and the velocities ( $\boldsymbol{\nu}$ ) can be defined as

$$\boldsymbol{\eta} := \begin{bmatrix} N \\ E \\ D \\ \psi \end{bmatrix} \in \mathbb{R}^3 \times \mathbb{S}^1 \quad , \quad \boldsymbol{\nu} := \begin{bmatrix} u \\ v \\ w \\ r \end{bmatrix} \in \mathbb{R}^{4 \times 1}. \quad (3.5)$$

The actuator forces vector is defined as

$$\boldsymbol{\tau}_{\text{actuators}} := \begin{bmatrix} X & Y & Z & N \end{bmatrix}^{\top} \in \mathbb{R}^{4 \times 1}, \quad (3.6)$$

giving the forces in {Surge, Sway, Heave} [N] and moment in {Yaw} [Nm] respectively. The remaining forces is collected in the force vector  $\boldsymbol{\tau}_{\text{other}}$ , and is the summation of all other forces affecting the vessel, and can for Minerva be given as

$${}^1\boldsymbol{\tau}_{\text{other}} = \boldsymbol{\tau}_{\text{umbilical}}. \quad (3.7)$$

Further the kinetic equation (ii) have the following components

- $\mathbf{M} = \mathbf{M}_{\text{RB}} + \mathbf{M}_{\text{A}}$ , is the mass matrix.
- $\mathbf{C} = \mathbf{C}_{\text{RB}} + \mathbf{C}_{\text{A}}$ , is the Coriolis and centripetal matrix due to the rotation between  $\{\text{b}\}$  and  $\{\text{n}\}$  frames.
- $\mathbf{D} = \mathbf{D}_{\text{L}} + \mathbf{D}_{\text{NL}}$ , is the linear and non-linear damping.
- $\mathbf{g}$ , is the restoring vector.
- $\mathbf{b}(t)$ , is a slowly varying bias vector, modelled as a first order Markov process, accounting for the current force.

By assuming starboard-port symmetry, with  $y_g = I_{xy} = I_{yz} = I_{zx} = 0$ , then according to Fossen (2011) the mass matrix can be defined as

$$\mathbf{M} = \begin{bmatrix} m - X_{\dot{u}} & 0 & -X_{\dot{w}} & 0 & mz_g - X_{\dot{q}} & 0 \\ 0 & m - Y_{\dot{v}} & 0 & -mz_g - Y_{\dot{p}} & 0 & mx_g - Y_{\dot{r}} \\ -X_{\dot{w}} & 0 & m - Z_{\dot{w}} & 0 & -mx_g - Z_{\dot{q}} & 0 \\ 0 & -mz_g - Y_{\dot{p}} & 0 & I_x - K_{\dot{p}} & 0 & -I_{zx} - K_{\dot{r}} \\ mz_g - X_{\dot{q}} & 0 & -mx_g - Z_{\dot{q}} & 0 & I_y - M_{\dot{q}} & 0 \\ 0 & mx_g - Y_{\dot{r}} & 0 & -I_{zx} - K_{\dot{r}} & 0 & I_z - N_{\dot{r}} \end{bmatrix} \quad (3.8)$$

With the ROV parameters as reported in Kirkeby (2010)

$$\begin{aligned} m &= 460 \quad [kg] \\ l &= 1,44 \quad [m] \\ b &= 0.82 \quad [m] \\ h &= 0.80 \quad [m] \\ I_x &= \frac{1}{12}m(b^2 + l^2) = 105,2633 \quad [kg\ m^2] \\ I_y &= \frac{1}{12}m(l^2 + h^2) = 104,0213 \quad [kg\ m^2] \\ I_z &= \frac{1}{12}m(b^2 + h^2) = 50,3087 \quad [kg\ m^2] \end{aligned} \quad (3.9)$$

and the calculated hydrodynamic added mass, assumed boxed-shaped body with values reduced by 20% to compensate for overestimation

$$\begin{aligned} X_{\dot{u}} &= -293 \quad [kg] \\ Y_{\dot{v}} &= -302 \quad [kg] \\ Z_{\dot{w}} &= -326 \quad [kg] \\ K_{\dot{p}} &= -52 \quad [kg\ m^2] \\ M_{\dot{q}} &= -52 \quad [kg\ m^2] \\ N_{\dot{r}} &= -57 \quad [kg\ m^2] \\ Y_{\dot{r}} &= -6 \quad [kg\ m] \end{aligned} \quad (3.10)$$

and  $X_{\dot{w}} = X_{\dot{q}} = Y_{\dot{p}} = Z_{\dot{q}} = K_{\dot{r}} = 0$ .

Further, according to Fossen (2011), assuming a diagonal damping matrix

---

<sup>1</sup> For a surface vessel also the wind ( $\boldsymbol{\tau}_{\text{wind}}$ ) and wave ( $\boldsymbol{\tau}_{\text{waves}}$ ) forces would be included in the  $\boldsymbol{\tau}_{\text{other}}$ .

$$\mathbf{D}(\nu) = \text{diag}\{ -X_u - X_{u|u}|u|, -Y_v - Y_{v|v}|v|, -Z_w - Z_{w|w}|w|, \\ -K_p - K_{p|p}|p|, -M_q - M_{q|q}|q|, -N_r - N_{r|r}|r|\} \quad (3.11)$$

with the calculated values from Kirkeby (2010)

$$\begin{aligned} -X_u &= -29 & -X_{u|u}|u| &= -292 \\ -Y_v &= -41 & -Y_{v|v}|v| &= -584 \\ -Z_w &= -254 & -Z_{w|w}|w| &= -635 \\ -K_p &= -34 & -K_{p|p}|p| &= -84 \\ -M_q &= -59 & -M_{q|q}|q| &= -148 \\ -N_r &= -0, 2 & -N_{r|r}|r| &= -1 \end{aligned} \quad (3.12)$$

The coriolis and centripetal matrix ( $\mathbf{C}$ ) can be parameterized in several ways, as a function of the mass matrix. See Fossen (2011) for further details.

### 3.3 Configuration, dynamical behavior and limitations

The ROV is rated for 700 m depth, but the current umbilical is of about 300 m length, and this limits the operation depth and radius accordingly. That being said, this is sufficient for the typical AUR-lab need. As there are no automatics connected with the umbilical cable spool, one must remain vigilant to ensure the ROV is not constrained by a too short umbilical feed. The ROV operator controls the umbilical feeding manually, this can end in critical situation where cable could be stuck in ROV, or degrade ROV performance.

Further the ROV is equipped with a range of sensors to provide the required measurements for the operation of the ROV. These are:

- DVL (Doppler Velocity Log): Gives velocity over ground (surge, sway, and heave) by utilizing the Doppler effect. The DVL is also equipped with an acoustic altimeter, which measure the ROV's height above the seabed. An important measurement as there may be deviations between depths in the map and the actual depth; And even more important as the depth sensor is of lesser quality. The aft mounting of the DVL causes a time delay in altitude readings.
- HiPAP (High Precision Acoustic Positioning): The HiPAP are mounted on the RV Gunnerus. A transponder mounted on the ROV gives the relative position of the ROV to RV Gunnerus.
- IMU (Inertial Measurement Unit): Can based on a 3-axis rate gyros and accelerometers (linear), give estimates of position and attitude (orientations) of the ROV. The IMU are also equipped with a 3-axis magnetometer.

- Pressure sensor: Gives the depth of the ROV based on hydrostatic pressure. The onboard pressure sensor was of pore quality, and needed often recalibration. An additional pressure sensor of high quality has been retrofitted.
- Magnetic compass: Gives heading angle of the ROV based on the earths magnetic field. The measurement from the magnetic compass is of low quality, due to soft and hard iron effects.
- Sonar (active): Gives a image of the ROV's surroundings, analogous with a radar image.

The ROV control station is also set up with the software *NaviPac* from EIVA. The software package is equipped with maps from CMAP and gives the position of the ROV. The software also have capabilities for filtering of measurements (?).

The ROV is also equipped with several cameras and light fixtures, all bow mounted; and with a hydraulic manipulator for intervention purposes. This is also bow mounted, on starboard side. None of this equipment is accounted for in the current Minerva model. The weight of the manipulator and camera equipment causes a constant roll angle on the ROV. This has been attempted compensated by the supplier, Sperre AS, by adding extra buoyancy on the starboard side. This is at best semi-successful. This roll angle introduces a coupling effect between heave and sway. In the ROV model, heave is assumed uncoupled, but as the ROV is tilted in a constant roll angle, motion in heave causes sway motion, and vice versa.

For prolonged deployments, the ROV becomes negatively buoyant. Thus a worst case scenario where the communication with the ROV is lost due to broken umbilical, the application do not fail to safe, to the potential cost of lost ROV. A update of the buoyancy element of the ROV may be in order. Also worth mentioning is the case where the ROV is operated with the manipulator extended, possibly with a hanging load. This most certainly alter the dynamical behavior negatively. This is not accounted for. This, according the PhD-candidate M.Candeloro associated with the AUR-lab, makes the ROV difficult to handle.

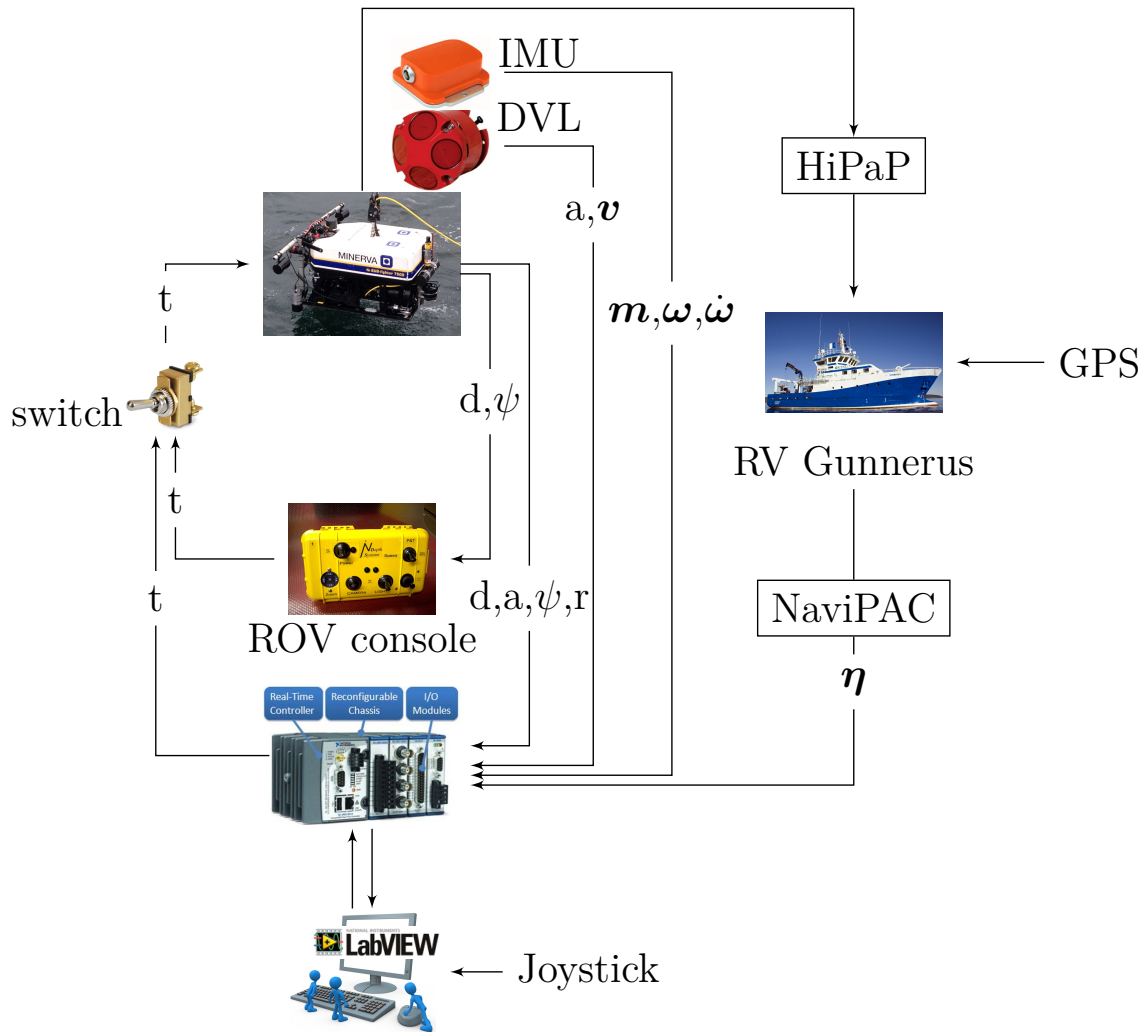
The umbilical excite the passively stable DOF's of the ROV, the main contribution is in pitch, but it also contribute in some extent to roll motion. Another concern with the umbilical is that it is no model of the cable in the control system, thus making it impossible to account for the cable induced drag forces beforehand. Pitch motion is also excited by the aft thrusters; They are mounted below the CO, and thus causing a pitch moment around CO. The thusters have a frequency limitation of 7 Hz.

Further, worth mentioning are that there are observed a resonance effect when operating the control system in DP mode. Another concern using the *NaviPac*, is that it is an proprietary software, i.e. it is unknown how the signals from HiPaP and GPS is filtered.

Lastly is the issue with the loading / unloading of the control container. One can argue that this is more related to RV Gunnerus, but even so, it would be extremely hard to operate the ROV without it's control system. The container has a water

leakage that causes the mass of the container to increase over time. A temporary quick fix to remedy the situation, was applied in the form of reducing the length of the loading chain. Thus reducing the loading moment on the crane, and thereby still having capacity for loading / unloading. The work to equip a new control container is initiated.

An illustration of ROV Minerva's sensor equipment is seen in Figure (3.3).



**Figure 3.3:** Illustration showing ROV Minerva's sensor equipment.

Picture courtesies: **RV Gunnerus**, Fredrik Skoglund; **IMU**, Xsens; **DVL**, Seatronics; **cRIO + LabVIEW logo**, NI; **ROV console**, iN-DepthSystems; **Switch**, WiringProducts; **PC**, Puzzledworld.

### 3.4 ROV Minerva simulation systems

The Minerva simulation systems represent the two first stepping stones on the way to test software on the LabVIEW control system for ROV Minerva. The structure of the control system can simplified be illustrated as in Figure (3.4), where the control system is running in the computer, connected by ethernet cable to the cRIO (Compact Reconfigurable Input-Output module) communicating with the ROV through the umbilical.



**Figure 3.4:** Illustration of the hardware structure for the ROV Minerva control system. Pictures with courtesy: **PC**, Puzzledworld; **cRIO + Logo**, NI; **Minerva**, TU.

The Minerva simulation system consist of the following simulators;

- Software simulator; MATLAB®/Simulink. (NonHIL).
- HIL simulator; LabVIEW.

where the difference in complexity is illustrated Figure 3.5 and Figure 3.6, respectively. The simulation system is discussed further in the following subsections.

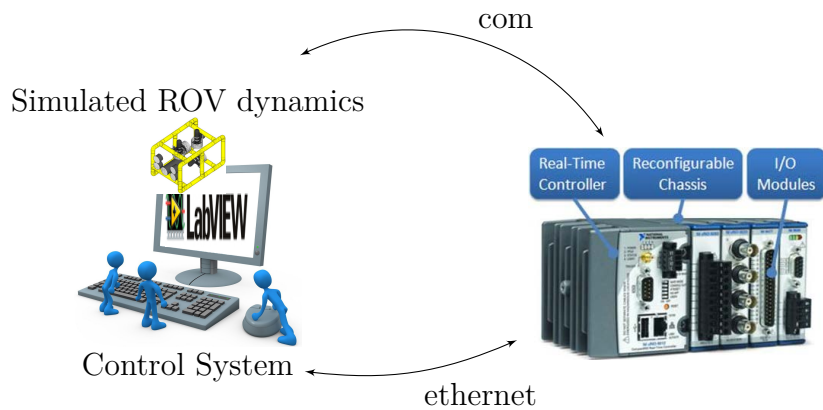


**Figure 3.5:** Illustration of the hardware structure for the software simulator of ROV Minerva. Pictures with courtesy: **PC**, Puzzledworld; **Logo**, MATLAB®.

There is also an alternative 6DOF software simulator in Simulink made by PhD-candidate Daniel de Fernandes, but this is not further discussed in this report.

Currently another software simulator, in LabVIEW, are being developed as part of the master thesis of a fellow MSc student, Martin Lauritzsen, at the Department of Marine Technology. This is achieved by adding the option of removing the cRIO dependency from the LabVIEW HIL simulator.

A major difference between the software simulators in Simulink and LabVIEW, is that while LabVIEW is real time software, Simulink is not.

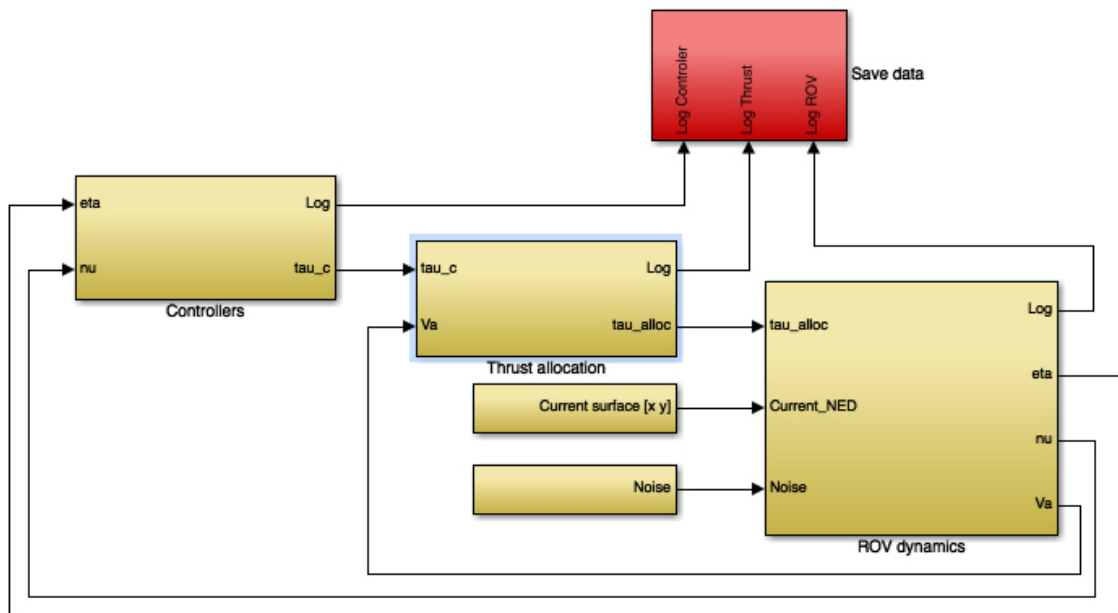


**Figure 3.6:** Illustration of the hardware structure for the HIL simulator for ROV Minerva. Pictures with courtesy: **PC**, Puzzledworld; **cRIO + Logo**, NI; **ROV sketch**, Wordpress.

### 3.4.1 MATLAB<sup>®</sup>/Simulink software simulator

The software simulator, or non-HIL simulator, described in the following, was developed as part of the master thesis of Kirkeby (2010). It has been further developed by MSc students and PhD-candidates associated with the AUR-lab.

The structure of the top-level block diagram of the software simulator can be seen in Figure (3.7),



**Figure 3.7:** Block diagram in SimuLink showing the top level for the software simulator of the ROV Minerva.



where the four main block of the model are;

- The Controller block, far left in the diagram.
- The Thrust Allocation block.
- The ROV Dynamics block, large block far right in the diagram.
- The Save Data block, red color.

The simulator is developed for discrete time, using Zero-Order-Hold (ZOH) discretizing, and the feedback variables are  $\boldsymbol{\eta}$  and  $\boldsymbol{\nu}$ . In the following a short description of the main blocks will be given.

The **controller block** contains a reference generator which takes way points (WP) as a step input. The input is smoothened through a reference filter, before the signal is fed to the LQ DP-controller. Further, the controller block takes the position/attitude ( $\boldsymbol{\eta}$ ) and velocities ( $\boldsymbol{\nu}$ ) as feedback input from the ROV Dynamics block.

The **ROV Dynamics block** computes the dynamics of the ROV according to equations given by Kirkeby (2010) as

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \mathbf{J}(\boldsymbol{\eta}) \boldsymbol{\nu} \quad , \quad \{\boldsymbol{\eta}, \boldsymbol{\nu}\} \in \mathbb{R}^{6 \times 1} \\ \dot{\mathbf{b}} &= -\mathbf{T}_b^{-1} \mathbf{b} + \mathbf{E}_b \mathbf{w} \\ \mathbf{M} \dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu}) \boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu}) \boldsymbol{\nu} + \mathbf{g} &= \mathbf{J}(\boldsymbol{\eta})^{-1} \mathbf{b} + \boldsymbol{\tau} \end{aligned} \quad (3.13)$$

with the components as discussed in Section 3.2. The ROV dynamics block are in great extent made of components from the MSS-toolbox. Thus, the required block diagram libraries (\*.slx) needed to run the model are included in the attached Minerva software CD in Appendix A.

The **Save Data** block sends all values from the model to the MATLAB<sup>®</sup> workspace. This is convenient if you need to access the data.

Lastly is the **Thrust Allocation** block which contains thrust allocation by pseudoinverse, modifying the commanded thrust for thrust loss factors, and converting allocated force to rpm. This block will be modified, and added the option of the thrust allocation algorithms to be presented in this thesis.

### 3.4.2 LabVIEW HIL simulator

The LabVIEW HIL simulator was based on the MATLAB<sup>®</sup>/Simulink created by Kirkeby (2010). The migration into LabVIEW was part of the master thesis of Tolpinrud (2012). The development of the dynamic positioning and tracking system for the ROV Minerva was done by PhD candidates and MSc students connected with the AUR-lab, as presented in Sørensen et al. (2012). The system is continuously being developed further by PhD candidates and MSc students associated with the Department of Marine Technology.

As illustrated in Figure 3.4, the control system Njord, communicates with the simulated ROV through the cRIO. This is also the case for the HIL simulator, illustrated

in Figure 3.6, but in this case the ROV is removed from the control loop and the dynamics are simulated.

### Hardware setup

As part of the work during this project, a new dedicated HIL-simulation station for the ROV Minerva, was set-up at the NTNU HIL-lab at Marintek. Use NTNU credentials, i.e. username and password to log on to the workstation.

The HIL simulator can be accessed / utilized in any of the following ways;

- Simulate at HIL-lab, room nr. 157 at Marintek (to access Marintek contact the reception).
- Connect to the HiL-station by "Remote access", and run system as one would do at the HIL-lab.
- Run control system (Njord) and Graphical User Interface (Frigg) on a windows<sup>2</sup> PC with LabVIEW 2012 SP1 + device drivers installed. Connect (over IP) to the HIL project running on the PC at HIL-lab.

**Table 3.2:** Summarization of HIL-simulator station information.

Component	Name	IP adress (fixed)	Network name
PC	-	129.241.140.109	hil2.ivt.ntnu.no
cRIO	Minerva-HILSim	129.241.140.108	hil1.ivt.ntnu.no

The cRIO has standalone capability, i.e. the control system or the HIL simulator, can be loaded into the cRIO and run from the onboard processing unit. This feature is currently not utilized.

### HIL-simulator

The Minerva HIL simulator consists of the following three LabVIEW projects;

1. AURLab ROV Control System.lvproj.
2. Graphical User Interface for Control System.lvproj.
3. HILpro.lvproj.

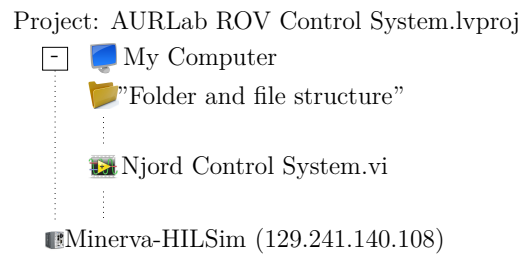
An illustration of the file structure of these three projects, with their main VI's, are seen in Figures 3.8 - 3.10.

To run the HIL simulator, follow these steps;

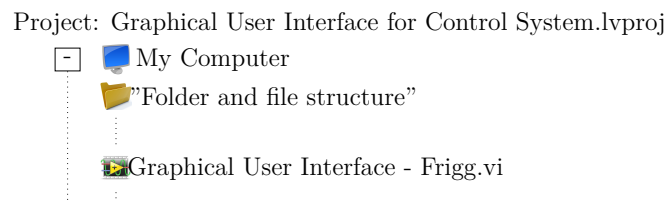
- Locate folders ***HIL simulator*** and ***ContrSyst*** in the ***Public Documents*** folder. Copy these into your own user documents folder.
- Locate and open the three projects; ***Njord***, ***Frigg*** and ***HIL Simulator***.

---

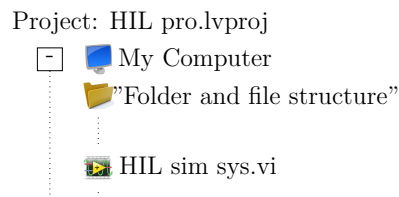
<sup>2</sup>The LabVIEW HIL simulator can not be run from mac OS, this is due too the required tollbox "Timeseries" is not a part of the software bundle for mac (per 2014).



**Figure 3.8:** Illustrating the file structure of the LabVIEW project "AURLab ROV Control System.lvproj", showing the main VI, "Njord Control System.vi" and the connection to the cRIO named "Minerva-HILSim".



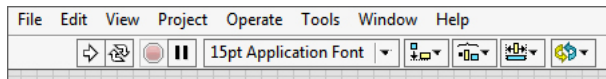
**Figure 3.9:** Illustrating the file structure of the LabVIEW project "Graphical User Interface for Control System.lvproj", showing the main VI, "Graphical User Interface - Frigg.vi".



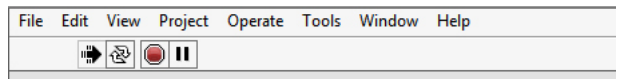
**Figure 3.10:** Illustrating the file structure of the LabVIEW project "HILpro.lvproj", showing the main VI, "HIL sim sys.vi".

- Open main VI's of the respective projects and press **Run**. See Figures 3.11 and 3.12 for how to identify whether or not the project are in working condition.
- On the GUI, press "Connect";
  - The Startup Panel.vi is triggered; from the "ROV Selector" choose Minerva.
- Press "Start Control Loop" on GUI;
  - The SetOriginPanel.vi is triggered; from the "Coordinate System" choose UTM.

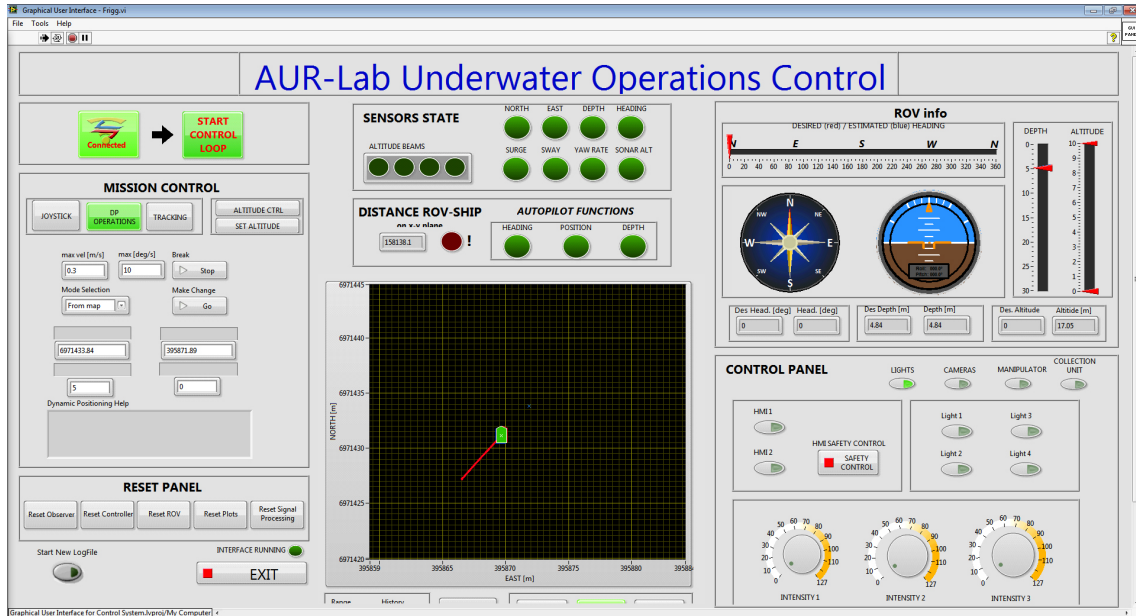
The system is now started up and ready to run. For simulation, choose between "DP OPERATION" or "TRACKING" mode. The "JOYSTICK" mode does not have a functionality in the HIL simulator. The system running in "DP OPERATION" mode is seen in Figure 3.13. To stop the system press "EXIT", this will break the connection and stop Njord and Frigg. The HIL simulator must be stopped manually.



**Figure 3.11:** Screenshot of a working LW project, before *Run* is pressed, indicated by solid white arrow.



**Figure 3.12:** Screenshot of a working LW project, after *Run* is pressed, indicated by solid black arrow.



**Figure 3.13:** Screenshot of the running GUI Frigg from the HIL simulation system. The map view in center of the GUI will show the ROV position, indicated by the solid green symbol, along with the desired position (red ROV trace) and the measured position (blue ROV trace).

The HIL simulator generate several log files; The most relevant log file to this project are the "Logfiles\_pro" which organize the logged parameters into separate files for  $\eta$ ,  $\nu$  and  $\tau$ . The additional logging of the allocated thrust, both force ( $f$ ) and rpm ( $n$ ) will be required. The HIL-simulator are attached in Appendix A.

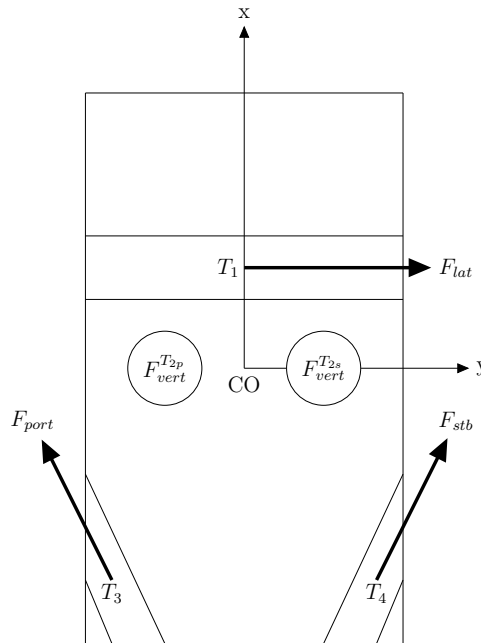
### 3.5 Thruster configuration, and current thrust allocation scheme

Thrust allocation is the task of distributing the generalized forces, to the thrusters of the ROV, to achieve the desired change in position and orientation. In this context the generalized forces denotes forces [N] and moments [Nm], and can be defined according to (3.14)

$$\boldsymbol{\tau} := \begin{bmatrix} \mathbf{f} \\ \mathbf{r} \times \mathbf{f} \end{bmatrix}. \quad (3.14)$$

The forces is represented by the thruster force vector,  $\mathbf{f} = [f_x \ f_y \ f_z]^\top$ , and the moment arms,  $\mathbf{r} = [l_x \ l_y \ l_z]^\top$ , giving the location of the thrusters with respect to CO of the ROV. The moment is obtained as the cross product of the thruster arm vector, and the thruster force vector.

An illustration of the ROV Minerva with thruster placement, and positive force direction of the thrusters, is given in Figure 3.14. As illustrated in the figure, thrusters  $T_3$  and  $T_4$  are fixed at a constant azimuth angel with respect to the  $\{b\}$ -frame, of  $-10^\circ$  and  $10^\circ$  respectively. These angels allow thrusters  $T_3$  and  $T_4$  to produce forces in both x- and y-direction, and is accounted for by decomposing these forces according to  $f_{x,i} = f_i \cos(\alpha_1)$  and  $f_{y,i} = f_i \sin(\alpha_1)$  when establishing the thruster configuration matrix.



**Figure 3.14:** Illustration of thruster layout, with positive force direction, for ROV Minerva

For the 6 DOF case, evaluated with the thruster parameters of Minerva, (3.14) can be expanded according to (3.15)

$$\begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} \mathbf{f}_x \\ \mathbf{f}_y \\ \mathbf{f}_z \\ \mathbf{f}_z \mathbf{l}_y - \mathbf{f}_y \mathbf{l}_z \\ \mathbf{f}_x \mathbf{l}_z - \mathbf{f}_z \mathbf{l}_x \\ \mathbf{f}_y \mathbf{l}_x - \mathbf{f}_x \mathbf{l}_y \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0.9848 & 0.9848 \\ 1 & 0 & -0.1736 & 0.1736 \\ 0 & 1 & 0.0 & 0.0 \\ -0.3 & 0 & 0.0521 & 0.0521 \\ 0 & 0 & -0.2954 & 0.2954 \\ 0.166 & 0 & 0.3353 & -0.3353 \end{bmatrix} \begin{bmatrix} f_{lat} \\ f_z \\ f_{port} \\ f_{stb} \end{bmatrix} = \mathbf{B}\mathbf{f}. \quad (3.15)$$

The thruster data utilized in the Minerva simulation and control systems, as given by Kirkeby (2010), are given in Table 3.3 and Table 3.3. In Table 3.3 the geometric properties of the thrusters are summarized, while the thrust related properties are summarized in Table 3.4.

**Table 3.3:** Summarization of the thruster data for ROV Minerva as implemented in the control and simulation systems.

Thruster	Symbol	Coordinates (x,y,z)	Azimuth angle [deg]
Lateral	$T_1$	$\mathbf{r}_1 = (0.166, 0, 0.30)$	$\alpha_1 = 90^\circ$
Vertical	$T_{2p}/T_{2s}$	$\mathbf{r}_2 = (0, 0, 0)$	-
Port	$T_3$	$\mathbf{r}_3 = (-0.57, -0.24, 0.30)$	$\alpha_3 = -10^\circ$
Starboard	$T_4$	$\mathbf{r}_4 = (-0.57, 0.24, 0.30)$	$\alpha_4 = 10^\circ$

**Table 3.4:** Summarization of the thrust parameters for ROV Minerva as implemented in the MATLAB<sup>®</sup>/Simulink simulator.

Parameters	Thrusters	Lateral ( $T_1$ )	Vertical ( $T_{2p}/T_{2s}$ )	Port ( $T_3$ )	Starboard ( $T_4$ )
Max revolutions [rpm]		1450	1450	1450	1450
Bollard Pull [N]		195	195	239	239
Diameter [m]		0,19	0,22	0,22	0,22
Thrust coeff.: $K_T^{fwd} / K_T^{aft}$ [-]		0,24 / 0,15	0,24 / 0,15	0,24 / 0,15	0,24 / 0,15
Thrus loss: $\theta_{fwd} / \theta_{bwd}$ [-]		1.04 / 1.04	0.58 / 0.58	0.72 / 0.53	0.72 / 0.53

The simulation, and control systems of Minerva are commanded in thruster revolutions (rpm), and thus the desired generalized force vector must be transformed to rpm. The mapping between force and shaft speed is achieved according to (3.16).

$$\begin{aligned} \mathbf{f} &= \rho \mathbf{K}_T \mathbf{D}^4 \boldsymbol{\theta} |\mathbf{n}| \mathbf{n} \quad [N] \\ \mathbf{n} &= \text{sign}(\mathbf{f}) \sqrt{\frac{|\mathbf{f}|}{\mathbf{c}}} \quad [rpm] \Rightarrow \mathbf{c} := \rho \mathbf{K}_T \mathbf{D}^4 \boldsymbol{\theta} \end{aligned} \quad (3.16)$$

Denoting  $\{\mathbf{c}^+, \mathbf{c}^-\}$  as the thrust coefficient in positiv and negativ rotational direction, respectively. Further, accounting for two thrusters i vertical direction by

mutliplying  $\{\mathbf{c}_{vert}^+, \mathbf{c}_{vert}^-\}$  with 2. The coefficients utilized in the mapping between forces and rpm, in the Minerva (LabVIEW) controll system, are given in Table 3.5.

**Table 3.5:** Summarization of the thrust coefficients utilized in the mapping between force and rpm, utilized in the control system of Minerva.

Thruster	Symbol	$\mathbf{c}^+$	$\mathbf{c}^-$
Lateral	$T_{lat}$	$0.943 \times 10^{-4}$	$0.943 \times 10^{-4}$
Vertical	$T_{vert}$	$2 \cdot 0.925 \times 10^{-4}$	$2 \cdot 0.74 \times 10^{-4}$
Port	$T_{port}$	$1.115 \times 10^{-4}$	$0.328 \times 10^{-4}$
Starboard	$T_{stb}$	$1.115 \times 10^{-4}$	$0.328 \times 10^{-4}$

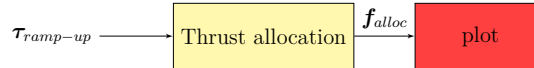
In chapters 4 - 6, the following thrust allocation algorithms;

1. Constrained 3-step recursive nullspace-based thrust allocation,
2. Constrained optimal thrust allocation, QP formulation,
3. Constrained optimal recursive thrust allocation, recursive QP formulation

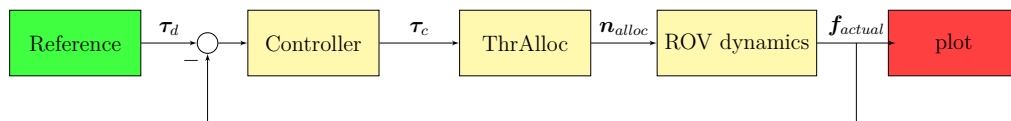
are applied to Minerva, and implemented into the MATLAB<sup>®</sup>/Simulink simulation system. The thrust allocation algorithms is then tested in the following two steps to show performance:

1. MATLAB<sup>®</sup> script; testing with a "ramp-up" commanded thrust vector.
2. MATLAB<sup>®</sup>/Simulink software simulation; test cases to be defined in the following

The proposed testing regime can be illustrated as seen in figures 3.15 - 3.16.



**Figure 3.15:** Illustration of script testing set-up



**Figure 3.16:** Illustration of MATLAB<sup>®</sup>/Simulink testing set-up

As seen from (3.15), the thruster configuration of Minerva have a one-to-one mapping for thrust allocation in heave. I.e., no improvements can be achieved for the thruster utilization i z-direction. Thus, the focus in this thesis will be on the in-plane forces and moment,  $\{X, Y, N\}$ .

In the Minerva control and simulation system, the unconstrained pseudoinverse method are utilized. The limitation of the control, are applied at thruster level, i.e. the commands are cut of at  $\pm 1450$  [rpm], before the command are given to the thrusters. The unconstrained thrust allocation clearly offers less control of the end result, as the obtained thrust will differ from the allocated thrust. Further, the thrust

configuration matrix utilized in the HIL-simulator/control system, seen in (3.15), are badly scaled. Thus, the pseudoinverse is found by single value decomposition (SVD).

For reference purposes, a constrained 1-step recursive nullspace-based thrust algorithm was implemented. This was later removed because of the implementation was in particular unbeneficial for the heave allocation. This is due to how the recursive nullspace-based thrust allocation algorithm are design; When a thruster saturates, all thrusters get limited to the current level of utilization. Thus, when a in-plane thruster saturates, the vertical thruster would also be limited to their current level. This applies to all DOFs being allocated within the current step, or any subsequent steps, of the recursive nullspace-based thrust allocation.

For that reason, a recursive 3-step nullspace-based thrust allocation algorithm are proposed for the Minerva; Allocating heave in the 1<sup>st</sup> step, yaw in the 2<sup>nd</sup> step, and lastly allocating surge and sway in the 3<sup>rd</sup> step. This will ensure full utilization of the vertical thruster, without any loss of yaw capacity. I.e. a recursive 3-step nullspace-based thrust allocation algorithm, prioritizing in this sequence {Z, N, XY}, are proposed and tested in the following. This prioritization will ensure full utilization in heave, and at the same time obtain the sought yaw prioritization.

The proposed recursive 1-step algorithm are attached in Appendix D, for reference.



## *Constrained recursive pseudoinverse-based thrust allocation*

Neglecting the coupling effect between {Roll, Pitch}, and the actuated DOF's {Surge, Sway, Heave, Yaw}, the thruster configuration matrix from from (3.15) can be simplified according to 4.1

$$\begin{aligned}
 \mathbf{B} &= \begin{bmatrix} \cos(\alpha_1) & 0 & \cos(\alpha_3) & \cos(\alpha_4) \\ \sin(\alpha_1) & 0 & \sin(\alpha_3) & \sin(\alpha_4) \\ 0 & 1 & 0 & 0 \\ l_{1x} & 0 & (l_{3x} \sin(\alpha_3) - l_{3y} \cos(\alpha_3)) & (l_{4x} \sin(\alpha_4) - l_{4y} \cos(\alpha_4)) \end{bmatrix} \\
 &= \begin{bmatrix} 0 & 0 & 0.9848 & 0.9848 \\ 1 & 0 & -0.1736 & 0.1736 \\ 0 & 1 & 0.0 & 0.0 \\ 0.166 & 0 & 0.3353 & -0.3353 \end{bmatrix} \in \mathbb{R}^{4 \times 4},
 \end{aligned} \tag{4.1}$$

giving the basis for which the thrust allocation algorithms are to be based.

### 4.1 Problem formulation

Considering the control allocation problem based on the linear thrust model from Skjetne and Kjerstad (2013);

$$\boldsymbol{\tau} = \mathbf{T} \mathbf{f}, \quad \mathbf{f} = \mathbf{K} \mathbf{u} \quad \Rightarrow \quad \boldsymbol{\tau} = \mathbf{T} \mathbf{K} \mathbf{u} = \mathbf{B} \mathbf{u}, \tag{4.2}$$

where  $\mathbf{f}$  is the constrained thruster vector and  $\mathbf{T}$  is the thruster configuration matrix from (3.15). The diagonal gain matrix  $\mathbf{K}$  is calculated such that the allocated thrust vector satisfies the constraint set  $\mathbb{U}$ , given according to

$$\mathbb{U} = \{\mathbf{u} \in \mathbb{R}^{4 \times 1} : \mathbf{A} \mathbf{u} \leq \mathbf{c}, \mathbf{A} \in \mathbb{R}^{8 \times 4}, \mathbf{c} \in \mathbb{R}^{8 \times 1}\}. \tag{4.3}$$

Saturating the thruster in force, with the parameters from Table 3.4, using  $n = n_{max} = 1.450$  [rpm] in (3.16)  $\mathbf{A} \mathbf{u} \leq \mathbf{c} = \mathbf{f}_{sat}$ , can be given as;

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{lat} \\ u_{vert} \\ u_{port} \\ u_{stb} \end{bmatrix} \leq \begin{bmatrix} -f_{1,min} \\ f_{1,max} \\ -f_{2,min} \\ f_{2,max} \\ -f_{3,min} \\ f_{3,max} \\ -f_{4,min} \\ f_{4,max} \end{bmatrix} [N] \quad (4.4)$$

## 4.2 Design of recursive nullspace-based thrust allocation for Minerva

For reasons discussed in section 3.5; A recursive 3-step nullspace-based thrust allocation algorithm are proposed. Allowing full utilization of thrust capability in heave (Z), while at the same time achieving the sought yaw moment (N) prioritization, over surge (X), and sway (Y) force allocation.

### 4.2.1 3-step recursive algorithm; { Z, N, XY }:

In the 3-step recursive nullspace-based thrust allocation algorithm { Z, N, XY }, the thrust is allocated to satisfy the following DOF's prioritization;

1. Heave (Z)
2. Yaw (N)
3. Surge and Sway (XY)

where the remaining thrust capability from the previous step, are allocated to satisfy the next step. As seen in (4.1), for the ROV Minerva Heave is uncoupled the horizontal DOF's { Surge, Sway, Yaw }, and thus no Yaw capability is lost in allocating Heave first.

Based on (3.15) and (4.1), defining  $\tau_z := \tau_Z$ ,  $\tau_\psi := \tau_N$  and  $\boldsymbol{\tau}_{xy} := \text{col}(\tau_X, \tau_Y)$  and rearranging the  $\mathbf{B}$ -matrix accordingly, such that

$$\begin{bmatrix} \tau_z \\ \tau_\psi \\ \boldsymbol{\tau}_{xy} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_z^\top \\ \mathbf{b}_\psi^\top \\ \mathbf{B}_{xy}^\top \end{bmatrix} \mathbf{u}_{3step} \quad , \quad \begin{cases} \mathbf{u}_{3step} & \in \mathbb{R}^{4 \times 1} \\ \{\mathbf{b}_z^\top, \mathbf{b}_\psi^\top\} & \in \mathbb{R}^{1 \times 4} \\ \mathbf{B}_{xy}^\top & \in \mathbb{R}^{2 \times 4} \end{cases} \quad (4.5)$$

The weighting matrices  $\{\mathbf{W}_z, \mathbf{W}_\psi, \mathbf{W}_{xy}\} \in \mathbb{R}^{4 \times 4}$  required for the algorithm, are left as tuning parameters to be defined in Section ??.

Let  $\mathbf{u}_{3step} = \mathbf{v}_{1a} + \mathbf{v}_{1n}$  and target the heave force,

$$\boldsymbol{\tau}_z = \mathbf{b}_z^\top (\mathbf{v}_{1a} + \mathbf{v}_{1n}). \quad (4.6)$$

Defining the weighted pseudoinverse for Z;

$$\left(\mathbf{b}_z^\top\right)_{w_z}^\dagger := \mathbf{W}_z^{-1}\mathbf{b}_z\left(\mathbf{b}_z^\top\mathbf{W}_z^{-1}\mathbf{b}_z\right)^{-1} \in \mathbb{R}^{4 \times 1}, \quad (4.7)$$

and the corresponding nullspace orthogonal projection matrix

$$\mathbf{Q}_z := \mathbf{I} - \left(\mathbf{b}_z^\top\right)_{w_z}^\dagger \mathbf{b}_z^\top \in \mathbb{R}^{4 \times 4}. \quad (4.8)$$

Further, defining

$$\mathbf{v}_{1a} := \left(\mathbf{b}_z^\top\right)^\dagger \boldsymbol{\tau}_z \quad \text{and} \quad \mathbf{v}_{1n} := \mathbf{Q}_z \mathbf{v}_{2a}, \quad \{\mathbf{v}_{1a}, \mathbf{v}_{1n}\} \in \mathbb{R}^{4 \times 1}, \quad (4.9)$$

such that

$$\boldsymbol{\tau}_z = \mathbf{b}_z^\top (\mathbf{v}_{1a} + \mathbf{v}_{1n}) = \mathbf{b}_z^\top \left(\mathbf{b}_z^\top\right)^\dagger \boldsymbol{\tau}_z + \mathbf{b}_z^\top \mathbf{Q}_z \mathbf{v}_{2a} \in \mathbb{R}^{4 \times 1} \quad (4.10)$$

is satisfied. Obtaining the Z candidate

$$\mathbf{v}_z = \left(\mathbf{b}_z^\top\right)_{w_z}^\dagger \boldsymbol{\tau}_z \in \mathbb{R}^{4 \times 1} \quad (4.11)$$

Checking each component of  $\mathbf{v}_z$  if the allocated thrust are feasible;

$$\mathbf{A}\mathbf{v}_z \leq \mathbf{f}_{sat}. \quad (4.12)$$

If  $\mathbf{v}_z$  is infeasible, the magnitude is adjusted with a gain,  $k_1 \in [0,1]$ ,

$$\begin{aligned} \kappa \mathbf{A}\mathbf{v}_z &= \mathbf{f}_{sat} \\ \Downarrow \\ \kappa_i &= \begin{cases} \frac{f_{sat,i}}{\mathbf{a}_i^\top \mathbf{v}_z}, & \text{if } \mathbf{a}_i^\top \mathbf{v}_z \neq 0 \\ 1, & \text{if } \mathbf{a}_i^\top \mathbf{v}_z = 0 \end{cases}, \quad i = 1, \dots, 8 \end{aligned} \quad (4.13)$$

where  $k_1$  is chosen as the minimum of the set

$$k_1 = \min_{i=1,\dots,8} \{\kappa_i\}. \quad (4.14)$$

Obtaining the Z allocation, satisfying the constraint set, as

$$\mathbf{u}_z = k_1 \mathbf{v}_z = k_1 \left(\mathbf{b}_z^\top\right)_{w_z}^\dagger \boldsymbol{\tau}_z \in \mathbb{R}^{4 \times 1}. \quad (4.15)$$

In the 2<sup>nd</sup> step, let  $\mathbf{v}_{1n} = \mathbf{v}_{2a} + \mathbf{v}_{2n}$  and target the yaw moment,

$$\begin{aligned} \boldsymbol{\tau}_\psi &= \mathbf{b}_\psi^\top (\mathbf{v}_{1a} + \mathbf{v}_{1n}) \\ &= \mathbf{b}_\psi^\top (\mathbf{u}_z + \mathbf{Q}_z (\mathbf{v}_{2a} + \mathbf{v}_{2n})) \\ &= \mathbf{b}_\psi^\top \mathbf{u}_z + \mathbf{b}_\psi^\top \mathbf{Q}_z (\mathbf{v}_{2a} + \mathbf{v}_{2n}). \end{aligned} \quad (4.16)$$

Defining the weighted pseudoinverse for N;

$$\left(\mathbf{b}_\psi^\top \mathbf{Q}_z\right)_{w_\psi}^\dagger := \mathbf{W}_\psi^{-1} \mathbf{Q}_z \mathbf{b}_\psi \left(\mathbf{b}_\psi^\top \mathbf{Q}_z \mathbf{W}_\psi^{-1} \mathbf{Q}_z \mathbf{b}_\psi\right)^{-1} \in \mathbb{R}^{4 \times 1}, \quad (4.17)$$

and the corresponding nullspace orthogonal projection matrix

$$\mathbf{Q}_\psi := \mathbf{I} - \left( \mathbf{b}_\psi^\top \mathbf{Q}_z \right)_{w_\psi}^\dagger \mathbf{b}_\psi^\top \mathbf{Q}_z \in \mathbb{R}^{4 \times 4}. \quad (4.18)$$

Further, defining

$$\mathbf{v}_{2a} := \left( \mathbf{b}_\psi^\top \mathbf{Q}_z \right)_{w_\psi}^\dagger \boldsymbol{\tau}_\psi \quad \text{and} \quad \mathbf{v}_{2n} := \mathbf{Q}_\psi \mathbf{v}_{3a}, \quad \{\mathbf{v}_{2a}, \mathbf{v}_{2n}\} \in \mathbb{R}^{4 \times 1}. \quad (4.19)$$

such that

$$\boldsymbol{\tau}_\psi = \mathbf{b}_\psi^\top \mathbf{u}_z + \mathbf{b}_\psi^\top \mathbf{Q}_z \left( \mathbf{b}_\psi^\top \mathbf{Q}_z \right)_{w_\psi}^\dagger \boldsymbol{\tau}_\psi + \mathbf{b}_\psi^\top \mathbf{Q}_z \mathbf{Q}_\psi \mathbf{v}_{3a} \quad (4.20)$$

is satisfied. Obtaining the N candidate

$$\mathbf{v}_\psi = \left( \mathbf{b}_\psi^\top \mathbf{Q}_z \right)_{w_\psi}^\dagger (\boldsymbol{\tau}_\psi - \mathbf{b}_\psi^\top \mathbf{u}_z) \in \mathbb{R}^{4 \times 1}. \quad (4.21)$$

Checking each component of  $\mathbf{v}_\psi$  if the allocated thrust are feasible;

$$\mathbf{A} \mathbf{Q}_z \mathbf{v}_\psi \leq (\mathbf{f}_{sat} - \mathbf{A} \mathbf{u}_z). \quad (4.22)$$

If  $\mathbf{v}_\psi$  is infeasible, the magnitude is adjusted with a gain,  $k_2 \in [0,1]$ ,

$$\begin{aligned} \kappa \mathbf{A} \mathbf{Q}_z \mathbf{v}_\psi &= (\mathbf{f}_{sat} - \mathbf{A} \mathbf{u}_z) \\ &\Downarrow \\ \kappa_i &= \begin{cases} \frac{(\mathbf{f}_{sat,i} - \mathbf{a}_i^\top \mathbf{u}_z)}{\mathbf{a}_i^\top \mathbf{Q}_z \mathbf{v}_\psi}, & \text{if } \mathbf{a}_i^\top \mathbf{v}_\psi \neq 0 \\ 1, & \text{if } \mathbf{a}_i^\top \mathbf{v}_\psi = 0 \end{cases}, \quad i = 1, \dots, 8 \end{aligned} \quad (4.23)$$

where  $k_2$  is chosen as the minimum of the set

$$k_2 = \min_{i=1, \dots, 8} \{\kappa_i\}. \quad (4.24)$$

Obtaining the N allocation, satisfying the constraint set, as

$$\mathbf{u}_\psi = k_2 \mathbf{v}_\psi = k_2 \left( \mathbf{b}_\psi^\top \mathbf{Q}_z \right)_{w_\psi}^\dagger (\boldsymbol{\tau}_\psi - \mathbf{b}_\psi^\top \mathbf{u}_z) \in \mathbb{R}^{4 \times 1}. \quad (4.25)$$

In the 3<sup>rd</sup> and final step, let  $\mathbf{v}_{2n} = \mathbf{v}_{3a}$  and target the surge and sway forces,

$$\begin{aligned} \boldsymbol{\tau}_{xy} &= \mathbf{B}_{xy}^\top (\mathbf{v}_{1a} + \mathbf{v}_{1n}) \\ &= \mathbf{B}_{xy}^\top (\mathbf{u}_z + \mathbf{Q}_z (\mathbf{v}_{2a} + \mathbf{v}_{2n})) \\ &= \mathbf{B}_{xy}^\top \mathbf{u}_z + \mathbf{B}_{xy}^\top \mathbf{Q}_z \mathbf{u}_\psi + \mathbf{B}_{xy}^\top \mathbf{Q}_z \mathbf{v}_{3a} \in \mathbb{R}^{2 \times 1}. \end{aligned} \quad (4.26)$$

Defining the weighted pseudoinverse for XY;

$$\left( \mathbf{B}_{xy}^\top \mathbf{Q}_z \right)_{w_{xy}}^\dagger := \mathbf{W}_{xy}^{-1} \mathbf{Q}_z \mathbf{B}_{xy} \left( \mathbf{B}_{xy}^\top \mathbf{Q}_z \mathbf{W}_{xy}^{-1} \mathbf{Q}_z \mathbf{B}_{xy} \right)^{-1} \in \mathbb{R}^{4 \times 2}. \quad (4.27)$$

Further, defining

$$\mathbf{v}_{3a} := \left( \mathbf{B}_{xy}^\top \mathbf{Q}_{z\psi} \right)_{w_{xy}}^\dagger \boldsymbol{\tau}_{xy} \in \mathbb{R}^{4 \times 1} \quad (4.28)$$

such that

$$\boldsymbol{\tau}_{xy} = \mathbf{B}_{xy}^\top \mathbf{u}_z + \mathbf{B}_{xy}^\top \mathbf{Q}_z \mathbf{u}_\psi + \mathbf{B}_{xy}^\top \mathbf{Q}_{z\psi} \left( \mathbf{B}_{xy}^\top \mathbf{Q}_{z\psi} \right)_{w_{xy}}^\dagger \boldsymbol{\tau}_{xy} \quad (4.29)$$

is satisfied. Obtaining the XY candidate

$$\mathbf{v}_{xy} = \left( \mathbf{B}_{xy}^\top \mathbf{Q}_{z\psi} \right)_{w_{xy}}^\dagger \left( \boldsymbol{\tau}_{xy} - \mathbf{B}_{xy}^\top (\mathbf{u}_z + \mathbf{Q}_z \mathbf{u}_\psi) \right) \in \mathbb{R}^{4 \times 1}. \quad (4.30)$$

Checking each component that the allocated thrust are feasible;

$$\mathbf{A} \mathbf{Q}_{z\psi} \mathbf{v}_{xy} \leq (\mathbf{f}_{sat} - \mathbf{A} \mathbf{u}_z - \mathbf{A} \mathbf{Q}_z \mathbf{u}_\psi). \quad (4.31)$$

If  $\mathbf{v}_{xy}$  is infeasible, the magnitude is adjusted with a gain,  $k_3 \in [0,1]$ ,

$$\begin{aligned} \kappa \mathbf{A} \mathbf{Q}_{z\psi} \mathbf{v}_{xy} &= (\mathbf{f}_{sat} - \mathbf{A} \mathbf{u}_z - \mathbf{A} \mathbf{Q}_z \mathbf{u}_\psi) \\ &\Downarrow \\ \kappa_i &= \begin{cases} \frac{(\mathbf{f}_{sat,i} - \mathbf{a}_i^\top \mathbf{u}_z - \mathbf{a}_i^\top \mathbf{u}_\psi)}{\mathbf{a}_i^\top \mathbf{Q}_{z\psi} \mathbf{v}_{xy}}, & \text{if } \mathbf{a}_i^\top \mathbf{v}_{xy} \neq 0 \\ 1, & \text{if } \mathbf{a}_i^\top \mathbf{v}_{xy} = 0 \end{cases}, \quad i = 1, \dots, 8 \end{aligned} \quad (4.32)$$

where  $k_3$  is chosen as the minimum of the set

$$k_3 = \min_{i=1, \dots, 8} \{\kappa_i\}. \quad (4.33)$$

Obtaining the N allocation, satisfying the constraint set, as

$$\mathbf{u}_{xy} = k_3 \mathbf{v}_{xy} = k_3 \left( \mathbf{B}_{xy}^\top \mathbf{Q}_{z\psi} \right)_{w_{xy}}^\dagger \left( \boldsymbol{\tau}_{xy} - \mathbf{B}_{xy}^\top (\mathbf{u}_z + \mathbf{Q}_z \mathbf{u}_\psi) \right) \in \mathbb{R}^{4 \times 1}. \quad (4.34)$$

The total allocated control vector is then

$$\mathbf{u}_{3step} = \mathbf{u}_z + \mathbf{Q}_z \mathbf{u}_\psi + \mathbf{Q}_z \mathbf{Q}_\psi \mathbf{u}_{xy} \in \mathbb{R}^{4 \times 1}. \quad (4.35)$$

### 4.3 Simulation to show performance of algorithm

To show performance of the thrust allocation algorithm, the following test regime, within the MATLAB®/Simulink software, are proposed to ensure functionality of the algorithm;

1. Open-loop; script testing, giving a commanded "ramp-up" thrust vector,  $\boldsymbol{\tau}_{c, ramp-up}$ .
2. Closed-loop; DP in set-point  $\boldsymbol{\eta} = [10 \ 10 \ 50 \ 45]^\top$  subject to a current velocity,  $\boldsymbol{v}_{current} = 1.15$  [knots] with a direction North (0 [deg] in the NED frame).

The same proposed test regime will be applied to the thrust allocation algorithms to be presented in following chapters. Then, a comparative simulation case study is carried out and presented in Chapter 7.

As there are no improvements to be gained in the heave (Z) allocation for Minerva, the focus of the following discussions will be on the horizontal DOF's {X Y N}.

Further, the recursive nullspace-based thrust allocation implemented in two versions; One recursive 3-step where the thruster weighting matrices are defined as the identity matrix. And one recursive 3-step where the thruster weighting matrix for the 2<sup>nd</sup> step, allocating the yaw moment, is given such that the thrusters are utilized to favor the yaw moment. The thruster weighting matrices utilized for the 3-step recursive, and the 3-step weighted recursive algorithm are given in (4.36).

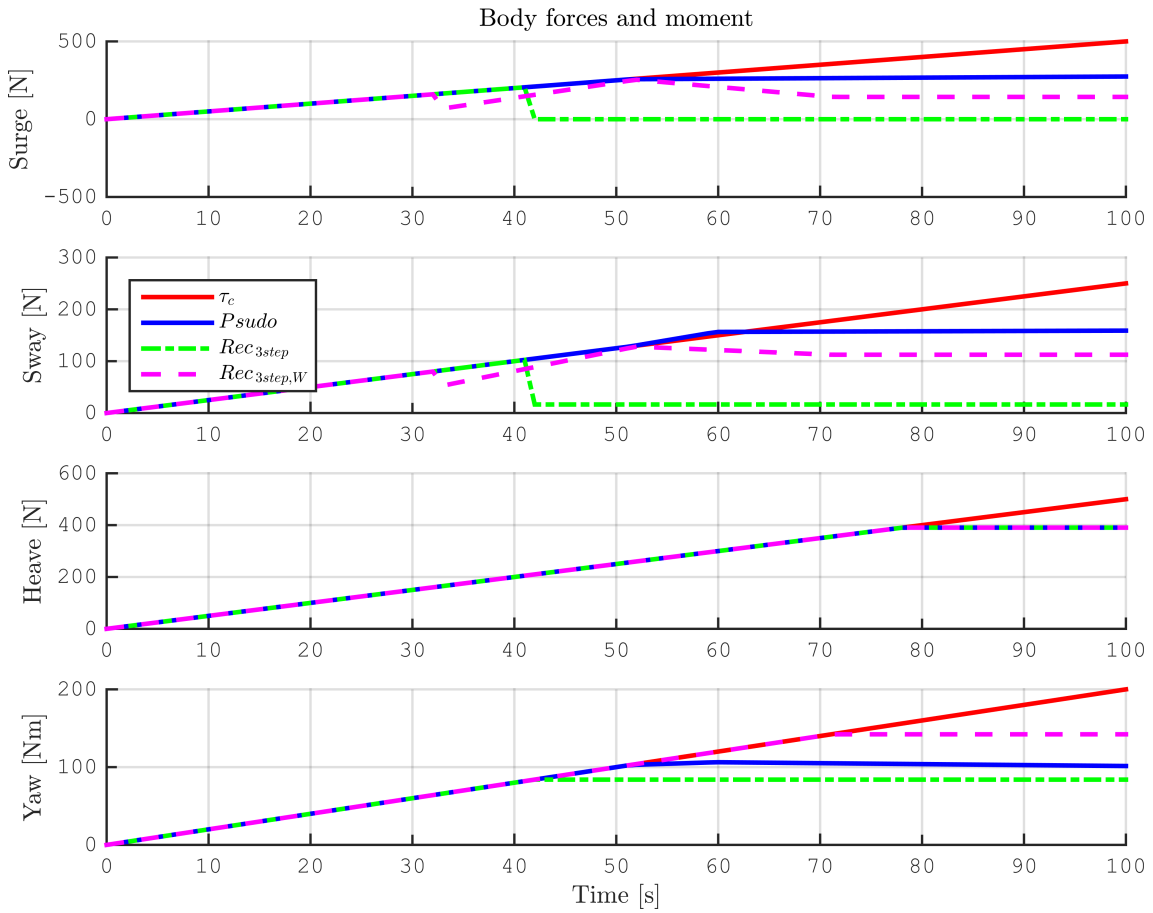
$$\begin{aligned} \text{3-step;} \quad & \{\mathbf{W}_z, \mathbf{W}_\psi, \mathbf{W}_{xy}\} = \mathbf{I}_{4 \times 4} \\ \text{weighted 3-step;} \quad & \{\mathbf{W}_z, \mathbf{W}_{xy}\} = \mathbf{I}_{4 \times 4}, \quad \mathbf{W}_\psi \neq \mathbf{I}_{4 \times 4} \end{aligned} \tag{4.36}$$

#### 4.3.1 MATLAB® script testing

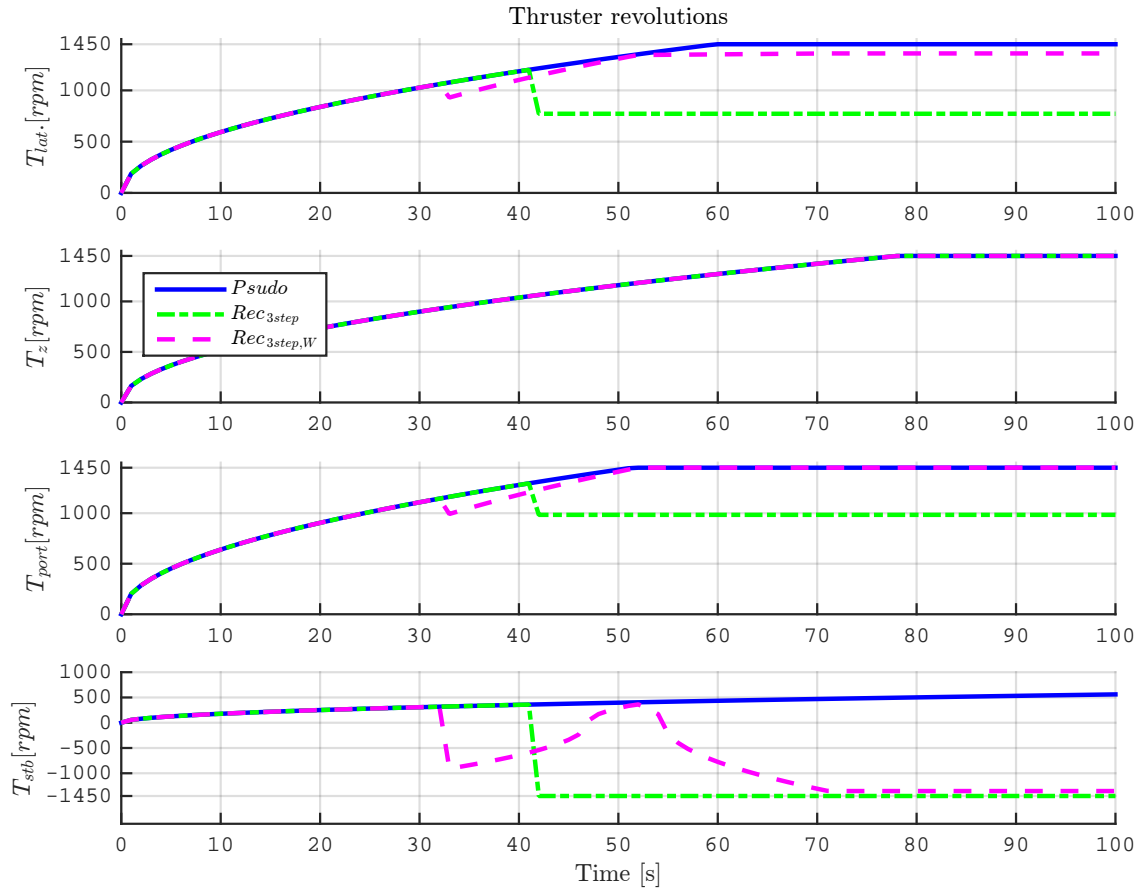
In the 1<sup>st</sup> level of testing, the 3-step recursive nullspace-based thrust allocation algorithm is tested with a  $\boldsymbol{\tau}_{ramp-up}$  commanded thrust vector. This open-loop testing enables the direct comparison of the performance of the algorithms. Thus, the recursive 3-step algorithms are compared with the existing pseudoinverse thrust allocation currently utilized in the Minerva simulation and control system.

Seen in Figure 4.1 are the  $\boldsymbol{\tau}_{ramp-up}$  in red, the resulting forces from  $\boldsymbol{\tau}_{pseudo}$  in blue, and the resulting forces from the recursive algorithms  $\boldsymbol{\tau}_{3step}$  and  $\boldsymbol{\tau}_{3step,W}$  in green and magenta respectively. Comparing the  $\boldsymbol{\tau}_{3step}$  with the  $\boldsymbol{\tau}_{3step,W}$ , it is clear that the thruster weighting matrix utilized in the yaw allocation step for Minerva are of great importance for the overall performance of the algorithm; The recursive 3-step algorithm, without weighting the thrusters usage in the 2<sup>nd</sup> step, achieves about 90 [Nm] yaw moment, whereas the pseudoinverse achieves 100 [Nm]. The weighted recursive 3-step algorithm, utilizing the thrusters to favour yaw allocation, achieves approx. 140 [Nm], close the maximum of 150 [Nm].

In Figure 4.2, showing the thruster usage (in rpm), giving the resulting forces from Figure 4.1; After approx. 42 seconds the recursive 3-step algorithm saturates the starboard thruster ( $T_{stb}$ ). As the algorithm reaches saturation, the allocation is kept at that level. This leaves the port thruster ( $T_{port}$ ) utilized at 1 000 [rpm], whilst the lateral thruster  $T_{lat}$  is utilized at approx. 1 400 [rpm]. Seen in Figure 4.1, when  $T_{stb}$  saturates at 42 seconds, the maximum yaw moment from the recursive 3-step algorithm is reach at approx. 90 [Nm], as previously stated. At the same time, both the surge and sway force are dropped to more or less zero force. This is expected, as this is how the recursive algorithm is designed.



**Figure 4.1:** 1<sup>st</sup> level of testing; Testing the 3-step recursive nullspace-based thrust allocation algorithm with a  $\tau_{ramp-up}$  control vector. Showing the resulting body forces and moment.



**Figure 4.2:** 1<sup>st</sup> level of testing; Testing the 3-step recursive nullspace-based thrust allocation algorithm with a  $\tau_{ramp-up}$  control vector. Showing the thruster responses in rpm.

Further, the weighted recursive 3-step algorithm, where the thruster weighting matrix for the 2<sup>nd</sup> step are implemented to favor the yaw moment, have a significant improved performance. Compared both to the recursive 3-step algorithm, and the pseudoinverse. Further, considering the weighted recursive 3-step algorithm; Looking at Figure 4.2,  $T_{stb}$  is allocated close to the saturation limit in negative rotational direction after approx. 32 seconds. At the same time  $T_{port}$  and  $T_{lat}$  starts to drop. Then, the allocation to  $T_{stb}$  is reduced, while  $T_{port}$  and  $T_{lat}$  are allocated further until  $T_{port}$  saturates after approx. 50 seconds. The saturation of  $T_{port}$  gives the maximum level of allocated surge, and sway force from the weighted recursive 3-step algorithm; The surge force is allocated at the same level as the surge allocation from the pseudoinverse, approx. 250 [N]. The sway force is allocated less than the pseudoinverse, approx. 150 [N] and 110 [N] respectively. After  $T_{port}$  saturates,  $T_{stb}$  is again allocated in negative rotational direction, until  $T_{stb}$  saturates after approx. 70 seconds. This is when the maximum allocated yaw moment from the weighted recursive 3-step algorithm is reached at approx. 140 [Nm].

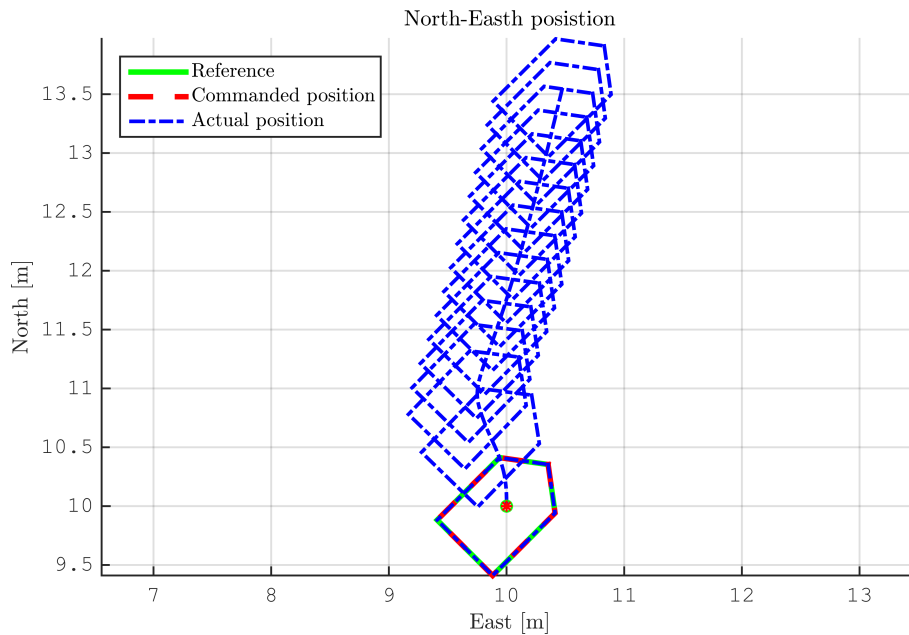
For the heave allocation, as expected all three algorithms allocate identically, since there are none improvements to be gained for Minerva in heave due to the thruster configuration.



### 4.3.2 MATLAB<sup>®</sup>/Simulink software simulation

Based on the findings in subsection 4.3.1, the weighted recursive 3-step nullspace-based thrust allocation algorithm is tested in the 2<sup>nd</sup> level of the proposed testing regime. The MATLAB<sup>®</sup>/Simulink simulation, running the algorithm in a closed loop system, prevents the direct comparison of the different thrust allocation algorithms (unlike the the case for the 1<sup>st</sup> level of testing) since the allocation in time step  $t_{n+1}$  is dependent on the response in time step  $t_n$ . Thus, the aim of the 2<sup>nd</sup> level of testing is to show that the the proposed thrust allocation algorithm can run the system in closed-loop, and the performance of the algorithm will be discussed. The errors from the algorithm will be discussed in in Chapter 7.

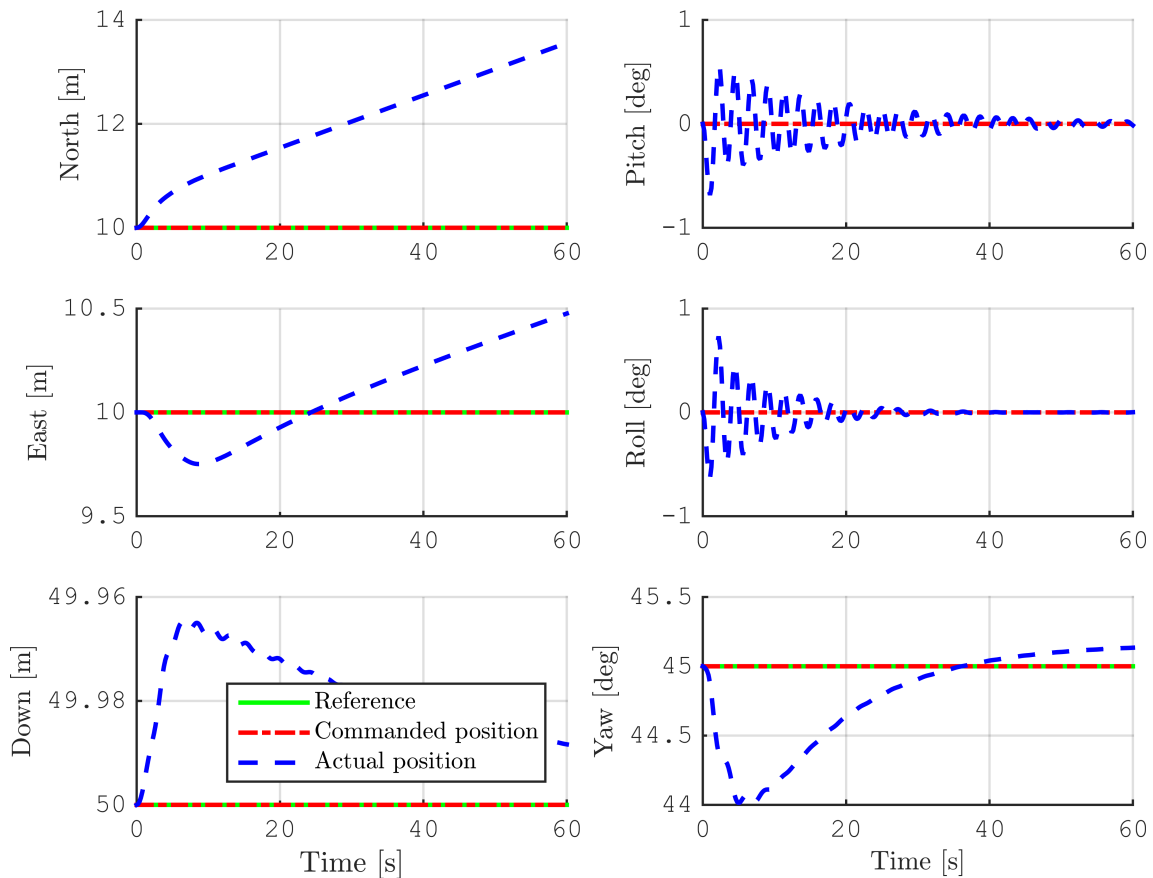
Conducting the 2<sup>nd</sup> level of testing, simulating DP in set-point  $\boldsymbol{\eta} = [10 \ 10 \ 50 \ 45]^T$  subject to a current velocity of  $\mathbf{v}_{current} = 1.15$  [knots] with a northern direction (0 [deg] in the NED frame).



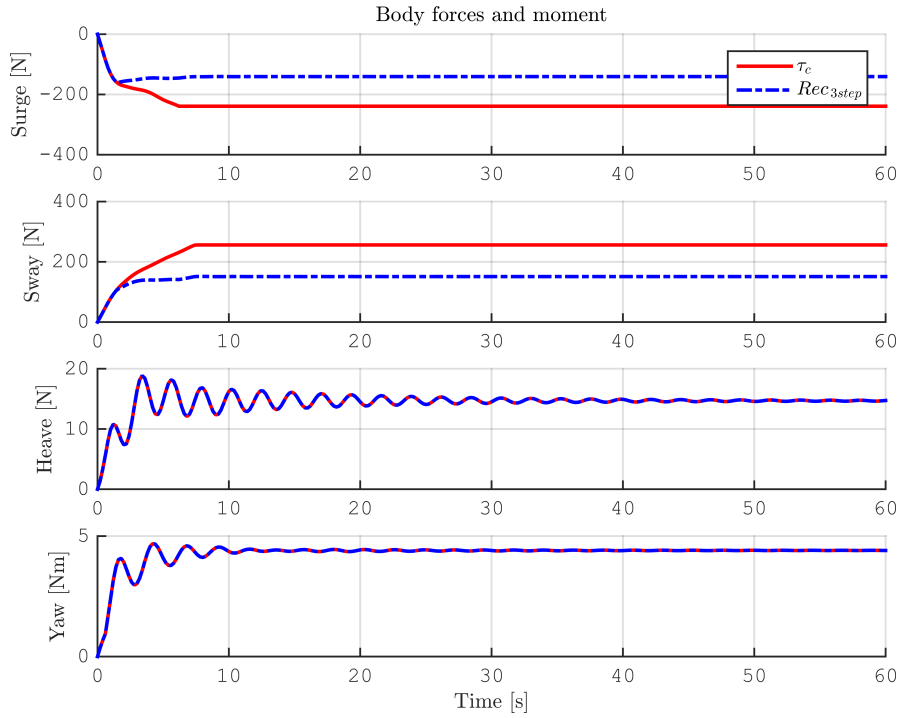
**Figure 4.3:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the weighted recursive 3-step nullspace-based thrust allocation algorithm in closed-loop. Showing the ROV position in the North-East plane.

Figure 4.3 show the position of the Minerva in the North-East plane. Seen in the figure; The current is too strong for the Minerva to be able to keep the commanded position. Thus, as yaw allocation has priority over surge and sway allocation, the ROV starts to drift while the thrust allocation algorithm utilizes the thrusters to obtain the commanded yaw angle.

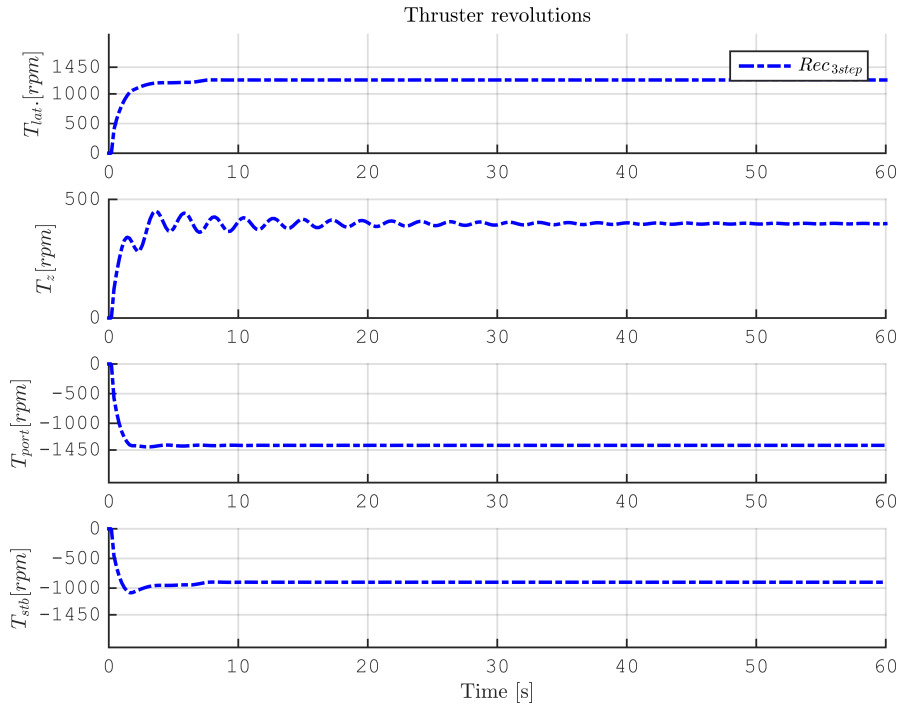
In Figure 4.4 the 6 DOF response of the Minerva are seen; The 3 translational DOF,  $\{1, 2, 3\}$  in the left column. And the 3 rotational DOF,  $\{4, 5, 6\}$  in the right column. As seen in the figure, the ROV are modeled with coupling effect between the un-actuated (DOF  $\{4, 5\}$ ), and the actuated (DOF  $\{1, 2, 3, 6\}$ ). Important to notice are that the un-actuated DOF are passively stable, with (small) exponential decaying responses. Further, looking at DOF 3; It is seen that the ROV is modeled positively buoyant. This is coherent with the case for the real Minerva. As the simulation is started, the ROV starts to rise, before the thrust allocation counteracts and keeps the ROV at the desired depth of 50 [m]. Then, looking at the responses in DOF  $\{1, 2, 6\}$ ; The north and east position is moving further and further away from the desired position, while the yaw angle converges to the desired heading of 45 [deg]. This is consistent with what is seen in Figure 4.3.



**Figure 4.4:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the weighted recursive 3-step nullspace-based thrust allocation algorithm in closed-loop. Showing the 6 DOF response of the ROV.



**Figure 4.5:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the weighted recursive 3-step nullspace-based thrust allocation algorithm in closed-loop. Showing the resulting body forces and moment.



**Figure 4.6:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the weighted recursive 3-step nullspace-based thrust allocation algorithm in closed-loop. Showing the allocated thruster revolutions.

Figure 4.5 shows the resulting body forces and moment, from the allocated thruster usage, seen in Figure 4.6. The commanded force vector ( $\boldsymbol{\tau}_c$ ) is seen in red, while the response from the recursive 3-step thrust allocation algorithm are seen in blue. When the port thruster ( $T_{port}$ ) saturates at -1450 [rpm] after approx. 2 seconds, further allocation to surge and sway stops. This is due to the fact that yaw are prioritized, and allocated in the step prior to the surge and sway allocation. Thus when  $T_{port}$  saturates allocating the yaw moment, all thrusters capacity (according to the recursive algorithm) are utilized, and no further surge and sway allocation are possible.

Further, looking at the surge and sway force in Figure 4.5; When the commanded forces are not obtained in time step  $t_n$ , the command is increased further in time step  $t_{n+1}$ . Thus, continuously increasing the commands until the limits of the controller are reached.

In Chapter 7, the position error  $\boldsymbol{\eta}_{error} = |\boldsymbol{\eta}_{des} - \boldsymbol{\eta}_{actual}|$ , and the forces error  $\boldsymbol{\tau}_{error} = |\boldsymbol{\tau}_{cmd} - \boldsymbol{\tau}_{alloc}|$  of the recursive nullspace-based thrust allocation algorithm are compared with the corresponding errors resulting from the pseudoinverse thrust allocation.

# *Constrained optimal thrust allocation*

In the constrained optimal thrust allocation problem, a standard quadratic programming (QP) problem formulation is chosen. In general, the QP can be formulated according to (5.1);

$$\underset{\mathbf{x}}{\operatorname{argmin}}\{ \mathbf{x}^\top \mathbf{H} \mathbf{x} + \mathbf{f}^\top \mathbf{x} \}, \quad (5.1)$$

subject to the linear equality, and inequality constraints

$$\begin{aligned} \mathbf{A} \mathbf{x} &= \mathbf{b}, \\ \mathbf{C} \mathbf{x} &\leq \mathbf{d}. \end{aligned} \quad (5.2)$$

As there are none linear term in the thrust allocation problem, the  $\mathbf{f}$  vector is a zero vector, i.e.  $\mathbf{f} \equiv \mathbf{0}$ . Thus, the quadratic term,  $\mathbf{x}^\top \mathbf{H} \mathbf{x}$ , is applied to the linear thruster model from (4.2).

## 5.1 Problem formulation

Based on (5.1), the optimal thrust allocation problem is set up as a standard QP problem where the thrust force is minimized with respect to thrust force squared. To allow for thruster saturation, slack variables  $\mathbf{s}$  are introduced. The equality constraint in (4.2) is extended to  $\mathbf{B} \mathbf{u} = \boldsymbol{\tau} + \mathbf{s}$ , i.e. ensuring  $\exists \mathbf{u}_d | \mathbf{B} \mathbf{u}_d = \boldsymbol{\tau}_c + \mathbf{s}$ . The cost function of the optimization problem is then given according to (5.3),

$$[\mathbf{u}_d, \mathbf{s}_d] = \underset{\mathbf{u}, \mathbf{s}}{\operatorname{argmin}}\{ \mathbf{u}^\top \mathbf{W} \mathbf{u} + \mathbf{s}^\top \mathbf{Q} \mathbf{s} \}, \quad (5.3)$$

where both  $\mathbf{W}$  and  $\mathbf{Q}$  are positive definite diagonal matrices. The slack variables should not assume values unless some thruster has saturated. To achieve this, slack variables are made much more expensive than  $\mathbf{u}$ , by selecting  $\mathbf{Q} \gg \mathbf{W}$ . By order of magnitude  $\geq 1\,000$  times. Further, the QP should satisfy the following linear equality, and inequality constraints;

$$\begin{aligned} \mathbf{B} \mathbf{u} &= \boldsymbol{\tau}_c + \mathbf{s}, \\ \mathbf{C} \mathbf{u} &\leq \mathbf{d}. \end{aligned} \quad (5.4)$$

Choosing the problem; quadratic in the states, and choosing the weighting matrices positive definite, i.e.  $\{\mathbf{W}, \mathbf{Q}\} > 0$ . guarantees the set to be convex, and a global minimum that solves the problem can be found.

## 5.2 Design of optimal thrust allocation for Minerva

Defining the new state vectors;

$$\begin{aligned} \mathbf{z} &:= [\mathbf{u}^\top \quad \mathbf{s}^\top]^\top \in \mathbb{R}^{8 \times 1}, \\ \mathbf{p} &:= [\boldsymbol{\tau}^\top \quad \mathbf{u}_{min}^\top \quad \mathbf{u}_{max}^\top]^\top \in \mathbb{R}^{12 \times 1}, \end{aligned} \quad (5.5)$$

transforming the optimization problem defined in (5.3), into a QP problem parameterized in  $\mathbf{z}$  according to (5.6);

$$\mathbf{z}_d = \underset{\mathbf{z}}{\operatorname{argmin}} \{ \mathbf{z}^\top \boldsymbol{\Phi} \mathbf{z} \} \quad (5.6)$$

subject to the constraints

$$\begin{aligned} \mathbf{A}_1 \mathbf{z} &= \mathbf{C}_1 \mathbf{p}, \\ \mathbf{A}_2 \mathbf{z} &\leq \mathbf{C}_2 \mathbf{p}, \end{aligned} \quad (5.7)$$

where the combined weighting matrix  $\boldsymbol{\Phi}$ , is given by (5.8),

$$\boldsymbol{\Phi} = \begin{bmatrix} \mathbf{W}_{4 \times 4} & \mathbf{0}_{4 \times 4} \\ \mathbf{0}_{4 \times 4} & \mathbf{Q}_{4 \times 4} \end{bmatrix} \in \mathbb{R}^{8 \times 8}. \quad (5.8)$$

The weighting matrices for thruster usage ( $\mathbf{W}$ ), and for the slack variables ( $\mathbf{Q}$ ), are chosen as diagonal matrices according to (5.9);

$$\mathbf{W} = \mathbf{I}_{4 \times 4} \quad , \quad \mathbf{Q} = 1\,000 \cdot \mathbf{I}_{4 \times 4}. \quad (5.9)$$

Further, the equality constraints matrices are given according to (5.10);

$$\begin{aligned} \mathbf{A}_1 &= [\mathbf{B}_{4 \times 4} \quad -\mathbf{I}_{4 \times 4}] \in \mathbb{R}^{4 \times 8}, \\ \mathbf{C}_1 &= [\mathbf{I}_{4 \times 4} \quad \mathbf{0}_{4 \times 4} \quad \mathbf{0}_{4 \times 4}] \in \mathbb{R}^{4 \times 12}, \end{aligned} \quad (5.10)$$

and lastly, the inequality constraints matrices are given according to (5.11);

$$\begin{aligned} \mathbf{A}_2 &= \begin{bmatrix} -\mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 4} \\ \mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 4} \end{bmatrix} \in \mathbb{R}^{8 \times 8}, \\ \mathbf{C}_2 &= \begin{bmatrix} \mathbf{0}_{4 \times 4} & -\mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 4} \\ \mathbf{0}_{4 \times 4} & \mathbf{0}_{4 \times 4} & \mathbf{I}_{4 \times 4} \end{bmatrix} \in \mathbb{R}^{8 \times 12}. \end{aligned} \quad (5.11)$$

The optimization problem is then solved numerically with the MATLAB<sup>®</sup> function `quadprog.m`, returning the  $\mathbf{z}_d$  solving the problem defined in (5.6)-(5.7).

The MATLAB<sup>®</sup> QP solver is called with the sequence;

$$[\mathbf{z}_d] = \operatorname{quadprog}(\mathbf{PHI}, \mathbf{f}, \mathbf{A}_2, \mathbf{b}_2, \mathbf{A}_1, \mathbf{b}_1), \quad (5.12)$$

where  $\mathbf{PHI} = \boldsymbol{\Phi}$ ,  $\mathbf{b}_1 := \mathbf{C}_1 \mathbf{p} \in \mathbb{R}^{4 \times 1}$  and  $\mathbf{b}_2 := \mathbf{C}_2 \mathbf{p} \in \mathbb{R}^{8 \times 1}$ .

### 5.3 Simulation to show performance of algorithm

To verify the functionality of the constrained optimal thrust allocation algorithm, the test cases proposed in section 4.3, are performed on the algorithm. Repeated here for readability;

1. Open-loop; script testing, giving a commanded "ramp-up" thrust vector,  $\boldsymbol{\tau}_{c, ramp-up}$ .
2. Closed-loop; DP in set-point  $\boldsymbol{\eta} = [10 \ 10 \ 50 \ 45]^T$  subject to a current velocity,  $\boldsymbol{v}_{current} = 1.15$  [knots] with a direction North (0 [deg] in the NED frame).

Further, for the optimal thrust allocation algorithm, DOF prioritization can be achieved, by penalizing the slack variable corresponding to the DOF to be prioritized. If this penalty is made much bigger (by order of magnitude  $\geq 1\ 000$  times) than the other slack variables, this will give a strict prioritization of this DOF. Thus, two optimal thrust allocation algorithms are implemented; One optimal algorithm, prioritize the DOFs {XYZN} equally. And one optimal weighted, prioritize the DOF's according to {N, XYZ}. The slack variable weighting matrices utilized for the optimal algorithm, and weighted optimal algorithm, are given in (6.23) as  $\mathbf{Q}$  and  $\mathbf{Q}_w$  respectively;

$$\mathbf{Q} = 1\ 000 \text{ diag} \left( \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \right) , \quad \mathbf{Q}_w = 1\ 000 \text{ diag} \left( \begin{bmatrix} 1 & 1 & 1 & 1000 \end{bmatrix} \right). \quad (5.13)$$

#### 5.3.1 MATLAB<sup>®</sup> script testing

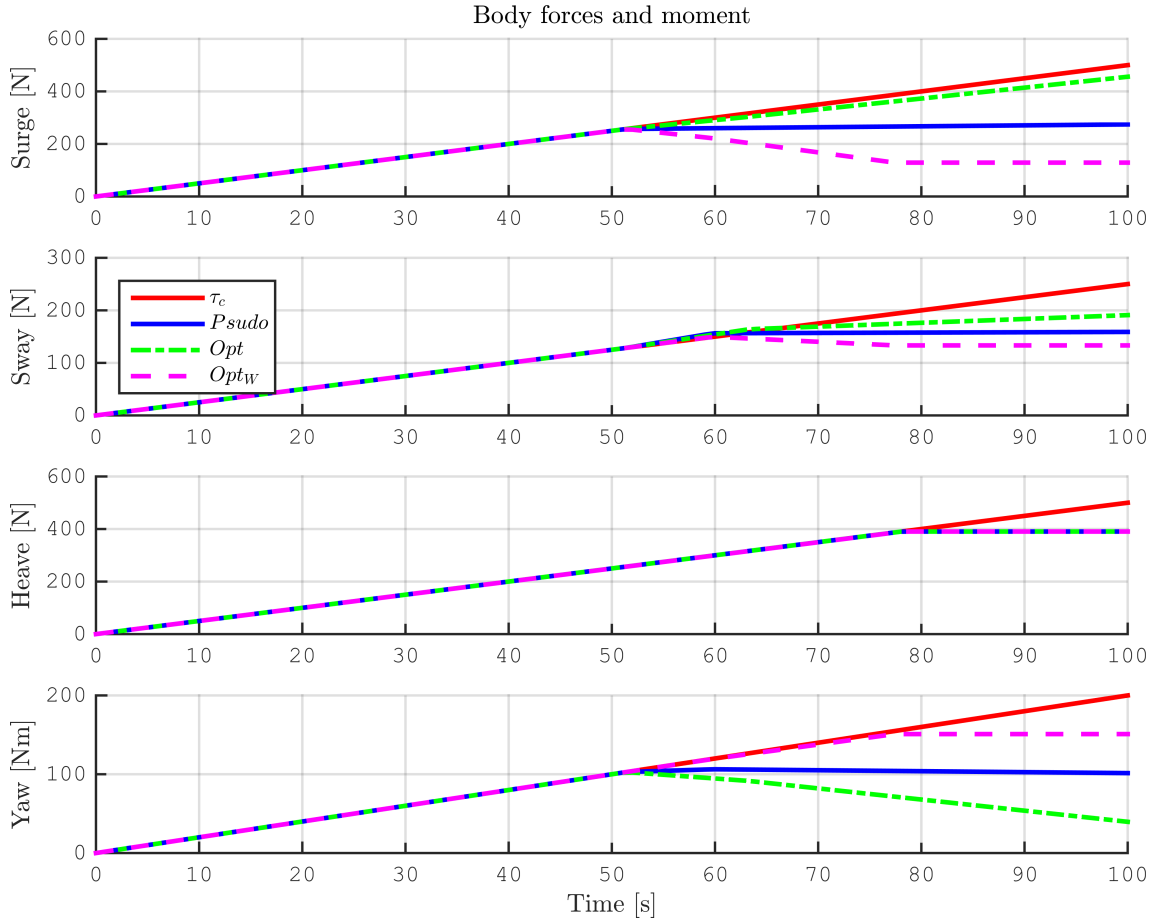
In the 1<sup>st</sup> level of testing, the optimal thrust allocation algorithms is tested with a  $\boldsymbol{\tau}_{ramp-up}$  commanded thrust vector. This open-loop testing, enables the direct comparison of the performance of the algorithms. Thus, the optimal algorithm is compared with the existing pseudoinverse thrust allocation currently utilized in the Minerva simulation, and control systems.

Seen in Figure 5.1 are the  $\boldsymbol{\tau}_{ramp-up}$  in red, the resulting forces from  $\boldsymbol{\tau}_{pseudo}$  in blue, and the resulting forces from the optimal algorithms  $\boldsymbol{\tau}_{opt}$  and  $\boldsymbol{\tau}_{opt,W}$  in green and magenta respectively.  $\boldsymbol{\tau}_{opt}$  prioritize the DOFs {XYZN} equally, while  $\boldsymbol{\tau}_{opt,W}$  prioritize according to {N, XYZ}, as previously stated. In Figure 5.2, the thruster usage (in rpm), giving the resulting body forces and moment in Figure 5.1, are seen.

Comparing the optimal algorithm with the pseudoinverse, it is seen that when the port thruster ( $T_{port}$ ) saturates at approx. 50 seconds, the maximum yaw moment, and surge force from the pseudoinverse algorithm is obtain. The sway force is allocated further, until approx. 60 seconds when the lateral thruster ( $T_{lat}$ ) saturates. This, leaving the starboard thruster ( $T_{stb}$ ), utilized at approx. 1/3 of the capacity in positiv direction. For the optimal algorithm,  $T_{port}$  also saturates at approx. 50 seconds. When saturating a thruster, the optimal algorithm start to allocated to the slack variable with the lowest cost. At 50 seconds with  $\tau_N < \tau_Y < \tau_X$ , the least cost for the optimal algorithm are to allocate to the slack variable corresponding to

yaw. This causes the max obtained yaw moment from the optimal algorithm to be achieved, at the same level as for the pseudoinverse, before declining in amplitude to approx. 1/3 of the pseudoinverse.

When the pseudoinverse keeps  $T_{stb}$  at a constant revolution, the optimal algorithm continues to utilize  $T_{stb}$  until it saturates at max rpm in positive direction, i.e. 1.450 [rpm]. This causes the optimal algorithm to achieve significant higher surge force, and somewhat improved sway force than the pseudoinverse, but at the expense of approx. 1/3 of the yaw moment.

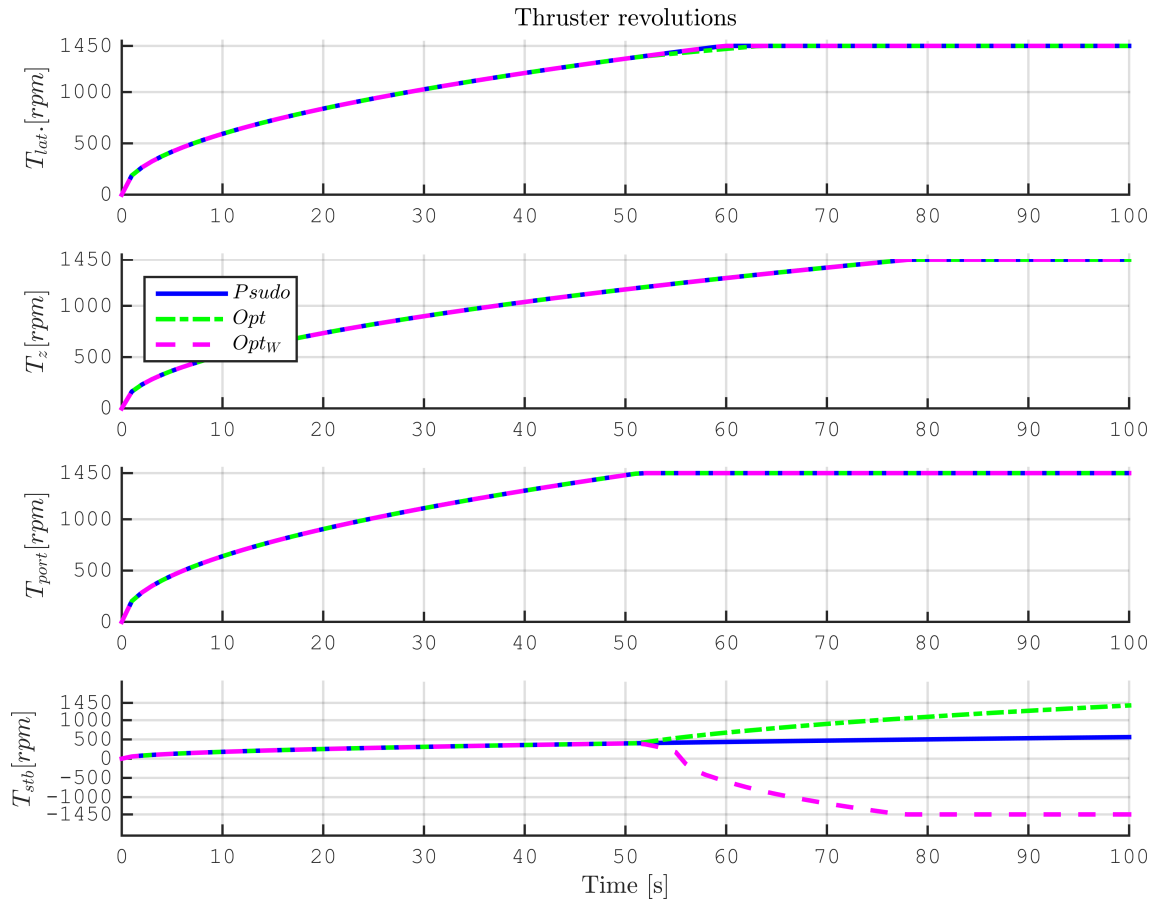


**Figure 5.1:** 1<sup>st</sup> level of testing; Testing the optimal thrust allocation algorithm with a  $T_{ramp-up}$  control vector. Showing the resulting body forces and moment.

Then, comparing the optimal, and the optimal weighted algorithms; It is seen from Figure 5.1 and Figure 5.2 that the saturation occurs at the same time for the optimal and optimal weighted algorithms. Where the optimal algorithm utilizes  $T_{stb}$  to its maximum capacity in positive rotational direction, the optimal weighted algorithm utilized  $T_{stb}$  to its maximum capacity in negative rotational direction. Further, where the optimal algorithm reaches max allocated yaw moment at approx. 50 seconds, when  $T_{port}$  saturates, the optimal weighted algorithm continues to allocate yaw moment until the max obtainable yaw moment for the Minerva is reached, at 150 [nm]. At the same time, the optimal weighted algorithm allocates less to surge; The allocated surge force from the optimal weighted algorithm declines until approx. 1/2 of



what the pseudoinverse algorithm achieves. And approx. 1/4 of what the optimal algorithm achieves. Further, the large yaw moment allocation from the weighted optimal algorithm, comes at the expence of approx. 1/4 less sway force than the optimal algorithm, and approx. 1/8 less than the pseudoinverse algorithm.



**Figure 5.2:** 1<sup>st</sup> level of testing; Testing the optimal thrust allocation algorithm with a  $\tau_{ramp-up}$  control vector. Showing the thrusters rpm's.

The main difference in the total thruster usage by the algorithm are that the optimal algorithm utilizes the full potential of  $T_{port}$  in positive rotation, the optimal weighted algorithm utilizes the full potential of  $T_{port}$  in negative rotation, whilst the pseudoinverse algorithm only utilize  $T_{port}$  at about 1/3 of the capacity.

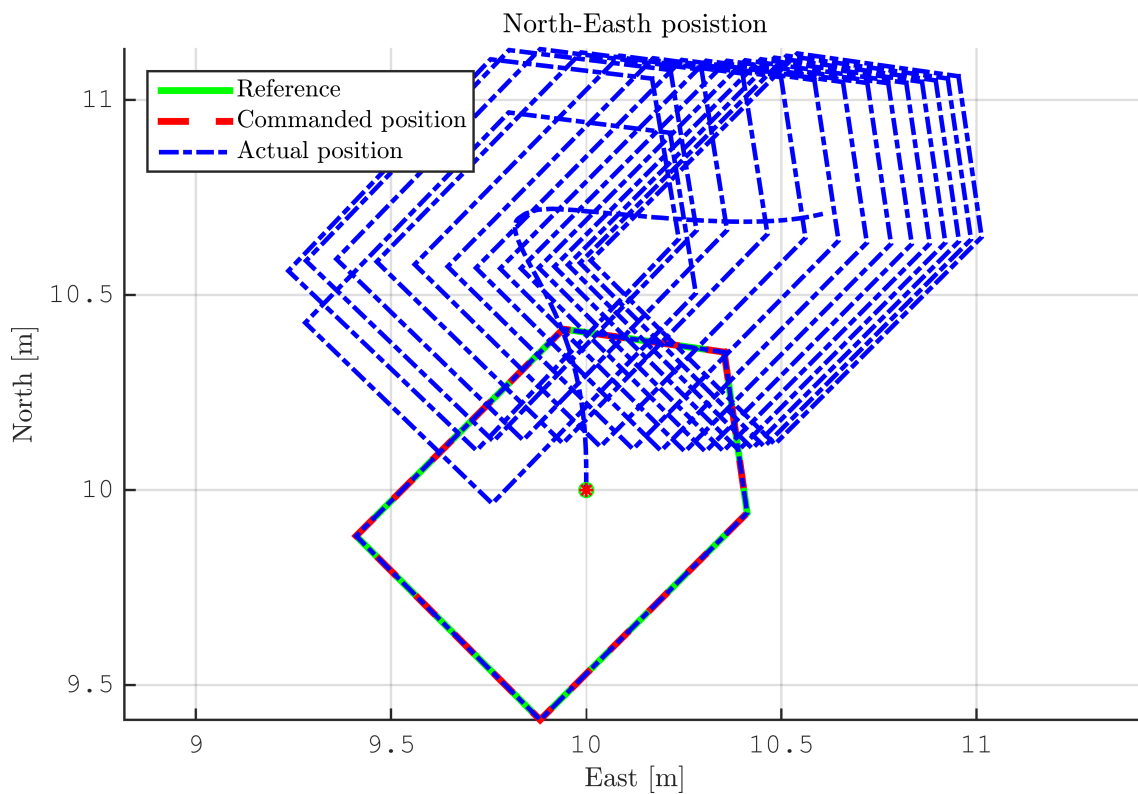
For the heave allocation, again as expected, all three algorithms allocates the heave identically.

### 5.3.2 MATLAB<sup>®</sup>/Simulink software simulation

Based on the findings in subsection 5.3.1, the optimal weighted thrust allocation algorithm is tested in the 2<sup>nd</sup> level of the proposed testing regime. As commented subsection 4.3.2; The MATLAB<sup>®</sup>/Simulink simulation, running the algorithm in a closed loop system, prevents the direct comparison of the different thrust allocation

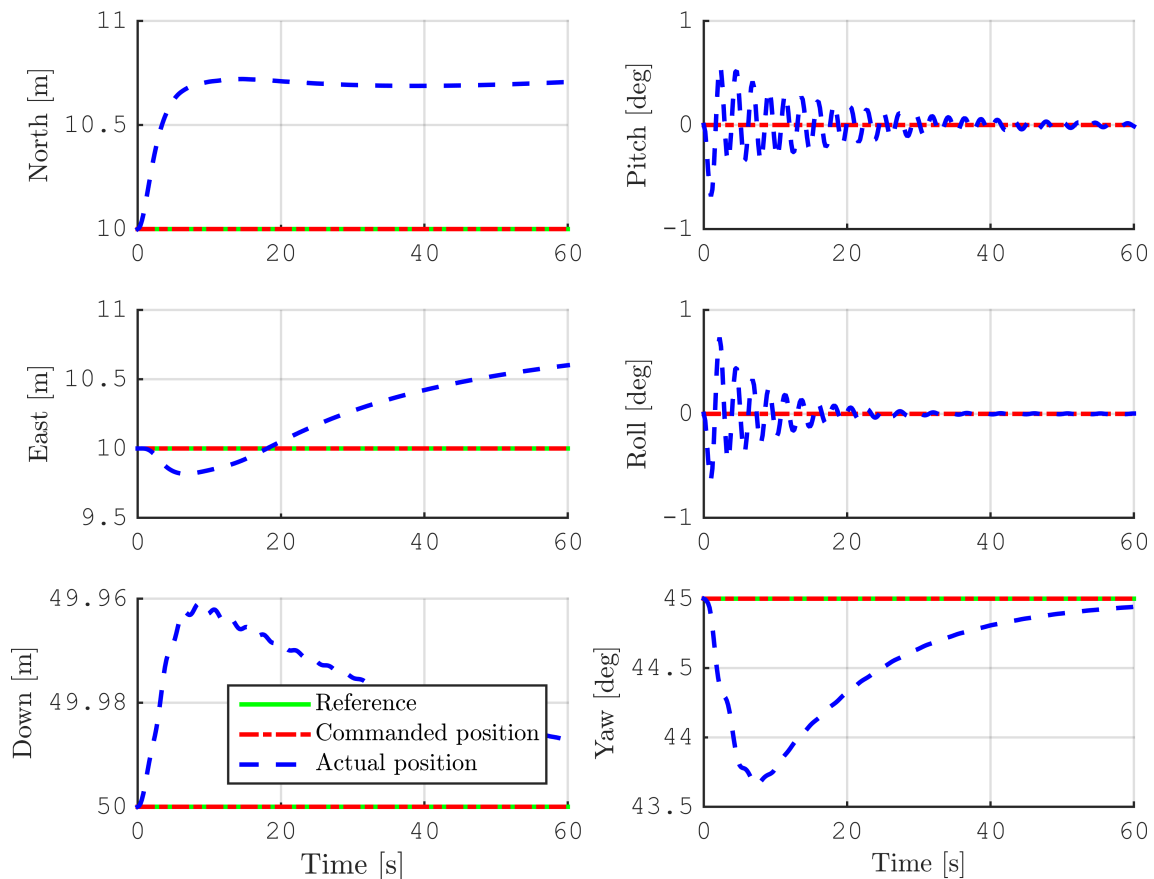
algorithms. Thus, the aim of the 2<sup>nd</sup> level of testing is to show that the the proposed thrust allocation algorithm can run the system in closed-loop, and the performance of the algorithm will be discussed. The errors from the algorithm will be discussed in in Chapter 7.

Conducting the 2<sup>nd</sup> level of testing, simulating DP in set-point  $\boldsymbol{\eta} = [10 \ 10 \ 50 \ 45]^T$  subject to a current velocity of  $\mathbf{v}_{current} = 1.15$  [knots] with a northern direction (0 [deg] in the NED frame).



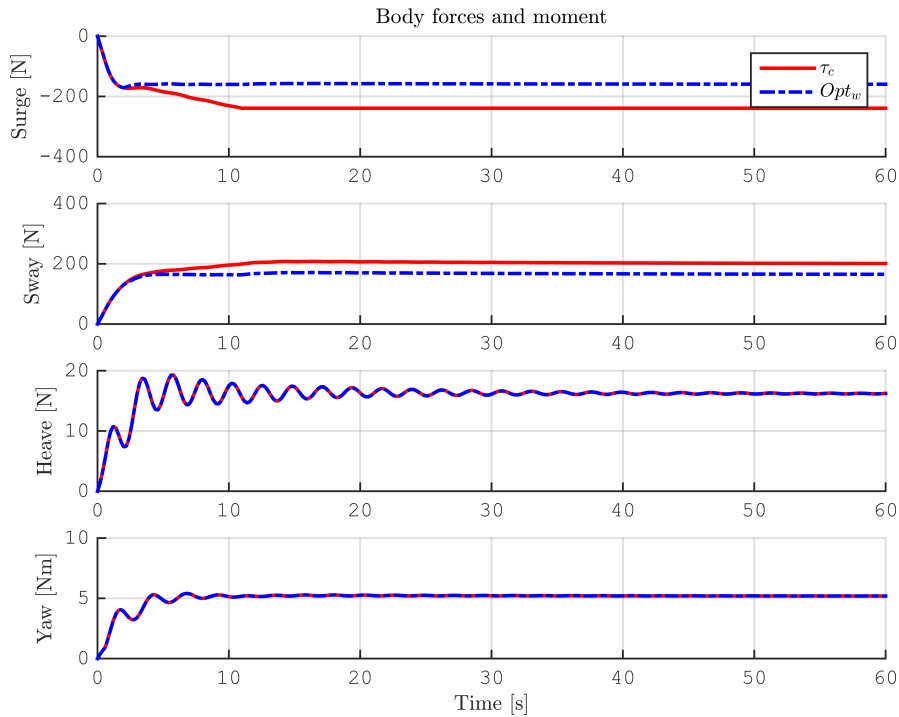
**Figure 5.3:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the optimal weighted thrust allocation algorithm in closed-loop. Showing the ROV position in the North-East plane.

Figure 5.3 show the position of the Minerva in the North-East plane. Seen in the figure; The current is too strong for the Minerva to be able to keep the commanded position. As opposed to the case seen in Figure 4.3, the drift off when running the system with the weighted optimal algorithm converges to a constant deviation. Further, as the weighted optimal algorithm prioritizes yaw over surge and sway, the heading angle converges to the desired heading. This heading priority comes from the large penalty put on the allocation to the slack variable corresponding to yaw.

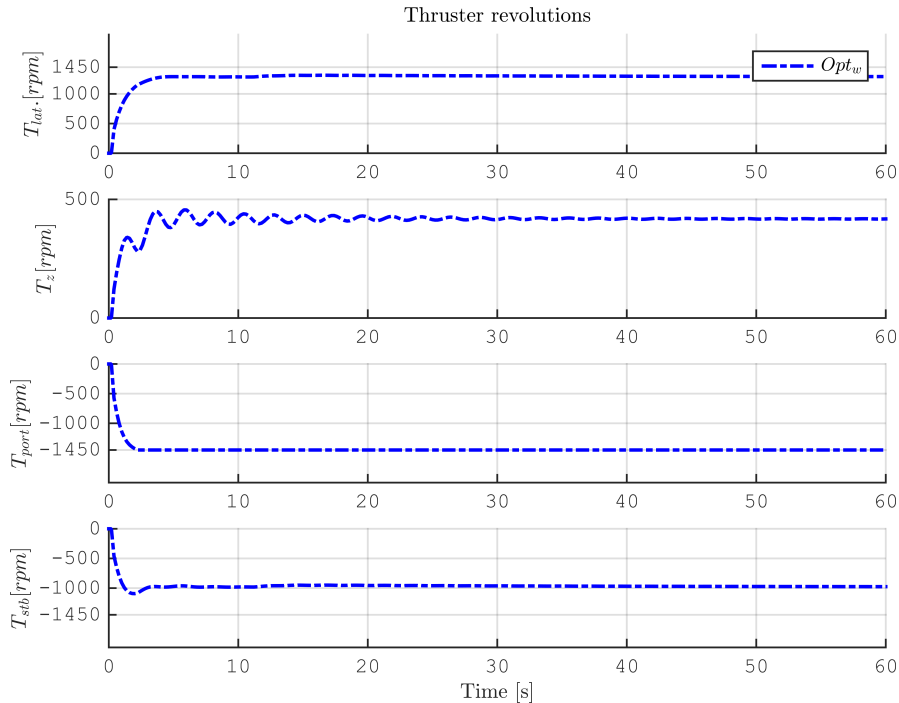


**Figure 5.4:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the optimal weighted thrust allocation algorithm in closed-loop. Showing the 6 DOF response of the ROV.

In Figure 5.4 the 6 DOF response of the Minerva are seen; The 3 translational DOF, {1, 2, 3} in the left column. And the 3 rotational DOF, {4, 5, 6} in the right column. As discussed in subsection 4.3.2; The excitation of the unactuated DOF due to coupling effects, are passively stable with (small) exponentially decaying responses. Further, the positively buoyant ROV starts to rise as the simulation is started, before the thrust allocation counteracts and keeps the ROV at the desired depth of 50 [m]. Then, looking at the responses in DOF {1, 2, 6}; The north and east position is moving away from the desired position, before converging to a constant deviation from the desired position, while the yaw angle converges to the desired heading of 45 [deg]. Again consistent with what is seen in the North-East plot, in Figure 5.3.



**Figure 5.5:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the optimal weighted thrust allocation algorithm in closed-loop. Showing the resulting body forces and moment.



**Figure 5.6:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the optimal weighted thrust allocation algorithm in closed-loop. Showing the allocated thruster revolutions.

Figure 5.5 shows the resulting body forces and moment, from the allocated thruster usage, seen in Figure 5.6. The commanded force vector ( $\boldsymbol{\tau}_c$ ) is seen in red, while the response from the weighted optimal thrust allocation algorithm are seen in blue.

When the port thruster ( $T_{port}$ ) saturates at -1450 [rpm] after approx. 2 seconds, further allocation to surge and sway stops. The cost of allocating surge and sway force have thus become more costly than allocating to the corresponding slack variables. The slack variable corresponding to yaw, have a cost 1000 times more expensive than those of surge and sway. Thus, ensuring the desired yaw moment are obtained.

Further, looking at the surge and sway force in Figure 5.5; When the commanded forces are not obtained in time step  $t_n$ , the command is increased further in time step  $t_{n+1}$ . Thus, continuously increasing the commands until the limits of the controller are reached.

In Chapter 7, the position error  $\boldsymbol{\eta}_{error} = |\boldsymbol{\eta}_{des} - \boldsymbol{\eta}_{actual}|$ , and the forces error  $\boldsymbol{\tau}_{error} = |\boldsymbol{\tau}_{cmd} - \boldsymbol{\tau}_{alloc}|$  of the weighted optimal thrust allocation algorithm are compared with the corresponding errors resulting from the 3-step recursive nullspace-based, and the pseudoinverse thrust allocation algorithms.



# *Constrained optimal recursive thrust allocation*

Combining the *constrained recursive nullspace-based thrust allocation algorithm* (Chapter 4), with the *constrained optimal thrust allocation algorithm* (Chapter 5), gives the basis for the *constrained optimal recursive thrust allocation algorithm* in this chapter.

## 6.1 Problem formulation

The problem formulation for the optimal recursive thrust allocation are identical to the problem formulation for the optimal thrust allocation algorithm, but restated here for readability.

Based on (5.1), the optimal thrust allocation problem is set up as a standard QP problem where the thrust force is minimized with respect to thrust force squared. Further, including slack variables in order to ensure the existence of a solution to the problem, i.e.  $\exists \mathbf{u}_d | \mathbf{B}\mathbf{u}_d = \boldsymbol{\tau}_c + \mathbf{s}$ , i.e.

$$[\mathbf{u}_d, \mathbf{s}_d] = \underset{\mathbf{u}, \mathbf{s}}{\operatorname{argmin}} \{ \mathbf{u}^\top \mathbf{W} \mathbf{u} + \mathbf{s}^\top \mathbf{Q} \mathbf{s} \}, \quad (6.1)$$

satisfying the following linear equality, and inequality constraints;

$$\begin{aligned} \mathbf{B} \mathbf{u} &= \boldsymbol{\tau}_c + \mathbf{s}, \\ \mathbf{C} \mathbf{u} &\leq \mathbf{d}. \end{aligned} \quad (6.2)$$

## 6.2 Design of optimal recursive thrust allocation for Minerva

Defining  $\boldsymbol{\tau}_{z\psi} := \operatorname{col}(\tau_Z, \tau_N)$  and  $\boldsymbol{\tau}_{xy} := \operatorname{col}(\tau_X, \tau_Y)$ , rearranging the  $\mathbf{B}$ -matrix from (4.1) accordingly, such that

$$\begin{bmatrix} \boldsymbol{\tau}_{z\psi} \\ \boldsymbol{\tau}_{xy} \end{bmatrix} = \begin{bmatrix} \mathbf{B}_{z\psi}^\top \\ \mathbf{B}_{xy}^\top \end{bmatrix} \mathbf{u}_{opt.rec.}, \quad \begin{cases} \mathbf{u}_{opt.rec.} & \in \mathbb{R}^{4 \times 1} \\ \mathbf{B}_{z\psi}^\top & \in \mathbb{R}^{2 \times 4} \\ \mathbf{B}_{xy}^\top & \in \mathbb{R}^{2 \times 4} \end{cases}. \quad (6.3)$$

Let  $\mathbf{u}_{opt.rec.} = \mathbf{v}_{1a} + \mathbf{v}_{1n}$  and target the heave force and yaw moment,

$$\boldsymbol{\tau}_{z\psi} = \mathbf{B}_{z\psi}^\top (\mathbf{v}_{1a} + \mathbf{v}_{1n}) \in \mathbb{R}^{2 \times 1}, \quad (6.4)$$

where  $\mathbf{v}_{1a}$  is the control vector to be allocated in the current step, and  $\mathbf{v}_{1n}$  is an auxiliary control for the next step.  $\mathbf{v}_{1n}$  is defined in the kernel of  $\mathbf{B}_{z\psi}^\top$  using the nullspace matrix,  $\mathbf{Q}_{z\psi} := \mathbf{I} - (\mathbf{B}_{z\psi}^\top)^\dagger \mathbf{B}_{z\psi}^\top \in \mathbb{R}^{4 \times 4}$ , i.e.

$$\mathbf{v}_{1n} \in \mathcal{N}(\mathbf{B}_{z\psi}^\top) \Rightarrow \mathbf{v}_{1n} := \mathbf{Q}_{z\psi} \mathbf{v}_2 \quad (6.5)$$

such that

$$\boldsymbol{\tau}_{z\psi} = \mathbf{B}_{z\psi}^\top (\mathbf{v}_{1a} + \mathbf{v}_{1n}) = \mathbf{B}_{z\psi}^\top \mathbf{v}_{1a} + \mathbf{B}_{z\psi}^\top \mathbf{Q}_{z\psi} \mathbf{v}_2 \in \mathbb{R}^{2 \times 1} \quad (6.6)$$

is satisfied. Further, defining the new state vectors;

$$\begin{aligned} \mathbf{z}_{z\psi} &:= \begin{bmatrix} \mathbf{u}_{1a}^\top & \mathbf{s}_{z\psi}^\top \end{bmatrix}^\top \in \mathbb{R}^{6 \times 1}, \\ \mathbf{p}_{z\psi} &:= \begin{bmatrix} \boldsymbol{\tau}_{z\psi}^\top & \mathbf{u}_{min}^\top & \mathbf{u}_{max}^\top \end{bmatrix}^\top \in \mathbb{R}^{10 \times 1}, \end{aligned} \quad (6.7)$$

transforming the 1<sup>st</sup> step of the optimization problem, defined in (6.1), into a QP problem parameterized in  $\mathbf{z}_{z\psi}$  according to (6.8);

$$\mathbf{z}_{z\psi,d} = \underset{\mathbf{z}_{z\psi}}{\operatorname{argmin}} \{ \mathbf{z}_{z\psi}^\top \boldsymbol{\Phi}_1 \mathbf{z}_{z\psi} \} \quad (6.8)$$

subject to the constraints

$$\begin{aligned} \mathbf{A}_{11} \mathbf{z}_{z\psi} &= \mathbf{C}_{11} \mathbf{p}_{z\psi}, \\ \mathbf{A}_{12} \mathbf{z}_{z\psi} &\leq \mathbf{C}_{12} \mathbf{p}_{z\psi}, \end{aligned} \quad (6.9)$$

where the combined weighting matrix  $\boldsymbol{\Phi}_1$  for ZN, is given by (6.10),

$$\boldsymbol{\Phi}_1 = \begin{bmatrix} \mathbf{W}_{4 \times 4} & \mathbf{0}_{4 \times 2} \\ \mathbf{0}_{2 \times 4} & \mathbf{Q}_{z\psi} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 2} \\ \mathbf{0}_{2 \times 4} & \mathbf{Q}_{z\psi} \end{bmatrix} \in \mathbb{R}^{6 \times 6}. \quad (6.10)$$

Further, the equality constraints matrices are given according to (6.11);

$$\begin{aligned} \mathbf{A}_{11} &= \begin{bmatrix} \mathbf{B}_{z\psi}^\top & -\mathbf{I}_{2 \times 4} \end{bmatrix} \in \mathbb{R}^{2 \times 6}, \\ \mathbf{C}_{11} &= \begin{bmatrix} \mathbf{I}_{2 \times 2} & \mathbf{0}_{2 \times 4} & \mathbf{0}_{2 \times 4} \end{bmatrix} \in \mathbb{R}^{2 \times 10}, \end{aligned} \quad (6.11)$$

and lastly, the inequality constraints matrices are given according to (6.12);

$$\begin{aligned} \mathbf{A}_{12} &= \begin{bmatrix} -\mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 2} \\ \mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 2} \end{bmatrix} \in \mathbb{R}^{8 \times 6}, \\ \mathbf{C}_{12} &= \begin{bmatrix} \mathbf{0}_{4 \times 2} & -\mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 4} \\ \mathbf{0}_{4 \times 2} & \mathbf{0}_{4 \times 4} & \mathbf{I}_{4 \times 4} \end{bmatrix} \in \mathbb{R}^{8 \times 10}. \end{aligned} \quad (6.12)$$

The 1<sup>st</sup> step of the optimization problem is then solved numerically with the MATLAB<sup>®</sup> function `quadprog.m`, returning the  $\mathbf{z}_{z\psi,d}$  solving the problem defined in (6.8)-(6.9).

The MATLAB<sup>®</sup> QP solver is called with the sequence

$$[\mathbf{z}_d] = \text{quadprog}(\text{PHI1}, \mathbf{f}, \mathbf{A12}, \mathbf{b12}, \mathbf{A11}, \mathbf{b11}), \quad (6.13)$$

where  $\text{PHI1} = \boldsymbol{\Phi}_1$ ,  $\mathbf{b}_{11} := \mathbf{C}_{11} \mathbf{p}_{z\psi} \in \mathbb{R}^{2 \times 1}$  and  $\mathbf{b}_{12} := \mathbf{C}_{12} \mathbf{p}_{z\psi} \in \mathbb{R}^{8 \times 1}$ .



In the  $2^{nd}$  step, let  $\mathbf{v}_{1n} = \mathbf{v}_2$  and target the surge and sway forces such that

$$\boldsymbol{\tau}_{xy} = \mathbf{B}_{xy}^\top (\mathbf{v}_{1a} + \mathbf{v}_{1n}) = \mathbf{B}_{xy}^\top \mathbf{u}_{z\psi} + \mathbf{B}_{xy}^\top \mathbf{v}_2 \in \mathbb{R}^{2 \times 1} \quad (6.14)$$

is satisfied. Further, defining the new state vectors;

$$\begin{aligned} \mathbf{z}_{xy} &:= \begin{bmatrix} \mathbf{u}_{2a}^\top & \mathbf{s}_{z\psi}^\top \end{bmatrix}^\top \in \mathbb{R}^{4 \times 1}, \\ \mathbf{p}_{xy} &:= \begin{bmatrix} (\boldsymbol{\tau}_{xy} - \mathbf{B}_{xy}^\top \mathbf{u}_{z\psi})^\top & (\mathbf{u}_{min} - \mathbf{u}_{z\psi})^\top & (\mathbf{u}_{max} - \mathbf{u}_{z\psi})^\top \end{bmatrix}^\top \end{aligned} \quad (6.15)$$

transforming the  $2^{nd}$  step of the optimization problem defined in (6.1), into a QP problem parameterized in  $\mathbf{z}_{xy}$ .

$$\mathbf{z}_{xy,d} = \underset{\mathbf{z}_{xy}}{\operatorname{argmin}} \{ \mathbf{z}_{xy}^\top \boldsymbol{\Phi}_2 \mathbf{z}_{xy} \} \quad (6.16)$$

subject to the constraints

$$\begin{aligned} \mathbf{A}_{21} \mathbf{z}_{xy} &= \mathbf{C}_{21} \mathbf{p}_{xy}, \\ \mathbf{A}_{22} \mathbf{z}_{xy} &\leq \mathbf{C}_{22} \mathbf{p}_{xy}, \end{aligned} \quad (6.17)$$

where the combined weighting matrix  $\boldsymbol{\Phi}_2$  for XY, is given by (6.18),

$$\boldsymbol{\Phi}_2 = \begin{bmatrix} \mathbf{W}_{4 \times 4} & \mathbf{0}_{4 \times 2} \\ \mathbf{0}_{2 \times 4} & \mathbf{Q}_{xy} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 2} \\ \mathbf{0}_{2 \times 4} & \mathbf{Q}_{xy} \end{bmatrix} \in \mathbb{R}^{6 \times 6}. \quad (6.18)$$

Further, the equality constraints matrices are given according to (6.19)

$$\begin{aligned} \mathbf{A}_{21} &= \begin{bmatrix} (\mathbf{B}_{xy}^\top \mathbf{Q}_{z\psi}) & -\mathbf{I}_{2 \times 4} \end{bmatrix} \in \mathbb{R}^{2 \times 6}, \\ \mathbf{C}_{21} &= \begin{bmatrix} \mathbf{I}_{2 \times 2} & \mathbf{0}_{2 \times 4} & \mathbf{0}_{2 \times 4} \end{bmatrix} \in \mathbb{R}^{2 \times 10}, \end{aligned} \quad (6.19)$$

and lastly, the inequality constraints matrices are given according to (6.20)

$$\begin{aligned} \mathbf{A}_{22} &= \begin{bmatrix} -\mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 2} \\ \mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 2} \end{bmatrix} \in \mathbb{R}^{8 \times 6}, \\ \mathbf{C}_{22} &= \begin{bmatrix} \mathbf{0}_{4 \times 2} & -\mathbf{I}_{4 \times 4} & \mathbf{0}_{4 \times 4} \\ \mathbf{0}_{4 \times 2} & \mathbf{0}_{4 \times 4} & \mathbf{I}_{4 \times 4} \end{bmatrix} \in \mathbb{R}^{8 \times 10}. \end{aligned} \quad (6.20)$$

The  $2^{nd}$  step of the optimization problem is then solved numerically with the MATLAB<sup>®</sup> function `quadprog.m`, returning the  $\mathbf{z}_{xy,d}$  solving the problem defined in (6.16)-(6.17).

The MATLAB<sup>®</sup> QP solver is called with the sequence

$$[\mathbf{z}_d] = \operatorname{quadprog}(\mathbf{PHI2}, \mathbf{f}, \mathbf{A}_{22}, \mathbf{b}_{22}, \mathbf{A}_{21}, \mathbf{b}_{21}), \quad (6.21)$$

where  $\mathbf{PHI2} = \boldsymbol{\Phi}_2$ ,  $\mathbf{b}_{21} := \mathbf{C}_{21} \mathbf{p}_{xy} \in \mathbb{R}^{2 \times 1}$  and  $\mathbf{b}_{22} := \mathbf{C}_{22} \mathbf{p}_{xy} \in \mathbb{R}^{8 \times 1}$ .

The total constrained optimal recursive allocated control vector is then

$$\mathbf{u}_{opt.rec.} = \mathbf{u}_{z\psi} + \mathbf{Q}_{z\psi} \mathbf{u}_{xy} \in \mathbb{R}^{4 \times 1}. \quad (6.22)$$

### 6.3 Simulation to show performance of algorithm

To verify the functionality of the constrained optimal recursive thrust allocation algorithm, the test cases proposed in section 4.3, are performed on the algorithm. Repeated here for readability;

1. Open-loop; script testing, giving a commanded "ramp-up" thrust vector,  $\boldsymbol{\tau}_{c, ramp-up}$ .
2. Closed-loop; DP in set-point  $\boldsymbol{\eta} = [10 \ 10 \ 50 \ 45]^\top$  subject to a current velocity,  $\boldsymbol{v}_{current} = 1.15$  [knots] with a direction North (0 [deg] in the NED frame).

For the optimal recursive algorithm, the cost of utilizing the slack variables are equal for all DOF; Thus, the yaw (and heave) priority are obtained by allocating them in the 1<sup>st</sup> step, having full thruster capacity available. Then, in the 2<sup>nd</sup> step surge and sway are allocated with the remaining thruster capacity from the 1<sup>st</sup> step. This is unlike the yaw priority obtained from the weighted optimal algorithm; The entire thrust allocation are solved within one step. And the yaw priority are achieved by penalizing allocation to the slack variable corresponding to yaw.

The slack weighting matrices utilized in the 1<sup>st</sup> and 2<sup>nd</sup> step of the algorithm,  $\{\boldsymbol{Q}_{z\psi}, \boldsymbol{Q}_{xy}\}$  respectively, are given in (6.23).

$$\boldsymbol{Q}_{z\psi} = \boldsymbol{Q}_{xy} = \begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix} \quad (6.23)$$

#### 6.3.1 MATLAB<sup>®</sup> script testing

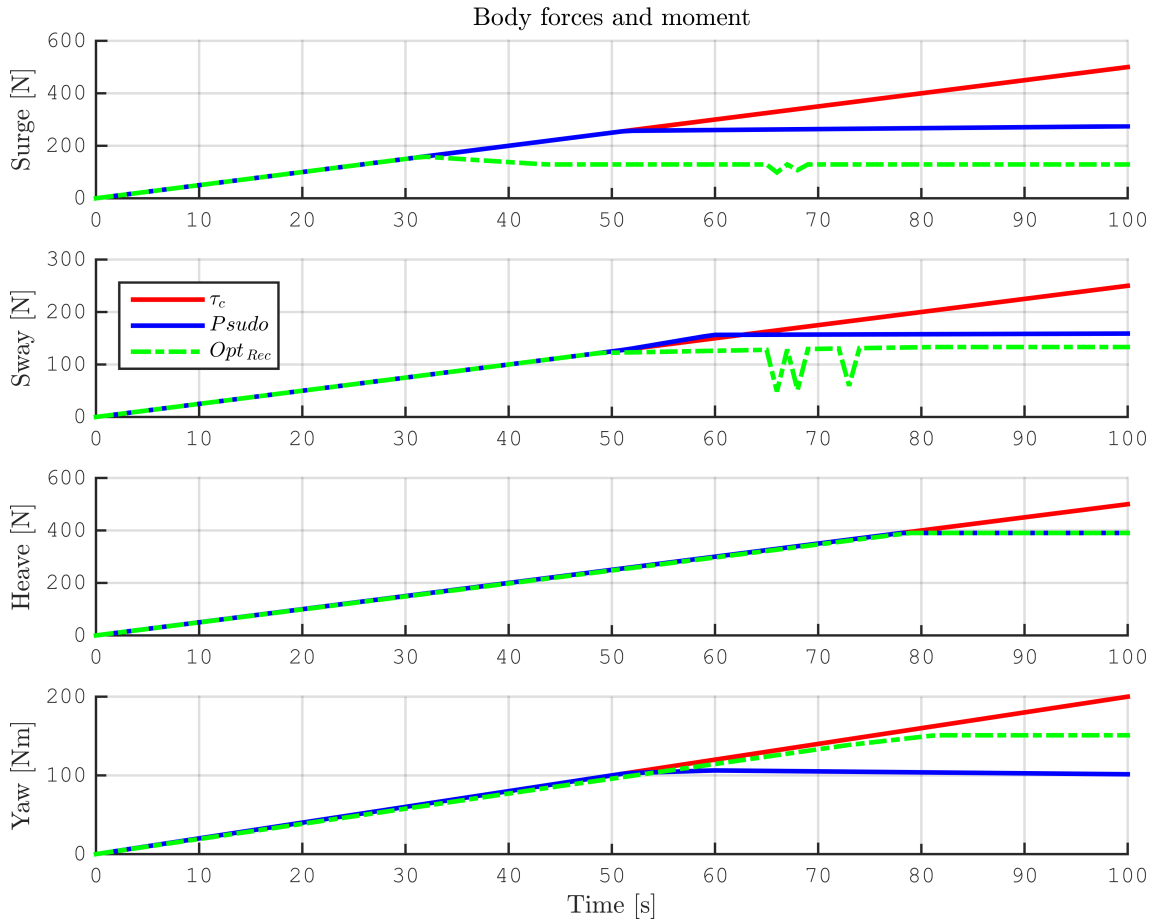
In the 1<sup>st</sup> level of testing, the optimal recursive thrust allocation algorithms is tested with a  $\boldsymbol{\tau}_{ramp-up}$  commanded thrust vector. This open-loop testing, enables the direct comparison of the performance of the algorithms. Thus, the optimal recursive algorithm is compared with the existing pseudoinverse thrust allocation currently utilized in the Minerva simulation, and control systems.

Seen in Figure 6.1 are the  $\boldsymbol{\tau}_{ramp-up}$  in red, the resulting forces from  $\boldsymbol{\tau}_{pseudo}$  in blue, and the resulting forces from the optimal recursive algorithm  $\boldsymbol{\tau}_{opt-rec}$  in green. The optimal recursive algorithm prioritize the DOFs according to  $\{ZN, XY\}$  as previously stated. In Figure 5.2, the thruster usage (in rpm), giving the resulting body forces and moment in Figure 5.1, are seen.

Comparing the optimal recursive algorithm with the pseudoinverse, it is seen that when the port thruster ( $T_{port}$ ) saturates at approx. 50 seconds, the maximum yaw moment, and surge force, from the pseudoinverse algorithm is obtained. Sway force is allocated further, until the lateral thruster ( $T_{lat}$ ) saturates at approx. 60 seconds. After the 2<sup>nd</sup> thruster saturates, the pseudoinverse is unable to allocate further, leaving the starboard thruster utilized in approx. 1/3 of the capacity, with positive rotation.

The optimal recursive algorithm, prioritizing heave and yaw, achieves full yaw allocation (for Minerva) after approx. 70 seconds. Seen in Figure 6.2, when  $T_{port}$  starts

to reach it's saturation limit, after approx. 30 seconds, the allocation to  $T_{port}$  is kept at approx. 1 200 [rpm]. At the same time, the allocation to  $T_{stb}$  is changed from positive to negative rotation, and allocated until  $T_{stb}$  saturates at approx. 70 seconds. When  $T_{lat}$  saturates after approx. 60 seconds,  $T_{port}$  is allocated further until  $T_{port}$  also saturates after approx. 70 seconds.

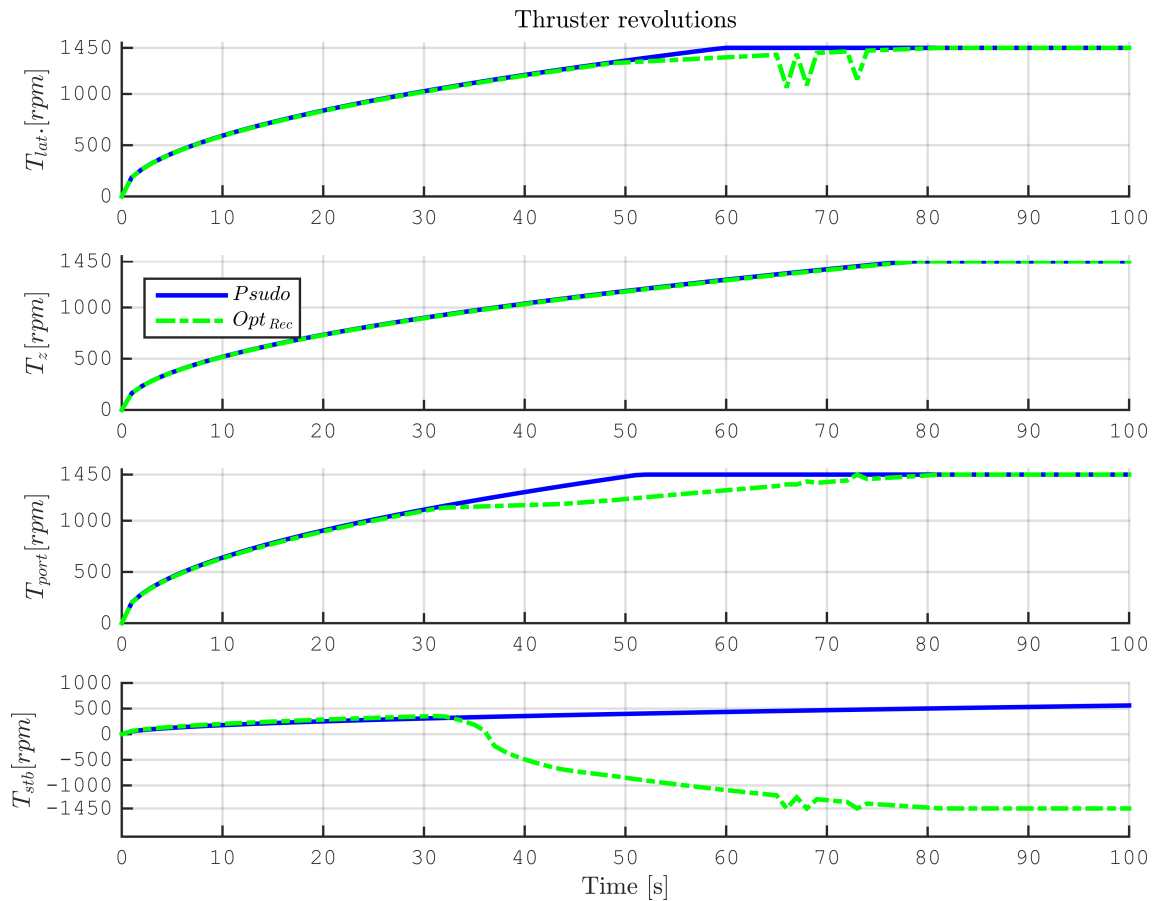


**Figure 6.1:** 1<sup>st</sup> level of testing; Testing the optimal recursive thrust allocation algorithm with a  $\tau_c$  ramp-up control vector. Showing the resulting body forces and moment.

Seen in Figure 6.1 at approx. 30 seconds, when  $T_{port}$  is kept at approx. 1 200 [rpm], and  $T_{stb}$  changes rotation direction, the optimal recursive algorithm reaches more or less the limit for surge force allocation. This, at a level of about 1/2 of what the pseudoinverse achieves. Further, at approx. 60 seconds when  $T_{port}$  and  $T_{stb}$  is close to saturating, in positive and negative rotational direction respectively, the sway force reaches the allocation limit. At a level slightly less than the pseudoinverse. Overall, the optimal recursive algorithm achieves a greater yaw moment of about 50 [Nm] more than the pseudoinverse algorithm. Allocating the maximum achievable yaw moment of Minerva, 150 [Nm]. This comes at the expense of a significantly less surge force allocation, about half of the approx. 280 [N] achieved by the pseudoinverse algorithm. Further, the optimal recursive algorithm achieve a slightly less sway force allocation. The main difference in total thruster usage, is that the optimal recursive algorithm utilizes the full potential of  $T_{port}$ , whilst the pseudoinverse algorithm only

utilize  $T_{port}$  at about 1/3 of the capacity.

Further, as seen in Figure 6.2; In the interval 60-70 second the  $T_{lat}$  have some unexplained drops in thruster usages. This manifests itself in the sway allocation, seen in Figure 6.1.



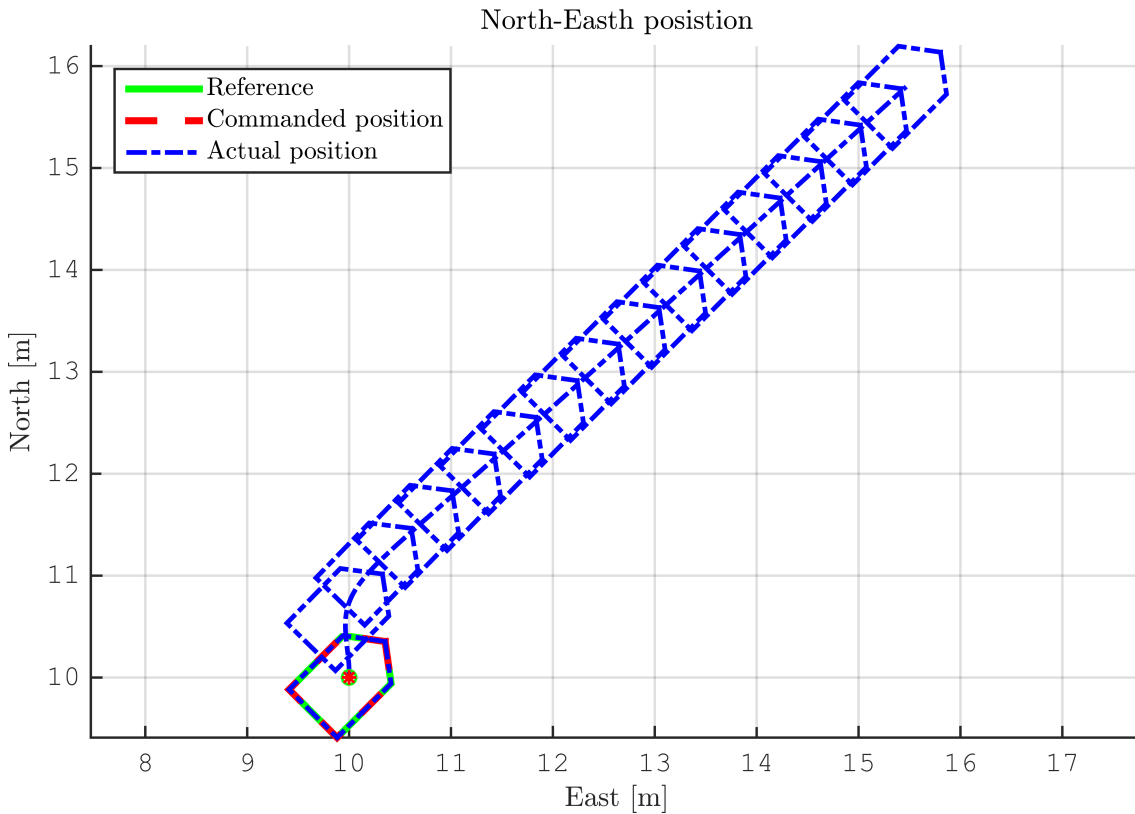
**Figure 6.2:** 1<sup>st</sup> level of testing; Testing the optimal recursive thrust allocation algorithm with a  $\tau_c$  ramp-up control vector. Showing the thrusters rpm's.

For the heave allocation, again as expected, both algorithms allocates the heave identically.

### 6.3.2 MATLAB<sup>®</sup>/Simulink software simulation

In this subsection the optimal recursive thrust allocation algorithm is tested in the 2<sup>nd</sup> level of the proposed testing regime. As commented subsection 4.3.2; The MATLAB<sup>®</sup>/Simulink simulation, running the algorithm in a closed loop system, prevents the direct comparison of the different thrust allocation algorithms. Thus, the aim of the 2<sup>nd</sup> level of testing is to show that the the proposed thrust allocation algorithm can run the system in closed-loop, and the performance of the algorithm will be discussed. The errors from the algorithm will be discussed in in Chapter 7.

Conducting the 2<sup>nd</sup> level of testing, simulating DP in set-point  $\boldsymbol{\eta} = [10 \ 10 \ 50 \ 45]^\top$  subject to a current velocity of  $\mathbf{v}_{current} = 1.15$  [knots] with a northern direction (0 [deg] in the NED frame).

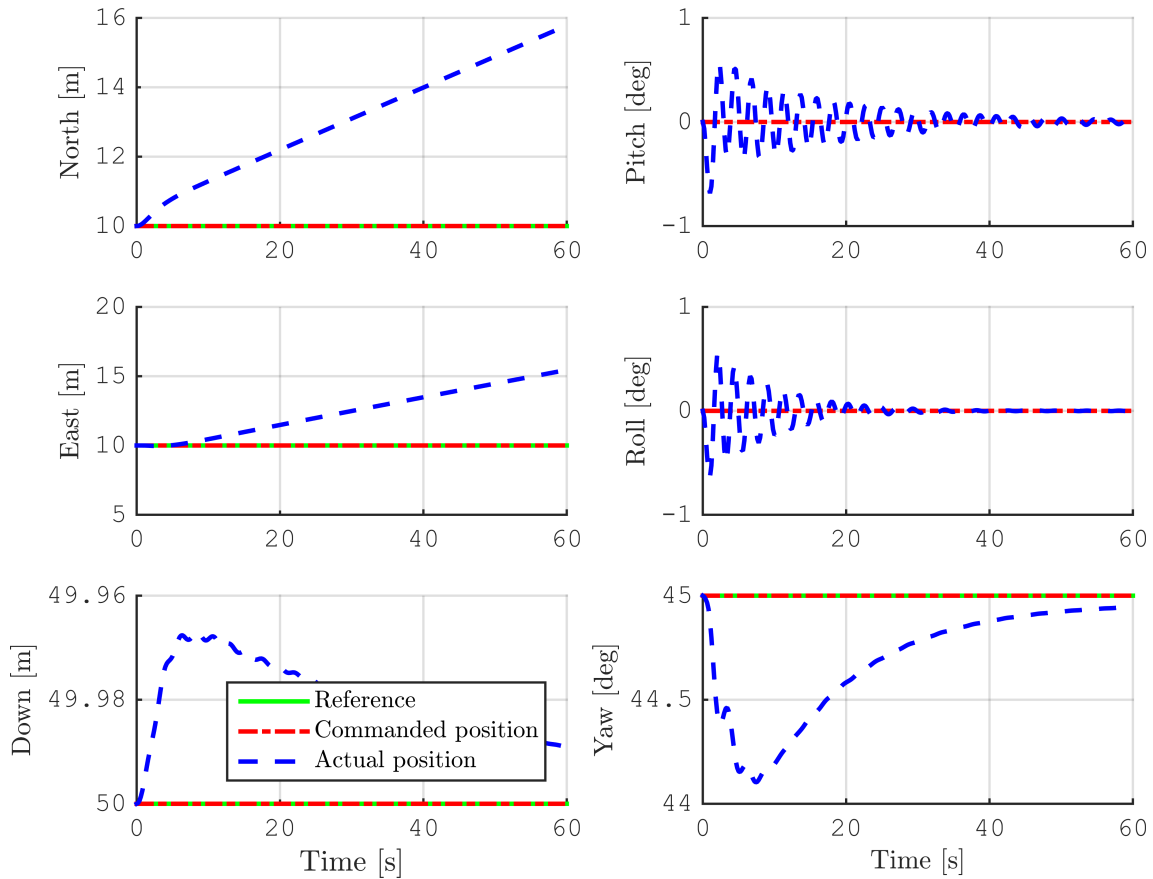


**Figure 6.3:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the optimal recursive thrust allocation algorithm in closed-loop. Showing the ROV position in the North-East plane.

Figure 6.3 show the position of the Minrva in the North-East plane. Seen in the figure; The current is too strong for the Minrva to be able to keep the commanded position. Thus, as yaw allocation has priority over surge and sway allocation, the ROV starts to drift while the thrust allocation algorithm utilizes the thrusters to obtain the commanded yaw angle. This is similar to the obtained results from the 3-step recursive nullspace-based algorithm.

In Figure 6.4 the 6 DOF response of the Minrva are seen; The 3 translational DOF,

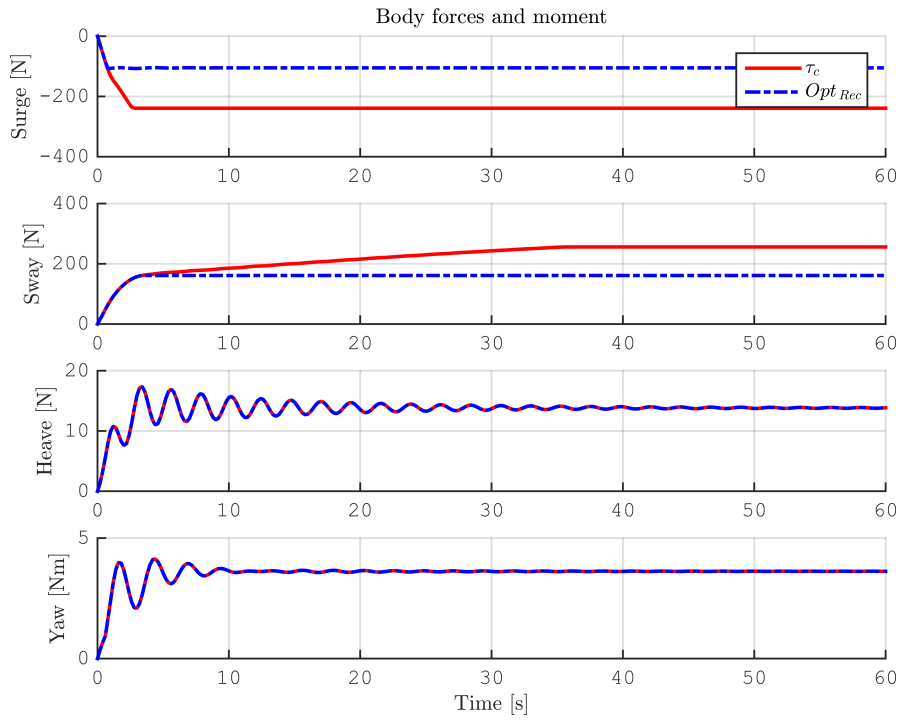
{1, 2, 3} in the left column. And the 3 rotational DOF, {4, 5, 6} in the right column. Again as discussed in subsection 4.3.2; The excitation of the unactuated DOF due to coupling effects, are passively stable with (small) exponential decaying responses. Further, the positively buoyant ROV start to rise as the simulation is started, before the thrust allocation counteracts and keeps the ROV at the desired depth of 50 [m]. Then, looking at the responses in DOF {1, 2, 6}; The north and east position is moving further and further away from the desired position, while the yaw angle converges to the desired heading of 45 [deg]. This is consistent with what is seen in Figure 4.3.



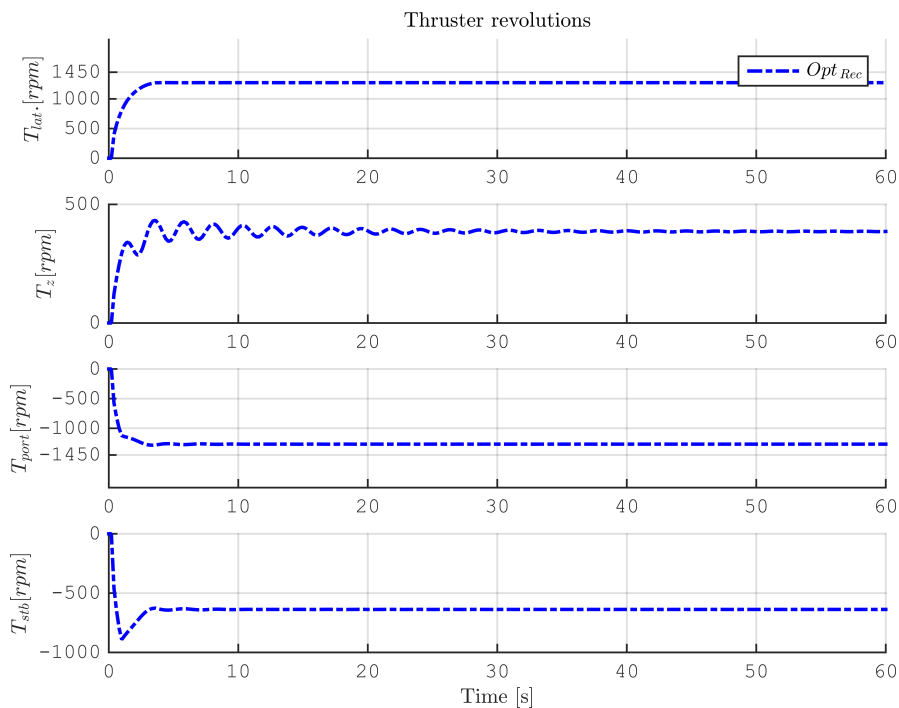
**Figure 6.4:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the optimal recursive thrust allocation algorithm in closed-loop. Showing the 6 DOF response of the ROV.

Figure 6.5 shows the resulting body forces and moment, from the allocated thruster usage, seen in Figure 6.6. The commanded force vector ( $\tau_c$ ) is seen in red, while the response from the optimal recursive thrust allocation algorithm are seen in blue.

After approx. 2 seconds the lateral thruster ( $T_{lat}$ ), and the port thruster ( $T_{port}$ ) seems to reach saturation, at approx. +1300 [rpm] and -1300 [rpm] respectively. This, corresponding to approx. a 10% loss of maximum thrust capacity. When this happens, further allocation to surge and sway stops.



**Figure 6.5:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the optimal recursive thrust allocation algorithm in closed-loop. Showing the resulting body forces and moment.



**Figure 6.6:** 2<sup>nd</sup> level of testing; Running the MATLAB<sup>®</sup>/Simulink simulator with the optimal recursive thrust allocation algorithm in closed-loop. Showing the allocated thruster revolutions.

Open-loop test of the constrained optimal recursive thrust allocation algorithm (seen in figures 6.1 - 6.2) shows that the proposed optimal recursive algorithm are able to fully utilize the thrusters, i.e. allocating the thrusters until saturation level of  $\pm 1450$  [rpm]. The open-loop test did however display some strange behaviour, with some "thruster drop" for the lateral thruster, in the interval 60-70 seconds, as previously commented.

The recursive algorithm is suboptimal, since the second step of the algorithm search a space that is limited by the first step. In the first step a solution is found, and this solution determines the set over which the second step can find its solution. Since the first step does not include all possible solutions, and then let the second step optimize over all solutions, the algorithm becomes suboptimal. In contrast does the weighted optimal algorithm find a global minima since it searches the entire space. However, what solution is found depend on the tuning of the cost function.

Further, what appear to be consived as the reduced thruster limits of approx.  $\pm 1300$  [rpm], are indeed significant part of the reason the optimal recursive algorithm are unable to reach and maintain a constant devoation from the set-point, as the weighted optimal algorithm did. Looking at Figure 6.5; When the commanded forces in surge and sway are not obtained in time step  $t_n$ , the command is increased further in time step  $t_{n+1}$ . Thus, continuously increasing the commands until the limits of the controller are reached.

In Chapter 7, the position error  $\boldsymbol{\eta}_{error} = |\boldsymbol{\eta}_{des} - \boldsymbol{\eta}_{actual}|$ , and the forces error  $\boldsymbol{\tau}_{error} = |\boldsymbol{\tau}_{cmd} - \boldsymbol{\tau}_{alloc}|$  of the weighted optimal thrust allocation algorithm are compared with the corresponding errors resulting from the 3-step recursive nullspace-based, and the pseudoinverse thrust allocation algorithms.



## *Simulation case study*

In this chapter a simulation case study is carried out in the MATLAB<sup>®</sup>/Simulink software. The aim of the case study is to compare the performance of the thrust allocation algorithms;

- 3-step weighted recursive nullspace-based thrust allocation algorithm,
- Weighted optimal thrust allocation algorithm,
- Optimal recursive thrust allocation algorithm,

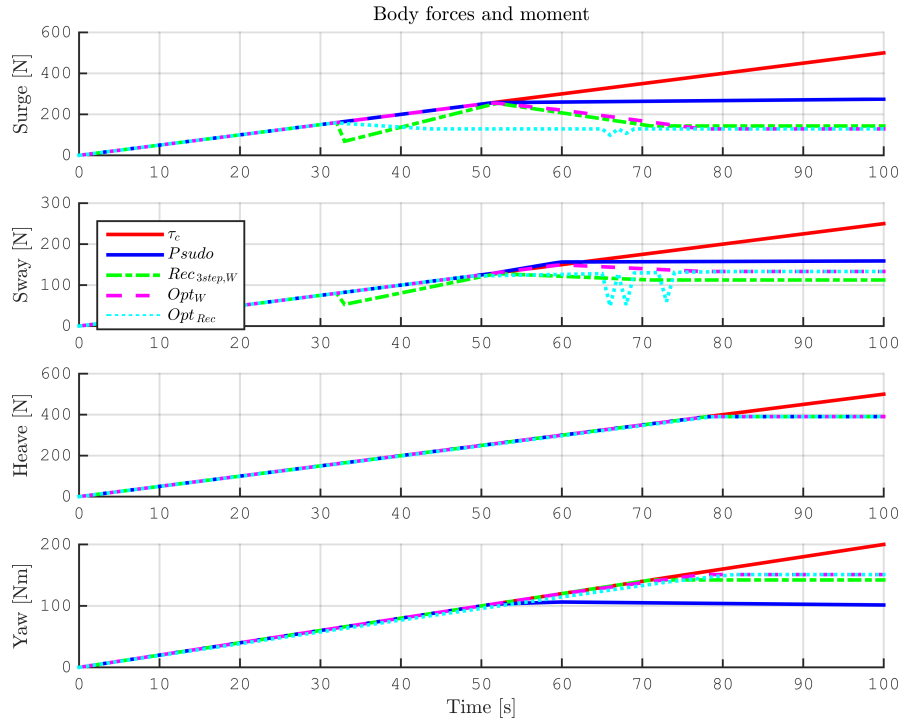
presented in chapters 4 - 6, respectively. The performance of these algorithms are compared with the pseudoinverse thrust allocation currently utilized in the Minerva control and simulation systems.

### 7.1 Comparison of thrust allocation algorithms

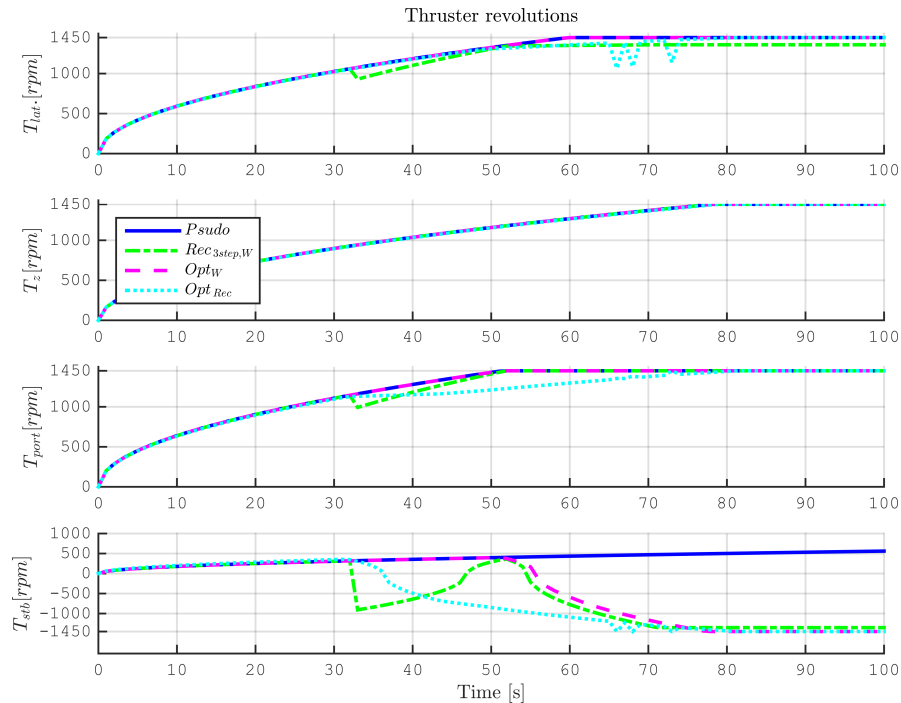
#### 7.1.1 MATLAB<sup>®</sup> script testing

First, the thrust allocation algorithms are compared in open-loop, commanding the algorithms with a  $\tau_{ramp-up}$ . As previously described, the open-loop testing allow for direct comparison of the allocated thrust. The aim of the open-loop comparison is to give an overview of the overall performance of the proposed algorithms, compared with the pseudoinverse thrust allocation.

Figure 7.1 shows the resulting body forces and moment, from the allocated thruster usage, seen in Figure 7.2. As seen in Figure 7.1; All three proposed thrust allocation algorithm achieve (or very close to) the maximum yaw moment of 150 [Nm]. The two recursive algorithms achieves this by allocating yaw in the step prior to surge and sway allocation, while the weighted optimal algorithm achieves the yaw priority by penalizing allocation to the slack variable corresponding to yaw. Compared with the pseudoinverse thrust allocation, the three proposed algorithms achieves greater allocated yaw moment, but at the expence of reduced surge force, and somewhat reduces sway force. Looking at the heave allocation, all algorithms achieve the same heave force, as they should due to the thruster configuration of Minerva where heave is decoupled. Considering the thruster usage of the algorithms, seen in Figure 7.2; Based on the overall thruster usage, the utilization of the starboard thruster ( $T_{stb}$ ), seems to differentiate the thrust allocation algorithms. The pseudoinverse thrust allocation utilizes ( $T_{stb}$ ) at approx. 1/3 of the capacity in positive rotational direction, the three propsed thrust allocation algorithms utilizes the starboard thruster ( $T_{stb}$ ) to its maximum capacity in negative rotational direction.



**Figure 7.1:** Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Open-loop testing with a  $\tau_{ramp-up}$ . Showing the resulting body forces and moment.



**Figure 7.2:** Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Open-loop testing with a  $\tau_{ramp-up}$ . Showing the allocated thruster revolutions.

### 7.1.2 MATLAB<sup>®</sup>/Simulink software simulation

In the following, the closed-loop performance of the three presented thrust allocation algorithms, are compared with the performance of the pseudoinverse thrust allocation. The following two cases are simulated;

- **Case 1:** DP in set-point  $\boldsymbol{\eta} = [10 \ 10 \ 50 \ 45]^\top$  subject to a current velocity,  $\mathbf{v}_{current} = 1.15$  [knots] with a direction North (0 [deg] in the NED frame).
- **Case 2:** Change in pose from  $\boldsymbol{\eta}_0 = [10 \ 10 \ 50 \ 20]^\top$  to  $\boldsymbol{\eta}_1 = [10 \ 10 \ 50 \ 65]^\top$  subject to a current velocity,  $\mathbf{v}_{current} = 1.05$  [knots] with a direction East (90 [deg] in the NED frame).

As previously discussed, there are no improvements to be gained in the heave allocation for Minerva. Thus, no focus are put on the discussion of heave actuation, nor the corresponding errors related to heave actuation in the following discussions.

#### Close-loop simulation; Case1. DP in set-point $\boldsymbol{\eta}_0$ .

Case 1 is identical with the 2<sup>nd</sup> level of testing for the individual algorithm testing, and thus only the following errors are discussed for this case;

- Position error:  $\boldsymbol{\eta}_{error} = |\boldsymbol{\eta}_{des} - \boldsymbol{\eta}_{actual}|$ ,
- Force error:  $\boldsymbol{\tau}_{error} = |\boldsymbol{\tau}_{cmd} - \boldsymbol{\tau}_{alloc}|$ .

The system responses from the simulation, running the system with the pseudoinverse thrust allocation, are attached in Appendix B, for reference. The following plots are seen in figures B.1 - B.4, respectively;

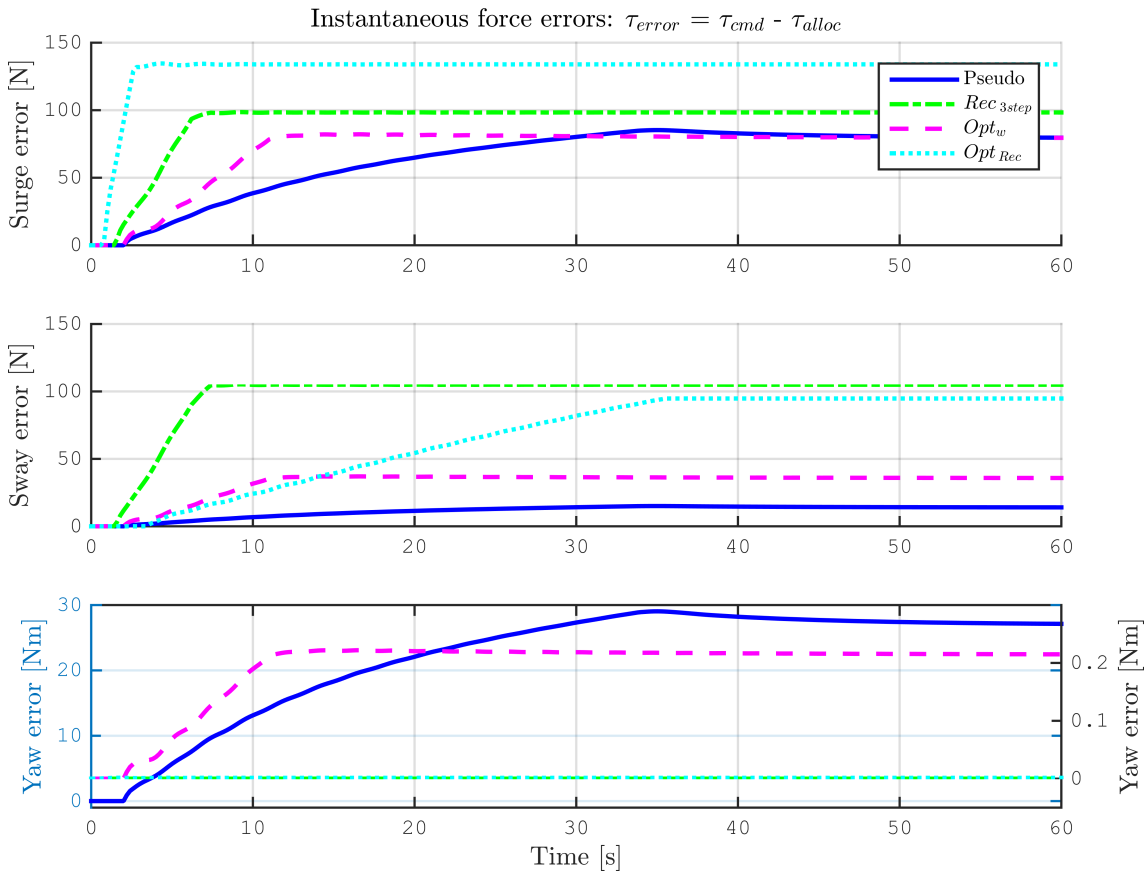
- North-East plot.
- 6 DOF responses.
- resulting body forces and moment.
- Allocated thruster rpm.

In Figure 7.3, the instantaneous error,  $\boldsymbol{\tau}_{error} = |\boldsymbol{\tau}_{cmd} - \boldsymbol{\tau}_{alloc}|$  are seen. Important to notice in the 3<sup>rd</sup> subplot; The yaw error from pseudoinverse allocation are plotted against the left y-axis, where  $y_{left} \in [0, 30]$ . The yaw errors from the {3-step recursive, weighted optimal, optimal recursive} algorithms are plotted against the right y-axis, where  $y_{right} \in [0, 0.3]$ . I.e. there is a magnitude difference of 100 times between the yaw error from the pseudoinverse allocation, and the yaw errors from {3-step recursive, weighted optimal, optimal recursive} allocations. The unit of both axis are moment [Nm]. Both the recursive algorithms, {3-step recursive, optimal recursive}, have a yaw error of approx. 0 [Nm], while the weighted optimal have a yaw error of approx. 0.2 [Nm]. The yaw errors of the algorithms {3-step recursive, weighted optimal, optimal recursive}, are all neglectable compared with the yaw error from the pseudoinverse, of approx. 28 [Nm].

The surge and sway error, in the 1<sup>st</sup> and 2<sup>nd</sup> subplot respectively, are both plotted against a common left y-axis, with unit force [N]. Looking at the sway error in the 2<sup>nd</sup>

subplot; The recursive algorithms {3-step recursive, optimal recursive} both have a significantly larger sway error than the error from the pseudoinverse. The sway error from the weighted optimal are also larger than the error from the pseudoinverse, but less than half the errors of the recursive algorithms. Where the sway error from the pseudoinverse have a magnitude of approx. 20 [N], the weighted optimal have a error magnitude of approx. 35 [N], and the recursive algorithms have a error magnitude of approx.100 [N].

Further, looking at the surge error in the 1<sup>st</sup> subplot; The surge errors from the pseudoinverse, and the weighted optimal algorithms are of same magnitude of approx. 80 [N]. Both recursive algorithms have a surge error of greater magnitude, approx. 100 [N] for the recursive 3-step, and approx. 135 [N] for the optimal recursive.

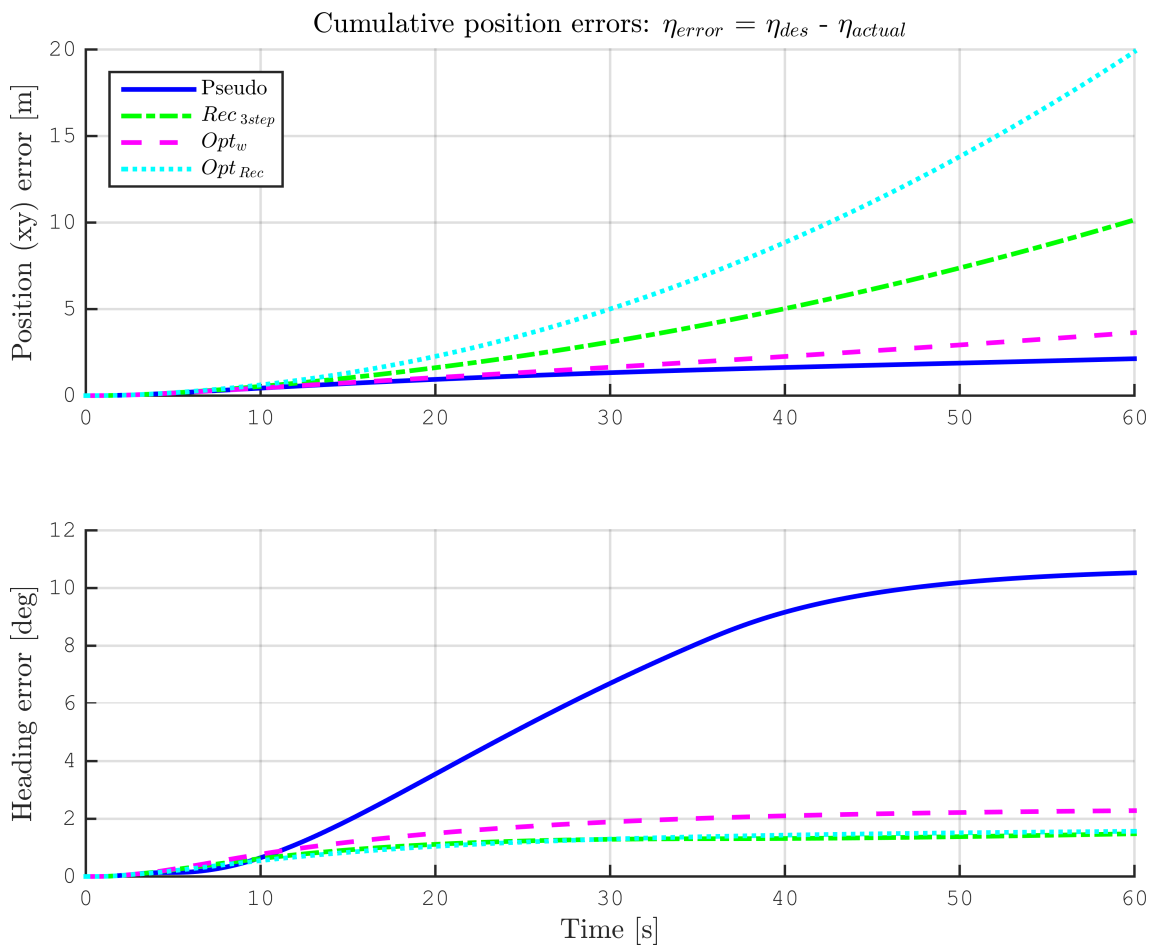


**Figure 7.3:** Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Case 1, DP in set-point. Comparing the instantaneous forces errors,  $\tau_{error}$ .

In Figure 7.4 the cumulative position error ( $\eta_{error}$ ) are seen. The 1<sup>st</sup> subplot show the xy-position error, measured in meter [m], while the 2<sup>nd</sup> subplot shows the error in heading, measured in degrees [deg]. Clearly the heading error are the focus in the thesis, but a measure of the overall performance of the algorithms are of interest, thus including a xy-position error plot.

Looking at the 2<sup>nd</sup> subplot; The cumulative heading error of the pseudoinverse allocation are approx. 10 degrees after about 60 seconds: While the cumulative

heading errors from the algorithms {3-step recursive, weighted optimal, optimal recursive} are approx. 2 degrees, i.e. the proposed algorithms have a yaw error of 1/5 of the pseudoinverse yaw error. Then, looking at the 1<sup>st</sup> subplot; The xy-position error between the proposed algorithms {3-step recursive, weighted optimal, optimal recursive} differs considerably. All larger than the xy-position error from the pseudoinverse, which are approx. 2 meters after about 60 seconds. The recursive algorithms {3-step recursive, optimal recursive} have a xy-position error of 10 meter, and 20 meters, respectively. While the weighted optimal xy-position error are approx. 3 meters. Thus, the xy-position error of the weighted optimal algorithm are slightly larger the error from the pseudoinverse, while the recursive algorithms {3-step recursive, optimal recursive} have an xy-position error that is 5 times, and of 10 times larger, respectively.



**Figure 7.4:** Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Case 1, DP in set-point. Comparing cumulative position errors,  $\eta_{error}$ .

**Close-loop simulation; Case 2. Change of pose from  $\boldsymbol{\eta}_0$  to  $\boldsymbol{\eta}_1$ .**

In the following, the 2<sup>nd</sup> simulation case of the simulation case study is presented. The simulation case is repeated here for readability;

- **Case 2:** Change in pose from  $\boldsymbol{\eta}_0 = [10 \ 10 \ 50 \ 20]^\top$  to  $\boldsymbol{\eta}_1 = [10 \ 10 \ 50 \ 65]^\top$  subject to a current velocity,  $\mathbf{v}_{current} = 1.05$  [knots] with a direction East (90 [deg] in the NED frame).

As previously discussed, the closed-loops system responses can not be compared directly. Thus, the focus of the following discussions will once again be on the following errors;

- Position error:  $\boldsymbol{\eta}_{error} = |\boldsymbol{\eta}_{des} - \boldsymbol{\eta}_{actual}|$ ,
- Force error:  $\boldsymbol{\tau}_{error} = |\boldsymbol{\tau}_{cmd} - \boldsymbol{\tau}_{alloc}|$ .

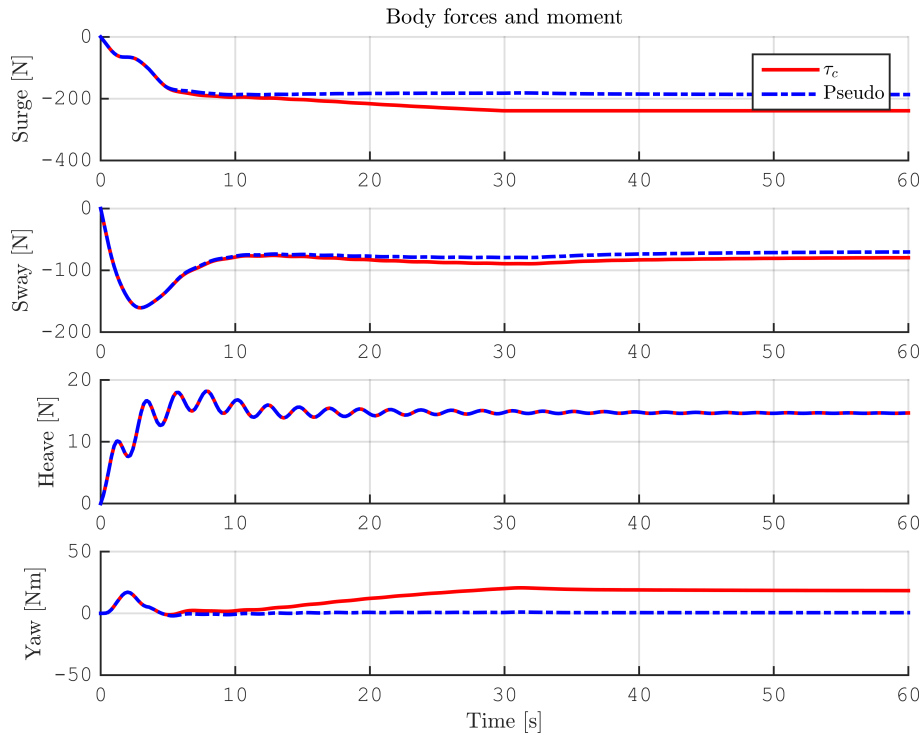
Before the errors are presented and discussed, the resulting body forces, and moment from the system, running simulation case 2 are presented shortly for better understanding of the error discussions. The resulting body forces, and moment corresponding to the proposed thrust allocation algorithms, {3-step recursive, weighted optimal, optimal recursive}, along with the resulting body forces, and moment from the pseudoinverse thrust allocation, are seen in figures 7.5 - 7.8, respectively.

Further, the following plots generated based on the simulations;

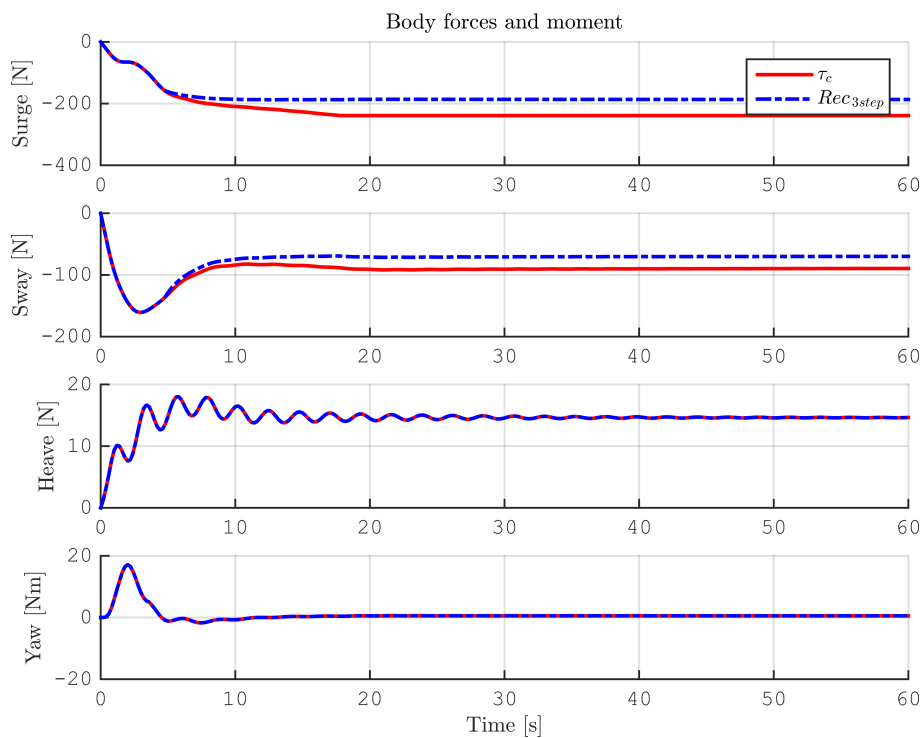
- North-East plot,
- 6 DOF responses,
- resulting body forces and moment,
- Allocated thruster rpm.

These are attached in Appendix C, for reference. Seen in figures C.1 - C.20;

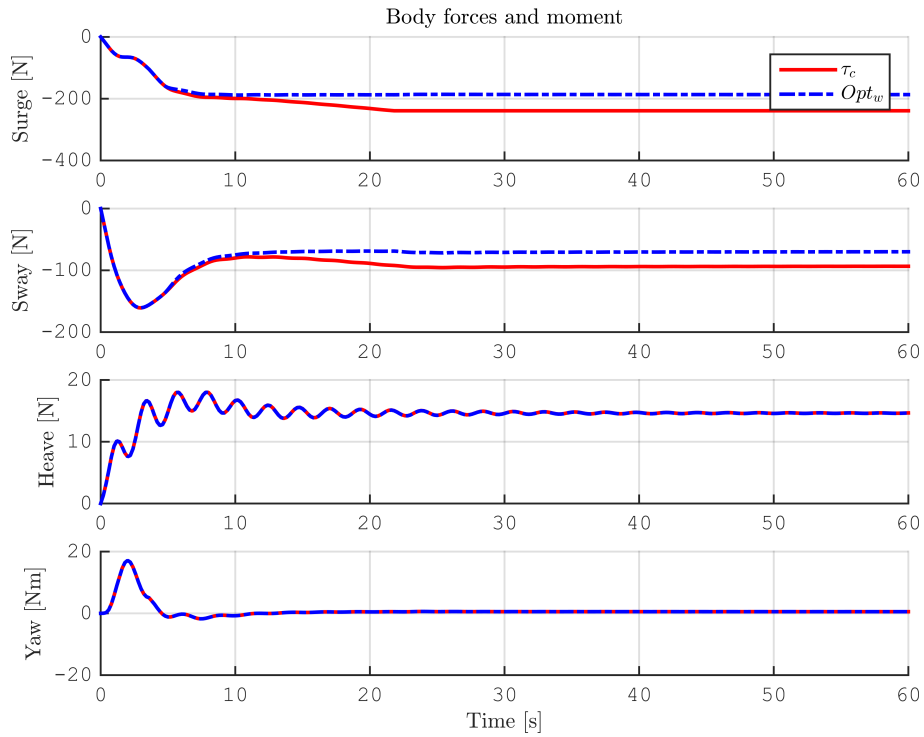
Looking at Figure 7.5, it is seen that the resulting body forces and moment are not met, running the system with the pseudoinverse thrust allocation. Then, looking at figures 7.6 - 7.8, corresponding to thrust allocation algorithms {3-step recursive, weighted optimal, optimal recursive} respectively; All the proposed thrust allocation algorithms achieve the sought yaw prioritization, but are not able to achieve the desired forces in surge and sway. This corresponds with what was seen in simulation case 1. Further, all algorithms achieve the desired heave allocation, as they should. As mentioned before the heave force is decoupled. The allocation is heave will therefore not differ between the algorithms, and the heave allocation is therefore not part of the remainder of the discussion. An important difference from simulation case 1, is seen in Figure 7.8, showing the resulting body forces and moment from the proposed optimal recursive thrust allocation algorithm. Although the sought yaw priority is obtained, the overall system performance of the thrust allocation algorithm are seen to be unacceptable. Simulating case 2 with reduced current velocity, the optimal recursive algorithm shows improved behaviour. These responses are seen in figures C.17 - C.20 in Appendix C.



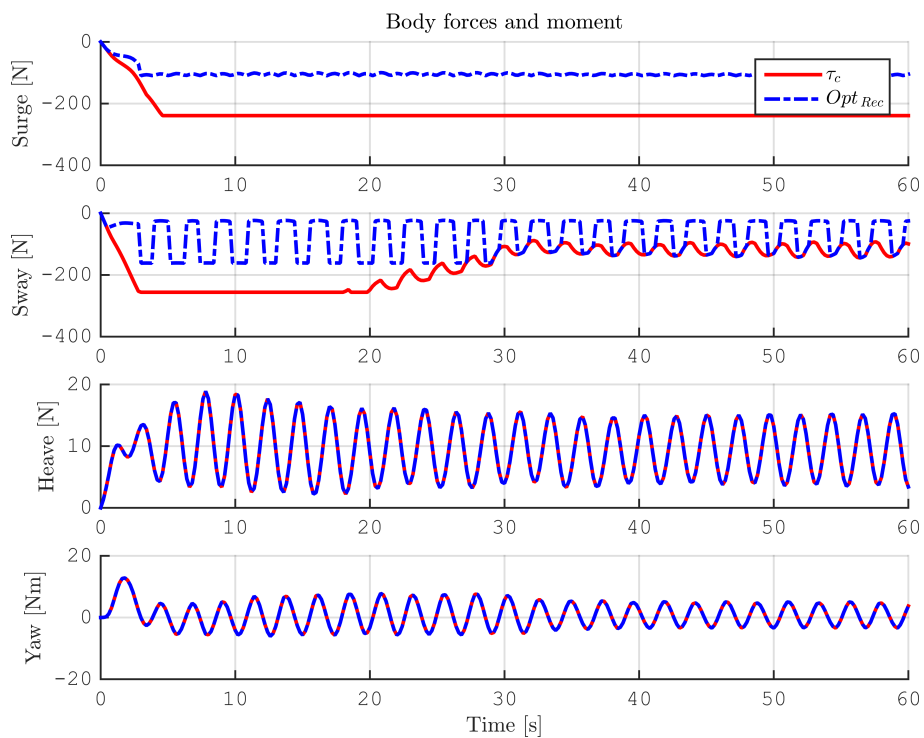
**Figure 7.5:** Case study; Case 2, change of pose. Showing the resulting body forces and moment, running the system with the pseudoinverse thrust allocation.



**Figure 7.6:** Case study; Case 2, change of pose. Showing the resulting body forces and moment, running the system with the 3-step recursive thrust allocation.



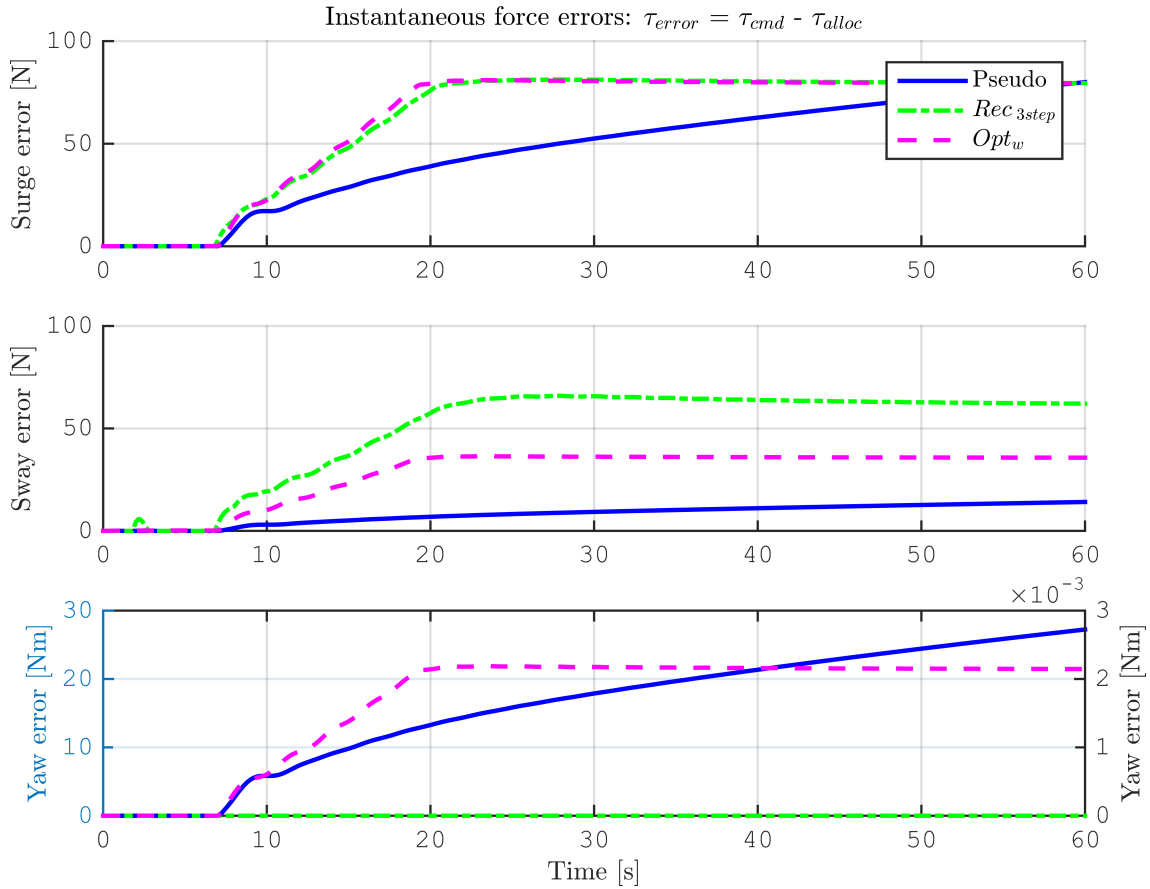
**Figure 7.7:** Case study; Case 2, change of pose. Showing the resulting body forces and moment, running the system with the weighted optimal thrust allocation.



**Figure 7.8:** Case study; Case 2, change of pose. Showing the resulting body forces and moment, running the system with the optimal recursive thrust allocation.



As the implementation of the recursive optimal algorithm is faulty, the algorithm are discarded from the following discussions.



**Figure 7.9:** Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Case 2, change of pose. Comparing the instantaneous forces errors,  $\tau_{error}$ .

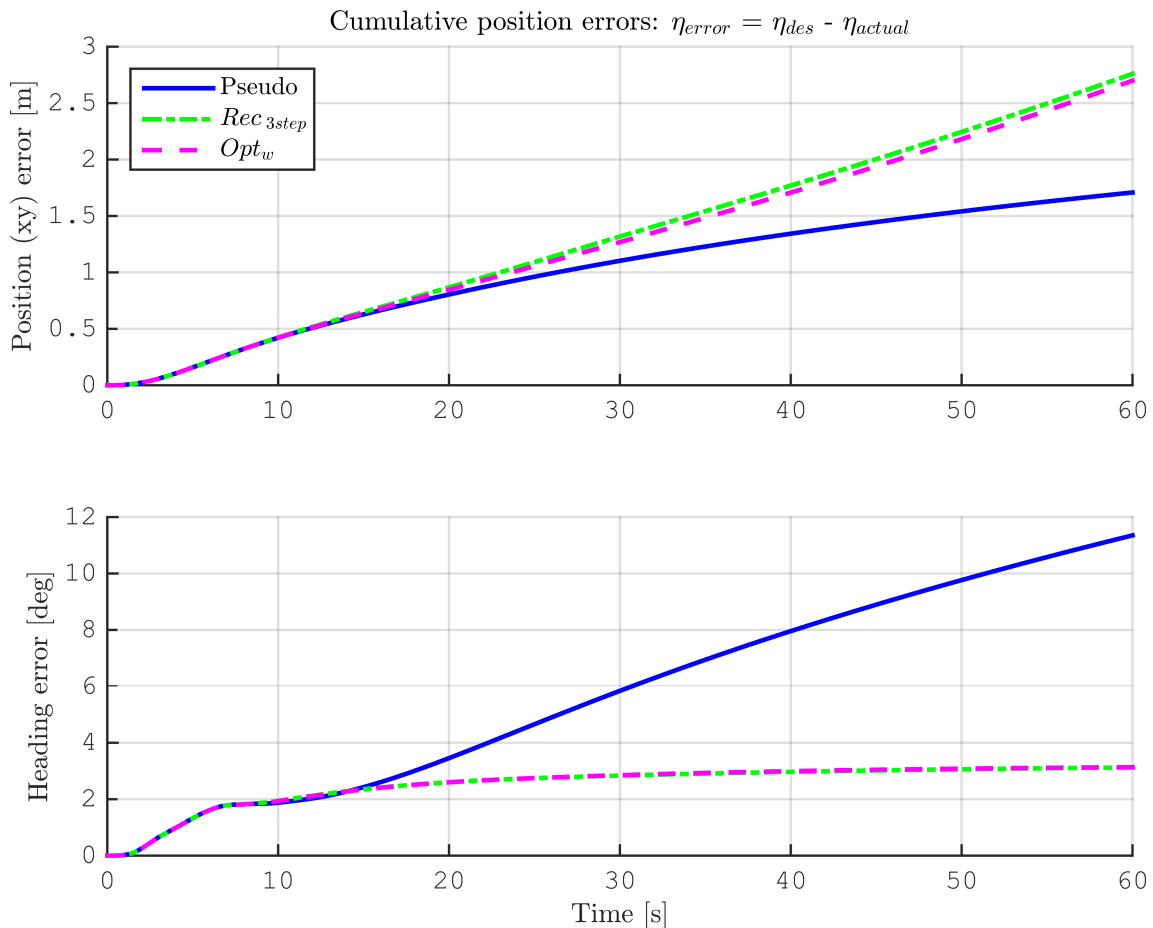
In Figure 7.9, the instantaneous error,  $\tau_{error} = |\tau_{cmd} - \tau_{alloc}|$  is seen. It is important to notice that in the 3<sup>rd</sup> subplot, the yaw error from the pseudoinverse allocation are plotted against the left y-axis, where  $y_{left} \in [0, 30]$ . The yaw errors from the 3-step recursive, and the weighted optimal algorithms are plotted against the right y-axis, where  $y_{right} \in [0, 0.003]$ . I.e. there is a magnitude difference of  $10^4$  between the yaw error from the pseudoinverse allocation, and the yaw errors from 3-step recursive, and weighted optimal recursive allocations. The unit of both axis are moment [Nm]. The 3-step recursive recursive algorithms have a yaw error of approx. 0 [Nm], while the weighted optimal have a yaw error of approx. 0.002 [Nm]. The yaw errors of the algorithms {3-step recursive, weighted optimal}, are both neglectable compared with the yaw error from the pseudoinverse, of approx. 28 [Nm].

The surge and sway error, in the 1<sup>st</sup> and 2<sup>nd</sup> subplot respectively, are both plotted against a common left y-axis, with unit force [N]. Looking at the sway error in the 2<sup>nd</sup> subplot; The sway errors of the proposed thrust allocation algorithms produce larger sway error, than the error from the pseudoinverse allocation. The error from the algorithms {3-step recursive, weighted optimal} are of approx. magnitude .60 [N], and 40 [N], respectively. Compared with the sway error from the pseudoinverse,

of approx. 18 [N], the error from the algorithms {3-step recursive, weighted optimal} are approx 3 times, and 2 times higher, respectively. Noteworthy is that the yaw error from the weighted optimal converges to the end value after approx. 20 second, while the yaw error from the pseudoinverse builds up over the whole period.

Further, looking at the surge error in the 1<sup>st</sup> subplot; All three algorithm end up with a surge error of approx. 80 [N]. The algorithms {3-step recursive, weighted optimal} converges to this value after approx. 20 seconds, while the error from the pseudoinverse builds up, and reaches this value after approx. 50 seconds.

In Figure 7.10 the cumulative position error ( $\eta_{error}$ ) are seen. The 1<sup>st</sup> subplot show the xy-position error, measured in meter [m], while the 2<sup>nd</sup> subplot shows the error in heading, measured in degrees [deg]. Again the heading error are the focus in the thesis, but a measure of the overall performance of the algorithms are of interest, thus including a xy-position error plot.



**Figure 7.10:** Case study, comparing the proposed thrust allocation algorithms with the pseudoinverse; Case 2, change of pose. Comparing cumulative position errors,  $\eta_{error}$ .

Looking at the 2<sup>nd</sup> subplot; The cumulative heading error of the pseudoinverse allocation are approx. 11 degrees after about 60 seconds: While the cumulative heading errors from the algorithms {3-step recursive, weighted optimal} are approx. 3 degrees, i.e. the proposed algorithms have a yaw error of approx. 1/4 of the

pseudoinverse yaw error.

Then, looking at the 1<sup>st</sup> subplot; The xy-position error between the proposed algorithms {3-step recursive, weighted optimal} are of approx. same magnitude, where the error from the 3-step recursive algorithm are slightly larger than the error from the weighted optimal algorithm. Both larger than the xy-position error of the pseudoinverse. Where the magnitude of the xy-position error from the algorithms {3-step recursive, weighted optimal} are approx. 2.75 [m], the xy-position error from the pseudoinverse a approx. 1.75 [m]. Thus, the xy-position errors from the algorithms {3-step recursive, weighted optimal} are close to 60 % larger than the error from the pseudoinverse.

## 7.2 Discussion

In the open-loop test, all three proposed thrust allocation algorithms {3-step recursive, weighted optimal, optimal recursive} displayed similar behaviour, achieving the desired yaw priority. The pseudoinverse allocation reach a yaw moment of 100 [Nm], 2/3 of max. achievable yaw moment for Minerva of 150 [Nm]. The three proposed thrust allocation algorithms all reached, or very close to, the full yaw moment of 150 [Nm]. The 3-step recursive obtained lightly less than max. yaw moment than the {weighted optimal, optimal recursive} algorithms. Then, looking at the surge force allocation; All the proposed thrust allocation algorithm reach approx. the same final surge force level, of approx. 145 [N]. The pseudoinverse reach a surge force level of approx. 265 [N]. I.e. approx. 45 % less surge force was achieved by the proposed algorithms, compared with the pseudoinverse. Further, it is worth mentioning that while the recursive optimal algorithm peaked at the surge force level of approx. 145 [N], the {3-step recursive, weighted optimal} algorithms peaked at the same surge force level as the pseudoinverse, at approx. 265 [N], before declining in magnitude to the same surge force level of approx. 145 [N] at the optimal recursive algorithm. Then, looking at the obtained sway force; The difference between the proposed thrust allocation algorithms {3-step recursive, weighted optimal, optimal recursive}, and the pseudoinverse was much less than what was seen for the yaw moment, and for the surge force. The sway force from the pseudoinverse peaked at approx. 160 [N], and the 3-step recursive algorithm at 112 [N], and the {weighted optimal, optimal recursive} algorithms at 130 [N]. Important to notice for the optimal recursive algorithm was the somewhat strange "drop-outs" of the lateral thruster utilization, resulting in "drop-outs" in the obtained sway force. This phenomenon occurred within the time interval 65-75 seconds.

Then, testing the proposed thrust allocation algorithms {3-step recursive, weighted optimal, optimal recursive} in closed-loop, comparing with the closed-loop performance of the pseudoinverse thrust allocation. The closed-loop testing differs from the open-loop test because the command in time step  $t_{n+1}$ , is dependent on the response of the system in time step  $t_n$ . Thus, the performance of the algorithms can not be compared directly, and this is why the following discussions focus on the errors  $\tau_{error}$ , and  $\eta_{error}$ . The  $\tau_{error}$  was chosen to be presented as the instantaneous

error, while the  $\boldsymbol{\eta}_{error}$  was chosen to be presented as the cumulative error. This choice was made because this was found to be the best way to present, and discuss the errors. Further, the closed-loop simulations show that the sooner a thruster saturated (and stays saturated), the larger the error  $\boldsymbol{\tau}_{error}$  will become. This is due to when the command  $\boldsymbol{\tau}_c^n$  is not achieved by the algorithms, the command in the following step  $\boldsymbol{\tau}_c^{n+1}$ , will be increased in order to achieve  $\boldsymbol{\tau}_d$ . This continues until  $\boldsymbol{\tau}_c^{max}$  of the controllers is reached.

Summarizing the findings from the two simulation cases; Case 1, DP in set-point, and case 2, change of pose. First, evaluating the performance of the proposed optimal recursive thrust allocation algorithm. In simulation case 1 the performance of the optimal recursive algorithm appeared to be acceptable. The desired yaw priority was obtained, with a yaw error of less than 1/100 compared with the yaw error from the pseudoinverse. For the surge, and sway force the optimal recursive algorithm performed poorer than the pseudoinverse, with an error of approx. 2 times, and 5 times the magnitude, respectively. This is clearly related to what appears to be conceived as the reduced saturation limit of  $\pm 1300$  [rpm] (seen in Figure 6.6) in the 2<sup>nd</sup> step of the optimal recursive algorithm. I.e. 10 % reduction of max. capacity of the thrusters. In the open-loop test, the optimal recursive algorithm reached the  $\pm 1450$  [rpm] saturation limits. Simulating case 2, the performance of the optimal recursive algorithm was even worse. Again the yaw priority was achieved, with a error of less than  $10^{-4}$  than the yaw error from the pseudoinverse. In the 2<sup>nd</sup> step, the optimal recursive algorithm utilizes the thrusters such that the rotational direction of the thrusters are constantly changed back and forth. Causing instability in the otherwise stable system, seen in figures C.14 - C.16. This is clearly increasing wear and tear of the thruster, and thereby reducing their life time. Further, this utilization produces inferior thrust allocation performance, than the pseudoinverse algorithm, as well as the other two proposed algorithms. The obvious conclusion from this, is that the proposed optimal recursive thrust allocation for Minerva, is infeasible.

Then, summarizing the findings on the {3-step recursive, weighted optimal} thrust allocation algorithms. Looking first at the  $\boldsymbol{\tau}_{error}$ . The yaw error from the proposed thrust allocation algorithms are significantly less than the corresponding yaw error from the pseudoinverse, 1/100 for case 1, and  $10^{-4}$  for case 2. The error from the weighted optimal algorithm are slightly larger than the error from the 3-step recursive, but both are neglectable compared with the yaw error from the pseudoinverse. The yaw error from the weighted optimal thrust allocation algorithm, can be reduced further, by increasing the cost of the corresponding slack variable. This will come at the expense of achieved surge, and sway force.

Further, looking at the sway errors, both the algorithms {3-step recursive, weighted optimal} produced larger sway errors, than the corresponding errors from the pseudoinverse. The sway errors from the 3-step recursive algorithm were for both cases larger than the sway error from the weighted optimal algorithm. 3 times higher for case 1, and 2 times higher for case 2. Looking at the surge errors, the differences were not as obvious; For case 1 the surge error from the weighted optimal was at the same level as the surge error from the pseudoinverse. The surge error from the

3-step recursive was slightly larger. In case 2, both the {3-step recursive, weighted optimal} algorithms produce very similar surge error, of same magnitude. For this case, also the surge error from the pseudoinverse reached the same level, but at a later time in the simulation.

Looking at the  $\boldsymbol{\eta}_{error}$ , and in particular on the heading error, both the {3-step recursive, weighted optimal} algorithms shows a significant improved performance, compared to the heading error of the pseudoinverse. For case 1 the heading error from the 3-step recursive was slightly less than the heading error from the weighted optimal. For case 2 the heading errors from both {3-step recursive, weighted optimal} algorithms were of same magnitude. Compared to the heading error from the pseudoinverse, the heading error from the {3-step recursive, weighted optimal} algorithms were approx 1/5, for both cases, after about 60 seconds. For the xy-position error, there were greater differences between the two cases. Looking at case 1, the xy-position error from the weighted optimal algorithm were slightly larger than the xy-position error from the pseudoinverse. The xy-position error from the 3-step recursive algorithm were approx. 5 times larger after about 60 seconds, than the xy-position error from the pseudoinverse. For case 2 the xy-position error were of same magnitude for both {3-step recursive, weighted optimal} algorithms. The xy-position error from the pseudoinverse were approx. 2/3 of the xy-position errors from the {3-step recursive, weighted optimal} algorithms, after about 60 seconds.

Overall, based on the testing carried out in this thesis, the performance of the weighted optimal algorithm, seemed better than the performance of the recursive 3-step algorithm. This is in accordance with the expectations to the results beforehand.



## *Conclusion and further work*

In this final chapter, concluding remarks on the material presented in this thesis will be made. Then, suggestions for further work will be given.

### 8.1 Conclusion

In this thesis three constrained thrust allocation algorithms, prioritizing the yaw moment, are proposed for the ROV Minerva. The three proposed thrust allocation algorithms are; 3-step recursive nullspace-based algorithm, the optimal (QP) algorithm, and the optimal recursive (recursive QP) algorithm. Before testing the thrust allocation algorithms in the Minerva MATLAB<sup>®</sup>/Simulink software simulator, the algorithms were tested in open-loop condition, in a MATLAB<sup>®</sup> script.

Testing of the 3-step recursive nullspace-based thrust allocation algorithm showed that the thruster weighting matrices utilized in the algorithm are of great significance for the performance of the algorithm, when applied to the Minerva design case. In particular, the thruster weighting matrix utilized in the 2<sup>nd</sup> step allocating yaw, is important. Leaving the thruster weighting matrix as the identity matrix causes the proposed recursive 3-step algorithm to perform poorer compared to the pseudoinverse algorithm. Both with respect to achieved yaw moment, but also with respect to the total achieved surge, and sway force. The choice of making the recursive nullspace-based thrust allocation a 3-step algorithm for the Minerva, ensured that the full heave potential could be utilized, and at the same time obtaining the desired yaw priority.

The optimal (QP) thrust allocation algorithm was intended to be an intermediate step on the path to merge the recursive nullspace-based thrust allocation algorithm, with a method to apply optimal constraints to each step of the algorithm. This will result in a optimal recursive thrust allocation algorithm. The weakness of the recursive nullspace-based thrust allocation algorithm is how the thrusts are cut off. Within each step a gain,  $k \in [0,1]$ , is calculated to reduce the allocated thrust if the magnitude exceeds the capacity of the thrusters. This constraining method might not yield the optimal result, motivating the merging of the recursive nullspace-based algorithm, with an optimal algorithm. From testing it was found that the optimal thrust allocation algorithm could achieve DOF priority by penalizing the slack variable corresponding to that DOF. Although the weighted optimal thrust allocation algorithm is not recursive, the sought yaw prioritizing, is achieved by penalizing the corresponding slack variable in the optimal thrust allocation algorithm. Further, how the weighted optimal thrust allocation prioritizes the DOFs, are easily modified by changing the cost on the individual slack variable.

For the optimal recursive (recursive QP) thrust allocation, a two-step recursive algorithm was proposed. Allocating {Heave, Yaw} in the 1<sup>st</sup> step, and {Surge, Sway} in the 2<sup>nd</sup> step. The open-loop test of the algorithm showed good promise of functionality. Some unexplained "drop outs" in the thruster usage was observed. Proceeding with the closed-loop test, the optimal recursive algorithm displayed unacceptable performance during the 2<sup>nd</sup> test case. The thrusters were allocated such that the rotational direction constantly was changed back and forth, inducing instability in an otherwise stable system. Further, this oscillatory allocation would indeed reduce the lifetime of the thrusters. Thus, concluding from this behavior that the proposed optimal recursive algorithm, as implemented was flawed.

Further consideration of the weighted optimal algorithm vs. the optimal recursive algorithm; Where the weighted optimal algorithm searches the entire solution space, finding a global minima, only the first step of the optimal recursive algorithm searches over the same solution space. In the first step a solution is found, and this solution determines the set over which the second step can find its solution. Since the first step does not include all possible solutions, and then let the second step optimize over all solutions, the algorithm becomes suboptimal. However, what solution is found depend on the tuning of the cost function.

The proposed optimal recursive thrust allocation algorithm as implemented for the Minerva are seen to be flawed. Therefor there is no point in HIL-testing of the algorithm, as proposed in this thesis.

## 8.2 Recommendations for further work

The proposed 2-step optimal recursive thrust allocation algorithm as implemented for the Minerva, is proven to be flawed. The erroneous implementation is believed to be related to the 2<sup>nd</sup> step of the algorithm. Further investigations as to why the proposed 2-step optimal recursive thrust allocation algorithm fails, should be considered. An extension to a 3-step optimal recursive, allocating with the following prioritization {Heave, Yaw, Surge & Sway}, should be considered. Due to the uncoupled heave allocation for Minerva, this prioritization will not lead to any loss of yaw capability. If an 3-step optimal recursive thrust allocation algorithm are tested, special attention should be focused towards the thruster weighting matrix in the 2<sup>nd</sup> step, as discussed for the 3-step recursive nullspace-based algorithm. Depending on simulation results from the MATLAB<sup>®</sup>/Simulink simulator, further HIL-testing, and possibly model tests on Minerva should be considered.

Although the proposed weighted optimal thrust allocation algorithm is not recursive, the sought yaw prioritization is achieved. Further HIL-testing of this algorithm should be considered. If satisfactory results from HIL-testing are obtained, further model tests on the Minerva should also be considered.

The reported issue with time-varying center of gravity (CG) when utilizing the manipulator (with heavy load) on Minerva, causing CG to move in front of the lateral thruster, should be investigated.



In the Minerva control system the thrust allocation is achieved by unconstrained pseudoinverse. The commanded thrusts are then constrained at thruster level, i.e. cutting off the command at  $\pm 1450$  [rpm]. Constraining directly on the thruster, rather on the commands to the thrusters, not considering the thruster capability, may cause unexpected behavior. Thus, producing different resulting body forces and moment than what is expected from the allocated thrust, due to thruster saturation. Further, the thruster configuration matrix in the control system is badly scaled. This is due to augmented expressions for the un-actuated DOFs, most likely done to obtain system responses in roll, and pitch. Thus, the pseudoinverse matrix for thrust allocation is found by single value decomposition. How these issues affect the system should be further investigated.



# *Bibliography*

- R. Kajanus. Recursive nullspace-based thrust allocation for the ROV Minerva. Technical report, Norwegian University of Science and Technology (NTNU), Spring 2014.
- M. Kirkeby. Comparison of Controllers for Dynamic Positioning and Tracking of ROV Minerva. Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Spring 2010.
- M. Candeloro. Design of Observers for DP and Tracking of ROV "Minerva" with Experimental Results. Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Spring 2011.
- A. J. Sørensen, F. Dukan, M. Ludvigsen, D. A. Fernandes, and M. Candeloro. Development of Dynamic Positioning and Tracking System for the ROV Minerva. Chapter 6 of Further Advances in Unmanned Marine Vehicles, 2012.
- E. M. Tolpinrud. Development of Computer-based Control System with ROV example. Technical report, Norwegian University of Science and Technology (NTNU), Trondheim, Spring 2011.
- E. M. Tolpinrud. Development and Implementation of Computer-based Control System for ROV with Experimental Results. Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Spring 2012.
- T. I. Fossen. Handbook of marine craft hydrodynamics and motion control. Wiley, Chichester, 2011. ISBN 978-1-119-99413-8.
- O. M. Faltinsen. Sea Loads on Ships and Offshore Structures. Cambridge University Press, 1990.
- R. Skjetne. Model analysis for Minerva. Norwegian University of Science and Technology (NTNU), November 2013.
- A. J. Sørensen. Marine Control Systems; Propulsion and Motion Control of Ocean Structures. Dept. of Marine Technology, NTNU, Trondheim, January 2011. Lecture Notes in TMR4240.
- T. I. Fossen. Nonlinear Modeling and Control of Underwater Vehicles. PhD thesis, Norwegian University of Science and Technology (NTNU), Trondheim, 1991.
- E. M. Lewandowski. The Dynamics of Marine Carft. World Scientific Publishing Co, Singapore, 2004.
- K. D. Do and J. Pan. Control of Ships and Underwater Vehicles. Springer-Verlag London Limited, 2009.






- A. Tsourdos, B. White, and M. Shanmugavel. Cooperative Path Planning of Unmanned Aerial Vehicles. Cambridge University Press, 2010.
- T. A. Johansen and T. I. Fossen. Control Allocation - A Survey. Automatica, 49(5):1087–1103, 2012.
- Ø. K. Kjerstad, R. Skjetne, and B. O. Berge. Constrained Nullspace-Based Thrust Allocation for Heading Prioritized Stationkeeping of Offshore Vessels in Ice. Espoo, Finland, June 2013. POAC.
- R. Skjetne and Ø. K. Kjerstad. Recursive Nullspace-Based Control Allocation with Strict Prioritization for Marine Craft. CAMS, 2013.
- M. Diehl. Numerical Optimal Control. Technical report, OPTEC - Center of Excellence Optimization in Engineering (OPTEC), June 2011.
- E. Ruth. Propulsion control and thrust allocation on marine vessels. PhD thesis, Norwegian University of Science and Technology (NTNU), Trondheim, 2008.
- H. E. Wold. Thrust allocation for DP in ice. Master’s thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Spring 2013.
- A. Veksler, T. A. Johansen, F. Borrelli, and B. Realtsen. Cartesian thrust allocation algorithm with variable direction thrusters, turn rate limits and singularity avoidance. In IEEE Multi-conference on Systems and Control, Antibes Juan-les-Pins, France, October 2014.
- National Instruments (fall 214). Quadratic programming, NI online documentation. URL [http://zone.ni.com/reference/en-XX/help/371361H-01/gmath/quadratic\\_programming/](http://zone.ni.com/reference/en-XX/help/371361H-01/gmath/quadratic_programming/).
- MathWorks (fall 214). Quadratic programming, R2014b documentation. URL <http://se.mathworks.com/help/optim/quadratic-programming.html>.

## Appendix A

# *Minerva software CD*

This appendix contains a CD with; MATLAB<sup>®</sup> scripts, and MATLAB<sup>®</sup>/Simulink software simulator, with required toolboxes, to run the software simulator. The software are set-up for testing of the proposed thrust allocation algorithms {recursive 3-step, weighted optimal, optimal recursive}. Also included are the LabVIEW HIL simulator.

The MATLAB<sup>®</sup>/Simulink scripts, and block diagrams, are tested with a mac running OSX 10.10 (Yosemite), with MATLAB<sup>®</sup> R2014b. Whilst the HIL-simulator is tested on a PC running on Windows 7 Enterprise, with National Instruments LabVIEW 2012.

- [-]  Minerva software
  -  1 Matlab scripts
  -  2 Simulink simulator [Non-HIL]
  -  3 LabVIEW simulator [HIL]
  -  4 Toolboxes





## Appendix B

# *Simulation case 1; Pseudoinverse thrust allocation*

In this appendix, the system responses from running the system with the pseudoinverse thrust allocation are attached. The simulation case are repeated for readability;

- **Case 1:** DP in set-point  $\boldsymbol{\eta} = [10 \ 10 \ 50 \ 45]^\top$  subject to a current velocity,  $\boldsymbol{v}_{current} = 1.15$  [knots] with a direction North (0 [deg] in the NED frame).

These responses are needed in order to be able to compare the errors;

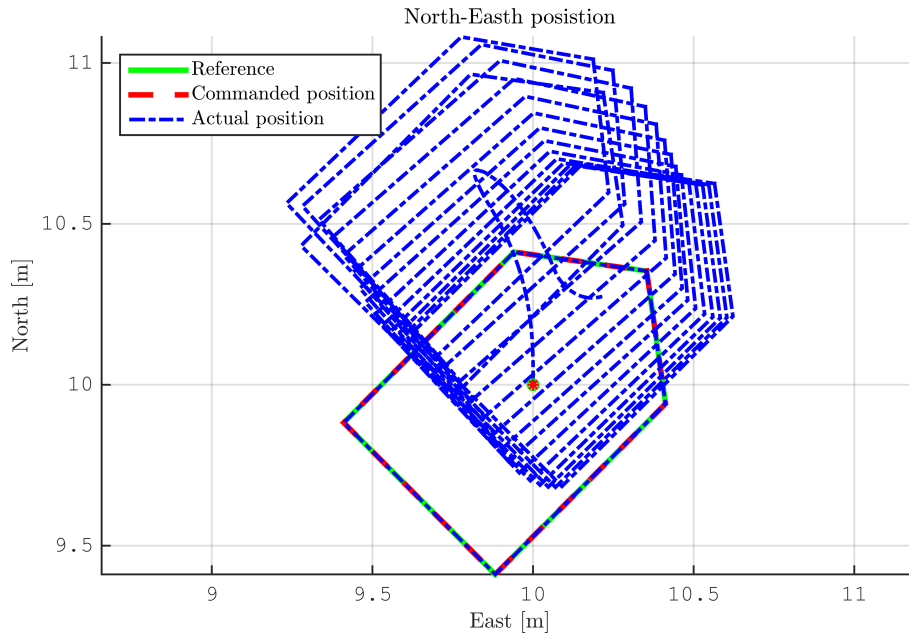
- Position error:  $\boldsymbol{\eta}_{error} = |\boldsymbol{\eta}_{des} - \boldsymbol{\eta}_{actual}|$ ,
- Force error:  $\boldsymbol{\tau}_{error} = |\boldsymbol{\tau}_{cmd} - \boldsymbol{\tau}_{alloc}|$ ,

as discussed in Chapter 7

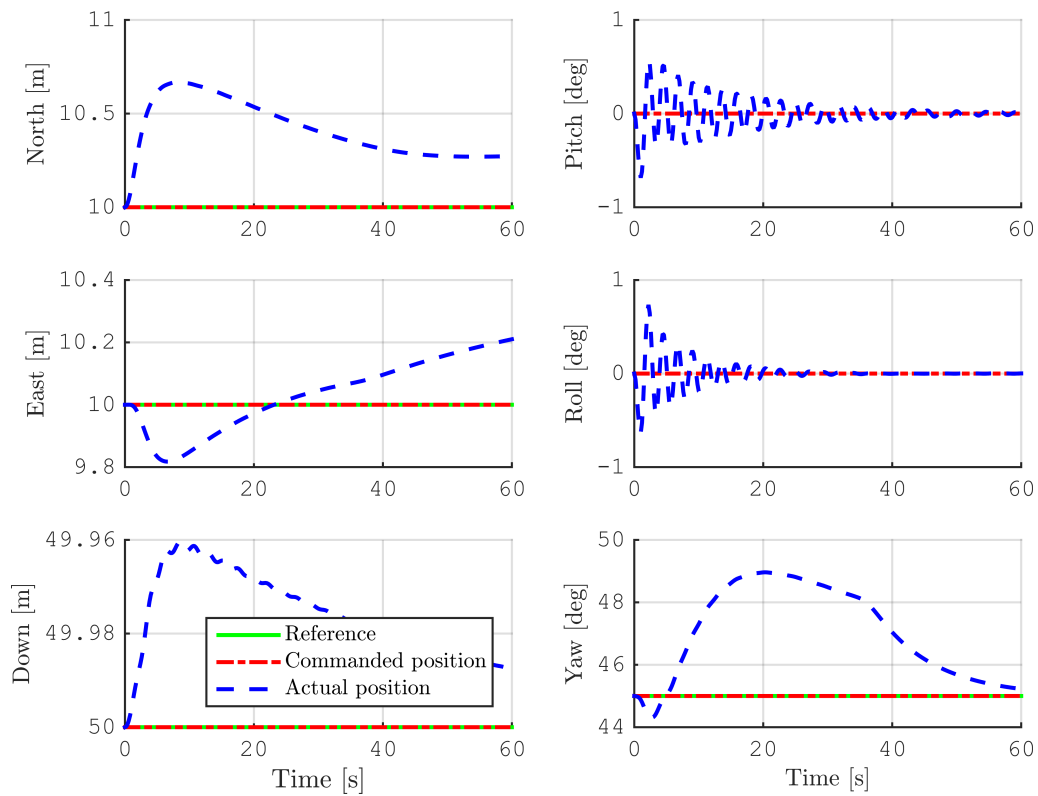
Figure B.1 shows the position of the ROV in the North-East plane. The Minerva are unable to keep the position as thruster saturation occurs, due to the strength of the current.

In Figure B.2 the 6 DOF response of the Minerva are seen; The 3 translational DOF, {1, 2, 3} in the left column. And the 3 rotational DOF, {4, 5, 6} in the right column. The responses in roll and pitch are small, and exponentially decaying. For the response in heave, it is seen that the ROV is rising towards the surface, before this is counteracted by the heave allocation. This is due to the modeled positive buoyancy of the ROV, as is the case for the actual ROV. Looking at the north and east positions, the ROV moves away from the set-point, before converging to a constant deviation in north position. The deviation in east position are steadily increasing. Further, the yaw angle the error increases, before converging towards the reference.

Figure B.3 shows the resulting body forces and moment, from the allocated thruster usage, seen in Figure B.4. When the starboard thruster ( $T_{stb}$ ) saturates after approx. 2 seconds, the pseudoinverse algorithm are unable to meet the desired forces and moment.

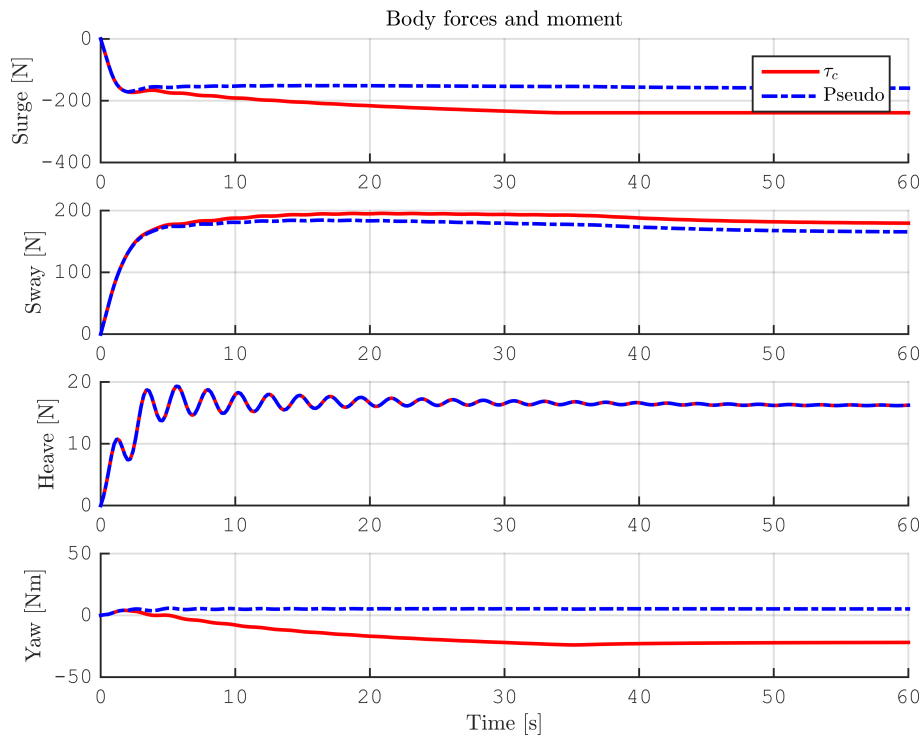


**Figure B.1:** Case 1, DP in set-point. Running the system with the pseudoinverse thrust allocation. Showing the ROV position in the North-East plane.

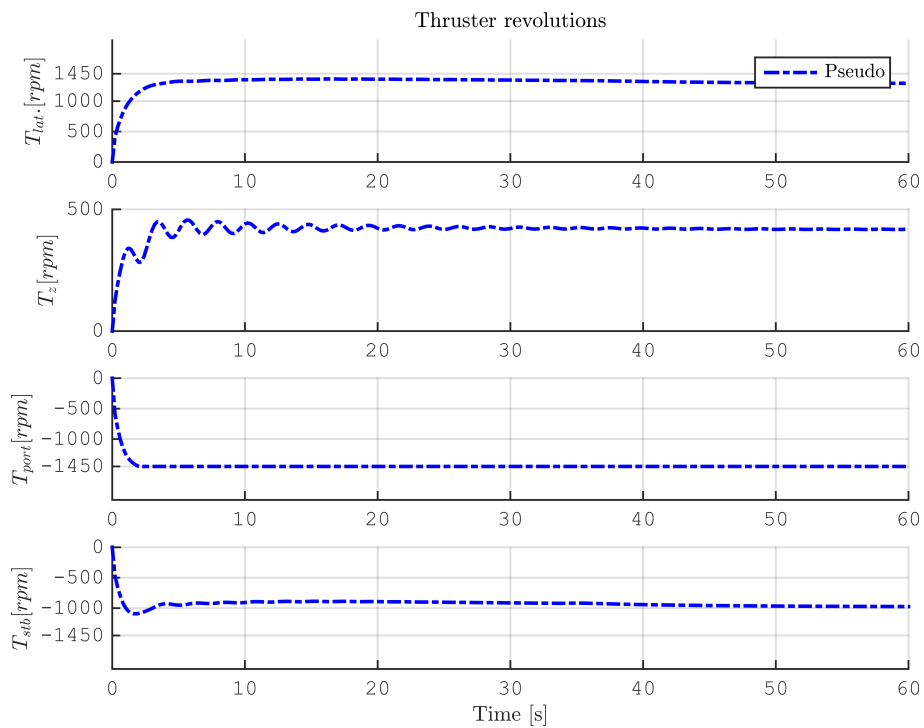


**Figure B.2:** Case 1, DP in set-point. Running the system with the pseudoinverse thrust allocation. Showing the 6 DOF response of the ROV.





**Figure B.3:** Case 1, DP in set-point. Running the system with the pseudoinverse thrust allocation. Showing the resulting body forces and moment.



**Figure B.4:** Case 1, DP in set-point. Running the system with the pseudoinverse thrust allocation. Showing allocated thruster revolutions.



## Appendix C

### *Simulation case 2;*

In this appendix additional plots from simulation of case 2, change in pose from  $\boldsymbol{\eta}_0 = [10 \ 10 \ 50 \ 20]^\top$  to  $\boldsymbol{\eta}_1 = [10 \ 10 \ 50 \ 65]^\top$  subject to a current velocity,  $\boldsymbol{v}_{current} = 1.05$  [knots] with a direction East (90 [deg] in the NED frame), from the cases study are attached. The following plots;

- North-East plot,
- 6 DOF responses,
- Resulting body forces,
- Allocated thruster rpm

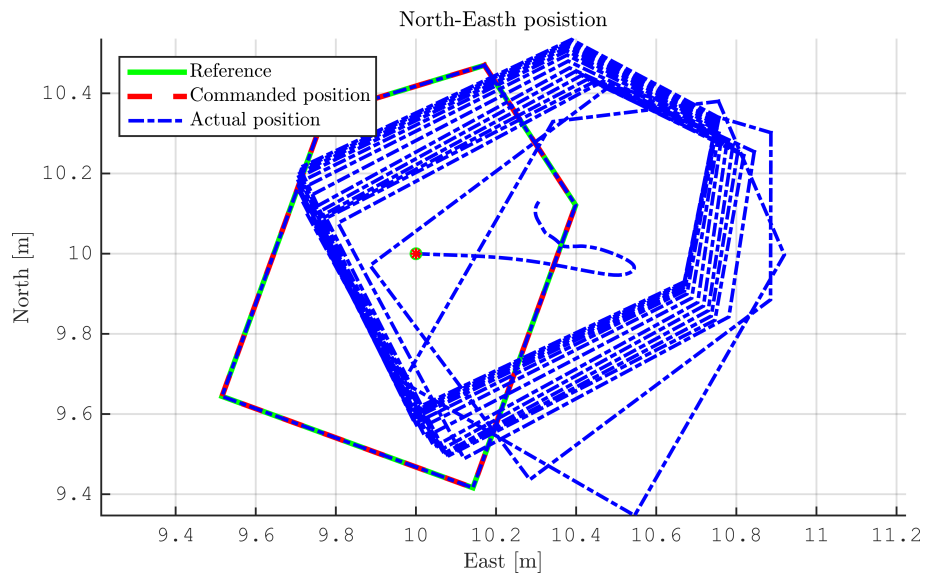
generated from running the system with the following thrust allocation algorithms;

- C.1 Pseudoinverse thrust allocation,
- C.2 3-step weighted recursive nullspace-based thrust allocation algorithm,
- C.3 Weighted optimal thrust allocation algorithm,
- C.4 Optimal recursive thrust allocation algorithm,

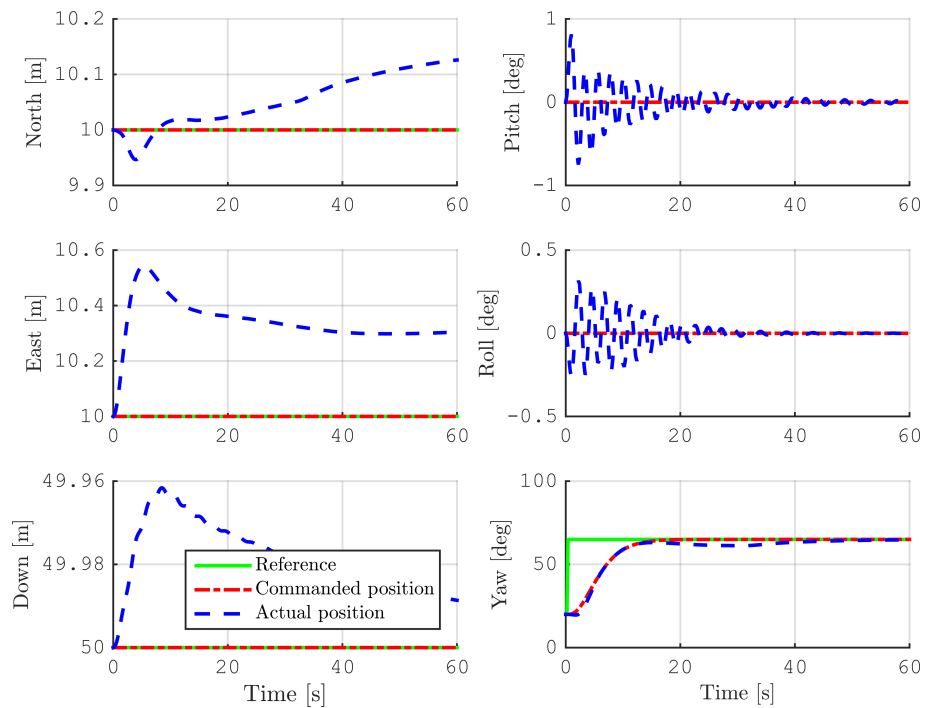
are seen in figures C.1 - C.20, respectively

For the optimal recursive algorithm in C.4, the results from simulating the case with a 50 % reduction in current velocity, has also been attached. As seen from figures C.17 - C.20, the performance of the algorithm improves, but are still displaying strange behavior.

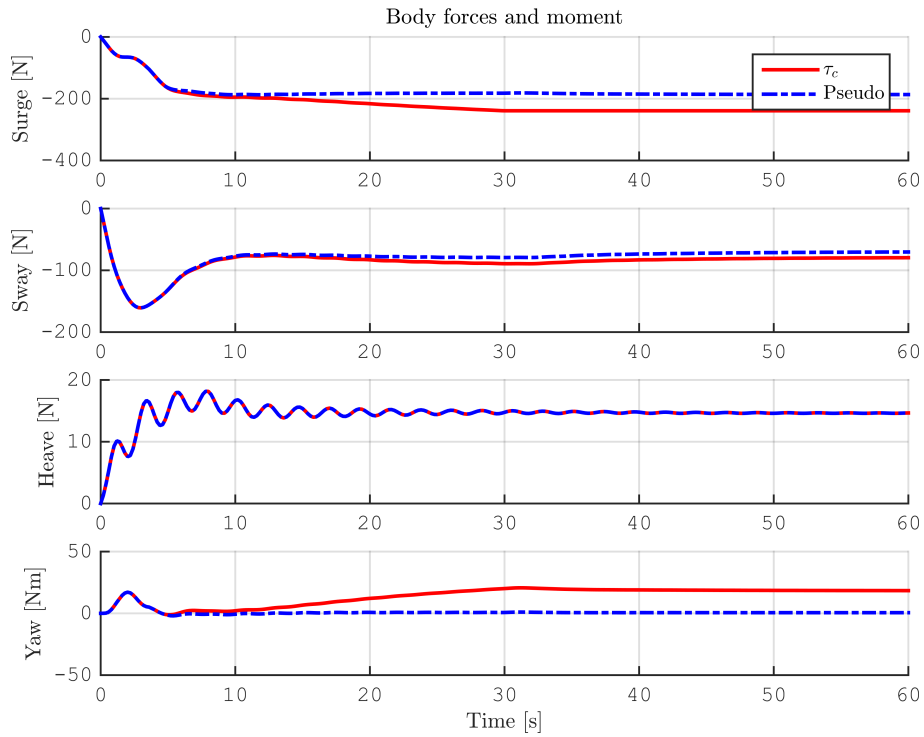
## C.1 Pseudoinverse thrust allocation



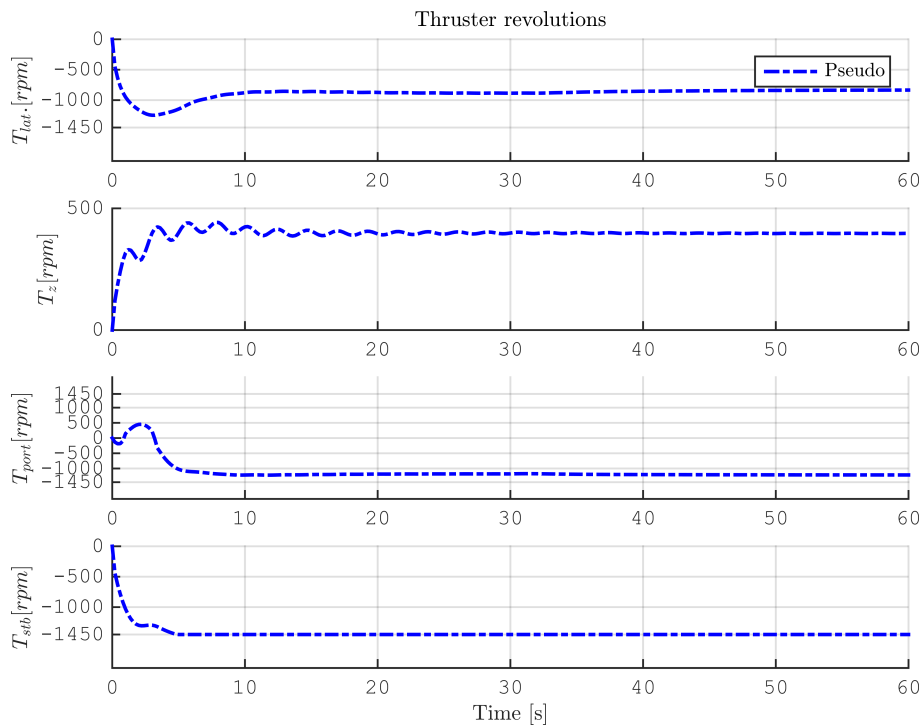
**Figure C.1:** Case 2, change in pose; Running the system with the pseudoinverse thrust allocation. Showing the ROV position in the North-East plane.



**Figure C.2:** Case 2, change in pose; Running the system with the pseudoinverse thrust allocation. Showing the 6 DOF response of the ROV.

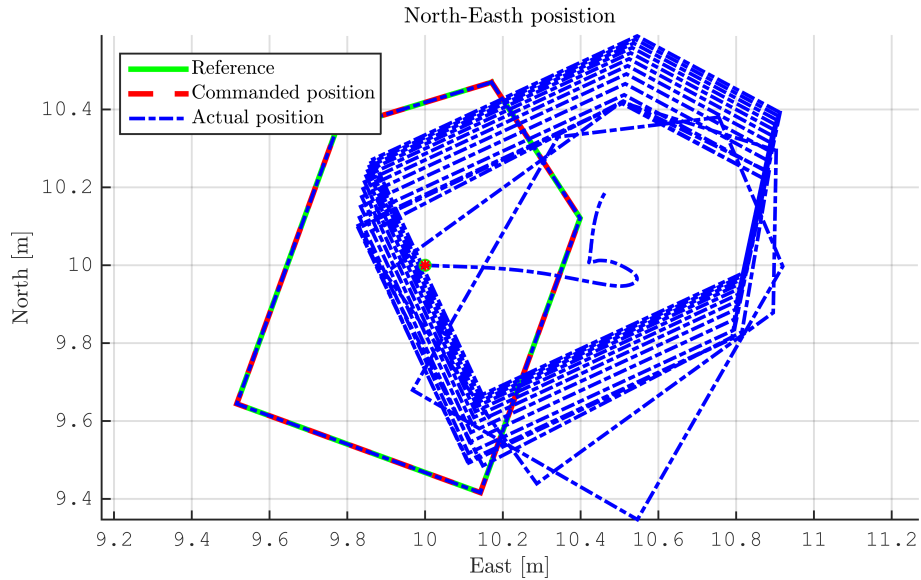


**Figure C.3:** Case 2, change in pose; Running the system with the pseudoinverse thrust allocation. Showing the resulting body forces and moment.

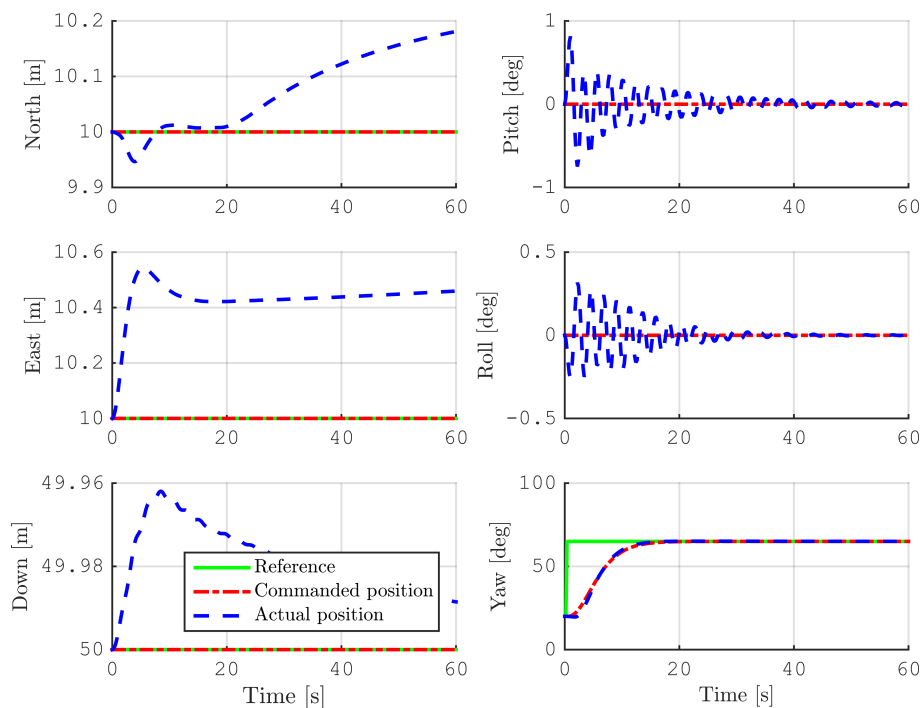


**Figure C.4:** Case 2, change in pose; Running the system with the pseudoinverse thrust allocation. Showing the allocated thruster revolution.

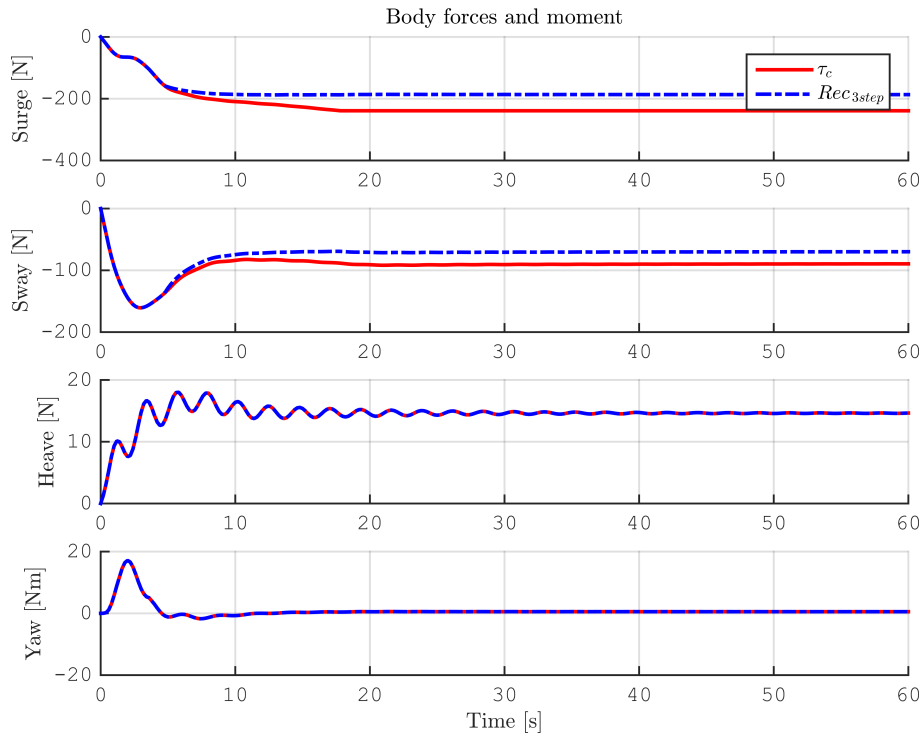
## C.2 Recursive 3-step nullspace-based thrust allocation



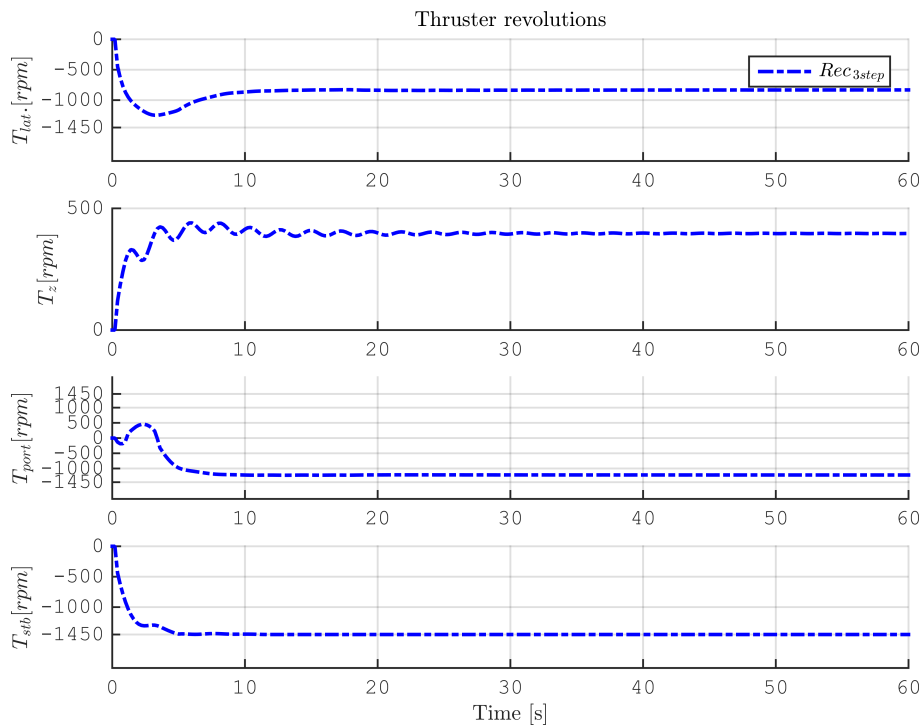
**Figure C.5:** Case 2, change in pose; Running the system with the recursive 3-step thrust allocation. Showing the ROV position in the North-East plane.



**Figure C.6:** Case 2, change in pose; Running the system with the recursive 3-step thrust allocation. Showing the 6 DOF response of the ROV.

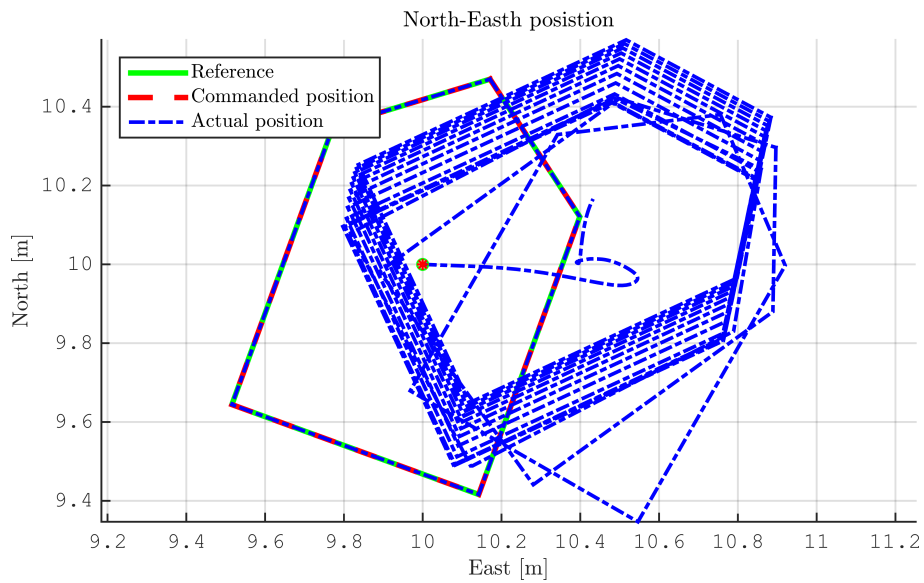


**Figure C.7:** Case 2, change in pose; Running the system with the recursive 3-step thrust allocation. Showing the resulting body forces and moment.

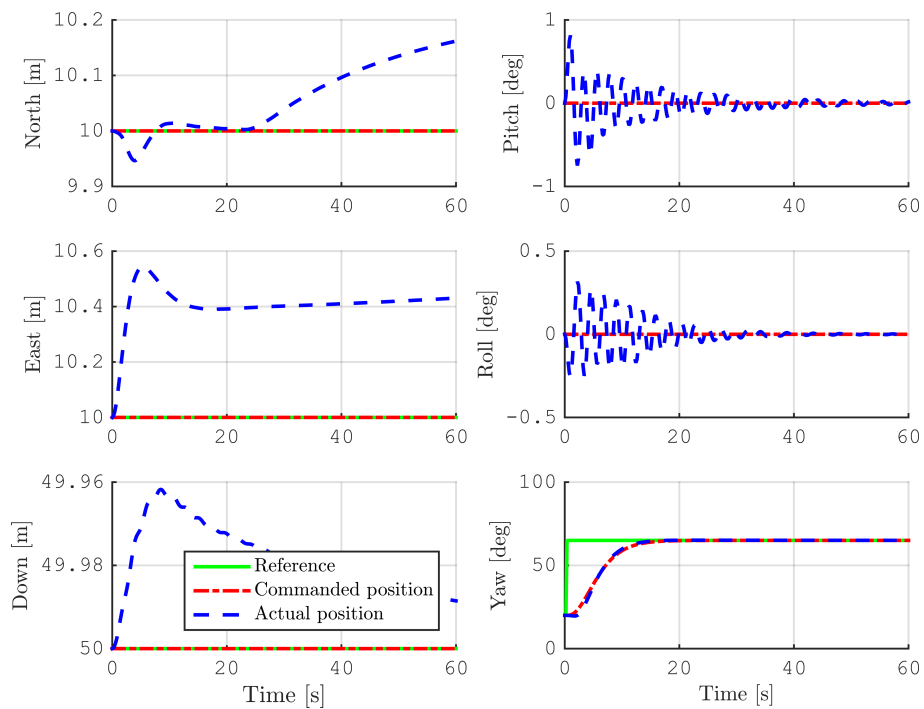


**Figure C.8:** Case 2, change in pose; Running the system with the recursive 3-step thrust allocation. Showing the allocated thruster revolution.

### C.3 Weighted optimal thrust allocation

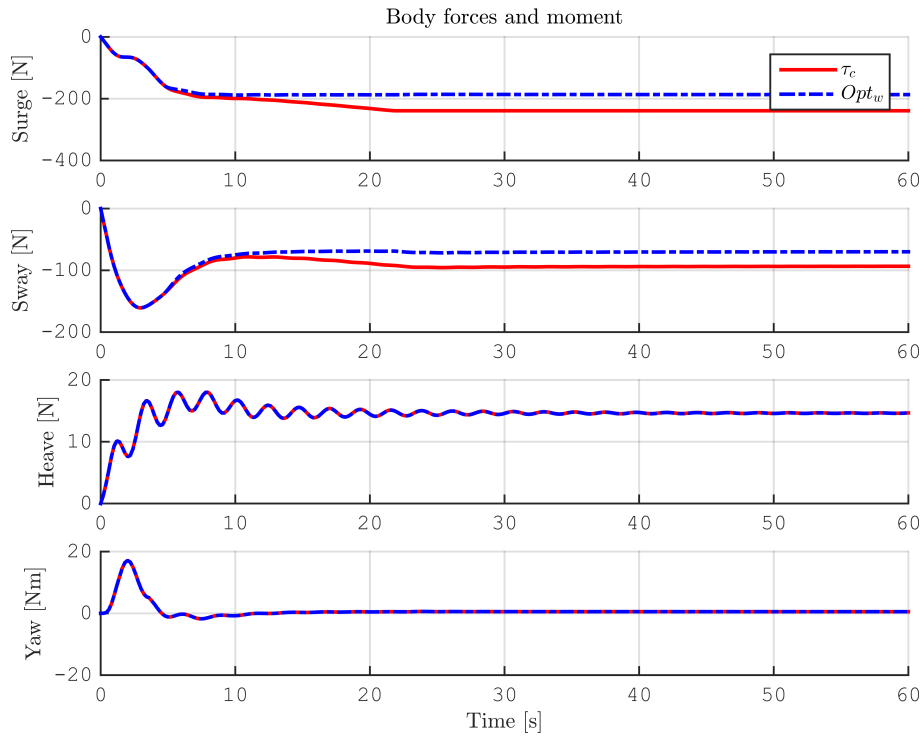


**Figure C.9:** Case 2, change in pose; Running the system with the weighted optimal thrust allocation. Showing the ROV position in the North-East plane.

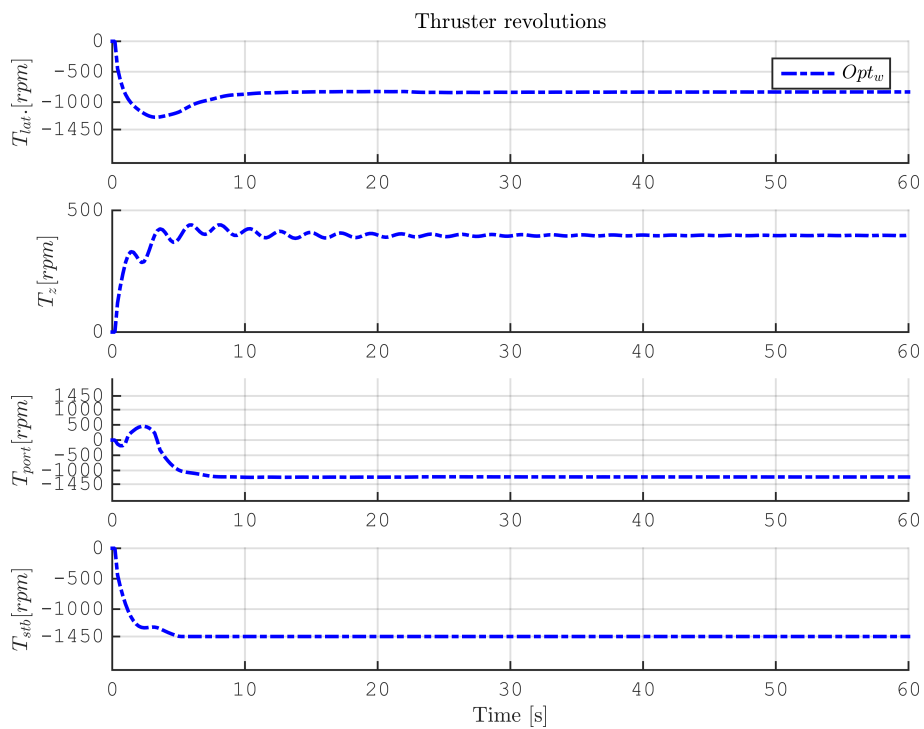


**Figure C.10:** Case 2, change in pose; Running the system with the optimal thrust allocation. Showing the 6 DOF response of the ROV.





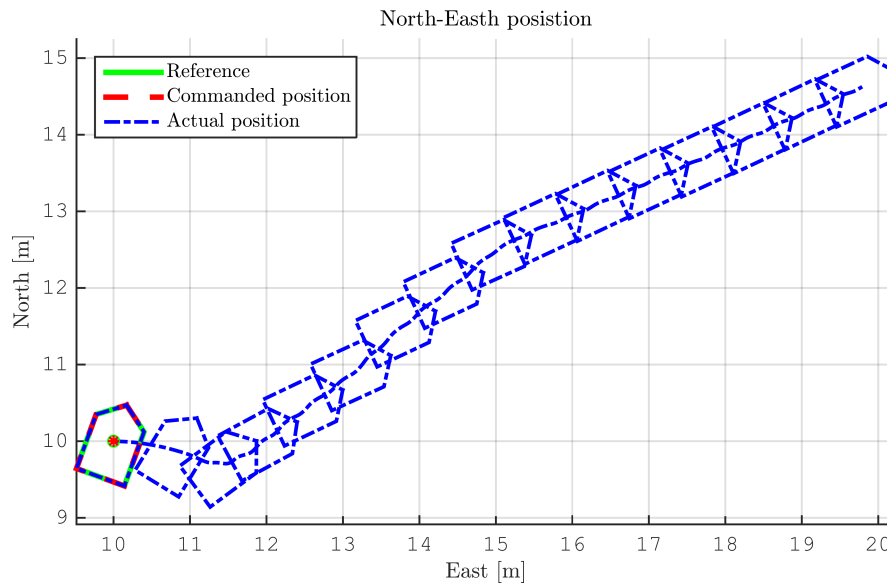
**Figure C.11:** Case 2, change in pose; Running the system with the optimal thrust allocation. Showing the resulting body forces and moment.



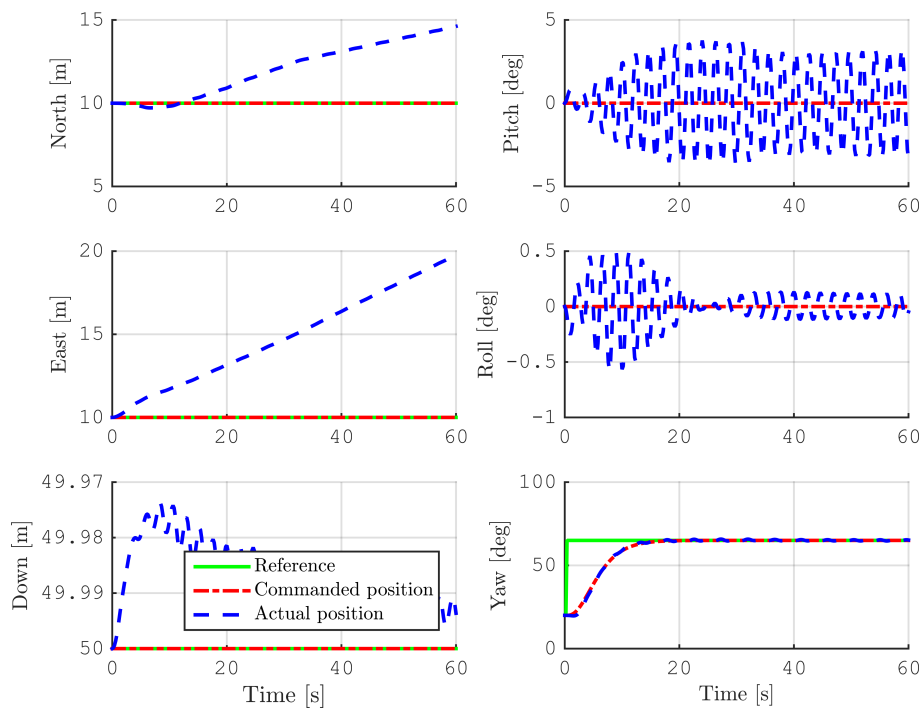
**Figure C.12:** Case 2, change in pose; Running the system with the optimal thrust allocation. Showing the allocated thruster revolution.

## C.4 Optimal recursive thrust allocation

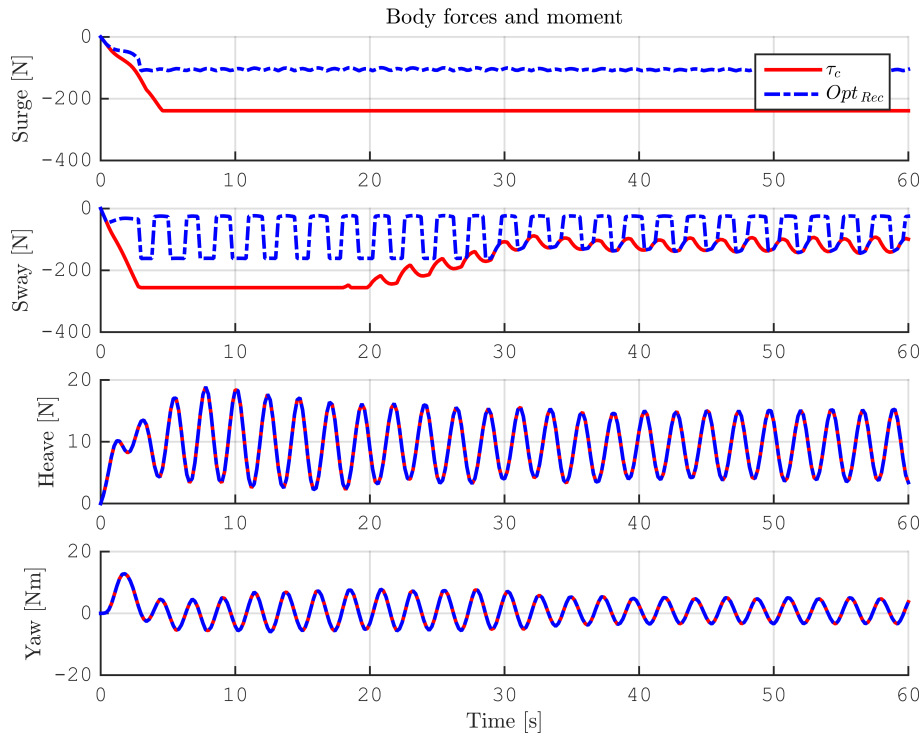
### C.4.1 Optimal recursive thrust allocation, full current velocity



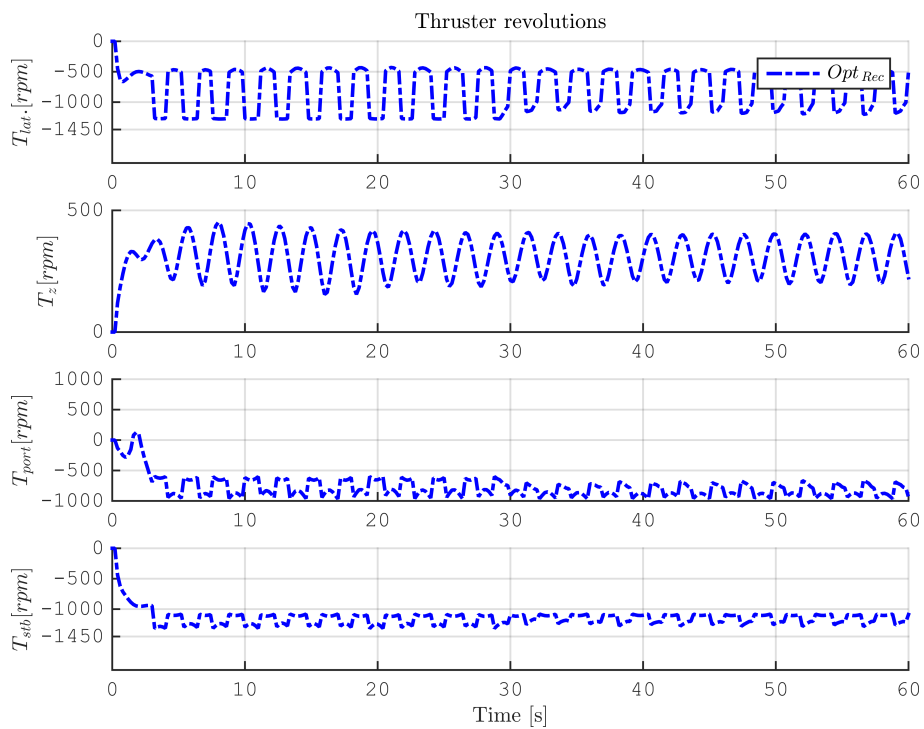
**Figure C.13:** Case 2, change in pose; Running the system with the optimal recursive thrust allocation. Showing the ROV position in the North-East plane.



**Figure C.14:** Case 2, change in pose; Running the system with the optimal recursive thrust allocation. Showing the 6 DOF response of the ROV.

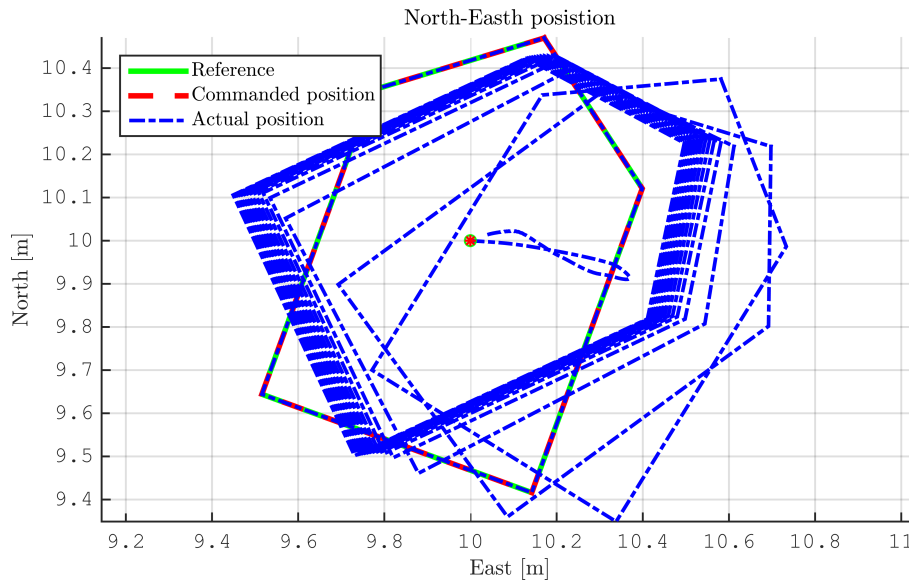


**Figure C.15:** Case 2, change in pose; Running the system with the Optimal recursive thrust allocation. Showing the resulting body forces and moment.

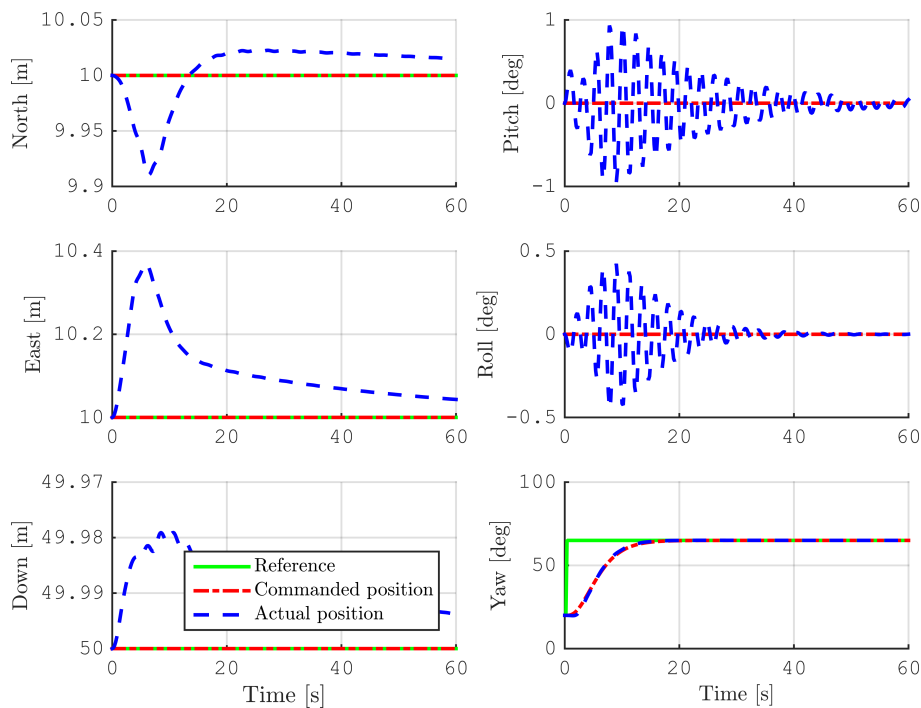


**Figure C.16:** Case 2, change in pose; Running the system with the Optimal recursive thrust allocation. Showing the allocated thruster revolution.

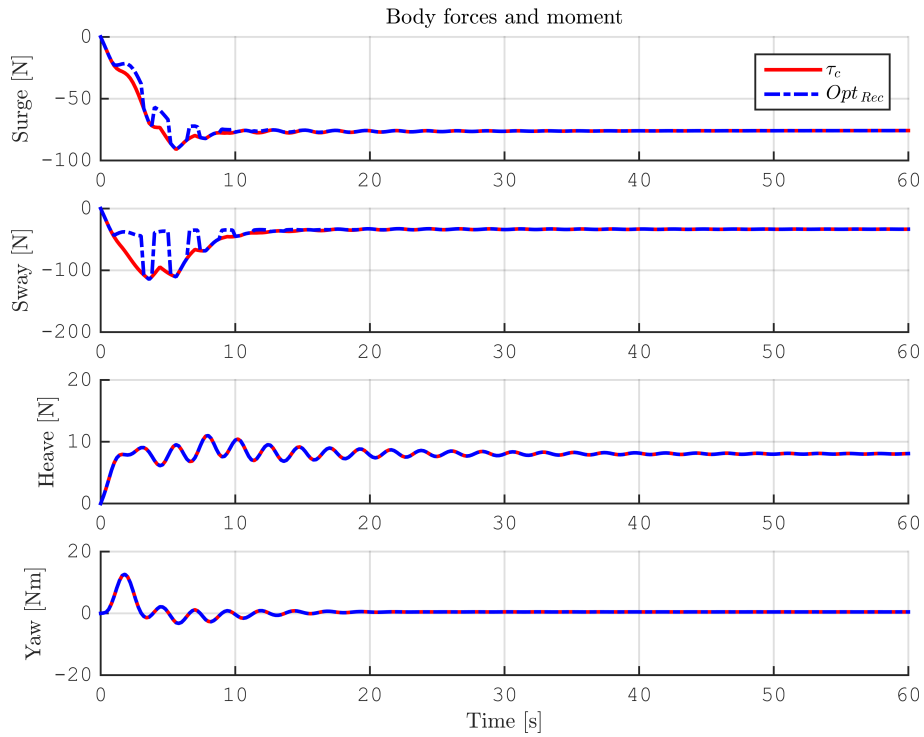
### C.4.2 Optimal recursive thrust allocation, reduced current velocity



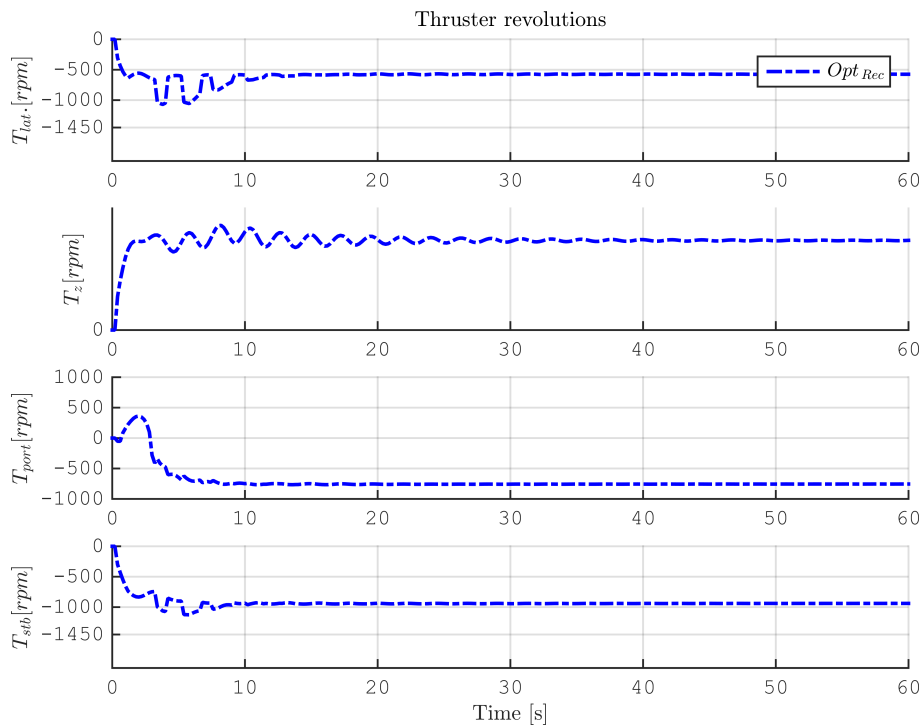
**Figure C.17:** Case 2, change in pose, reduces current velocity; Running the system with the optimal recursive thrust allocation. Showing the ROV position in the North-East plane.



**Figure C.18:** Case 2, change in pose, reduces current velocity; Running the system with the optimal recursive thrust allocation. Showing the 6 DOF response of the ROV.



**Figure C.19:** Case 2, change in pose, reduces current velocity; Running the system with the Optimal recursive thrust allocation. Showing the resulting body forces and moment.



**Figure C.20:** Case 2, change in pose, reduces current velocity; Running the system with the Optimal recursive thrust allocation. Showing the allocated thruster revolution.



## *Recursive 1-step nullspace-based thrust allocation*

### 1-step DOF prioritization, { XYZN }:

In the 1-step DOF prioritization, { XYZN }, the DOF's are equally prioritized. Left unconstrained, and defining the weighting matrix ( $\mathbf{W}$ ) as the identity matrix ( $\mathbf{I}$ ), the 1-step recursive algorithm produces the same result as thrust allocation by pseudoinverse.

Defining the weighted pseudoinverse for

$$\left(\mathbf{B}\right)_w^\dagger := \mathbf{W}^{-1}\mathbf{B}\left(\mathbf{B}^\top\mathbf{W}^{-1}\mathbf{B}\right)^{-1} \in \mathbb{R}^{4 \times 4}, \quad (\text{D.1})$$

The weighting matrices ( $\mathbf{W}$ ) are chosen as the identity matrix ( $\mathbf{I}_{4 \times 4}$ ) for simplicity.

$$\mathbf{v} = \left(\mathbf{B}\right)_w^\dagger \boldsymbol{\tau} \in \mathbb{R}^{4 \times 1}. \quad (\text{D.2})$$

Then, checking each actuator that the allocated thrust are feasible;

$$\mathbf{A}\mathbf{v} \leq \mathbf{f}_{sat}. \quad (\text{D.3})$$

If  $\mathbf{v}$  is infeasible, the magnitude is adjusted with a gain,  $k \in [0,1]$ , calculated from

$$\boldsymbol{\kappa}\mathbf{A}\mathbf{v} = \mathbf{f}_{sat} \Rightarrow \kappa_i = \begin{cases} \frac{f_{sat,i}}{\mathbf{a}_i^\top \mathbf{v}} & , \text{ if } \mathbf{a}_i^\top \mathbf{v} \neq 0 \\ 1 & , \text{ if } \mathbf{a}_i^\top \mathbf{v} = 0 \end{cases}, \quad i = 1, \dots, 8 \quad (\text{D.4})$$

where  $k$  is chosen as the minimum of the set

$$k = \min_{i=1,\dots,6} \{\kappa_i\}. \quad (\text{D.5})$$

Obtaining the allocated thrust vector, satisfying the constraint set as

$$\mathbf{u} = k\mathbf{v} = k \left(\mathbf{B}\right)_w^\dagger \boldsymbol{\tau} \in \mathbb{R}^{4 \times 1}. \quad (\text{D.6})$$